Kinematic Control of Continuum Manipulators Using a Fuzzy-Model-Based Approach

Peng Qi, *Student Member, IEEE*, Chuang Liu, Ahmad Ataka, H. K. Lam, *Senior Member, IEEE*, and Kaspar Althoefer, *Member, IEEE*

***Abstract*—Continuum manipulators are a rapidly emerging class of robots. However, due to the complexity of their mathematical models and modeling inaccuracies, the development of effective control systems is a particularly challenging task. This paper presents the first attempt on kinematic control of continuum manipulators using a fuzzy-model-based approach. A fuzzy controller is proposed for autonomous execution of end-effector trajectory tracking tasks for a continuum manipulator. Particularly, membership functions are employed to combine the linearized state-space models, to achieve, overall, a fuzzy model. The fuzzy model can help design the fuzzy controller; in our approach, this process is supported by a thorough stability analysis. This control methodology enables a solution with low computational requirements to this motion control problem—there is no need to continuously update the Jacobian of the continuum manipulator. The superior performance of this controller is validated in MATLAB simulations and compared with those of classical controllers found in the literature. The experiments on a rapid-prototyped continuum manipulator further verify the feasibility and the advantages of this fuzzy controller in the presence of modeling discrepancies and hardware inaccuracies.**

***Index Terms*—Closed-loop tracking control, continuum manipulators, fuzzy-model-based control, H_∞ performance, Jacobian, kinematics, nonlinear systems.**

I. Introduction

CONTINUUM manipulators are mainly characterized by their ability to continuously bend along the length of their structure; further, due to the inherent compliance, these manipulators demonstrate appealing flexibility and allow safe interactions in constrained environments. Although continuum robotics is still in its infancy, considerable current research is focusing on the development of both hardware and machine learning methods for such continuum robots, including design, modeling, control, and learning [1]. Continuum robot manipulators have made inroads into a rapidly growing number of applications across different sectors, ranging from industrial operations [2] to health care and domestic environments [3]. Compared with conventional manipulators with segmented rigid-links, the architecture concept and actuation principles for continuum manipulators are fundamentally different—they often mimic biological trunk or tentacle behaviors and manipulate objects in ways similar to how the biological role models do it. Particularly, continuum manipulators emphasise "whole arm manipulation" of a wide range of objects [4], which is even performed without prior knowledge of the shape of the object. Detailed surveys of the state-of-the-art and continuum manipulator designs are given in [1] and [5]. Frequently applied continuum manipulator structures are tendon-driven flexible backbone designs [6], pneumatically actuated bellow-integrated designs [7], concentric tube designs [8], and soft body structures with locally actuated cells [9]. Significant progress has not only been made in design but also in modeling, including both kinematics and dynamics. Chirikjian published their pilot research in the 1990s on continuum robot kinematics and dynamics [10]. Hannan and Walker provided the general kinematic model for continuum manipulators using the well-established Denavit–Hartenberg convention [11]. This approach adopted the modeling methodology originally used for traditional rigid-link manipulators to establish the continuum kinematics via virtual rigid-link kinematics. Other approaches focusing on static modeling give insight into the mechanics of continuum manipulators based on the elastic beam theory [12]. Among different kinematic models, the underlying methodology is the use of constant-curvature approximation [5]. It provides closed-form position and velocity kinematics which is the basis for real-time control and further motion planning.

There are several different approaches for the robotic control of continuum manipulators. Penning *et al.* investigated closed-loop control in both task space and joint space, resulting in improved end-point positioning accuracy of robot catheters [13]. Due to the nonlinear behavior and high flexibility of continuum manipulators, the system performance has shown to benefit from closed-loop control. Regarding the selection of task space or joint space control, generally, the task space controllers that employ a feedback loop to directly minimize task errors show some advantages [14]. In terms of kinematic control versus dynamic control, it is noted that kinematic control embedding the velocity-level kinematics is commonly utilized [15]–[17]; dynamic control has also been studied [18]; however, the lack of a well-understood efficient dynamic model of continuum manipulators limits its implementation. In [19], considering the problems of steady-state positioning errors and undesirable dynamic behaviors of continuum manipulators, a combined control system incorporating a position feedback and a modal-space

Manuscript received July 16, 2015; revised December 20, 2015; accepted March 17, 2016. Date of publication April 14, 2016; date of current version July 08, 2016. This work was supported in part by the European Commission's Seventh Framework Program under Grant 287728 in the framework of EU Project STIFF-FLOP and in part by the China Scholarship Council.

The authors are with the Department of Informatics, King's College London, London, WC2R 2LS, U.K. (e-mail: peng.qi@kcl.ac.uk; chuang.liu@kcl.ac.uk; ahmad_ataka_awwalur.rizqi@kcl.ac.uk; hak-keung.lam@kcl.ac.uk; kaspar.althoefer@kcl.ac.uk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIE.2016.2554078

controller was proposed and shown to be effective. At the intelligent control level, a distributed fuzzy controller was introduced as part of a control law in [20], which avoids the difficulties determined by the complexity of nonlinear integral-differential equations. Recently, a fuzzy logic methodology was proposed in [21] to design a nonlinear controller to regulate the end-effector of a continuum manipulator to a constant desired position. Furthermore, a neural network-based tracking controller was presented for a wide range of continuum manipulators [22], and it does not require an accurate manipulator dynamic model. In [23], an adaptive neural network controller was implemented achieving the end-effector position tracking control in real time with high accuracy. Likewise, considering the situation in which continuum manipulators interact with unknown obstacles and environments, a task space closed-loop controller, only based on empirical estimates of the real-time Jacobian but without using a model, is used for overcoming these disturbances [24]. For the future, we foresee autonomous execution of command tracking tasks based on practical control strategies approaching more new applications in the presence of continuum manipulators. Recently, an impressive implementation of a motion controller for a catheter (realized as a type of a continuum manipulator) for beating heart intracardiac surgery has been reported in [25], [26].

In this paper, we present a fuzzy-model-based approach for controlling a continuum manipulator. The controller was designed based on stability analysis for general continuous-time nonlinear systems [27]. We first derive the kinematic model and analyze its successive state-space model. Then, a fuzzy model is established to represent this state-space model by using a local approximation technique [28]. We design the fuzzy controller based on the stability conditions proposed in [27]. This controller enables the states of our continuum manipulator to track a desired reference model. The fuzzy-model-based approach can suppress the tracking error according to H_∞ performance based on the Lyapunov stability theory. Compared with open-loop feedforward control which is highly dependent on the accuracy of the model in real-time control, our closed-loop control is adequate to accommodate the online trajectory adjustments and has effective trajectory tracking capabilities. Although there commonly exists a certain modeling error between the established fuzzy model and the physical nonlinear model, the stability and performance of the specified tracking task can still be accomplished. Compared with other (pseudo)inverse Jacobian-based kinematic control systems, the proposed method does not require online updating of the Jacobian, nor rely on continuously updated estimations of the Jacobian. It responses to sensor inputs, thus also providing a closed-form low-computation solution of a motion control problem with respect to continuum manipulators. To the best of our knowledge, this is the first work of achieving task space closed-loop control proposed with respect to a continuum manipulator using a fuzzy-model-based approach.

The remainder of the paper is organized as follows: Section II presents a general continuum manipulator kinematics under the constant-curvature approximation. Section III introduces the methodology of the fuzzy-model-based approach for kinematic control of the continuum manipulator. A fuzzy controller is developed. In Section IV, simulation examples are given to show the feasibility and merits of the proposed controller. Section V reports the demonstration of the controller in real-time tracking tasks employing a rapid-prototyped continuum manipulator [29]. Two other traditional Jacobian-based control methods are implemented for comparison. Concluding remarks and a plan for future work are given in Section VI.

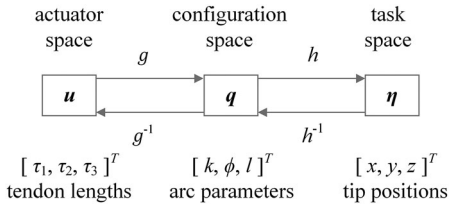


Fig. 1. Kinematic mapping and its decomposition of a continuum manipulator modeled using constant-curvature theory.

II. General Continuum Manipulator Kinematics

In this section, we give an overview of the constant-curvature-based continuum manipulator kinematics. The derived models form the basis for robot controller development.

The constant-curvature arc approximation has been frequently applied to the kinematic modeling of many continuum manipulators [11]. Different modeling approaches producing equivalent results of constant-curvature forward kinematics are reviewed and unified in [5]. Due to its simplifications in modeling, it enables an analytical closed-form relationship between actuator inputs and arc parameters useful for real-time control. Extensions of fundamental concept of constant-curvature kinematics, piecewise constant-curvature, and finite-fragmentation curvature modeling are proposed to fit the physical model of manipulators with a multi-section backbone or a variable section curvature [15]. The latter considers the backbone shape comprised of a finite number of small curved units and it is equivalent to modeling a single section with piecewise constant-curvature approximation. Unlike conventional rigid-link discrete manipulators [30], the use of joint variables and link parameters does not directly yield continuum kinematics. The elastic bending feature of a continuum manipulator leads kinematics to be decomposed into two submappings that link together with configuration variables (see Fig. 1). In order to lighten the notation, we drop all time-varying variable notations in this paper.

The two decomposed submapping portions are, respectively, described by a manipulator-specific kinematics $g : \mathbb{R}^n \to \mathbb{R}^n$, $\boldsymbol{u} \mapsto \boldsymbol{q}$, and a manipulator-independent kinematics $h : \mathbb{R}^n \to \mathbb{R}^n$, $\boldsymbol{q} \mapsto \boldsymbol{\eta}$. The former varies with different actuation manners (although sometimes there exists certain correlation among common actuation strategies), while the latter is totally general and applies to all the individual sections of a continuum manipulator under the assumption of constant curvature. Hence, the complete kinematics mapping that computes the end-effector's pose $\boldsymbol{\eta}$ depending on the actuator state $\boldsymbol{u}$ is given by $f = h(g(\cdot))$.

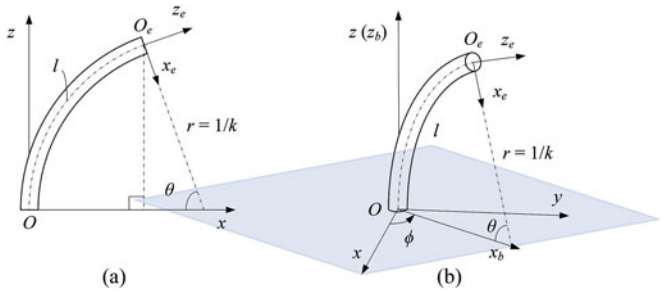


Fig. 2. Diagram of a continuum manipulator bending in (a) 2-D space and in (b) 3-D space. The configuration variables and different coordinate systems are illustrated.

The function f defines a chaining process where the output of the function g becomes the input of the function h.

Without limiting generality, the actuator space variables are chosen as the most direct actuation—tendon-driven design, where an arc is shaped by tendons. Herein, the three tendon lengths are written in vector form $\boldsymbol{u} = [\tau_1, \tau_2, \tau_3]^T$.

The arc parameters are represented by the curvature (k), rotational angle (ϕ), and arc length (l) (see Fig. 2). The configuration space triplets above are all functions of the actuator variables, i.e., $\boldsymbol{q} = [k(\boldsymbol{u}), \phi(\boldsymbol{u}), l(\boldsymbol{u})]^T$.

Furthermore, the arc geometry provides the relationships $\theta = k \cdot l$ and $r = 1/k$, which enable the calculation of the arc bending angle θ and radius r. Regarding the pose representation, we only specify the position of the end-effector for the purpose of motion control, which in 3-D Euclidean space is defined by the vector $\boldsymbol{\eta} = [x, y, z]^T$.

A. Coordinate Systems

With different coordinate frame choices, the derived kinematic mapping would be diverse in form. In order to describe the position of the end-effector in the universe, the reference coordinate system must be first established. Likewise, additional moving frames attached to the continuum manipulator are introduced. All the coordinate systems with respect to a single-section continuum manipulator (see Fig. 2) are described below.

1) *Reference coordinate system* $\{xyz\}$, for convenience, is fixed to the proximal end of the continuum manipulator with its z-axis tangent to the backbone curve of the bending manipulator and pointing toward the distal end. The xy plane is perpendicular to the bending plane.
2) *Bendingsystem* $\{x_b y_b z_b\}$ is defined such that continuum manipulator always bends in the $x_b z_b$ plane. The origin O_b coincides with the origin O and the z_b-axis is collinear with the z-axis.
3) *End-effector coordinate system* $\{x_e y_e z_e\}$ is attached to the tip of the continuum manipulator. The origin O_e is at the center of the tip cross section and the z_e-axis tangent to the backbone curve, or equivalently normal to the tip cross section. For convenience, the $x_e z_e$ plane is coplanar with the bending plane $x_b z_b$.

B. Manipulator-Independent Submapping

Once the aforementioned coordinate systems are established, the problem of deriving the manipulator-independent submapping is transformed into solving the mathematics to describe the end-effector coordinate system $\{x_e y_e z_e\}$ relative to the reference coordinate system $\{xyz\}$. Thus, a parameterized homogenous transformation can be used as

$$ {}^{o}\mathbf{T}_e(\boldsymbol{q}) = \begin{bmatrix} {}^{o}\mathbf{R}_e(\boldsymbol{q}) & {}^{o}\boldsymbol{p}_e(\boldsymbol{q}) \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{1} $$

where the 4×4 homogeneous transformation matrix ${}^{o}\mathbf{T}_e(\boldsymbol{q})$ constitutes the standard representation of the special Euclidean group $SE(3)$ with the effect of transforming the coordinate frame $\{x_e y_e z_e\}$ to the reference coordinate frame $\{xyz\}$, the matrix ${}^{o}\mathbf{T}_e(\boldsymbol{q})$ is constructed by a 3×3 rotation matrix ${}^{o}\mathbf{R}_e(\boldsymbol{q})$ and a 3×1 position vector ${}^{o}\boldsymbol{p}_e(\boldsymbol{q})$, ${}^{o}\mathbf{R}_e(\boldsymbol{q})$ is an element of the special orthogonal group $SO(3)$ and denotes the orientation of the coordinate frame $\{x_e y_e z_e\}$ relative to the reference coordinate frame $\{xyz\}$, and the three-vector ${}^{o}\boldsymbol{p}_e(\boldsymbol{q})$ is an element of the translation group $T(3)$ and denotes the position of the origin O_e relative to the reference coordinate frame $\{xyz\}$.

Each component of the homogeneous transformation matrix is derived as follows. The columns of the rotation matrix ${}^{o}\mathbf{R}_e(\boldsymbol{q})$ can be obtained by writing the unit vectors that define directions of the principle axes of end-effector coordinate system $\{x_e y_e z_e\}$ in reference coordinate frame $\{xyz\}$. ${}^{o}\boldsymbol{p}_e(\boldsymbol{q})$ is a pure position vector translating the point in space. The operations of $SE(3)$ can be performed through the matrix multiplication, with the transformation composition implemented. Therefore, ${}^{o}\mathbf{T}_e(\boldsymbol{q}) = {}^{o}\mathbf{T}_b(\boldsymbol{q}){}^{b}\mathbf{T}_e(\boldsymbol{q})$. Note here that the operations of $SE(3)$ is noncommutative, hence the order for composition is important. Composition of the homogeneous transformation matrix ${}^{o}\mathbf{T}_e(\boldsymbol{q})$ is accomplished as

$$ \begin{aligned} {}^{o}\mathbf{T}_e(\boldsymbol{q}) &= \begin{bmatrix} {}^{o}\mathbf{R}_b(\phi) & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} {}^{b}\mathbf{R}_e(k,l) & {}^{b}\boldsymbol{p}_e(k,l) \\ \mathbf{0}^T & 1 \end{bmatrix} \\ &= \begin{bmatrix} {}^{o}\mathbf{R}_b(\phi){}^{b}\mathbf{R}_e(k,l) & {}^{o}\mathbf{R}_b(\phi){}^{b}\boldsymbol{p}_e(k,l) \\ \mathbf{0}^T & 1 \end{bmatrix} \end{aligned} \tag{2} $$

where one homogeneous transformation matrix describing the frame $\{x_b y_b z_b\}$ relative to the reference frame $\{xyz\}$ is

$$ {}^{o}\mathbf{T}_b(\boldsymbol{q}) = \begin{bmatrix} {}^{o}\mathbf{R}_b(\phi) & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix} $$

and another one describing the frame $\{x_e y_e z_e\}$ relative to the frame $\{x_b y_b z_b\}$ is

$$ {}^{b}\mathbf{T}_e(\boldsymbol{q}) = \begin{bmatrix} {}^{b}\mathbf{R}_e(k,l) & {}^{b}\boldsymbol{p}_e(k,l) \\ \mathbf{0}^T & 1 \end{bmatrix}. $$

Referring to Fig. 2(a), in the 2-D bending model of a continuum manipulator, the position vector ${}^{b}\boldsymbol{p}_e(k,l)$ can be

written as

$$ {}^{b}\boldsymbol{p}_e(k,l) = \begin{bmatrix} \dfrac{1-\cos(k\cdot l)}{k} \\ 0 \\ \dfrac{\sin(k\cdot l)}{k} \end{bmatrix} \tag{3} $$

and both the rotation matrix ${}^{b}\mathbf{R}_e(k\cdot l)$ representing a rotation of $\theta(=k\cdot l)$ about y_b-axis and the rotation matrix ${}^{o}\mathbf{R}_b(\phi)$ representing the bending plane rotation of ϕ about z-axis can be written in the form of the corresponding rotation matrices.

Therefore, the complete homogeneous transformation matrix ${}^{o}\mathbf{T}_e(\boldsymbol{q})$ can be calculated. When in the case of only considering the end-effector position representation, the manipulator-independent submapping h only takes the first three elements of the last column of ${}^{o}\mathbf{T}_e(\boldsymbol{q})$, i.e., $h = {}^{o}\boldsymbol{p}_e(\boldsymbol{q})$

$$ h = {}^{o}\mathbf{R}_b(\phi)\,{}^{b}\boldsymbol{p}_e(k,l) = \begin{bmatrix} \dfrac{\cos\phi(1-\cos(k\cdot l))}{k} \\ \dfrac{\sin\phi(1-\cos(k\cdot l))}{k} \\ \dfrac{sin(k\cdot l)}{k} \end{bmatrix}. \tag{4} $$

So far, we complete the derivation of the manipulator-independent submapping based on a homogeneous transformation. Furthermore, using the derived 4×4 homogeneous transformation matrix ${}^{o}\mathbf{T}_e(\boldsymbol{q})$, any vectors ${}^{e}\boldsymbol{s}$ expressed relative to the end-effector coordinate system $\{x_e y_e z_e\}$ can be expressed relative to the reference coordinate system $\{xyz\}$. Thus,

$$ \begin{bmatrix} {}^{o}\boldsymbol{s} \\ 1 \end{bmatrix} = {}^{o}\mathbf{T}_e(\boldsymbol{q}) \begin{bmatrix} {}^{e}\boldsymbol{s} \\ 1 \end{bmatrix}. \tag{5} $$

Equation (5) can be used to solve for the pose representation including end-effector positions and orientations.

C. Manipulator-Specific Submapping

Now to find the manipulator-specific submapping, we decide to adopt the three-tendon-driven actuation strategy. First, we assume that all the three tendons are in tension during the manipulator articulation and there is no slack. Referring to [5] and [15], all the defined configuration variables can be expressed with respect to tendon actuation variables as written by [5, eqs. (17)–(19)].

This mathematic model is most frequently applied to tendon-driven continuum manipulators and also to any continuously bending actuator, for example, the bellow-like actuators in Festo's bionic handling assistant [7], [15]. Hereby, we complete the manipulator-specific submapping g and end the forward kinematics. Upon the analytical kinematic modeling, the inverse mapping can be further derived and in case of the current simplified model, both the submappings g^{-1} and h^{-1} can be produced analytically by solving the nonlinear equations defined by forward mappings g and h.

D. Jacobian

The Jacobian is a multidimensional form of partial derivatives with respect to time of the forward kinematics. It reveals the velocity-level forward kinematics that the actuator velocities to the spatial velocity of the end-effector. Given the forward kinematics of the form

$$ \boldsymbol{\eta} = f(\boldsymbol{u}) = h(g(\boldsymbol{u})) \tag{6} $$

then, the velocity kinematics is derived as

$$ \dot{\boldsymbol{\eta}} = \frac{\partial f}{\partial \boldsymbol{u}}\dot{\boldsymbol{u}} = \frac{\partial h}{\partial \boldsymbol{q}}\frac{\partial g}{\partial \boldsymbol{u}}\dot{\boldsymbol{u}}. \tag{7} $$

This yields the Jacobian matrix equal to

$$ \mathbf{J}(\boldsymbol{u}) = \frac{\partial h}{\partial \boldsymbol{q}}\frac{\partial g}{\partial \boldsymbol{u}}, \Rightarrow \dot{\boldsymbol{\eta}} = \mathbf{J}(\boldsymbol{u})\dot{\boldsymbol{u}} \tag{8} $$

where $\mathbf{J}(\boldsymbol{u})$ is a time-varying 3×3 matrix, whose elements are nonlinear functions of instant actuator states expressed by $\boldsymbol{u}$.

In (8), the left component of the Jacobian represents the Jacobian $\mathbf{J}_h(\boldsymbol{q})$ of the manipulator-independent portion of kinematics and the right component of the Jacobian represents the Jacobian $\mathbf{J}_g(\boldsymbol{u})$ of the manipulator-independent portion of kinematics. We get both explicit Jacobian matrices as (9) and (10), shown at the bottom of the page, where d is the radius of the

$$ \mathbf{J}_h(\boldsymbol{q}) = \begin{bmatrix} \dfrac{\cos\phi(k\cdot l\sin(k\cdot l)+\cos(k\cdot l)-1)}{k^2} & -\dfrac{\sin\phi(1-\cos(k\cdot l))}{k} & \cos\phi\sin(k\cdot l) \\ \dfrac{\sin\phi(k\cdot l\sin(k\cdot l)+\cos(k\cdot l)-1)}{k^2} & \dfrac{\cos\phi(1-\cos(k\cdot l))}{k} & \sin\phi\sin(k\cdot l) \\ \dfrac{k\cdot l(\cos(k\cdot l)-\sin(k\cdot l)}{k^2} & 0 & \cos(k\cdot l) \end{bmatrix} \tag{9} $$

$$ \mathbf{J}_g(\boldsymbol{u}) = \begin{bmatrix} \dfrac{3(\tau_1\tau_2+\tau_1\tau_3-\tau_2^2-\tau_3^2)}{d\tau_{\text{sum}}^2\tau_{\text{sqrt}}} & -\dfrac{3(\tau_1^2-\tau_1\tau_2-\tau_2\tau_3+\tau_3^2)}{d\tau_{\text{sum}}^2\tau_{\text{sqrt}}} & -\dfrac{3(\tau_1^2-\tau_1\tau_3-\tau_2\tau_3+\tau_2^2)}{d\tau_{\text{sum}}^2\tau_{\text{sqrt}}} \\ \dfrac{\sqrt{3}(\tau_3-\tau_2)}{2\tau_{\text{sqrt}}^2} & \dfrac{\sqrt{3}(\tau_1-\tau_3)}{2\tau_{\text{sqrt}}^2} & \dfrac{\sqrt{3}(\tau_2-\tau_1)}{2\tau_{\text{sqrt}}^2} \\ \dfrac{1}{3} & \dfrac{1}{3} & \dfrac{1}{3} \end{bmatrix} \tag{10} $$

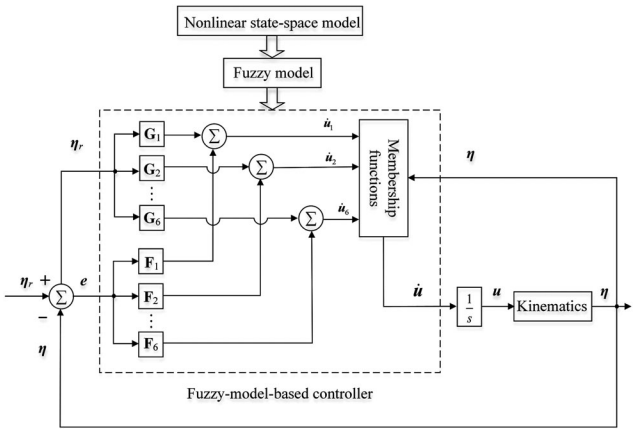


Fig. 3. Overview of a task space closed-loop tracking control system using the fuzzy-model-based approach for continuum manipulators. $\boldsymbol{\eta}_r$ represents the desired end-effector trajectory in task space. $\mathbf{G}_j$ and $\mathbf{F}_j$, (j = 1, 2, ..., 6), are feedback gains. ($\dot{}$) is a time derivative and $\dot{\boldsymbol{u}}$ denotes the end-effector motion velocity. The feedback information is acquired with position sensors.

cross section of continuum manipulator; $\tau_{\text{sum}} = \tau_1 + \tau_2 + \tau_3$;

$$\tau_{\text{sqrt}} = \sqrt{\tau_1^2 + \tau_2^2 + \tau_3^2 - \tau_1\tau_2 - \tau_1\tau_3 - \tau_2\tau_3}.$$

III. Kinematic Control Using the Fuzzy-Model-Based Approach

The objective of the kinematic control task is to find a solution with respect to the actuation space variables to enable the end-effector of the continuum manipulator to track a desired trajectory. Fig. 3 illustrates the control architecture using fuzzy-model-based approach. Afterward, we first introduce the design procedures step-by-step according to the literature. Then, the state-space model with respect to the continuum manipulator is proposed, based on which, the fuzzy model can be developed. We specify two different trajectory tracking tasks and accordingly design two sets of feedback gains in the fuzzy controllers. Details about control synthesis are shown in the following subsections.

A. Polynomial Fuzzy-Model-Based Stability Conditions

The methodology of the fuzzy-model-based control is summarized in this subsection based on [27].

1) Polynomial Fuzzy Model: In order to apply the fuzzy-model-based stability analysis, the polynomial fuzzy model is employed to represent the system state model of the continuum manipulator. The polynomial fuzzy model is constructed by using membership functions to blend the local polynomial models. The p-rule polynomial fuzzy model describing the behavior of a general nonlinear model can be defined as [31]

$$\dot{\boldsymbol{x}} = \sum_{i=1}^{p} w_i(\boldsymbol{y})(\mathbf{A}_i(\boldsymbol{x})\hat{\boldsymbol{x}}(\boldsymbol{x}) + \mathbf{B}_i(\boldsymbol{x})\boldsymbol{v}) \tag{11}$$

$$\boldsymbol{y} = \mathbf{C}\hat{\boldsymbol{x}}(\boldsymbol{x}) \tag{12}$$

where $\boldsymbol{x}$ denotes the system state vector, $\boldsymbol{y}$ denotes the output vector, $w_i(\boldsymbol{y})$ is the normalized grade of membership, $\mathbf{A}_i(\boldsymbol{x})$ and $\mathbf{B}_i(\boldsymbol{x})$ are the known polynomial system and input matrices, respectively, $\hat{\boldsymbol{x}}(\boldsymbol{x})$ is a vector of monomials in $\boldsymbol{x}$, $\boldsymbol{v}$ is the input vector, and $\mathbf{C}$ is a constant output matrix. It is assumed that $\hat{\boldsymbol{x}}(\boldsymbol{x}) = \mathbf{0}$ iff $\boldsymbol{x} = \mathbf{0}$.

2) Reference Model: The reference model mathematically describes the desired trajectory. It is specified by users and later is utilized in the fuzzy-model-based stability analysis for the tracking control of a continuum manipulator. The reference model is defined as follows [27]:

$$\dot{\boldsymbol{x}}_r = \mathbf{A}_r\hat{\boldsymbol{x}}_r(\boldsymbol{x}_r) + \mathbf{B}_r\boldsymbol{r} \tag{13}$$

$$\boldsymbol{y}_r = \mathbf{C}\hat{\boldsymbol{x}}_r(\boldsymbol{x}_r) \tag{14}$$

where $\boldsymbol{x}_r$ denotes the state vector of the reference model; $\mathbf{A}_r$ and $\mathbf{B}_r$ are the constant system and input matrices, respectively; $\hat{\boldsymbol{x}}_r(\boldsymbol{x}_r)$ is a vector with monomials in $\boldsymbol{x}_r$ as the entries; $\boldsymbol{r}$ denotes the reference input vector; and $\boldsymbol{y}_r$ denotes the output vector of the reference model.

3) Output-feedback Polynomial Fuzzy Controller: The basic idea of trajectory tracking is to continuously reduce the discrepancies between the desired position and the actual position. A polynomial fuzzy controller is employed here to track the trajectory without the online computation of a (pseudo-)inverse Jacobian matrix. This fuzzy controller is designed based on the concept of the parallel distributed compensation [32]. In other words, the membership functions integrated in the fuzzy controller are the same as those in (11). The output-feedback polynomial fuzzy controller is defined as follows [27]:

$$\boldsymbol{v} = \sum_{j=1}^{p} w_j(\boldsymbol{y})(\mathbf{F}_j(\boldsymbol{h})\mathbf{C}\hat{\boldsymbol{e}} + \mathbf{G}_j(\boldsymbol{h})\mathbf{C}\hat{\boldsymbol{x}}_r(\boldsymbol{x}_r)) \tag{15}$$

where we define $\boldsymbol{h} = \begin{bmatrix}\boldsymbol{y}^T & \boldsymbol{y}_r^T\end{bmatrix}^T$, $\mathbf{F}_j(\boldsymbol{h})$ and $\mathbf{G}_j(\boldsymbol{h})$ are the polynomial feedback gains to be determined, and the tracking error is defined by $\hat{\boldsymbol{e}} = \hat{\boldsymbol{x}}(\boldsymbol{x}) - \hat{\boldsymbol{x}}_r(\boldsymbol{x}_r)$.

4) H_∞ Performance of Tracking Control: The tracking performance can be governed by an H_∞ performance index which can be adjusted by the user to minimize the tracking error $\hat{\boldsymbol{e}}$ in (15). It origins from the Lyapunov-based stability analysis. The H_∞ performance of tracking control is defined as follows [27]:

$$\frac{\int_0^{t_f} \boldsymbol{z}_1^T\boldsymbol{z}_1 dt - V(0)}{\int_0^{t_f} (\sigma_1^2\boldsymbol{z}_2^T\boldsymbol{z}_2 + \sigma_2^2\boldsymbol{z}_3^T\boldsymbol{z}_3)dt} \leq 1 \tag{16}$$

where t_f is the termination time of tracking control; σ_1 and σ_2 are the predefined scalars; $\boldsymbol{z}_1 = \mathbf{X}(\tilde{\boldsymbol{x}})^{-1}\boldsymbol{\Gamma}^{-1}\hat{\boldsymbol{e}}$, $\boldsymbol{z}_2 = \mathbf{X}(\tilde{\boldsymbol{x}})^{-1}\boldsymbol{\Gamma}^{-1}\boldsymbol{x}_r$, $\boldsymbol{z}_3 = \boldsymbol{r}$;

$$\mathbf{X}(\tilde{\boldsymbol{x}}) = \begin{bmatrix}\mathbf{X}_{11} & \mathbf{0} \\ \mathbf{0} & \mathbf{X}_{22}(\tilde{\boldsymbol{x}})\end{bmatrix}$$

is termed as a symmetric decision variable which can be obtained in MATLAB; $\boldsymbol{\Gamma} = \begin{bmatrix}\mathbf{C}^T(\mathbf{C}\mathbf{C}^T)^{-1} & \text{ortc}(\mathbf{C}^T)\end{bmatrix}$ and ortc(.) denotes the orthogonal complement; $V(t) = \boldsymbol{\rho}^T\mathbf{X}(\tilde{\boldsymbol{x}})^{-1}\boldsymbol{\rho}$ is the polynomial Lyapunov function candidate, and $\boldsymbol{\rho} = \boldsymbol{\Gamma}^{-1}\hat{\boldsymbol{e}}$; $\tilde{\boldsymbol{x}} = \begin{bmatrix}x_{j_1}, x_{j_2}, ..., x_{j_q}, x_{r_{k_1}}, x_{r_{k_2}}, ..., x_{r_{k_s}}\end{bmatrix}^T$, where the subscripts

$j_1, j_2, ..., j_q$ are the row indices that the entries of the entire row of $\mathbf{B}_i(\boldsymbol{x})$ for all i are all zeros, and the subscripts $k_1, k_2, ..., k_s$ are the row indices that the entries of the entire row of $\mathbf{B}_r$ are all zeros.

5) Stability Conditions of the polynomial Fuzzy-Model-Based control Systems: The defining feature and also the superiority of the fuzzy-model-based approach is that various control problems such as trajectory tracking and H_∞ performance can be systematically analyzed while ensuring the system stability. This gives the theoretical support to physically implement the designed controller. It is derived based on the Lyapunov stability theory.

Before proceeding further, we first describe the following notations which will be employed in the theorem. A polynomial $p(\boldsymbol{x})$ is a sum of squares (SOS) if it can be written as $p(\boldsymbol{x}) = \sum_{j=1}^{m} q_j(\boldsymbol{x})^2$ where $q_j(\boldsymbol{x})$ is polynomial and m is a nonzero integer. Thus, if the condition "$p(\boldsymbol{x})$ is an SOS" holds, then we have $p(\boldsymbol{x}) \geq 0$. SOSTOOLS is a third-party MATLAB toolbox to numerically find solutions to SOS conditions [33].

Theorem 1 (see[27]): The designed polynomial fuzzy controller in (15) is guaranteed to enable the states of the polynomial fuzzy model in (11) representing the physical nonlinear system to track a desired reference model in (13) subject to an H_∞ performance of (16) if there exists decision variables $\mathbf{X}(\tilde{\boldsymbol{x}})$ referring to (16), $\mathbf{M}_j(\boldsymbol{h})$ and $\mathbf{N}_j(\boldsymbol{h})$, $(j = 1, 2, ..., p)$, such that the following SOS conditions are satisfied:

$$\boldsymbol{\nu}_1^T(\mathbf{X}(\tilde{\boldsymbol{x}}) - \epsilon_1(\tilde{\boldsymbol{x}})\mathbf{I})\boldsymbol{\nu}_1 \text{ is } SOS$$
$$-\boldsymbol{\nu}_2^T(\boldsymbol{\Xi}_{\text{ij}}(\boldsymbol{x}, \boldsymbol{x}_r) + \boldsymbol{\Xi}_{\text{ji}}(\boldsymbol{x}, \boldsymbol{x}_r) + \epsilon_2(\boldsymbol{x}, \boldsymbol{x}_r)\mathbf{I})\boldsymbol{\nu}_2 \text{ is SOS}$$
$$\forall j = 1, 2, ..., p; i < j$$

where $\boldsymbol{\nu}_1$ and $\boldsymbol{\nu}_2$ are arbitrary vectors independent of $\boldsymbol{x}$ and $\boldsymbol{x}_r$ and $\epsilon_1(\tilde{\boldsymbol{x}})$ and $\epsilon_2(\boldsymbol{x}, \boldsymbol{x}_r)$ are predefined positive polynomials; the technical details of $\boldsymbol{\Xi}_{\text{ij}}(\boldsymbol{x}, \boldsymbol{x}_r)$ can be found from [27, eq. (29)]. The feedback gains can be obtained by

$$\mathbf{F}_j(\boldsymbol{h}) = \mathbf{M}_j(\boldsymbol{h})\mathbf{X}_{11}^{-1},\ \mathbf{G}_j(\boldsymbol{h}) = \mathbf{N}_j(\boldsymbol{h})\mathbf{X}_{11}^{-1}.$$

B. State-Space Representation

The derived Jacobian in (8) reveals the velocity-level kinematics and fully describes the continuous-time dynamic system. From the control aspect, the mathematical description of the system is expressed as

$$\dot{\boldsymbol{\eta}} = \mathbf{J}(g^{-1}(h^{-1}(\boldsymbol{\eta})))\dot{\boldsymbol{u}} \iff \dot{\boldsymbol{\eta}} = \mathbf{J}(\boldsymbol{\eta})\dot{\boldsymbol{u}} \tag{17}$$

where the equation is known as the state-space model and $\boldsymbol{u} = g^{-1}(h^{-1}(\boldsymbol{\eta}))$ can be obtained analytically by solving their respective parts of forward kinematics.

With the substitution of the above state-space model in the time domain, the state-space controller design techniques such as [27], [28], are enabled toward a dynamic system for the continuum manipulator.

C. Fuzzy Model Construction via the Local Approximation

In order to represent the continuum manipulator state-space model embodied in (17) by a fuzzy model, a local approximation technique is utilized. In our case, the task space in Cartesian coordinate system for a continuum manipulator with 0.01-m diameter is specified as

$$\mathbb{D}^3 = \{x, y, z | x \in [0.015, 0.075],\ y \in [-0.075, 0.075],\ z \in [0.015, 0.15]\} \text{ (Unit: m)}. \tag{18}$$

Based on this range of interest, we approximate the state-space model at six different local sets of system states, i.e., $\boldsymbol{\eta}_1 = [0.015, -0.075, 0.075]^T$, $\boldsymbol{\eta}_2 = [0.015, 0, 0.075]^T$, $\boldsymbol{\eta}_3 = [0.015, 0.075, 0.075]^T$, $\boldsymbol{\eta}_4 = [0.075, -0.075, 0.075]^T$, $\boldsymbol{\eta}_5 = [0.075, 0, 0.075]^T$, and $\boldsymbol{\eta}_6 = [0.075, 0.075, 0.075]^T$. Note that more local sets of system states can be used to establish a more accurate fuzzy model. However, it will lead to higher computational demand. Other advanced fuzzy modeling techniques can be employed to find a better tradeoff between the accuracy and computational burden. In this paper, the system state z is only approximated at one point to lower the computational demand. Then, the local state-space models with respect to each set of system states can be obtained as

$$\mathbf{A}_i = \mathbf{0}; \mathbf{B}_i = \mathbf{J}(\boldsymbol{\eta}_i),\ i = 1, 2, ..., 6 \tag{19}$$

where the derived input matrices are

$$\mathbf{B}_1 = \begin{bmatrix} -0.50 & 5.77 & -5.15 \\ -3.91 & 2.11 & 1.17 \\ -6.15 & 2.29 & 4.49 \end{bmatrix},$$
$$\mathbf{B}_2 = \begin{bmatrix} 0.06 & 4.34 & -4.21 \\ -5.07 & 2.53 & 2.53 \\ 0.32 & -0.83 & 1.48 \end{bmatrix}$$
$$\mathbf{B}_3 = \begin{bmatrix} -0.50 & -5.15 & 5.77 \\ 3.91 & -1.17 & -2.11 \\ -6.15 & 4.49 & 2.29 \end{bmatrix},$$
$$\mathbf{B}_4 = \begin{bmatrix} -2.24 & 5.70 & -2.96 \\ -5.16 & 4.42 & 0.26 \\ -6.04 & -2.11 & 8.64 \end{bmatrix}$$
$$\mathbf{B}_5 = \begin{bmatrix} 0.21 & 3.36 & -2.93 \\ -6.37 & 3.18 & 3.18 \\ 0.21 & -5.30 & 5.73 \end{bmatrix},$$
$$\mathbf{B}_6 = \begin{bmatrix} 2.57 & 3.29 & -5.37 \\ -4.84 & 0.58 & 4.74 \\ 6.37 & -8.31 & 2.43 \end{bmatrix}.$$

After deriving the six local state-space models of (19), we then define six fuzzy rules to smoothly combine them to form the overall fuzzy model. Six fuzzy rules are described as

$$\text{Rule } i: \text{ IF } x \text{ is } M_1^i \text{ and } y \text{ is } M_2^i, \quad \text{THEN } \dot{\boldsymbol{\eta}} = \mathbf{B}_i\dot{\boldsymbol{u}}$$

where $M_1^i, i = 1, 2, ..., 6$, is the fuzzy term of rule i corresponding to the premise variable x, $M_1^1 = M_1^2 = M_1^3 =$ "around 0.015" and $M_1^4 = M_1^5 = M_1^6 =$ "around 0.075"; M_2^i, $i = 1, 2, ..., 6$, is another fuzzy term of rule i corresponding to the premise

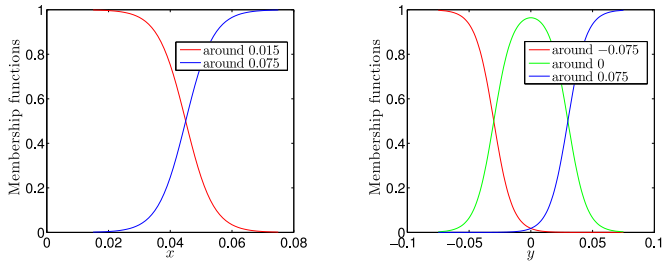


Fig. 4. Illustration of the employed membership functions.

variable y, $M_2^1 = M_2^4 =$ "around -0.075" and $M_2^2 = M_2^5 =$ "around 0," $M_2^3 = M_2^6 =$ "around 0.075." Since z is approximated at only one point, the transition between local models does not depend on z. Consequently, the premise variables are only x and y.

In order to enable the transitions among the six separate fuzzy rules, we propose the following membership functions. $\mu_{M_1^i}(x)$ and $\mu_{M_2^i}(x), i = 1, 2,...,6$, are grades of membership corresponding to the fuzzy terms M_1^i and M_2^i, respectively (see Fig. 4). They are defined as

$$\mu_{M_1^i}(x) = 1 - \frac{1}{1+\exp(-\frac{x-0.045}{0.0045})}, \quad i = 1,2,3 \tag{20}$$

$$\mu_{M_1^i}(x) = 1 - \mu_{M_1^1}(x), \quad i = 4,5,6 \tag{21}$$

$$\mu_{M_2^i}(y) = 1 - \frac{1}{1+\exp(-\frac{y+0.03}{0.0075})}, \quad i = 1,4 \tag{22}$$

$$\mu_{M_2^i}(y) = \frac{1}{1+\exp(-\frac{y-0.03}{0.0075})}, \quad i = 2,5 \tag{23}$$

$$\mu_{M_2^i}(y) = 1 - \mu_{M_2^1}(y) - \mu_{M_2^2}(y), \quad i = 3,6. \tag{24}$$

The membership functions for the local state-space models are then derived by

$$w_i(\boldsymbol{\eta}) = \mu_{M_1^i}(x)\mu_{M_2^i}(y), \quad i = 1,2,...,6 \tag{25}$$

where $w_i(\boldsymbol{\eta}), i = 1,2,...,6$, are the employed membership functions and they possess the following property $\sum_{i=1}^{6} w_i(\boldsymbol{\eta}) = 1, w_i(\boldsymbol{\eta}) \geq 0, \forall i$.

Here, we consider the full state-feedback control instead of output-feedback control; thus, $\mathbf{C} = \mathbf{I}$ which leads to $\boldsymbol{\Gamma} = \mathbf{I}$. So far, the fuzzy model is established by substituting the derived $\mathbf{A}_i, \mathbf{B}_i, \mathbf{C}$, and $w_i(\boldsymbol{\eta})$ into (11) and (12) as

$$\dot{\boldsymbol{\eta}} = \sum_{i=1}^{6} w_i(\boldsymbol{\eta})\mathbf{B}_i\dot{\boldsymbol{u}}. \tag{26}$$

The difference of each entry between the original state-space model in (17) and the fuzzy model in (26) is measured by the mean absolute error (MAE) for all system states in the range of interest described in (18), which is defined by

$$\beta_{\mathrm{mn}} = \frac{1}{N}\sum_{j=1}^{N}|\mathbf{J}_{\mathrm{mn}}(\boldsymbol{\eta}_j) - \tilde{\mathbf{B}}_{\mathrm{mn}}(\boldsymbol{\eta}_j)|, \quad m,n = 1,2,3 \tag{27}$$

where N is the number of a series of dense system states in the range of interest, m and n define the (m,n)th entry of the corresponding matrix, $\boldsymbol{\eta}_j$ is the sampled system state, and $\tilde{\mathbf{B}}(\boldsymbol{\eta}) = \sum_{i=1}^{6} w_i(\boldsymbol{\eta})\mathbf{B}_i$.

The calculated MAEs are $\beta_{11} = 0.4149, \beta_{12} = 2.4850, \beta_{13} = 2.6373, \beta_{21} = 2.6893, \beta_{22} = 1.2943, \beta_{23} = 1.4764, \beta_{31} = 1.4137, \beta_{32} = 1.2130$, and $\beta_{33} = 0.7695$. It can be seen that the fuzzy modeling error exists due to the high nonlinearity of the original state-space model and the limited number of fuzzy rules, which can be reduced in the future.

D. Fuzzy Controller Design

We first define two trajectory tracking cases to specify the reference model: one is to track a straight line in task space and the other is to follow an ellipse.

The straight line reference model in the form of (13) is given by

$$\dot{\boldsymbol{\eta}}_r = \mathbf{B}_r r \tag{28}$$

where $\mathbf{B}_r = \begin{bmatrix} -0.0005 & -0.002 & 0.0018 \end{bmatrix}^T$

$$r = \begin{cases} 1, & t \leq 60\,\mathrm{s} \\ 0, & t > 60\,\mathrm{s}. \end{cases}$$

Given the above task, the corresponding fuzzy controller is designed by applying Theorem 1. Choosing the decision variables $\mathbf{X}, \mathbf{M}_j, \mathbf{N}_j$ as constant matrices, $\epsilon_1 = \epsilon_2 = 0.001$, and $\sigma_1 = \sigma_2 = 0.1$ in (16), the feedback gains are obtained as follows:

$$\mathbf{F}_1 = \begin{bmatrix} -1.14 & -0.22 & -0.39 \\ -1.31 & -0.35 & -0.55 \\ -1.03 & -0.16 & -0.62 \end{bmatrix}$$

$$\mathbf{F}_2 = \begin{bmatrix} -0.43 & 0.34 & -0.09 \\ -0.36 & 0.26 & -0.17 \\ -0.34 & 0.36 & -0.23 \end{bmatrix}$$

$$\mathbf{F}_3 = \begin{bmatrix} -2.75 & -1.26 & -1.85 \\ -2.47 & -1.67 & -2.12 \\ -2.50 & -1.25 & -2.42 \end{bmatrix}$$

$$\mathbf{F}_4 = \begin{bmatrix} -0.50 & 0.23 & 0.51 \\ -0.52 & 0.19 & 0.48 \\ -0.50 & 0.24 & 0.45 \end{bmatrix}$$

$$\mathbf{F}_5 = \begin{bmatrix} -0.28 & 0.34 & 0.03 \\ -0.30 & 0.28 & -0.00 \\ -0.28 & 0.32 & -0.05 \end{bmatrix}$$

$$\mathbf{F}_6 = \begin{bmatrix} -0.98 & 0.03 & -0.84 \\ 0.93 & -0.11 & -0.92 \\ -0.93 & 0.02 & -1.04 \end{bmatrix}.$$

Here, $\mathbf{G}_i \approx \mathbf{0}, i = 1,2,...,6$ (the magnitude of the entries of the matrix $\mathbf{G}_i$ is less than 10^{-14}).

The designed fuzzy controller in the form of (15) can be acquired to comply with the fuzzy model in (26) as

$$\dot{\boldsymbol{u}} = \sum_{j=1}^{6} w_j(\boldsymbol{\eta})(\mathbf{F}_j \Delta\boldsymbol{\eta} + \mathbf{G}_j \boldsymbol{\eta}_r) \tag{29}$$
$$\Delta\boldsymbol{\eta} \equiv \boldsymbol{\eta} - \boldsymbol{\eta}_r.$$

The ellipse reference model in the form of (13) is given by

$$\dot{\boldsymbol{\eta}}_r = \mathbf{A}_r \boldsymbol{\eta}_r + \mathbf{B}_r r \tag{30}$$

where

$$\mathbf{A}_r = \begin{bmatrix} 0 & -0.1 & 0 \\ 0.1 & 0 & 0 \\ 0.2 & 0 & -0.1 \end{bmatrix}$$
$$\mathbf{B}_r = \begin{bmatrix} 0 & -0.0045 & -0.001 \end{bmatrix}^T, \quad r = 1.$$

Given this task, the corresponding fuzzy controller is designed similarly by applying Theorem 1. Choosing the decision variables $\mathbf{X}, \mathbf{M}_j, \mathbf{N}_j$ as constant matrices, $\epsilon_1 = \epsilon_2 = 0.001$, and $\sigma_1 = \sigma_2 = 1$ (different from the straight line tracking case) in (16), the feedback gains are then correspondingly obtained as follows:

$$\mathbf{F}_1 = \begin{bmatrix} -397 & -246 & -124 \\ -402 & -252 & -127 \\ -377 & -232 & -121 \end{bmatrix}$$
$$\mathbf{F}_2 = \begin{bmatrix} -69.1 & -35.5 & -22.3 \\ -65.9 & -35.0 & -22.5 \\ -51.4 & -24.2 & -18.9 \end{bmatrix}$$
$$\mathbf{F}_3 = \begin{bmatrix} -1227 & -774 & -395 \\ -1237 & -789 & -402 \\ -1201 & -758 & -396 \end{bmatrix}$$
$$\mathbf{F}_4 = \begin{bmatrix} -23.73 & -10.03 & 0.53 \\ -36.80 & -18.55 & 3.95 \\ -24.02 & -9.61 & -0.46 \end{bmatrix}$$
$$\mathbf{F}_5 = \begin{bmatrix} 5.51 & 11.06 & 3.09 \\ -12.78 & -1.20 & -2.93 \\ -8.36 & 2.31 & -2.29 \end{bmatrix}$$
$$\mathbf{F}_6 = \begin{bmatrix} -362 & -219 & -120 \\ -379 & -233 & -126 \\ -366 & -222 & -124 \end{bmatrix}$$
$$\mathbf{G}_1 = \begin{bmatrix} 0.06 & -0.03 & -0.04 \\ 0.10 & -0.04 & -0.05 \\ 0.10 & -0.04 & -0.06 \end{bmatrix}$$
$$\mathbf{G}_2 = \begin{bmatrix} 0.15 & -0.06 & -0.10 \\ 0.16 & -0.07 & -0.10 \\ 0.17 & -0.05 & -0.11 \end{bmatrix}$$
$$\mathbf{G}_3 = \begin{bmatrix} -0.20 & 0.09 & 0.09 \\ -0.21 & 0.09 & 0.11 \\ -0.22 & 0.10 & 0.10 \end{bmatrix}$$
$$\mathbf{G}_4 = \begin{bmatrix} 0.05 & -0.07 & -0.04 \\ 0.07 & -0.08 & -0.04 \\ 0.09 & -0.07 & -0.05 \end{bmatrix}$$
$$\mathbf{G}_5 = \begin{bmatrix} 0.16 & -0.08 & -0.07 \\ 0.16 & -0.09 & -0.06 \\ 0.18 & -0.08 & -0.08 \end{bmatrix}$$
$$\mathbf{G}_6 = \begin{bmatrix} 0.29 & -0.10 & -0.15 \\ 0.28 & -0.11 & -0.14 \\ 0.28 & -0.10 & -0.15 \end{bmatrix}.$$

The fuzzy controller in the form of (29) can be acquired but with feedback gains $\mathbf{F}_j, \mathbf{G}_j, j = 1, 2, ..., 6$, designed for this ellipse trajectory.

IV. Simulation Examples and Analysis

We implement the proposed fuzzy controller in MATLAB simulation to investigate its performance. The simulation environment contains the aforementioned task space [see (18)] with respect to a continuum manipulator. The manipulator's mathematical model described in (8) is utilized and `ode23` function command in MATLAB is executed to generate the continuous navigation path. In order to include the modeling inaccuracies and other real-time errors in simulation and validate the robust performance of the designed fuzzy controller, we introduce an additive term $\Delta\mathbf{J}$ to the analytically derived Jacobian matrix, i.e.,

$$\dot{\boldsymbol{\eta}} = (\mathbf{J}(\boldsymbol{u}) + \Delta\mathbf{J})\dot{\boldsymbol{u}}. \tag{31}$$

Two different types of reference models, respectively, describing the straight line and ellipse tracking trajectories are utilized in the simulation. To compare with other controllers, we implement all the controllers in the same situation, where the same additive term in the disturbed model is considered.

A. Straight Line Trajectory Tracking Task

In the simulation, the initial states of the disturbed model in (31) and the specified straight tracking trajectory in (28) are defined as $\boldsymbol{\eta}(0) = \boldsymbol{\eta}_r(0) = \begin{bmatrix} 0.06 & 0.06 & 0.03 \end{bmatrix}^T$. The additive disturbance term is chosen as

$$\Delta\mathbf{J} = \begin{bmatrix} 0.2 & 0.2 & 0.2 \\ 0.2 & 0.2 & 0.2 \\ 0.2 & 0.2 & 0.2 \end{bmatrix}$$

in the case. This value is chosen based on expert knowledge and it is approximately the mean value to simulate disturbance in our experimental setup. Implementing the designed fuzzy controller in (29), the simulation results are shown in Fig. 5(a). We can see that the trajectory tracking task is effectively achieved by the proposed fuzzy controller.

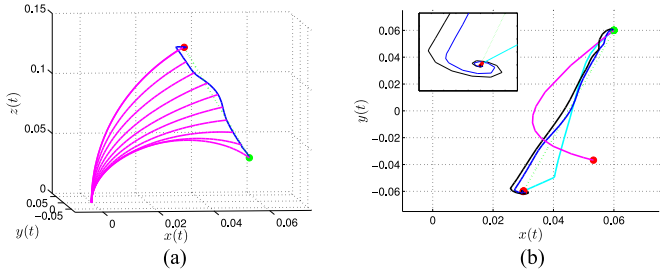


Fig. 5. (a) Performance of the designed fuzzy controller used to track a straight line trajectory (Unit: m). The trajectory of the continuum manipulator tip in 3-D task space are captured and illustrated in blue line and the central backbone shape of the continuum manipulator is illustrated in pink. The green dot indicates the initial position $[0.06, 0.06, 0.03]^T$ and the red dot indicates termination position $[0.03, -0.06, 0.135]^T$. The green-dotted line shows the specified reference trajectory. (b) Performance comparisons of the four controllers shown via the xy plane view of 3-D task space. The blue, black, cyan, and pink trajectories indicate the trajectories based on the controllers "I-A," "I-B," "I-C," "I-D," respectively. A "zoom-in" view around the target is presented, which shows the spiral phenomenon associated to our fuzzy controllers.

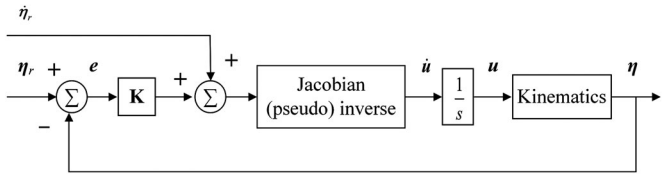


Fig. 6. Overview of a closed-loop control that is based on the (pseudo) inverse Jacobian method. Control input includes both the desired time-varying trajectory and the preplanned task space velocity with respect to the desired task (trajectory) [15], [34]. $\mathbf{K}$ is a diagonal matrix, and if $\mathbf{K} = \mathbf{0}$, then this control architecture becomes an open-loop control.

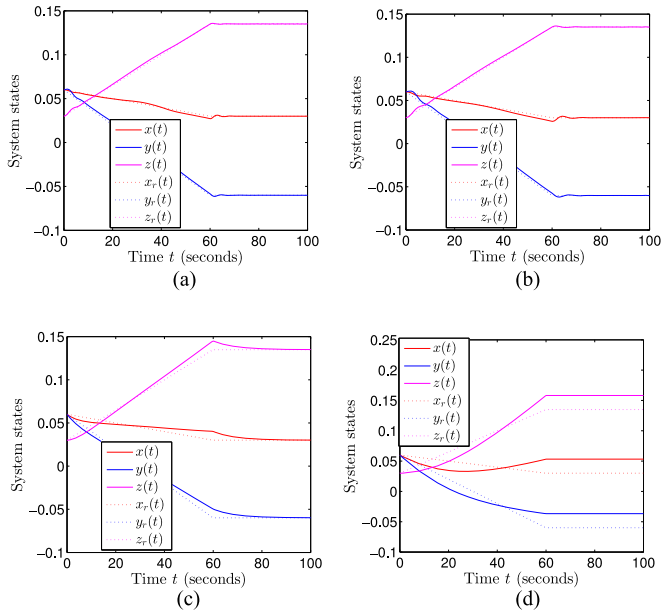


Fig. 7. Illustrations of time responses with respect to each of the four controllers: (a) I-A, (b) I-B, (c) I-C, and (d) I-D, respectively.

TABLE I
NUMERICAL COMPARISONS OF DIFFERENT CONTROLLERS' STRAIGHT LINE TRACKING PERFORMANCES VIA PERFORMANCE INDICES

Controllers	IAE	ITAE	IAV	Execution Rate
I-A	0.3513	11.4858	0.1871	23.83 kHz
I-B	0.4279	14.3184	0.1906	22.24 kHz
I-C	0.9997	45.9150	0.1981	13.54 kHz
I-D	4.6803	282.7586	0.2741	13.97 kHz

We further compare our designed controller (labeled by "I-A") with three other types of controllers: fuzzy controller with different H_∞ performance (labeled by "I-B"), closed-loop Jacobian-based controller (labeled by "I-C") [15], [34] (see Fig. 6), and open-loop Jacobian-based controller (labeled by "I-D") [35].

To design a fuzzy controller with different H_∞ performances, we choose $\sigma_1 = \sigma_2 = 100$ in (16), while other controller design parameters remain the same in Theorem 1. Therefore, the feedback gains can be similarly obtained.

The closed-loop Jacobian-based controller is designed based on (40) in [15] with $\mathbf{W} = \mathbf{I}, \mathbf{K} = 0.1\mathbf{I}$ resulting $\dot{\boldsymbol{u}} = \mathbf{J}(\boldsymbol{u})^{-1}(\dot{\boldsymbol{\eta}}_r + 0.1(\boldsymbol{\eta}_r - \boldsymbol{\eta}))$. The open-loop Jacobian-based controller is simply given by $\dot{\boldsymbol{u}} = \mathbf{J}(\boldsymbol{u})^{-1}\dot{\boldsymbol{\eta}}_r$. The performance comparisons among the total four controllers are illustrated in Fig. 5(b) and Fig. 7. The proposed fuzzy controller demonstrates the best performance with the minimum tracking errors. Compared with the additional fuzzy controller with different H_∞ performance, the results imply that the smaller the values of σ_1 and σ_2, the better the H_∞ tracking performance governed by (16). Both implemented fuzzy controllers exhibit a spiral phenomenon and this converged spiral decreases the tracking error around the target. The open-loop and closed-loop Jacobian-based controllers suffer from the modeling inaccuracies. Although the closed-loop controller can reduce the tracking error based on the real-time feedback information, large modeling error results in poor performances. Both Jacobian-based controllers need online updates of the Jacobian which causes a computational burden, which could be particularly problematic in a real-time system; on the other hand, our fuzzy controllers are very efficient and have a low computational load.

The performance indices are calculated by using the integral absolute error (IAE) $\bar{h}_1 = \int_0^{t_f}(|x - x_r| + |y - y_r| + |z - z_r|)dt$, integral of time multiply absolute error (ITAE) $\bar{h}_2 = \int_0^{t_f} t(|x - x_r| + |y - y_r| + |z - z_r|)dt$, and integral of the absolute value of the control input (IAV) $\bar{h}_3 = \int_0^{t_f}(|\dot{\tau}_1| + |\dot{\tau}_2| + |\dot{\tau}_3|)dt$, where $t_f = 100$ s and $\dot{\tau}_1$, $\dot{\tau}_2$, and $\dot{\tau}_3$ are control inputs representing the controlled tendon speeds of three respective tendons. The results are shown in Table I, and they further illustrate the superiority of the proposed fuzzy controller.

B. Ellipse Trajectory Tracking Task

In this simulation, the initial states of the disturbed model in (31) and the ellipse tracking trajectory in (30) are $\boldsymbol{\eta}(0) = \boldsymbol{\eta}_r(0) = [0.065, 0, 0.1]^T$. The additive disturbance term, 3×3 matrix $\Delta\mathbf{J}(t)$, is chosen by assigning each of all nine entries as $0.1\sin(\frac{\pi t}{30})$. Implementing the designed fuzzy controller in (29)

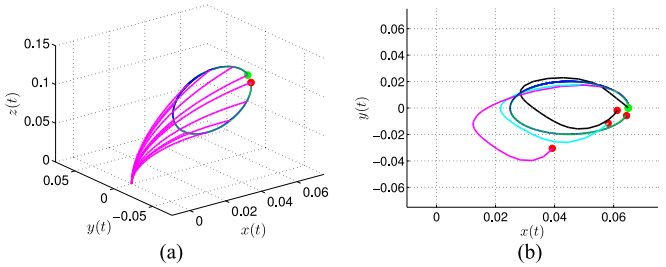


Fig. 8. Simulated performances for the ellipse trajectory tracking task. The same drawing convention applies here as used for Fig. 5. In this case, the initial position is $[0.065, 0, 0.1]^T$. The blue, black, cyan, and pink trajectories indicate the trajectories based on the controllers "II-A," "II-B," "II-C," "II-D," respectively.

with the feedback gains derived in Section III-D for the ellipse trajectory case, the simulation results are shown in Fig. 8(a). The ellipse tracking task is accomplished perfectly by the designed fuzzy controller.

Here, we also compare our designed fuzzy controller (labeled by "II-A") with three other types of controllers: closed-loop Jacobian-based controller (labeled by "II-C") and open-loop Jacobian-based controller (labeled by "II-D") are same as those used for straight line trajectory tracking task; another linear controller (labeled by "II-B") is designed with the same methodology as to design the fuzzy controller but choosing only one operating point $\boldsymbol{\eta} = [0.04, 0, 0.075]^T$ (a special case of the fuzzy model). The linear controller is described as

$$\dot{\boldsymbol{u}} = \mathbf{F}\Delta\boldsymbol{\eta} + \mathbf{G}\boldsymbol{\eta}_r \tag{32}$$

where

$$\mathbf{F} = \begin{bmatrix} -0.85 & 0.02 & 0.27 \\ -0.85 & -0.02 & 0.28 \\ -0.85 & -0.02 & 0.26 \end{bmatrix}$$

$$\mathbf{G} = \begin{bmatrix} -0.08 & -0.12 & 0.04 \\ -0.08 & -0.12 & 0.04 \\ -0.07 & -0.12 & 0.03 \end{bmatrix}.$$

The comparisons of the four controllers are illustrated in Figs. 8(b) and 9. Similar results with those for straight line tracking case can be obtained. We can see that although the actual trajectories from both the open-loop and closed-loop Jacobian-based controllers are ellipse-like, they quickly run away from the defined the ellipse after the starting position. The performance of the linear controller obtained from one operating point is worse than the fuzzy controller obtained from six operating points. One operating point is not enough to represent the original nonlinear model. The numerical comparison results using performance indices (i.e., IAE, ITAE, IAV) are given in Table II. As indicated by IAE and ITAE, the proposed fuzzy controller with the lowest cost provides the best performance. The larger value of the IAV index for the proposed fuzzy controller indicates that there is a cost of control effort to achieve the smaller error. In both Table I and II, the execution rates of all implemented controllers are listed for comparison, which indicates that the fuzzy controller

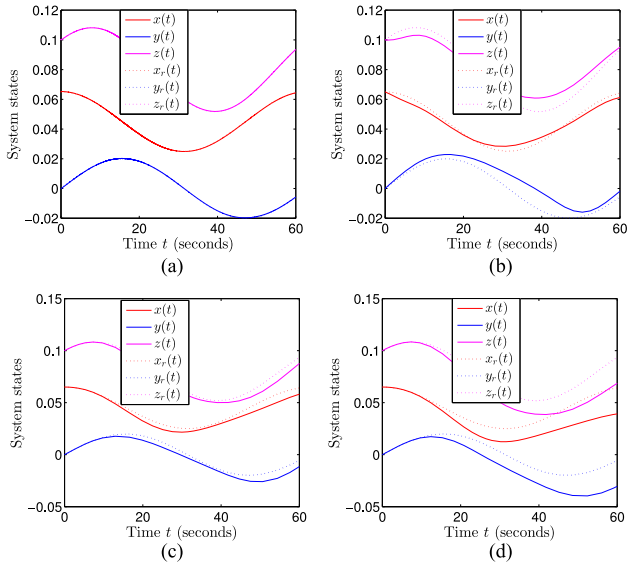


Fig. 9. Illustrations of time responses with respect to each of four controllers: (a) II-A, (b) II-B, (c) II-C, and (d) II-D, respectively.

TABLE II
NUMERICAL COMPARISONS OF DIFFERENT CONTROLLERS' ELLIPSE TRAJECTORY TRACKING PERFORMANCES VIA PERFORMANCE INDICES

Controllers	*IAE*	*ITAE*	*IAV*	*Execution Rate*
II-A	0.0214	0.6555	0.8120	40.55 kHz
II-B	0.7800	25.5389	0.3570	52.42 kHz
II-C	0.6271	24.8722	0.4170	13.35 kHz
II-D	2.0268	85.0483	0.3893	13.74 kHz

operates faster and has the superiority of a low computational cost compared to conventional Jacobian-based methods.

V. EXPERIMENTS

The proposed fuzzy controller is implemented on a tendon-driven 3-D-printed continuum manipulator, whose design was presented in [29]. This continuum manipulator demonstrates an effectively decoupled bending with contraction via three tendons at the periphery, thus, in line with the previously presented kinematic model for a general continuum manipulator. The contraction capability enables the length of the manipulator to vary from the original full length to contract to a length of about 70%. In order to measure the manipulator's tip position, a commercial electromagnetic (EM) tracking system NDI Aurora is used. One 0.8-mm diameter × 11-mm length sensor coil is integrated in the head of the continuum manipulator to track real-time tip positions and orientations. Each tendon is actuated via a DC motor (Maxon Motor). For comparative purposes, both the traditional Jacobian-based open-loop and closed-loop controllers are implemented under the same condition; Fig. 10 illustrates the experimental setup.

A. System Description

All the three controllers and the reference trajectory generator are implemented in robot operating system (ROS) Environment

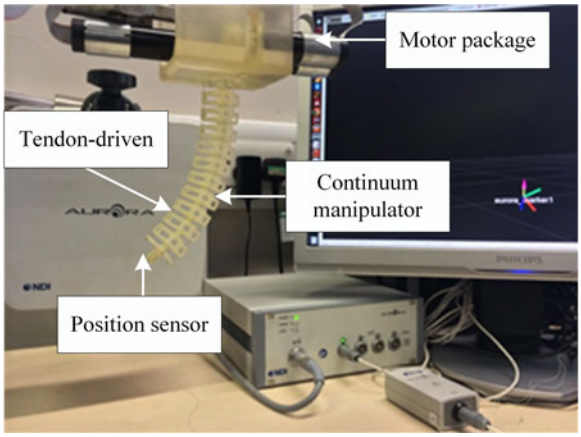


Fig. 10. Experimental setup. The controllers are implemented on a tendon-driven continuum manipulator with validated constant-curvature bending performance. The NDI Aurora EM position sensor is used for the purpose of tracking the manipulator tip.

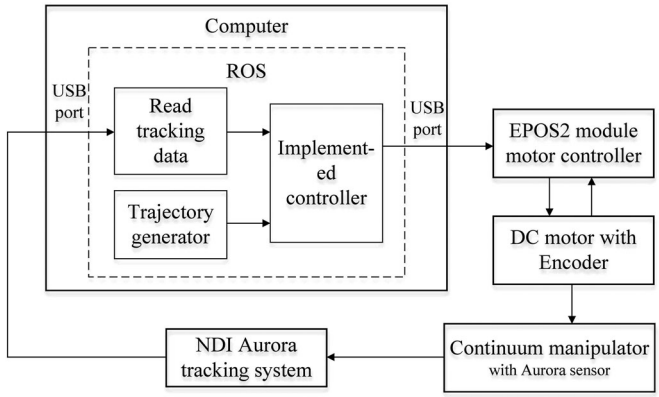


Fig. 11. Block diagram of the experimental system integration.

on an Intel Core i3 @2.40 GHz and 1.5-GB RAM-based platform running Linux Mint 13 Operating System. The control signal is fed to the EPOS2 Module motor controller to control the velocities of three Maxon DC Motors, each equipped with encoders to ensure precise velocity control. The motors are connected to the tendons via a gearbox with a reduction ratio of 128:1. The rotation of the motors moves each tendon with the desired velocity. A change in tendon length will move the tip of the continuum manipulator; the motor control is based on the kinematic model.

An Aurora sensor coil, embedded in the tip of the manipulator, will give the position of the tip with respect to another sensor coil embedded in the base of the manipulator. This information is fed to the computer via the NDI Aurora tracking system and used as a feedback to the fuzzy controller. A standard Jacobian-based closed-loop controller receives Aurora signals in the same way during the comparative experiments. The reference trajectory in this experiment is chosen to be a straight line measured with respect to the base of the robot. The reference trajectory position, as well as the manipulator tip's position and the tendon velocity as control signal are also recorded via ROS to enable further analysis and documentation. The block diagram of the experimental system integration is shown in Fig. 11. The parameters of the experiment are as follows: the length of manipulator $l = 143$ mm, cross-sectional radius $d = 13.4$ mm, the specified workspace range $x : [7, 31.5],\ y : [-31.5, 31.5],\ z : [110, 134]$, where the coordinate system is in accordance with the global coordinate system as shown in Fig. 2(b) (unit: mm), velocity $\mathbf{B}_r = [0.250, 0.750, 0.267]$, and 70 s duration.

The control programs are first tested offline using simulated tracking sensor feedback. These tests validate the advantage of the fuzzy controller in terms of its reduced amount of necessary calculations. When executed on our computer system, the proposed fuzzy controller operates at an execution rate of 168.79 Hz (0.0059 s execution time per iteration) as opposed to 40.90 Hz (0.0244 s execution time per iteration) for the Jacobian-based methods. This big difference in execution time is caused by the fact that the fuzzy controller's feedback gains do not need to be updated. However, the Jacobian-based controllers need to execute numerical integrations to estimate the current length of each tendon and update the Jacobian matrix in every control iteration. Besides, the matrix inversion operation—a complex mathematical process—to inverse the derived Jacobians is needed in Jacobian-based controllers and slows down the computations. Based on the analyzed execution rate above, the system is thus determined to be executed with a sampling rate of 40 Hz.

B. Experimental Results and Analysis

To implement the fuzzy-model-based control for experimental studies, six different local operating points [7, −31.5, 122], [7, 0, 122], [7, 31.5, 122], [31.5, −31.5, 122], [31.5, 0, 122], and [31.5, 31.5, 122] (Unit: mm), with respect to the specified workspace range, are chosen to approximate the state-space model. Accordingly, membership functions are then derived and utilized for both fuzzy model construction and fuzzy controller design. The experimental results are illustrated in Fig. 12. Despite the fact that model discrepancies and hardware tolerances exist, the proposed fuzzy-model-based approach still accomplishes the tracking task. As shown in Fig. 12(a), the final stage of the experimental recorded trajectory presents a converged spiral, which indicates the feasibility of the controller. For comparison purposes, we also implemented two other traditional Jacobian-based controllers and tested in an experimental study. Fig. 12(c) and (d) shows tracking performance with an open-loop Jacobian-based controller; Fig. 12(e) and (f) shows the tracking performance with a closed-loop Jacobian-based controller ($\mathbf{K} = 0.1\mathbf{I}$), whose control architecture is shown in Fig. 6. From Fig. 12, we can see that both traditional controllers achieve the tracking tasks but there exist a significant distance between the end-point to the target. Based on Fig. 12(c) and (e), the performance with the closed-loop controller is better than the performance with the open-loop controller. The open-loop control execution leads to an accumulation of the tracking errors, and it can be seen that the experimental recorded trajectory gradually moves away from the reference without any trend to decrease the error. The closed-loop Jacobian-based controller

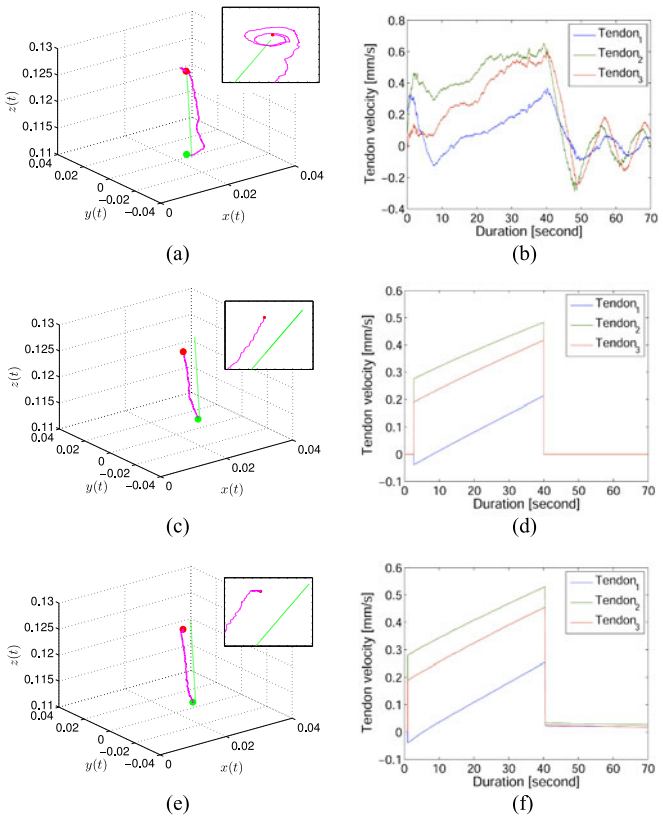


Fig. 12. Experimental results of a trajectory tracking execution via fuzzy-model-based approach [shown by (a) and (b)], open-loop Jacobian-based approach [shown by (c) and (d)], and closed-loop Jacobian-based approach [shown by (e) and (f)]. The figures in the left column show the experimental recorded trajectory in 3-D space and the zoom-in view around the target. The green dot indicates the initial position and red dots indicate the termination positions. The green line shows the specified reference trajectory. The magenta trajectory indicates the trajectory based on the applied controller. The figures in right column show the tendon speed control signals.

TABLE III
NUMERICAL COMPARISON OF DIFFERENT CONTROLLERS TACKING PERFORMANCES IN EXPERIMENTS

Controllers	Targeting Precision	IAE	ITAE	IAV
IV-A	0.7165 mm	0.2446	6.5240	48.0182
IV-B	3.1 mm	0.2968	11.8942	35.7483
IV-C	5.2 mm	0.4311	18.2555	29.3033

IV-A: the proposed fuzzy controller; IV-B: closed-loop Jacobian-based; IV-C: open-loop Jacobian-based.

keeps to a trajectory that is almost parallel with respect to the reference. After 40 s when the reference model terminates at the target point, the closed-loop control will drive the tip of the manipulator to gradually approach the steady target, while the open-loop control terminates at exactly 40 s. Due to the delay on the ROS Node initialization, the controller does not start to produce control signals immediately, rather it lags by a small duration of time at the beginning of the experiments. The numerical comparisons regarding the performances with these three different controllers are given in Table III. It can be seen that, with regard to the targeting precision and these performance indices of the proposed fuzzy-model-based controller shows advantages. It is also to be noted that the performance of implementing the open-loop controller reflects the accuracy of our kinematic model and the hardware shortcomings. Both the closed-loop Jacobian-based controller and the proposed fuzzy-model-based controller still have space to be further improved so that a better tracking performance can be expected. These experiments in this paper validate the feasibility of the fuzzy-model-based controller to be implemented for continuum manipulators with some appealing advantages. This is the first work of achieving task space closed-loop control with a fuzzy-model-based approach. Besides, the (fuzzy-model-based) PI control can be achieved by adding an integral term. The analysis will remain more or less the same but the integral term will increase the dimensions of the (augmented) system and input matrices. Given that the current performance is acceptable in simulations and experiments, and, in order to avoid complicating the controller, the integral action is not considered in this study.

VI. CONCLUSION AND FUTURE WORK

A fuzzy controller has been proposed for autonomous execution of end-effector trajectory tracking tasks of a continuum manipulator, overcoming model complexities and uncertainty issues that plague other types of controllers. In MATLAB simulations, the proposed controller was implemented and compared with three other controllers. The results showed that the designed fuzzy controller had the best performance with regards to the minimum tracking errors and accomplished both tracking tasks efficiently. The other Jacobian-based controllers suffered from model inaccuracies. Experiments were conducted employing a rapid-prototyped continuum manipulator. The results verified the feasibility of the controller in presence of modeling discrepancies and hardware tolerances.

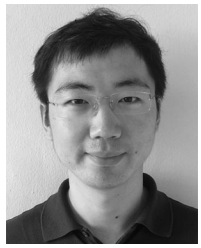
Some limitations are discussed here. The proposed fuzzy approach has a high computational complexity when deriving the controller gains, especially when we choose more linearization points; this could be solved by using high-performance computers. Besides, we did not use the exact original nonlinear model (we linearized it) when deriving the controller, and more rules are required to achieve a more accurate model. Therefore, it is critical to find a balance between the better performance and more rules.

Future work will include refining the controller design and testing the controller with practical continuum manipulator systems. A future work on both dynamic model and elastic material's hysteresis issue will help further understand and control such continuum manipulators. Multisection continuum manipulators with more degrees of freedom are to be analyzed and controlled with the fuzzy-model-based approach.

REFERENCES

[1] I. D. Walker, "Continuous backbone "continuum" robot manipulators," *ISRN Robot.*, vol. 2013, pp. 1–19, 2013.
[2] S. Hirose, *Biologically Inspired Robots, Snake-Like Locomotors and Manipulators*. Oxford, U.K.: Oxford Univ. Press, 1993.

[3] N. Simaan, K. Xu, A. Kapoor, W. Wei, P. Kazanzides, P. Flint, and R. H. Taylor, "Design and integration of a telerobotic system for minimally invasive surgery of the throat," *Int. J. Robot. Res.*, vol. 28, no. 9, pp. 1134–1153, 2009.
[4] J. K. Salisbury, "Whole arm manipulation," presented at the 4th Symp. Robot. Res., Santa Cruz, CA, USA, 1987.
[5] R. J. Webster and B. A. Jones, "Design and kinematic modeling of constant curvature continuum robots: A review," *Int. J. Robot. Res.*, vol. 29, no. 13, pp. 1661–1683, 2010.
[6] Y.-J. Kim, S. Cheng, S. Kim, and K. Iagnemma, "A novel layer jamming mechanism with tunable stiffness capability for minimally invasive surgery," *IEEE Trans. Robot.*, vol. 29, no. 4, pp. 1031–1042, Aug. 2013.
[7] Festo Corporate, "Bionic Handling Assistant—Flexible and compliant movement," from *Bionic Learning Network*. 2010. [Online]. Available: http://www.festo.com/cms/en_corp/9655_10218.htm
[8] R. J. Webster, R. Joseph, and N. J. Cowan, "Mechanics of precurved-tube continuum robots," *IEEE Trans. Robot.*, vol. 25, no. 1, pp. 69–78, Feb. 2009.
[9] M. Cianchetti, T. Ranzani, G. Gerboni, I. De Falco, C. Laschi, and A. Menciassi, "STIFF-FLOP surgical manipulator: Mechanical design and experimental characterization of the single module," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2013, pp. 3576–3581.
[10] G. S. Chirikjian, "Hyper-redundant manipulator dynamics: A continuum approximation," *Adv. Robot.*, vol. 9, no. 3, pp. 217–243, 1994.
[11] M. W. Hannan and I. D. Walker, "Kinematics and the implementation of an elephant's trunk manipulator and other continuum style robots," *J. Robot. Syst.*, vol. 20, no. 2, pp. 45–63, 2003.
[12] D. C. Rucker and R. J. Webster, "Statics and dynamics of continuum robots with general tendon routing and external loading," *IEEE Trans. Robot.*, vol. 27, no. 6, pp. 1033–1044, Dec. 2011.
[13] R. S. Penning, J. Jung, J. A. Borgstadt, N. J. Ferrier, and M. R. Zinn, "Towards closed loop control of a continuum robotic manipulator for medical applications," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2011, pp. 4822–4827.
[14] R. S. Penning, J. Jung, N. J. Ferrier, and M. R. Zinn, "An evaluation of closed-loop control options for continuum manipulators," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2012, pp. 5392–5397.
[15] T. Mahl, A. Hildebrandt, and O. Sawodny, "A variable curvature continuum kinematics for kinematic control of the bionic handling assistant," *IEEE Trans. Robot.*, vol. 30, no. 4, pp. 935–949, Aug. 2014.
[16] Y. Bailly, Y. Amirat, and G. Fried, "Modeling and control of a continuum style microrobot for endovascular surgery," *IEEE Trans. Robot.*, vol. 27, no. 5, pp. 1024–1030, Oct. 2011.
[17] V. K. Chitrakaran, A. Behal, D. M. Dawson, and I. D. Walker, "Setpoint regulation of continuum robots using a fixed camera," *Robotica*, vol. 25, no. 05, pp. 581–586, 2007.
[18] A. Kapadia and I. D. Walker, "Task-space control of extensible continuum manipulators," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2011, pp. 1087–1092.
[19] R. S. Penning and M. Zinn, "A combined modal-joint space control approach for continuum manipulators," *Adv. Robot.*, vol. 28, no. 16, pp. 1091–1108, 2014.
[20] M. Ivanescu, "Position dynamic control for a tentacle manipulator," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2002, pp. 1531–1538.
[21] P. Qi, C. Liu, L. Zhang, S. Wang, H.-K. Lam, and K. Althoefer, "Fuzzy logic control of a continuum manipulator for surgical applications," in *Proc. IEEE Int. Conf. Robot. Biomimetics*, 2014, pp. 413–418.
[22] D. Braganza, D. M. Dawson, I. D. Walker, and N. Nath, "A neural network controller for continuum robots," *IEEE Trans. Robot.*, vol. 23, no. 6, pp. 1270–1277, Dec. 2007.
[23] A. Melingui, O. Lakhal, B. Daachi, J. B. Mbede, and R. Merzouki, "Adaptive neural network control of a compact bionic handling arm," *IEEE/ASME Trans. Mechatronics*, vol. 20, no. 6, pp. 2862–2875, Dec. 2015.
[24] M. C. Yip and D. B. Camarillo, "Model-less feedback control of continuum manipulators in constrained environments," *IEEE Trans. Robot.*, vol. 30, no. 4, pp. 880–889, Aug. 2014.
[25] S. B. Kesner and R. D. Howe, "Position control of motion compensation cardiac catheters," *IEEE Trans. Robot.*, vol. 27, no. 6, pp. 1045–1055, Dec. 2011.
[26] S. G. Yuen, D. T. Kettler, P. M. Novotny, R. D. Plowes, and R. D. Howe, "Robotic motion compensation for beating heart intracardiac surgery," *Int. J. Robot. Res.*, vol. 28, no. 10, pp. 1355–1372, 2009.
[27] H. K. Lam and H. Li, "Output-feedback tracking control for polynomial fuzzy-model-based control systems," *IEEE Trans. Ind. Electron.*, vol. 60, no. 12, pp. 5830–5838, Dec. 2013.
[28] K. Tanaka and H. O. Wang, *Fuzzy Control Systems Design and Analysis: A Linear Matrix Inequality Approach*. Hoboken, NJ, USA: Wiley, 2004.
[29] P. Qi, C. Qiu, H. Liu, J. S. Dai, L. Seneviratne, and K. Althoefer, "A novel continuum-style robot with multilayer compliant modules," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2014, pp. 3175–3180.
[30] J. J. Craig, *Introduction to Robotics: Mechanics and Control*. Englewood Cliffs, NJ, USA: Prentice-Hall, 2005.
[31] K. Tanaka, H. Yoshida, H. Ohtake, and H. O. Wang, "A sum-of-squares approach to modeling and control of nonlinear dynamical systems with polynomial fuzzy systems," *IEEE Trans. Fuzzy Syst.*, vol. 17, no. 4, pp. 911–922, Aug. 2009.
[32] H. O. Wang, K. Tanaka and M. F. Griffin, "An approach to fuzzy control of nonlinear systems: Stability and design issues," *IEEE Trans. Fuzzy Syst.*, vol. 4, no. 1, pp. 14–23, Feb. 1996.
[33] S. Prajna, A. Papachristodoulou, and P. A. Parrilo, *SOSTOOLS—Sum of Squares Optimization Toolbox, User's Guide*, The MathWorks Inc., Natick, MA, USA, 2002.
[34] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. London, U.K.: Springer-Verlag, 2009.
[35] T. Mahl, A. E. Mayer, A. Hildebrandt, and O. Sawodny, "A variable curvature modeling approach for kinematic control of continuum manipulators," in *Proc. Amer. Control Conf.*, 2013, pp. 4945–4950.



Peng Qi (S'11) received the B.Eng. degree in automation from Beijing Jiaotong University, Beijing, China, in 2010, the M.S. degree in electrical engineering from the KTH Royal Institute of Technology, Stockholm, Sweden, in 2012, and the Ph.D. degree in robotics from King's College London, London, U.K., in 2016.

He is currently a Research Fellow at the National University of Singapore. His research interests include design, modeling, and control of continuum/soft manipulators as surgical assistants.

Chuang Liu received the B.Eng. degree in mechanical engineering from Tsinghua University, Beijing, China, in 2011, and the M.Sc. degree in robotics from King's College London, London, U.K., in 2013, where he is currently working toward the Ph.D. degree.

His research interests include fuzzy-model-based control and its applications.

Ahmad Ataka received the Bachelor's degree in electrical engineering from Gadjah Mada University, Yogyakarta, Indonesia, in 2014. Since January 2015, he has been working toward the M.Phil./Ph.D. degrees in robotics at King's College London, London, U.K.

His research interests include motion planning, control, obstacle avoidance, and continuum manipulators.

Mr. Ataka is among the recipients of the first Indonesia Presidential Scholarships from the Indonesia Endowment Fund for Education.

H. K. Lam (M'98–SM'10) received the B.Eng. (Hons.) and Ph.D. degrees from the Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hong Kong, in 1995 and 2000, respectively.

In 2005, he joined King's College London, London, U.K., as a Lecturer, and he is currently a Reader. His current research interests include intelligent control systems and computational intelligence.

Kaspar Althoefer (M'03) received the Ph.D. degree from King's College London, London, U.K., in 1996.

In the same year, he joined the Department of Mechanical Engineering, King's College London, as a Lecturer, where he is currently a Professor of robotics and intelligent systems in the Department of Informatics. His current research interests include force and tactile sensors for medical applications, miniaturized optic-fiber-based sensing, medical robotics, soft and continuum robots, and neurofuzzy sensor signal analysis and classification.