frontiers in Robotics and AI

REVIEW
published: 24 September 2021
doi: 10.3389/frobt.2021.730330

A Survey for Machine Learning-Based Control of Continuum Robots

Xiaomei Wang[1,2], *Yingqi Li*[1] *and Ka-Wai Kwok*[1]*

[1]Department of Mechanical Engineering, The University of Hong Kong, Pokfulam, Hong Kong, SAR China, [2]Multi-Scale Medical Robotics Center Limited, Hong Kong, Hong Kong, SAR China

Soft continuum robots have been accepted as a promising category of biomedical robots, accredited to the robots' inherent compliance that makes them safely interact with their surroundings. In its application of minimally invasive surgery, such a continuum concept shares the same view of robotization for conventional endoscopy/laparoscopy. Different from rigid-link robots with accurate analytical kinematics/dynamics, soft robots encounter modeling uncertainties due to intrinsic and extrinsic factors, which would deteriorate the model-based control performances. However, the trade-off between flexibility and controllability of soft manipulators may not be readily optimized but would be demanded for specific kinds of modeling approaches. To this end, data-driven modeling strategies making use of machine learning algorithms would be an encouraging way out for the control of soft continuum robots. In this article, we attempt to overview the current state of kinematic/dynamic model-free control schemes for continuum manipulators, particularly by learning-based means, and discuss their similarities and differences. Perspectives and trends in the development of new control methods are also investigated through the review of existing limitations and challenges.

Keywords: continuum robots, data-driven control, inverse kinematics (IK), kinematic/dynamic model-free control, learning-based control, machine learning, reinforcement learning, soft robots

OPEN ACCESS

Edited by:
Noman Naseer,
Air University, Pakistan

Reviewed by:
Deepak Trivedi,
General Electric, United States
Muhammad Jawad Khan,
National University of Sciences and Technology (NUST), Pakistan

***Correspondence:**
Ka-Wai Kwok
kwokkw@hku.hk

Specialty section:
This article was submitted to Biomedical Robotics, a section of the journal Frontiers in Robotics and AI

Received: *24 June 2021*
Accepted: *17 August 2021*
Published: *24 September 2021*

Citation:
Wang X, Li Y and Kwok K-W (2021) A Survey for Machine Learning-Based Control of Continuum Robots. Front. Robot. AI 8:730330. doi: 10.3389/frobt.2021.730330

1 INTRODUCTION

Bioinspired by snakes, elephant trunks, and octopus tentacles, continuum robots are designed to structurally mimic their inherent dexterity and adaptability (Webster and Jones, 2010). In contrast to conventional rigid-link manipulators, "continuum" mechanisms leverage a series of continuous arcs without a skeletal structure to produce a bending motion (Robinson and Davies, 1999). Such design initially focuses on large-scale grasping, locomotion, and positioning in industrial applications (Robinson and Davies, 1999) or even urban search and rescue operations in confined environments (Jones and Walker, 2006a). The trade-off between high flexibility and low payload induces strict structural requirements. Gradually, with the reduced scale of continuum robots, the concerns are also diverted to the delicate steering of the slim robot body. The flexible characteristics of continuum robots are appropriate for surgical field applications. Enabling infinite degree-of-freedom (DoF) manipulations within small scales, continuum robots endow the target with flexible access and the patient with less invasion (Burgner-Kahrs et al., 2015). Moreover, pliable interventional devices with broad-range functions, such as catheters (Lee et al., 2018; Wang et al., 2018), afford much inspiration to the development of robotic continuum manipulators. Besides the mechanism of a robot, proper controllers and corresponding sensors are also necessary to guarantee accurate control performance.

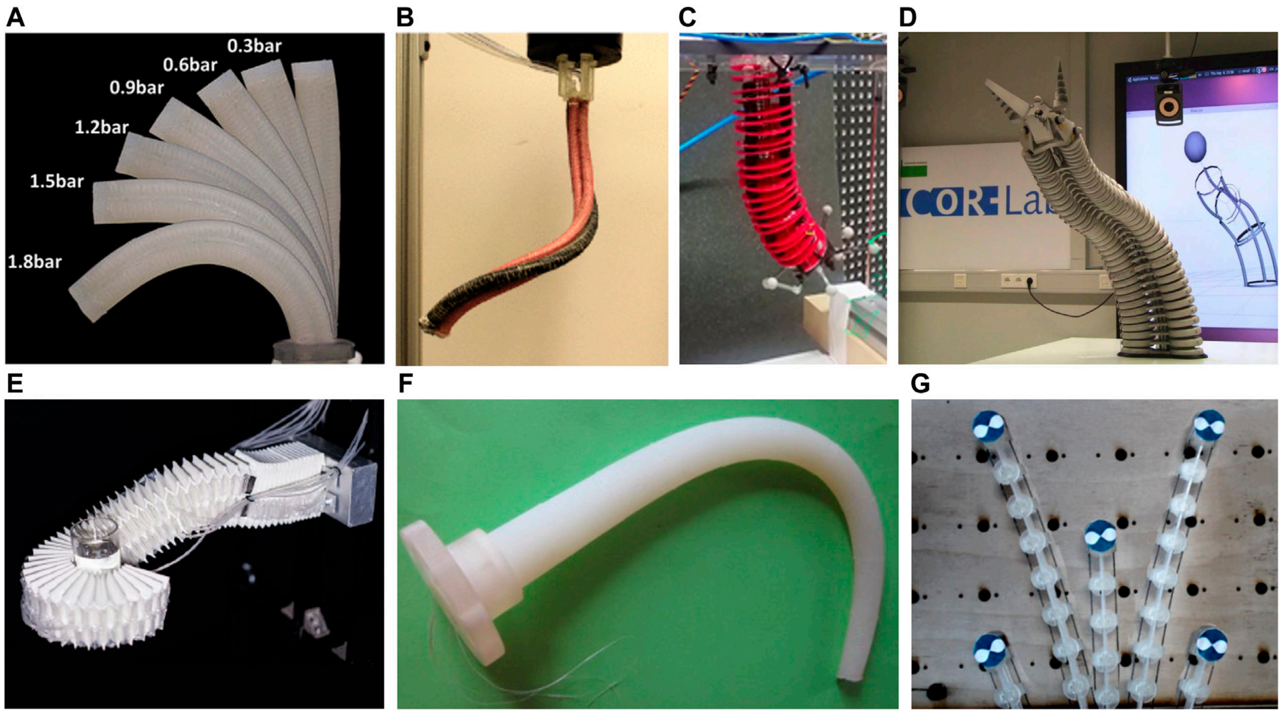

FIGURE 1 | Continuum robots driven by **(A–E)** pneumatics or **(F-G)** tendons. Image sources for **(A)–(G)** in sequence: (Lee et al., 2017a; Satheeshbabu et al., 2019; Ansari et al., 2017a; Rolf and Steil, 2013; Jiang et al., 2016; Giorelli et al., 2015a; Yip and Camarillo, 2014).

Conventional rigid-link robots could be controlled directly by the commands of motors on each joint. Once the joint angles and the link lengths are available, the pose of all points, including the end-effector, can be fully determined. Different from rigid-link mechanisms which have definite kinematic mapping, soft robots meet great challenges in accurate analytical modeling, considering the nonlinear deformation induced by the actuation, material elasticity, and susceptibility to contacts with surroundings. Generally speaking, the actuation mechanism of continuum robots (Rolf and Steil, 2013; Yip and Camarillo, 2014; Giorelli et al., 2015a; Ansari et al., 2017a; Lee et al., 2017a; Satheeshbabu et al., 2019; Jiang et al., 2021) can be categorized as intrinsic and extrinsic (Robinson and Davies, 1999), based on the location of the actuator. The intrinsic mechanism means that the actuators are located inside and form as part of the mechanism (Fu et al., 2020). One example could be pneumatic-driven robots (**Figures 1A–E**), whose deformation is induced by the inflation of internal elastic chambers. Extrinsic mechanisms use external components to distort the robot body (**Figures 1F,G**), such as tendons/cables dragged by motors. Due to the high compliance of continuum manipulators, constraints imposed by obstacle interactions may deform the robot body into undesired shapes regardless of the actuation status. These effects are generally difficult to completely sense and feed back into the model, leading to unstable behaviors. In addition, the individual variation also leads to modeling uncertainties. For example, even with the same design prototype, different degrees of fabrication errors may require repetitive parameter tuning for all robots and their actuators (Nordmann et al., 2012). The characteristics of a specific robot may also change (e.g., wear-out effects) over the course of time.

Machine learning approaches provide a promising way out for the control of continuum robots. As the controller or inverse kinematic mapping is identified by experimental sensory data, people also title it as data-driven control. Sometimes, to distinguish it from the quantitative modeling which parameterizes the system using compact representations (e.g., differential equations) (Lunze, 1998), there were researchers using qualitative modeling to describe such circumstances (Melingui et al., 2014a; Melingui et al., 2014b). Apart from circumventing prior analytical modeling and computational complexity, another advantage of the learning-based control is the simplified requirement on sensors. Representative models include the (piecewise) constant-curvature (PCC)-based kinematic model (Jones and Walker, 2006b; Webster and Jones, 2010), Cosserat rod theory (Trivedi et al., 2008; Rucker et al., 2010; Giorelli et al., 2012), and spring-mass model (Kang et al., 2013) would require the measurements of actuator length/strain/curvature for more accurate use of the model, if not relying on the feedback. Their effectiveness in the feed-forward loop could not be guaranteed, since the analytic shape reconstructed from the set of possible length combinations or other configuration parameters is unknown and non-stationary

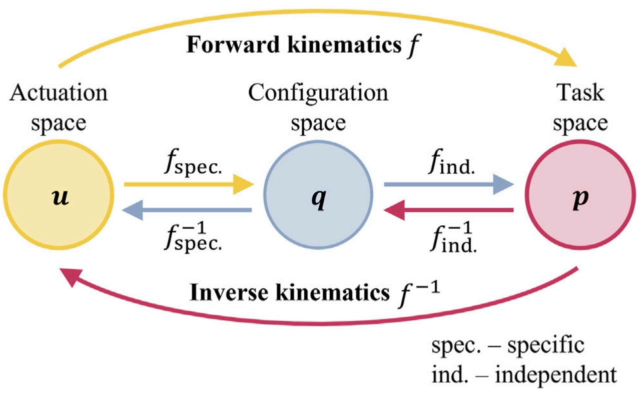

FIGURE 2 | Mapping relationship between the three spaces (i.e., $\boldsymbol{u}$: actuation space, $\boldsymbol{q}$: configuration space, and $\boldsymbol{p}$: task space) in soft robot control.

(Nordmann et al., 2012). Fortunately, direct learning of the end-to-end mapping only needs the information of actuators (e.g., motor position/command) and a sensor that can capture the end-effector (or other control objectives like the shape) in the task space. Therefore, the burden for multiple types of sensors is alleviated and even eliminated.

Various machine learning techniques have the potential or have been validated for continuum robot control in existing works. In this article, we aim to summarize and discuss their representative implementations. Different from previous reviews such as the one by George Thuruthel et al. (2018) that provides the outlines of all possible control strategies, here machine learning–based approaches are intensively investigated with details and systematic analysis. Starting with the iterative machine learning methods, supervised learning methods follow as a main part. Since supervised learning techniques, which utilize known input and output datasets for mapping establishment, accord suitably with the thought of robot control, lots of research works have been conducted. Several examples showing the combination of learning-based and analytical models are also introduced. However, as semi-supervised learning and unsupervised learning are designed to learn classifications from small-portion tagged data and patterns from untagged data, respectively, few types have been applied in robot manipulation, which targets definite and accurate regression results. Although they are possible to implement in specific control units in the future, we will not discuss such categories in this article. Finally, the attempts at reinforcement learning on continuum robots are also elaborated upon for the first time. Reinforcement learning refers to a family of learning-based algorithms where the agent autonomously learns to deal with new tasks during the interaction with its environment. Compared to supervised learning where the model learns from the "answer key" in training data, reinforcement learning enables the model to discover the optimal behavior policy from experience. Nowadays, reinforcement learning applied in the control of soft robots has been attracting lots of interest and developing fast since it could avoid prior knowledge of robot configuration. Additionally, not limited to the modeling in a specific workspace, reinforcement learning is expected to extend the manipulation adaptivity to a complex and dynamic environment (Polydoros and Nalpantidis, 2017). At the end of this article, potential trends in the development of machine learning–based control methods are also discussed after a summary of existing works.

2 ITERATION-BASED KINEMATIC MODEL-FREE CONTROL

In model-based controllers, the kinematic model can be decomposed into two mappings (Jones and Walker, 2006b), namely, robot-independent mapping and robot-specific mapping. These mappings link three spaces separately. The robot-independent mapping depicts the relation between the configuration and task spaces, while the robot-specific mapping represents the relation between the actuation and configuration spaces (**Figure 2**). The task space means the feasible region of the control object (usually the workspace of the end-effector $\boldsymbol{p}$). The actuation space represents the command $\boldsymbol{u}$ on motors or other actuation types which could be instructed quantitatively. In rigid-link robots, the kinematic mapping could be definitely established between the actuation and task spaces, since the configuration space is mostly linearly related to the actuation space. However, in soft continuum robots, the configuration parameters characterizing the arc idealize the robot deformation by making several assumptions. The material elasticity or fluid dynamics would induce large nonlinearity between the actuation and configuration spaces, which is difficult to involve in the modeling. That is also why the actuation-related part is called robot-specific mapping. There are model-free approaches utilizing the idea of iterative optimization to refine the desired mapping relationship and conduct control. Some representative control theories also involve similar ideas, such as the model predictive control (Tang et al., 2019a). In the following sections, representative methods to iteratively learn the mapping or its inverse format will be introduced. The conventional control idea utilizes the inverse Jacobian matrix to build the mapping between the actuation and task spaces. Although the Jacobian matrix may vary nonlinearly during the robot motion, someone rationally hypothesized that such variations within a single control interval are minimal and linear, assuming the slow movement of continuum robots.

2.1 Optimization-Based Jacobian Matrix Estimation

Yip and Camarillo (2014), Yip and Camarillo (2016), and Yip et al. (2017) proposed a series of model-free closed-loop controllers based on the optimal control strategy. Different from deducing the Jacobian matrix regarding the analytical model (Siciliano and Khatib, 2016), one advantage of such methods is the quick initialization of the Jacobian matrix; during the execution of the robot, the Jacobian matrix can also be updated. Using the symbols in **Figure 2**, the actuator input (at equilibrium) is represented as $\boldsymbol{u}(k) \in \mathbb{U}^m$ at time step k, where $\mathbb{U}^m$ denotes the m-dimensional actuation space, while the task space (e.g., the end-effector position in the Cartesian space $\in \mathbb{R}^3$) is denoted by $\boldsymbol{p}(k)$. With the inverse Jacobian matrix, the discretized relationship from the task space to the actuation space can be as follows:

$$\Delta \boldsymbol{u}(k) = \boldsymbol{J}^{-1}\Delta \boldsymbol{p}(k). \tag{1}$$

Assume that the robot actuation is designed as 3 DoFs which are independent from each other. The initialization of $\boldsymbol{J}$ can be finished by successively actuating the 3 DoFs in turn with an incremental amount Δu_i, $i = 1, 2, 3$ (i.e., the actuation commands are $[\Delta u_1 \quad 0 \quad 0]$, $[0 \quad \Delta u_2 \quad 0]$, and $[0 \quad 0 \quad \Delta u_3]$ in turn), and measuring the corresponding displacements $\Delta \boldsymbol{p_i}$, $i = 1, 2, 3$. The initial Jacobian matrix could be constructed as follows:

$$\boldsymbol{J} = [\boldsymbol{J}_1 \quad \boldsymbol{J}_2 \quad \boldsymbol{J}_3], \tag{2}$$

where $\boldsymbol{J}_i = \Delta \boldsymbol{p}_i / \Delta u_i$, $i = 1, 2, 3$. The Jacobian matrix $\boldsymbol{J}$ is updated by quadratic programming as follows:

$$\begin{array}{ll} \text{minimize} & \|\Delta \boldsymbol{J}(k)\| \\ \text{subject to} & \Delta \boldsymbol{p}(k) = \boldsymbol{J}(k)\Delta \boldsymbol{u}(k) \\ & \boldsymbol{J}(k) = \boldsymbol{J}(k-1) + \Delta \boldsymbol{J}(k) \end{array} \tag{3}$$

They also proposed to use force sensors to measure the tension of each tendon, therefore optimizing the actuation command with the minimal change. Besides the optimization-based construction of inverse kinematic mapping, machine learning–based methods have also been employed in various works, which are summarized in the following section.

2.2 Methods Utilizing Adaptive Kalman Filter

Li et al. (2017) considered the change of the Jacobian entries (forming system states $\boldsymbol{x}(k) \in \mathbb{R}^{m\times n}$ of the robot) as the process noise of the stochastic system and used an adaptive Kalman filter to estimate the entries, avoiding the kinematic modeling procedure. The state and measurement (displacement of the robot end-effector $\boldsymbol{y}(k) = \Delta \boldsymbol{p}(k) \in \mathbb{R}^m$) of the stochastic system are constructed as follows:

$$\begin{aligned} \boldsymbol{x}(k+1) &= \boldsymbol{x}(k) + \boldsymbol{\eta}(k) \\ \boldsymbol{y}(k) &= \boldsymbol{H}(k)\boldsymbol{x}(k) + \boldsymbol{\delta}(k), \end{aligned} \tag{4}$$

where the block-diagonal measurement matrix $\boldsymbol{H}(k) = \text{diag}(\Delta \boldsymbol{u}(k)^T, m) \in \mathbb{R}^{m\times nm}$ is composed by the change of actuation pressure/voltage $\Delta \boldsymbol{u}(k) = \boldsymbol{u}(k) - \Delta \boldsymbol{u}(k-1) \in \mathbb{R}^n$; The process and measurement noises are represented by $\boldsymbol{\eta}(k)$ and $\boldsymbol{\delta}(k)$, respectively, which are assumed to be Gaussian noises with covariance matrices of $\boldsymbol{Q}$ and $\boldsymbol{R}$. The two-step recursive formulas contribute to the calculation of the Jacobian matrix by the following:

$$\begin{aligned} \hat{\boldsymbol{x}}_p(k+1) &= \hat{\boldsymbol{x}}(k) + \boldsymbol{\eta}(k) \\ \boldsymbol{P}_p(k+1) &= \boldsymbol{P}(k) + \boldsymbol{Q}(k), \end{aligned} \tag{5}$$

and

$$\begin{aligned} \boldsymbol{K}(k) &= \boldsymbol{P_p}(k)\boldsymbol{H}^{\mathrm{T}}(k)\left[\boldsymbol{H}(k)\boldsymbol{P_p}(k)\boldsymbol{H}^{\mathrm{T}}(k) + \boldsymbol{R}\right]^{-1} \\ \boldsymbol{P}(k) &= [\boldsymbol{I} - \boldsymbol{K}(k)\boldsymbol{H}(k)]\boldsymbol{P_p}(k) \\ \hat{\boldsymbol{x}}(k) &= \hat{\boldsymbol{x}}_p(k) + \boldsymbol{K}(k)\left[\boldsymbol{y}(k) - \boldsymbol{H}(k)\hat{\boldsymbol{x}}_p(k)\right] \end{aligned} \tag{6}$$

where $\hat{\boldsymbol{x}}_p(k)$ and $\boldsymbol{P_p}(k)$ are separately the predicted state estimation and error covariance, their corresponding updates are $\hat{\boldsymbol{x}}(k)$ and $\boldsymbol{P}(k)$. The optimal Kalman gain is $\boldsymbol{K}(k)$. The detailed calculation approach can be found in the study by Li et al. (2017), where the use of an adaptive covariance matrix for the process noise $\boldsymbol{\eta}(k)$ improves the filter's convergence and tracking performance against system uncertainties. Based on the updated Jacobian matrix, an optimal vector is further implemented into the controller to determine the appropriate configuration allowing for the lowest deformation of the robot (i.e., minimal actuation variation).

3 SUPERVISED LEARNING OF INVERSE STATICS/KINEMATICS

This section will summarize the methods to learn the desired mapping in control offline, that is, using learning-based algorithms to approximate the statics/kinematics of the entire mapping from the actuation space to the task space. Here, the difference between statics models and kinematics models is briefly explained. A statics model depicts the robot configuration by assuming all forces on the manipulator at rest under equilibrium conditions. A kinematics model describes the robot motion only based on the geometric relationship, without considering the applied forces. For example, in the case of soft cable-driven manipulators, the direct forward statics model maps the cable tensions onto the tip position, while the inverse statics model calculates the cable tensions in order to make the tip on the desired position (Giorelli et al., 2013a; Thuruthel et al., 2016b). It is an absolute mapping scheme. The kinematics can be regarded as an approach to represent relative/incremental movements, where the change from the current robot status to the desired one is focused, or it can also be described as a mapping between the actuator velocity and the end-effector velocity. Accurate inverse statics can be implemented in open-loop control, and the inverse kinematics can be coordinated with the feedback in closed-loop control. As the most straightforward applications of learning-based algorithms, there are lots of articles from 2013 to 2020 (Giorelli et al., 2013a; Giorelli et al., 2013b; Melingui et al., 2014a; Giorelli et al., 2015a; Giorelli et al., 2015b; Chen and Lau, 2016; Thuruthel et al., 2016b; Thuruthel et al., 2016a; Lee et al., 2017a; Xu et al., 2017; Ho et al., 2018; Fang et al., 2019; Bern et al., 2020) utilizing this kind of thought to handle the continuum robot control. Details of these works are summarized in **Table 1**. For the convenience of showing their similarities and conducting systematic comparisons and discussions, we organized the following sections: **Section 3.1** and **Section 3.2**.

TABLE 1 | Sampling of inverse statics/kinematics learning-based control for continuum robots. Intended to be exemplary, not comprehensive.

Literature	Classification criteria							
	Model	**Robot structure**	**Actuation (length)**	**Act. & TaskDoFs**	**Mapping to be learned**	**Samples**	**Task**	**Accuracy/ mm (mean/ STD/max)**
Giorelli et al.	1-hidden-layer FNN							
a. Giorelli et al. (2013a)	a. 21 neurons	Silicone conical robot	Tendon (310 mm/ 280 mm)	a. 2 & 2	$\boldsymbol{u}(k) = f_s^{-1}(\boldsymbol{p}^*(k))$	500 (8:2)	Simulation: path generation. Discrete points	2.27/ 1.70/9.30
b. Giorelli et al. (2013b)	b. 34 neurons			b. 3 & 3		500 (8:2)		4.2/2.8/12.3
c. Giorelli et al. (2015a)	c. 6 neurons			c. 2 & 2		405 (8:2)		22.88/11.80/ 59.79
d. Giorelli et al. (2015b)	d. 28 neurons			d. 3 & 3		395 (8:2)		7.35/--/22.22
Melingui et al. (2014a)	Forward: FNN. Inverse: 1-hidden-layer NN in DSL	CBHA	Pneumatic (NA)	[a]3 × 2 & 3	Forward: $\boldsymbol{p}(k) = f_s(\boldsymbol{u}(k))$ Inverse: $\boldsymbol{u}(k) = f_s^{-1}(\boldsymbol{p}^*(k))$	4,096 (7: 1.5:1.5)	Comparison: robot and model postures (under the same actuation)	Inverse: 1.1 e^{-4} (MLP) ~ 4.1 e^{-4} (RBF)/--/--
Thuruthel et al.	1-hidden-layer NN.							
a. Thuruthel et al. (2016a)	a. 20 neurons	a. BHA	a. Pneumatic (0.9 m)	a. 3 × 3 & 3	$\boldsymbol{q}(k) = \hat{f}_s^{-1}(\boldsymbol{q}(k-1), \boldsymbol{p}^*(k))$	a. 10,000 (7:3)	Simulation: continuous path following in terms of position (P)/ orientation (O)	p: <1/ 1.504/--
b. Thuruthel et al. (2016b)	b. 40 neurons	b. Silicone conical	b. Tendon (31 cm)	b. 12 & 6	$\Delta\boldsymbol{q}(k) = \hat{f}_k^{-1}(\boldsymbol{q}(k-1), \Delta\boldsymbol{\theta}^*(k))$	b. 14,000 (8:2)		p: 8.5/2.8/-- O/°: 3.21/1.71/--
Chen and Lau (2016), Xu et al. (2017)	ELM, GMR, or KNNR	Silicone serpentine	Tendon (NA)	[a]2 & 2	$\boldsymbol{u}(k) = f_s^{-1}(\boldsymbol{p}^*(k))$	20,000	Simulation & real robot: path following	2.1275~ 2.5556/–/–
Lee et al. (2017a)	LWPR *online update*	Silicone cylindrical	Pneumatic (93 mm)	3 & 2	$\Delta\boldsymbol{u}(k) = f_k^{-1}(\boldsymbol{n}(k-1), \boldsymbol{u}(k-1), \Delta\boldsymbol{n}^*(k))$	>1,000 for initialization [FEA Lee et al. (2017b)]	Angular path following (2D) + external forces	Free space/°: 0.90/ 0.65/2.80 Disturbed/°: 2.49/1.74/ 11.03
Ho et al. (2018)	LWPR *online update*	Silicone cylindrical	Pneumatic (155 mm)	3 × 2 & 3	$\Delta\boldsymbol{u}(k) = f_k^{-1}(\boldsymbol{u}(k-1), \Delta\boldsymbol{p}^*(k))$	—	Path following (3D) + tip load (72% robot mass)	With load: 0.98/0.26/--
Fang et al. (2019)	LGPR *online update*	Silicone cylindrical	Pneumatic (67 mm)	3 & 2	$\Delta\boldsymbol{u}(k) = f_k^{-1}(\boldsymbol{u}(k-1), \Delta\boldsymbol{z}^*(k))$	300	Path following (2D visual servo) + tip load	Free space/ pixel: 5.4/--/11.5
Bern et al. (2020)	a. 2-hidden-layer FNN (20 × 2)	Silicone cylindrical	Tendon (20 cm)	a. 3 & 2	$\boldsymbol{u}(k) = f_s^{-1}(\boldsymbol{p}^*(k))$	a. 308 (9:1)	a. Real robot	Real robot: 6.2~9.2/–/–
	b: 3-hidden-layer FNN (25 × 3)			b. 3 × 2 & 2		b. 15414 (8: 1:1)	b. Simulation 2D path following	—

[a]*Only the actuation dimensions that are related to the end-effector control are considered; "×2" or "×3" means the number of segments in the manipulator.*
(F)NN, (feed-forward) neural network; GMR, Gaussian mixture regression; FEA, finite element analysis; DSL, distal supervised learning; KNNR, K-nearest neighbors regression; STD, standard deviation; (C)BHA, (compact) bionic handling assistant; LWPR, locally weighted projection regression; MLP, multilayer perceptron; ELM, extreme learning machine; LGPR, locally Gaussian process regression; RBF, radial basis function.

3.1 Mapping to Be Learned

As in **Figure 2**, successive mappings between the actuation and configuration spaces, the configuration and task spaces, or directly from the actuation space to the task space have to be defined as the forward kinematics. The actuator input (at equilibrium) is represented as $\boldsymbol{u}(k) \in \mathbb{U}^m$ at time step k, where

$\mathbb{U}^m$ denotes the m-dimensional actuation space. Let $\boldsymbol{q}(k)$ be the manipulator configuration parameters under input $\boldsymbol{u}(k)$, which corresponds to a specific task space status such as the end-effector position $\boldsymbol{p}(k) \in \mathbb{R}^3$ and the orientation normal $\boldsymbol{n}(k) \in \mathbb{R}^3$ in the Cartesian space. The collective pose variable $\boldsymbol{\theta}(k) = [\boldsymbol{p}(k), \boldsymbol{n}(k)] \in \mathbb{R}^6$ can be thus denoted. It should be noticed that for continuum robots, not only can the pose of an appointed point be defined as the control objective but also the entire manipulator shape can be represented by more DoFs. Here, we take the three-dimensional (3D) position $\boldsymbol{p}(k)$ as an example to show the representation of the absolute forward robot-independent mapping as follows:

$$\boldsymbol{p}(k) = f_{ind.}(\boldsymbol{q}(k)), \tag{7}$$

and that of absolute forward statics as follows:

$$\boldsymbol{p}(k) = f_s(\boldsymbol{u}(k)). \tag{8}$$

With quasi-static movements, the forward transition model can be expressed in the incremental format as follows:

$$\Delta\boldsymbol{p}(k) = \widehat{f}_k(\boldsymbol{q}(k-1), \Delta\boldsymbol{u}(k)), \tag{9}$$

where $\Delta\boldsymbol{u}(k) = \boldsymbol{u}(k) - \boldsymbol{u}(k-1)$ is the difference of inputs between time step k and $(k+1)$, and $\Delta\boldsymbol{p}(k) = \boldsymbol{p}(k) - \boldsymbol{p}(k-1)$ denotes the incremental displacement. The control objective of inverse statics is to generate an actuation command $\boldsymbol{u}(k)$, thus steering the manipulator to the desired $\boldsymbol{p}^\star(k)$ in the task space as follows:

$$\boldsymbol{u}(k) = f_s^{-1}(\boldsymbol{p}^\star(k)). \tag{10}$$

In inverse kinematics for closed-loop control, the change of actuation command $\Delta\boldsymbol{u}(k)$, thus achieving the desired movement $\Delta\boldsymbol{p}^\star(k) = \boldsymbol{p}^\star(k) - \boldsymbol{p}(k-1)$ in the task space, is calculated. Therefore, mapping **Eq. 11** is deduced to approximate the inverse kinematics of **Eq. 9**, as follows:

$$\Delta\boldsymbol{u}(k) = \widehat{f}_k^{-1}(\boldsymbol{q}(k-1), (\boldsymbol{p}^\star(k)). \tag{11}$$

The inverse transition $\widehat{f}_s^{-1}$ heavily depends on the last robot configuration $\boldsymbol{q}(k-1)$ that is supposed to be unknown during the training of an operation. However, since during quasi-static movements, the robot configuration $\boldsymbol{q}(k-1)$ can be represented/defined by the corresponding actuation $\boldsymbol{u}(k-1)$, the inverse kinematics function **Eq. 11** can be approximated as follows:

$$\Delta\boldsymbol{u}(k) = f_k^{-1}(\boldsymbol{u}(k-1), \Delta\boldsymbol{p}^\star(k)). \tag{12}$$

here, as long as the sensory information of the task space variable $\boldsymbol{p}$ and the encoded actuation command $\boldsymbol{u}$ are available, the inverse kinematics can be learned to accomplish various control tasks, without the need of analytical/quantitative modeling. According to the control object, the task space can be specified as the 2D/3D position $\boldsymbol{p}(k)$, the 2D/3D orientation $\boldsymbol{n}(k)$, or be defined in other coordinate frames. For example, in the visual-servoing tasks (Fang et al., 2019), the position of the end-effector $\boldsymbol{z}(k)$ in the 2D camera frame/view is desired to stay focused or follow paths. For some tasks like whole-arm grasping, the entire robot body should be considered and commanded.

3.2 Learning Approaches

As can be seen from **Table 1**, neural networks (NNs) are the most commonly used regression model to approximate the mapping. The feed-forward NN (FNN) is the basic type, where the information always flows from the input side to the output side with the weighted calculation of hidden layers (Svozil et al., 1997). If not specified, an NN usually indicates an FNN. Specific types of FNNs like the extreme learning machine (ELM) were also applied. Besides, some regression methods perform satisfying results in continuum robot control, and representative ones can be the locally weighted projection regression (LWPR) (Lee et al., 2017a; Ho et al., 2018) and (locally) Gaussian process regression (GPR) (Fang et al., 2019). It can be found that FNNs are adaptive to find the absolute relationships (Giorelli et al., 2013a; Giorelli et al., 2013b; Melingui et al., 2014a; Giorelli et al., 2015a; Giorelli et al., 2015b; Thuruthel et al., 2016a; Chen and Lau, 2016; Xu et al., 2017; Bern et al., 2020), for example, forward and inverse statics relations as in **Eq. 8** and **Eq. 10**. Such cases correspond to open-loop controls, where online sensory devices are not available. For the learning of relative mappings (Thuruthel et al., 2016b; Lee et al., 2017a; Ho et al., 2018; Fang et al., 2019) such as **Eq. 9** and **Eq. 11**, regression-based methods usually guarantee more stable convergence. Combined with corresponding sensing feedback, these mappings enable higher accuracy since the control loop can be closed. Most of the models were confirmed in advance using selected training samples. Related discussions of the data exploration can be found in **Section 3.3.1**. Regression models also support the online update using newly collected sensory data, but the trade-off will be the calculation load. For example, GPR requires the matrix inversion calculation (the matrix dimension is related to the sample amount) for each time of model refinement; therefore, the sampling window could not be too large. Locally, GPR models were therefore implemented in the study by Fang et al. (2019), where the whole workspace can be divided into several sections (≤300 samples for each) for independent model training and updating. Such k-means–guided localization could guarantee high computation speed (>20 Hz).

3.3 Problems to Be Considered

3.3.1 Data Exploration

For the offline trained models, collection and selection of the training data are crucial for the accuracy of the model. Requirements of the samples are as follows: 1) covering the whole workspace of the robot end-effector and 2) evenly distributed in the task space to ensure consistent estimation performance in all areas. Most existing works used a kind of motor babbling approach (Thuruthel et al., 2016b), applying the interpolation in the actuation or configuration space. The number of optimal samples will be dependent on the workspace range and motion step size. Sample (input–output) pairs were collected by incrementally actuating the motors (or other specific mechanisms) with a fixed increment and saving the corresponding end-effector status. With the actuation command in the safe range, this procedure will result in an ergodic dataset. De-noising and filtering were usually needed to abstract high-quality and nonrepetitive samples. To fully exploit the advantage of learning-based approaches, normalizations of the input and output data (into [−1,1] or [0,1]) were conducted before training. For the training of relative mappings, an additional process will be finding out all possible displacements between two random points and filtering out motions in a fixed range of step size (Fang et al., 2019). Samples were usually divided into training, testing and validation sets, where the training dataset usually occupies more than 70% of them. The concept of goal babbling (Rolf et al., 2010) was also proposed but more like a standalone control approach rather than a means of data exploration. It depended on an iterative bootstrapping and refinement procedure of the inverse kinematics estimates, by a path-based sampling approach. The training data were generated along paths (a set of linearly interpolated target points) *via* repeating the "trying to reach" process. This thought has been validated on the simulation of a bionic handling assistant (BHA) robot (Rolf and Steil, 2013).

3.3.2 Structural Optimization of Neural Networks

Although no prior knowledge of the robot modeling is required in data-driven approaches, there are hyper-parameters to be tuned, especially in the NNs, since LWPR and LGPR would not require the manual tuning of any hyper-parameters. The structural optimization of NNs focused on the number of hidden layers and neurons once a specific model was selected. Previous works of Giorelli et al. (2013a), Giorelli et al. (2013b), Giorelli et al. (2015a) and Bern et al. (2020) discussed the heuristic procedure. They observed that small-size networks usually had poor performances, which may not be expressive enough to capture the robot's physical characteristics. Increasing the size of the network yields greater accuracy; however, it may encounter a bottleneck or even induce a new trend of accuracy decrease once past a certain size. A hypothesis is that the gathered dataset was insufficiently large to properly train such a large NN. Overfitting is also a factor to be avoided. Considering the actuation DoFs, one hidden layer would be enough for single-segment robots, while the multi-segment redundant manipulators may require a similar increasing number of hidden layers.

3.3.3 Actuation/Configuration Redundancy

In the learning-based kinematic control, redundancy is the feature which is possible to generate inconsistent samples with the same effector pose but different joint angles or actuation commands. Learning from such examples will lead to invalid solutions (Lunze, 1998). The redundancy may result from the configuration space, that is, the robot-independent mapping (the configuration space to the task space) is redundant. This case exists in the multi-segment continuum robots, where the actuation space is 6D or 9D, but the task space only involves 3D positional information. Another cause of redundancy may derive from the actuation space, and this is a specific circumstance in tendon-driven or pneumatic-driven mechanisms. As sometimes the overall elongation change can be ignored, the relationship from the actuation space to the configuration space (robot-specific mapping) is also a multiple-to-single mapping. Both kinds of problems had been discussed and considered in several literatures. One way to resolve the redundancy in the actuation/configuration space is manually biasing the original training data to only allow a single inverse solution (Fang et al., 2019). This method is an optimal way out in single-segment manipulator control, since it can reduce the unnecessary complexity of training data without reducing the robot flexibility. However, for kinematically redundant (configuration-redundant) manipulators, the benefits of using multi-segment/DoFs are mostly lost and will result in improperly accomplished tasks (Peters and Schaal, 2008). For such manipulators, the alternative method is to introduce a reward/cost function to draw the system to a desired solution (Mahler et al., 2014). This is the most widely accepted approach, and an example can be found in the study by Ho et al. (2018), where the task was formulated as a constrained optimization problem. However, drawbacks of the cost function using the sum of the squared error Giorelli et al. (2013b) were raised in the study by Melingui et al. (2014b), pointing out that it is possible to yield nonconvex inverse mappings. They solved it by implementing a squared penalty term in the objective function of the inverse NN, thus selecting one particular inverse model from the redundancy manifold. In addition, the weighting scheme (Nordmann et al., 2012) and method taking both/all-segment trajectories into account (Melingui et al., 2015) had also been tested.

3.4 Combination of Analytical Model and Learning-Based Component

Either in general or specific tasks, there are hybrid controllers proposed, combining the analytical dynamics/kinematics model and learning-based approaches (e.g., NN) to accomplish robust control performance. Methods combining the learning-based and conventional components also appeared in several works. For example, Braganza et al. (2007) proposed a framework comprising a neural network feed-forward component and a nonlinear feedback component, where the NN based on augmented back propagation was utilized as a feed-forward compensator for the nonlinear uncertain dynamics. The proposed scheme enables continuous and asymptotic tracking without any prior knowledge of the robot dynamic model, and the back propagation technique enables fast online training of the NN weight matrices referring to the tracking error signal. Similarly, Queißer et al. (2014) presented the idea to superimpose a learning-based inverse equilibrium dynamics model for the feed-forward control and then combined it with a feedback controller. Subudhi and Morris (2009) implemented a hybrid fuzzy neural control (HFNC) scheme for a multi-link flexible manipulator. The control actions were determined by both a fuzzy controller (the primary loop) and an NN controller (the secondary loop) to compensate for the coupling effects. Tang et al. (2019a) and Tang et al. (2019b) proposed a control framework combining the model-free iterative learning (Rong-Hu and Zhong-Sheng, 2007) and model predictive control for trajectory-tracking control of a wearable soft robotic glove. The integration of the kinematic model and the machine learning trained model was also validated, and most of the learning-based parts acted as an error compensator of the analytical model, such as in the works of Reinhart and Steil (2016) and Reinhart et al. (2017).

4 REINFORCEMENT LEARNING STRATEGIES

With the development of artificial intelligence, reinforcement learning is emerging in the robotics community, which is a natural application for learning-based control since the interaction between robots and the environment is necessary. Reinforcement learning offers appealing tools enabling to complete sophisticated tasks and accommodate complex environments, which may be limited in conventional control strategies.

Reinforcement learning is regarded as a Markov decision process (MDP), represented using a tuple $(S, A, p(s'|s,a), R, \gamma)$. In the agent's interaction with the environment, S is the set of the agent's possible states, where s is the current state and s' is the next state after the agent transition. A presents the set of the agent's actions, where a is the action. $p(s'|s,a)$ is the state-transition probability of the agent transiting from the current state s to the future state s' after the implementation of action a. The states and actions constitute the trajectory $\tau = (s_0, a_0, s_1, a_1, \ldots)$. $R(s,\ a)$ defines the reward function after the agent executes action a at state s, and for convenience, $r(s,a)$ represents the immediate reward of one transition and R is the accumulated reward or expected return of the whole trajectory, as written in **Eq. 13**.

$$R(\tau) = \sum_{t=0}^{T} r(s_t, a_t), \tag{13}$$

where T is the number of time steps in the trajectory τ. For the infinite-horizon reinforcement learning problem, the effect of a future reward on the present decision could be considered with the reward discount factor γ ranging from 0 to 1, which is common for classical reinforcement learning (Kober et al., 2013). The accumulated reward is written as follows:

$$R(\tau) = \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t). \tag{14}$$

For the finite-horizon reinforcement learning problem, the average reward function is considered as shown in **Eq. 15**.

$$R(\tau) = \frac{1}{T}\sum_{t=0}^{T-1} r(s_t, a_t). \tag{15}$$

Policy $\pi(a|s)$ is the mapping from the state s to the action a, namely, given the current state, it could suggest the next step to obtain an optimal reward. The value function could evaluate the quality of the policy, offering the quantitative metric for the behavior decision maker. One of the value functions is called the state–value function $V^{\pi}(s)$, which defines the value of a state s under the policy π.

$$V^{\pi}(s) = E_{a\sim\pi}[R(\tau)|s_t = s]. \tag{16}$$

Another one is the action–value function $Q^{\pi}(s,a)$, which could assess the action a at state s under the policy π.

$$Q^{\pi}(s,a) = E_{a\sim\pi}[R(\tau)|s_t = s, a_t = a]. \tag{17}$$

Using **Eq. 13** and **Eq. 14** and the Bellman equation (Sutton and Barto, 2018), value functions could be written as follows:

$$V^{\pi}(s) = \sum_{a} \pi(a|s)\left[r(s,a) + \gamma \sum_{s'} P\left(s'\middle|s,a\right)V^{\pi}(s')\right], \tag{18}$$

$$Q^{\pi}(s,a) = r(s,a) + \gamma \sum_{s'} P\left(s'\middle|s,a\right)V^{\pi}(s'). \tag{19}$$

4.1 The Goal of Reinforcement Learning

In the context of mathematics, the goal of reinforcement learning is to explore an optimal policy that could instruct actions based on the present observation. The objective is to maximize the accumulated reward (**Eq. 13**), which determines the learning task. When considered in robot control, the goal of reinforcement learning is to figure out a control strategy that could generate optimal instruction for robot action in order to accomplish the specified task effectively. The reward function is designed manually to train the robot with certain characteristics, for example, penalizing the times of transition to enable the robot to reach the target in as few movements as possible.

For instance, there is a soft planar robot planned to touch a designated point in 2D space, where reinforcement learning can be explained as below. Sensors on the robot provide the observation about its relative position to the target, as well as moving velocity and direction, which describe the current state s. The soft robot is actuated using several inflating air chambers so that it could elongate or contract, thus steering the robot to the left and the right, indicating the action set A. Reward function $R(s,\ a)$ is designed manually, for example, the reward on short relative distance and the penalty on transition times could accelerate the learning convergence process. Policy $\pi(a|s)$ gives the action suggestion based on the observation of the current state to maximize the cumulative reward. **Figure 3** describes the pipeline of reinforcement learning algorithms in soft manipulator control. In the training stage (**Figure 3A**), trajectories calculated from the forward kinematic model or the simulation environment contribute to training the control policy. In the application stage (**Figure 3B**), every time upon receiving the sensor-observed state and target information, the learned control policy would give instructed actions, which would be executed by the actuator. Specificities of the application in continuum robots will be introduced in the following section.

4.2 Reinforcement Learning in Soft Robot Manipulation

Compared with the conventional joint-linked robots, the application of reinforcement learning in soft robots may face a number of challenges requiring specific attention. Reinforcement learning enables robots to learn from experience, which demands thousands of interactions with the environment. In addition to the tedious data collection, the noisy data, incomplete observation in practice, and the highly frequent movement could even damage the soft robot since it is mainly actuated hydraulically or pneumatically.

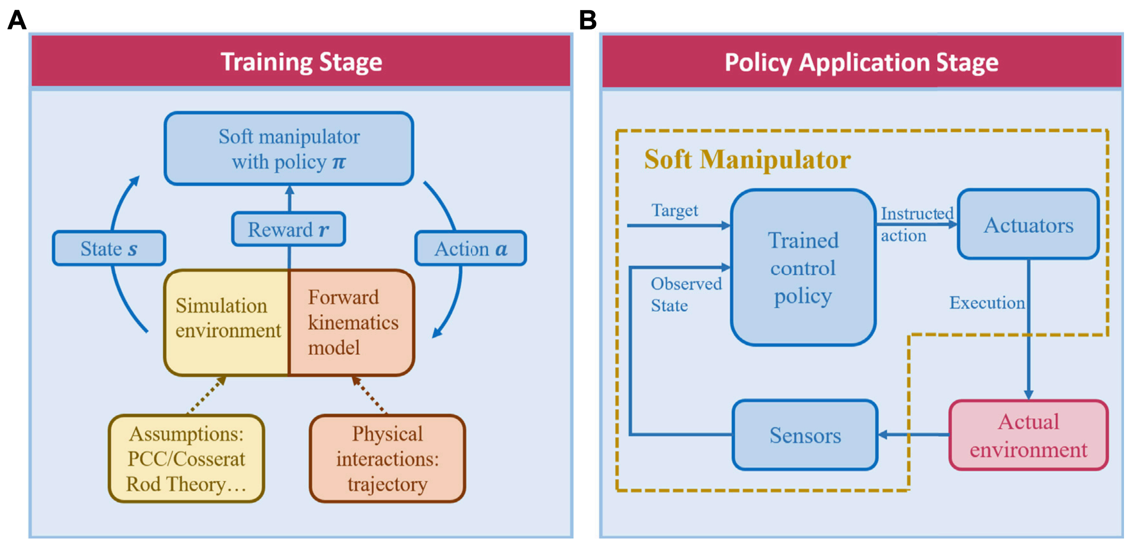

FIGURE 3 | Schematics of reinforcement learning in soft robot manipulation, with **(A)** the policy training stage and **(B)** the application stage separately shown.

To improve the effectiveness of training data collection, the model-free reinforcement learning approach has arisen, which obtains the learning experience from simulations. However, when transferred from the simulation to the prototype, the model-free algorithm might perform with large deviations since the modeling without real data could be inaccurate. Besides, the compliance and flexibility of soft robots cause high dimensional and continuous space, as to which the appropriate state and space discretization methods are expected in reinforcement learning. Furthermore, it is noticed that the soft manipulator would experience attenuation when exerted by external loads or disturbances. Not only could it impede the modeling of robots but it also makes the loading robustness experiment a necessity for the validation of reinforcement learning.

Referring to various criteria, reinforcement learning could be classified into different categories, such as model-based/model-free, policy-based/value-based, and averaged/discounted return function algorithms. The subsequent sections will introduce the first two kinds of categories in detail.

4.2.1 Model-Based Vs. Model-Free Reinforcement Learning

Upon the employment of explicit transition functions, reinforcement learning could be classified into two categories: model-based and model-free algorithms. In robotic applications, model-based methods mostly concentrate on forward kinematics/dynamics models which require prior knowledge on robots or environments (Moerland et al., 2020). Unlike the joint-linked robots soft robot modeling relies on data-driven methods or geometric assumptions to establish the forward model. Subsequently, the forward kinematics/dynamics model could generate robot trajectory data for policy optimization.

Model-free reinforcement learning would exploit the virtual training data for policy learning. That is, the data are obtained *via* robot simulations. It is worth noting that the modeling of the robot and the environment in the simulation is based on simplified assumptions such as PCC (Webster and Jones, 2010; Lee et al., 2017a) and Cosserat rod theory (Antman, 2005) rather than the real interaction data, and such modeling would not be counted as a model-based approach. In the following sections, for the sake of clear expression, methods with kinematics/dynamics model denote model-based reinforcement learning; while those without such model represent model-free approaches.

4.2.1.1 Reinforcement Learning With Kinematics/Dynamics Model

The policy trained on kinematics/dynamics model-based methods can perform stably, while in model-free methods, the derivation may be large when the algorithm is implemented from the simulation into the real robot. This is reasonable as the data from physical interaction are more realistic and persuasive than the one from simulation, where the virtual model is established using many simplifications and assumptions. Moreover, when the robot interacts with the environment in a circumstance that never appeared before, the policy might be invalid.

Thuruthel et al. (2018) leveraged the model-based policy learning algorithm on a simulated tendon-driven soft manipulator capable of touching dynamic targets. A nonlinear autoregressive network with exogenous inputs (NARX) was employed to establish the forward dynamic models using the observed data. Policy iteration from the sampled trajectories was used to give the optimal action directly. Wu et al. (2020) accomplished the position control of a cable-driven soft arm employing Deep Q-learning. Similar to the procedure in the study by Thuruthel et al. (2018), experiment data was collected to model the manipulator in simulation.

However, the shortcoming of kinematics/dynamics model-based reinforcement learning is that the physical interactions would be time-consuming and the data may be noisy; they might even bring more mechanical wear on the robot prototype, particularly the vulnerable soft robot (Polydoros and Nalpantidis, 2017). As a result, the approach not requiring physical data has drawn lots of attention from soft robot researchers.

4.2.1.2 Reinforcement Learning Without Kinematics/ Dynamics Model

In recent years, kinematics/dynamics model-free reinforcement learning implemented on the physical robot has attracted interest in the soft robot community, since it could circumvent the accuracy requirement of analytical modeling, which is hampered by the intrinsic nonlinearity and uncertain external disturbances. In general, the trajectory data are generated in the simulation environment for model-free reinforcement learning, and subsequently, the trained model would be transferred to the physical robot.

You et al. (2017) investigated a model-free reinforcement learning control strategy for a multi-segment soft manipulator, called honeycomb pneumatic networks (HPNs), capable of physically reaching the target in 2D space. The control policy was learned using Q-learning with the simulation data, demonstrating its effectiveness and robustness in simulation and practice. Jiang et al. (2021) adopted the same soft arm and developed a hierarchical control algorithm for complex tasks such as opening a drawer and rotating a handwheel. The control architecture was inspired by human decision-making process. The system consisted of three levels, sequentially low-level motion controller, high-level behavior controller and behavior planner. Q-learning was implemented in the low-level motion controller. The research demonstrated the feasibility of a relatively simple control algorithm on interaction tasks in unstructured environments. Considering that simple Q-learning cannot handle the high-dimension workspace and action of soft robots, Deep Q-learning which leverages the advantage of deep learning to learn policy is prevailing in continuous control. Satheeshbabu et al. (2019) proposed an open-loop position control strategy for a spatial soft arm named BR^2 based on Deep-Q Network (DQN). Cosserat rod formulation–based simulation is used for training data generation. The positioning accuracy (>94%) in simulation and the real environment, even with the external load, indicated that the control approach was effective and robust. You et al. (2019) introduced a control strategy enabling the soft catheter to move in a heart model using Dueling DQN (DDQN). This algorithm defines the Q-value as the summation of the state value and the advantage function. Isolating the value enables more precise state approximation and higher learning efficiency (Wang et al., 2016).

Ansari et al. (2017b) exploited model-free multi-agent reinforcement learning (MARL) on a soft arm to complete an assistive bathing task, where each actuator of the manipulator is regarded as an agent, sharing the common environment. The actor-critic algorithm consists of two parts. The actor performs as policy π, accepting the current state, thus generating the next action. The critic assesses the state-action tuple through the state-value function. In the study by Ansari et al. (2017b), the state-action-reward-state-action (SARSA) algorithm was adopted as the critic to evaluate the policy. In contrast to Q-learning, SARSA employs the same policy to sample and optimize, which is an on-policy learning algorithm. In the study by Satheeshbabu et al. (2020), the BR^2 soft continuum arm was improved with vision feedback and deep deterministic policy gradient (DDPG), which is a family of actor–critic algorithms. Compared to the previous open-loop control scheme, the closed-loop control method could not only decrease the error obviously but also enable the soft manipulator to track the relatively complex curve path. This research indicates that state feedback and closed-loop control could offset the real-world derivation in model-free reinforcement learning. Liu et al. (2020) presented a control strategy incorporating proximal policy optimization (PPO) and a central pattern generator (CPG) for soft robot snakes, which could track the planar changing goals. Comparison of training steps/time among these different tasks can be found in **Table 2**.

4.2.2 Policy-Based Vs. Value-Based Reinforcement Learning

In addition to the perspective of modeling, reinforcement learning algorithms could also be categorized in terms of the solution of optimal policy: the policy search, value-based, and actor–critic methods (Polydoros and Nalpantidis, 2017; Najar and Chetouani, 2020). Policy search methods could generate optimal policy $\pi^*(a|s)$ parameterized with θ directly using various methods: the gradient-based, sampling-based, and informatic theory methods (Polydoros and Nalpantidis, 2017). In the gradient-based approach, the policy function is maximized with gradient-descent iteratively (Thuruthel et al., 2018; Liu et al., 2020). In contrast to policy search reinforcement learning, value-based methods generate the optimal control policy by optimizing the value function, including SARSA (Ansari et al., 2017b), Q-learning (You et al., 2017; Jiang et al., 2021), DQN (Satheeshbabu et al., 2019; Wu et al., 2020) and its various extensions (e.g., DDQN (You et al., 2019) and Double DQN). The actor–critic approach is a combination of policy-based and value-based reinforcement learning, where the actor executes referring to the policy; thereby the critic calculates the value function to evaluate the actor (Satheeshbabu et al., 2020). Some algorithms (Satheeshbabu et al., 2019; Satheeshbabu et al., 2020; Wu et al., 2020) can be regarded as deep reinforcement learning, which means complex deep neural networks were applied in the control policy, rather than a simple state-action-reward table. This is common in the control for flexible manipulators with high-dimensional action and state variables. Deep reinforcement learning is capable of processing more complicated input formats including images (You et al., 2019) while storing more states and actions, which is particularly significant for continuous control.

5 DISCUSSION AND CONCLUSION

In this article, we surveyed the state-of-the-art machine learning–based control strategies of continuum robots. Compared with conventional modeling, learning-based mappings provide effective substitutes for analytical models in

TABLE 2 | Sampling of reinforcement learning control for continuum robots, including the algorithm, task, and training duration. Intended to be exemplary, not comprehensive.

Literature	Reinforcement learning algorithm	Task	Training steps/time	Distance error/ success rate
Thuruthel et al. (2018)	Policy search	3D position reaching	8,000 s	Without load: 0.009~0.017 m With load: 0.022 m
Wu et al. (2020)	Q-learning	2D position reaching	1,000 iterations	Without load: <0.5 cm With load: <1 cm
You et al. (2017)	Q-learning	2D position reaching	1,000 iterations	<10 mm
Jiang et al. (2021)	Q-learning	Interaction tasks including drawer opening and handwheel rotating	120 iterations (about 60 s) and 20,000 iterations (about 11 h) with/without the method of virtual goals	Task success rate 98.86%
Satheeshbabu et al. (2019)	DQN	3D position reaching	5,000 episodes[a]	3.05 cm
You et al. (2019)	DDQN	3D position reaching	100 episodes	6.58 ± 5.6 mm
Ansari et al. (2017b)	Actor–critic	3D position control	300 episodes	—
Satheeshbabu et al. (2020)	DDPG	3D path tracking	10,000 episodes	≤3 cm
Liu et al. (2020)	PPO	2D tracking with changing goals	6,400 episodes	—

[a]*One episode in reinforcement learning means a sequence of states, actions, and rewards, which ends with the terminal state. The time length of one episode depends on the specific task.*

TABLE 3 | A conclusive comparison of analytical-modeling-based and machine-learning-based control.

Aspects		Analytical modeling	Supervised learning*	Reinforcement learning
Human Intervention (Single robot)	Model derivation	✓	✗	✗
	Parameter tuning	✓	✗	✗
	Data collection	✗	✓ (offline)	✓ (online)
	Training	✗	✓	✓
Generalization (Same prototype)	Model derivation	✗	✗	✗
	Parameter tuning	✓	✗	✗
	Data collection	✗	✓ (offline)	✓ (online)
	Training	✗	✓	✓
Dependence on data		Low	high	high
Online refinement		✗	✓	✓
Adaptability to un-modeled disturbances		✗	✓	✓

**Cells marked with gray shading indicate advantages.*

the feed-forward control loop, without the need of manual model construction and calibration (note: a brief summary of their comparisons can be found in **Table 3**). The correction load utilizing the feedback loop is therefore reduced. The format of learning objects also varies a lot; in addition to the direct forward/ inverse kinematics/statics relationships, learning-based components also contribute to the optimization or compensation units in several works. For example, in the metaheuristics-assisted approach presented in the study by Amouri et al. (2017), particle swarm optimization (PSO) and the genetic algorithm (GA) were used to solve the optimization of PCC-deduced endpoint coordinates, and the constraints involved obstacle avoidance and tracking trajectories. Some learning-based models focus more on intermediate shape modeling rather than the final goal of end-effector control (Holsten et al., 2019).

All the methods mentioned in **Sections 3** and **4** facilitate the release of sensor variety demand. However, for those approaches that solely rely on sensory data, a drawback will be the high requirement on the quality of feedback information (i.e., task space sensing). No matter for iterative- or regression-based machine learning techniques, the distribution and accuracy of sensory data would play an important role, which has been discussed in **Section 3.3**. To adapt to the unstructured external conditions (e.g., contact forces), online updates should be an emphasis in future applications. This is also one major advantage of learning-based control, while relevant attempts have been made in recent works (Lee et al., 2017a; Ho et al., 2018; Fang et al., 2019). However, for such schemes like the ones in the studies by Yip and Camarillo (2014), Lee et al. (2017a), the question of how to characterize and resolve the outliers with low confidence of the ground truth needs to be carefully considered. Although excessive addition of sensors is not recommended, several self-contained measurement devices such as fiber Bragg gratings (FBGs) which can appropriately accommodate the flexible continuum body have the potential for sensor fusion with positional sensors (Lun et al., 2019; Wang et al., 2020; Wang et al., 2021). Meanwhile, the combination of analytical and data-driven models will be another trend for continuum robot control. Although the analytical approaches encounter challenges in parameter characterization and modeling uncertainties, the convergence of their solutions can usually be guaranteed. Its combination with online learning could leverage both of their respective advantages, which are convergent performance, no prior data exploration, and online control error compensation.

Section 5 concludes the various deployments of reinforcement learning in soft continuum robots and reveals its prospect in dealing with complex learning tasks automatically. Nevertheless, it is observed that most previous works concentrated on simplified tasks such as trajectory tracking and goal reaching, which only take little advantage of the powerful learning tool. Developing the learning ability in sophisticated applications is the main challenge of reinforcement learning in soft robots, even the whole robotics field. Additionally, the effectiveness of interaction data collection also hinders further development. Provided with kinematics/dynamics models, enhancing the validity and efficacy of data collection during environmental interactions will be a significant contribution to reinforcement learning. Without such models, the deviation between simulated and actual manipulators would be a big obstacle for reinforcement learning's application. To handle these cases, Sim-to-Real transfer approaches (Zhao et al., 2020) such as domain randomization and domain adaptation may drive a new research focus in the recent future.

AUTHOR CONTRIBUTIONS

Conceived the format of the review: XW and K-WK; curated the existing bibliography, contributed to figures/materials and the writing of the manuscript: XW and YL; proofreading and revision: XW, YL, and K-WK.

FUNDING

This study received funding in part from the Research Grants Council (RGC) of Hong Kong (17206818, 17205919, 17207020, and T42-409/18-R), in part from the Innovation and Technology Commission (ITC) (MRP/029/20X and UIM/353), and in part from Multi-Scale Medical Robotics Center Limited, InnoHK, under Grant EW01500.

REFERENCES

Amouri, A., Mahfoudi, C., Zaatri, A., Lakhal, O., and Merzouki, R. (2017). A Metaheuristic Approach to Solve Inverse Kinematics of Continuum Manipulators. *Proc. Inst. Mech. Eng. J. Syst. Control. Eng.* 231 (5), 380–394. doi:10.1177/0959651817700779

Ansari, Y., Manti, M., Falotico, E., Cianchetti, M., and Laschi, C. (2017). Multiobjective Optimization for Stiffness and Position Control in a Soft Robot Arm Module. *IEEE Robotics Automation Lett.* 3 (1), 108–115. doi:10.1109/LRA.2017.2734247

Ansari, Y., Manti, M., Falotico, E., Mollard, Y., Cianchetti, M., and Laschi, C. (2017). Towards the Development of a Soft Manipulator as an Assistive Robot for Personal Care of Elderly People. *Int. J. Adv. Robotic Syst.* 14 (2), 1729881416687132. doi:10.1177/1729881416687132

Antman, S. S. (2005). "Problems in Nonlinear Elasticity," in *Nonlinear Problems of Elasticity* New York City: Springer, 513–584.

Bern, J. M., Schnider, Y., Banzet, P., Kumar, N., and Coros, S. (2020). "Soft Robot Control with a Learned Differentiable Model," in 2020 3rd IEEE International Conference on Soft Robotics, Yale University, USA, May 15–July 15, 2020 (RoboSoft), 417–423. doi:10.1109/robosoft48309.2020.9116011

Braganza, D., Dawson, D. M., Walker, I. D., and Nath, N. (2007). A Neural Network Controller for Continuum Robots. *IEEE Trans. Robot.* 23 (6), 1270–1277. doi:10.1109/tro.2007.906248

Burgner-Kahrs, J., Rucker, D. C., and Choset, H. (2015). Continuum Robots for Medical Applications: A Survey. *IEEE Trans. Robot.* 31 (6), 1261–1280. doi:10.1109/tro.2015.2489500

Chen, J., and Lau, H. Y. (2016). "Learning the Inverse Kinematics of Tendon-Driven Soft Manipulators with K-Nearest Neighbors Regression and Gaussian Mixture Regression," in 2016 2nd International Conference on Control, Hong Kong, China, Apr 28–30, 2016 (Automation and Robotics (ICCAR), 103–107. doi:10.1109/iccar.2016.7486707

Fang, G., Wang, X., Wang, K., Lee, K.-H., Ho, J. D. L., Fu, H.-C., et al. (2019). Vision-based Online Learning Kinematic Control for Soft Robots Using Local Gaussian Process Regression. *IEEE Robot. Autom. Lett.* 4 (2), 1194–1201. doi:10.1109/lra.2019.2893691

Fu, H.-C., Ho, J. D. L., Lee, K.-H., Hu, Y. C., Au, S. K. W., Cho, K.-J., et al. (2020). Interfacing Soft and Hard: A spring Reinforced Actuator. *Soft Robotics* 7 (1), 44–58. doi:10.1089/soro.2018.0118

George Thuruthel, T., Ansari, Y., Falotico, E., and Laschi, C. (2018). Control Strategies for Soft Robotic Manipulators: A Survey. *Soft robotics* 5 (2), 149–163. doi:10.1089/soro.2017.0007

Giorelli, M., Renda, F., Calisti, M., Arienti, A., Ferri, G., and Laschi, C. (2012). "A Two Dimensional Inverse Kinetics Model of a cable Driven Manipulator Inspired by the octopus Arm," in 2012 IEEE international conference on robotics and automation, Saint Paul, MN, USA, May 14–19, 2012, 3819–3824. doi:10.1109/icra.2012.6225254

Giorelli, M., Renda, F., Calisti, M., Arienti, A., Ferri, G., and Laschi, C. (2015a). Neural Network and Jacobian Method for Solving the Inverse Statics of a Cable-Driven Soft Arm with Nonconstant Curvature. *IEEE Trans. Robot.* 31 (4), 823–834. doi:10.1109/tro.2015.2428511

Giorelli, M., Renda, F., Calisti, M., Arienti, A., Ferri, G., and Laschi, C. (2015b). Learning the Inverse Kinetics of an Octopus-Like Manipulator in Three-Dimensional Space. *Bioinspir. Biomim.* 10 (3), 035006. doi:10.1088/1748-3190/10/3/035006

Giorelli, M., Renda, F., Ferri, G., and Laschi, C. (2013a). "A Feed Forward Neural Network for Solving the Inverse Kinetics of Non-constant Curvature Soft Manipulators Driven by Cables," in Dynamic Systems and Control Conference, Palo Alto, CA, USA, Oct 21–23, 2013 (Stanford University), V003T38A001. doi:10.1115/dscc2013-3740

Giorelli, M., Renda, F., Ferri, G., and Laschi, C. (2013b). "A Feed-Forward Neural Network Learning the Inverse Kinetics of a Soft Cable-Driven Manipulator Moving in Three-Dimensional Space," in 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, Nov 3–7, 2013, 5033–5039. doi:10.1109/iros.2013.6697084

Ho, J. D. L., Lee, K.-H., Tang, W. L., Hui, K.-M., Althoefer, K., Lam, J., et al. (2018). Localized Online Learning-Based Control of a Soft Redundant Manipulator under Variable Loading. *Adv. Robotics* 32 (21), 1168–1183. doi:10.1080/01691864.2018.1528178

Holsten, F., Engell-Nørregård, M. P., Darkner, S., and Erleben, K. (2019). "Data Driven Inverse Kinematics of Soft Robots Using Local Models," in 2019 International Conference on Robotics and Automation (ICRA), Montreal, Canada, May 20–24, 2019, 6251–6257. doi:10.1109/icra.2019.8794191

Jiang, H., Liu, X., Chen, X., Wang, Z., Jin, Y., and Chen, X. (2016). "Design and simulation analysis of a soft manipulator based on honeycomb pneumatic networks," in 2016 IEEE International Conference on Robotics and Biomimetics (ROBIO), Qingdao, China, December 3–7, 2016, 350–356. doi:10.1109/ROBIO.2016.7866347

Jiang, H., Wang, Z., Jin, Y., Chen, X., Li, P., Gan, Y., et al. (2021). Hierarchical Control of Soft Manipulators towards Unstructured Interactions. *Int. J. Robotics Res.* 40 (1), 411–434. doi:10.1177/0278364920979367

Jones, B. A., and Walker, I. D. (2006). Kinematics for Multisection Continuum Robots. *IEEE Trans. Robot.* 22 (1), 43–55. doi:10.1109/tro.2005.861458

Jones, B. A., and Walker, I. D. (2006). Practical Kinematics for Real-Time Implementation of Continuum Robots. *IEEE Trans. Robot.* 22 (6), 1087–1099. doi:10.1109/tro.2006.886268

Kang, R., Branson, D. T., Zheng, T., Guglielmino, E., and Caldwell, D. G. (2013). Design, Modeling and Control of a Pneumatically Actuated Manipulator Inspired by Biological Continuum Structures. *Bioinspir. Biomim.* 8 (3), 036008. doi:10.1088/1748-3182/8/3/036008

Kober, J., Bagnell, J. A., and Peters, J. (2013). Reinforcement Learning in Robotics: A Survey. *Int. J. Robotics Res.* 32 (11), 1238–1274. doi:10.1177/0278364913495721

Lee, K.-H., Fu, D. K. C., Leong, M. C. W., Chow, M., Fu, H.-C., Althoefer, K., et al. (2017a). Nonparametric Online Learning Control for Soft Continuum Robot: An Enabling Technique for Effective Endoscopic Navigation. *Soft robotics* 4 (4), 324–337. doi:10.1089/soro.2016.0065

Lee, K.-H., Leong, M. C., Chow, M. C., Fu, H.-C., Luk, W., Sze, K.-Y., Yeung, C.-K., and Kwok, K.-W. (2017b). "FEM-Based Soft Robotic Control Framework for Intracavitary Navigation," in 2017 IEEE International Conference on Real-time Computing and Robotics (RCAR), Okinawa, Japan, Jul 14–18, 2017, 11–16. doi:10.1109/rcar.2017.8311828

Lee, K.-H., Fu, K. C. D., Guo, Z., Dong, Z., Leong, M. C. W., Cheung, C.-L., et al. (2018). MR Safe Robotic Manipulator for MRI-Guided Intracardiac Catheterization. *Ieee/asme Trans. Mechatron.* 23 (2), 586–595. doi:10.1109/tmech.2018.2801787

Li, M., Kang, R., Branson, D. T., and Dai, J. S. (2017). Model-Free Control for Continuum Robots Based on an Adaptive Kalman Filter. *IEEE/ASME Trans. Mechatronics* 23 (1), 286–297. doi:10.1109/TMECH.2017.2775663

Liu, X., Gasoto, R., Jiang, Z., Onal, C., and Fu, J. (2020). "Learning to Locomote with Artificial Neural-Network and Cpg-Based Control in a Soft Snake Robot," in 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, USA, Oct 25–29, 2020, 7758–7765. doi:10.1109/iros45743.2020.9340763

Lun, T. L. T., Wang, K., Ho, J. D. L., Lee, K.-H., Sze, K. Y., and Kwok, K.-W. (2019). Real-Time Surface Shape Sensing for Soft and Flexible Structures Using Fiber Bragg Gratings. *IEEE Robot. Autom. Lett.* 4 (2), 1454–1461. doi:10.1109/lra.2019.2893036

Lunze, J. (1998). Qualitative Modelling of Dynamical Systems: Motivation, Methods, and Prospective Applications. *Mathematics Comput. Simulation* 46 (5-6), 465–483. doi:10.1016/s0378-4754(98)00077-9

Mahler, J., Krishnan, S., Laskey, M., Sen, S., Murali, A., Kehoe, B., Patil, S., Wang, J., Franklin, M., and Abbeel, P. (2014). "Learning Accurate Kinematic Control of Cable-Driven Surgical Robots Using Data Cleaning and Gaussian Process Regression," in 2014 IEEE international conference on automation science and engineering (CASE), Taipei, Taiwan, Aug 18–22, 2014, 532–539. doi:10.1109/coase.2014.6899377

Melingui, A., Escande, C., Benoudjit, N., Merzouki, R., and Mbede, J. B. (2014). Qualitative Approach for Forward Kinematic Modeling of a Compact Bionic Handling Assistant Trunk. *IFAC Proc. Volumes* 47 (3), 9353–9358. doi:10.3182/20140824-6-za-1003.01758

Melingui, A., Lakhal, O., Daachi, B., Mbede, J. B., and Merzouki, R. (2015). Adaptive Neural Network Control of a Compact Bionic Handling Arm. *Ieee/asme Trans. Mechatron.* 20 (6), 2862–2875. doi:10.1109/tmech.2015.2396114

Melingui, A., Merzouki, R., Mbede, J. B., Escande, C., Daachi, B., and Benoudjit, N. (2014). "Qualitative Approach for Inverse Kinematic Modeling of a Compact Bionic Handling Assistant Trunk," in 2014 International Joint Conference on Neural Networks (IJCNN), Beijing, China, Jul 6–11, 2014, 754–761. doi:10.1109/ijcnn.2014.6889947

Moerland, T. M., Broekens, J., and Jonker, C. M. (2020). Model-Based Reinforcement Learning: A Survey. arXiv preprint arXiv:.16712.

Najar, A., and Chetouani, M. (2020). Reinforcement Learning with Human Advice. A Survey. arXiv preprint arXiv:.11016

Nordmann, A., Rolf, M., and Wrede, S. (2012). "Software Abstractions for Simulation and Control of a Continuum Robot," in International Conference on Simulation, Modeling, and Programming for Autonomous Robots, Tsukuba, Japan, Nov 5–8, 2012, 113–124. doi:10.1007/978-3-642-34327-8_13

Peters, J., and Schaal, S. (2008). Learning to Control in Operational Space. *Int. J. Robotics Res.* 27 (2), 197–212. doi:10.1177/0278364907087548

Polydoros, A. S., and Nalpantidis, L. (2017). Survey of Model-Based Reinforcement Learning: Applications on Robotics. *J. Intell. Robot Syst.* 86 (2), 153–173. doi:10.1007/s10846-017-0468-y

Queißer, J. F., Neumann, K., Rolf, M., Reinhart, R. F., and Steil, J. J. (2014). "An Active Compliant Control Mode for Interaction with a Pneumatic Soft Robot," in 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, IL, USA, Sep 14–18, 2014, 573–579.

Reinhart, R. F., and Steil, J. J. (2016). Hybrid Mechanical and Data-Driven Modeling Improves Inverse Kinematic Control of a Soft Robot. *Proced. Technol.* 26, 12–19. doi:10.1016/j.protcy.2016.08.003

Reinhart, R., Shareef, Z., and Steil, J. (2017). Hybrid Analytical and Data-Driven Modeling for Feed-Forward Robot Control †. *Sensors* 17 (2), 311. doi:10.3390/s17020311

Robinson, G., and Davies, J. B. C. (1999). "Continuum Robots-A State of the Art," in Proceedings 1999 IEEE international conference on robotics and automation (Cat. No. 99CH36288C), Detroit, MI, USA, May 10–15, 1999, 2849–2854.

Rolf, M., and Steil, J. J. (2013). Efficient Exploratory Learning of Inverse Kinematics on a Bionic Elephant Trunk. *IEEE Trans. Neural networks Learn. Syst.* 25 (6), 1147–1160. doi:10.1109/TNNLS.2013.2287890

Rolf, M., Steil, J. J., and Gienger, M. (2010). Goal Babbling Permits Direct Learning of Inverse Kinematics. *IEEE Trans. Auton. Ment. Dev.* 2 (3), 216–229. doi:10.1109/tamd.2010.2062511

Rong-Hu, C., and Zhong-Sheng, H. (2007). Dual-Stage Optimal Iterative Learning Control for Nonlinear Non-Affine Discrete-Time Systems. *Acta Automatica Sinica* 33 (10), 1061–1065. doi:10.1360/aas-007-1061

Rucker, D. C., Jones, B. A., and Webster III, R. J., III (2010). A Geometrically Exact Model for Externally Loaded Concentric-Tube Continuum Robots. *IEEE Trans. Robot.* 26 (5), 769–780. doi:10.1109/tro.2010.2062570

Satheeshbabu, S., Uppalapati, N. K., Chowdhary, G., and Krishnan, G. (2019). "Open Loop Position Control of Soft Continuum Arm Using Deep Reinforcement Learning," in 2019 International Conference on Robotics and Automation (ICRA), Montreal, Canada, May 20–24, 2019, 5133–5139. doi:10.1109/icra.2019.8793653

Satheeshbabu, S., Uppalapati, N. K., Fu, T., and Krishnan, G. (2020). "Continuous Control of a Soft Continuum Arm Using Deep Reinforcement Learning," in 2020 3rd IEEE International Conference on Soft Robotics (RoboSoft), United States, May 15–July 15, 2020 (Yale University), 497–503.

Siciliano, B., and Khatib, O. (2016). *Springer Handbook of Robotics*. Berlin: Springer.

Subudhi, B., and Morris, A. S. (2009). Soft Computing Methods Applied to the Control of a Flexible Robot Manipulator. *Appl. Soft Comput.* 9 (1), 149–158. doi:10.1016/j.asoc.2008.02.004

Sutton, R. S., and Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT press.

Svozil, D., Kvasnicka, V., and Pospichal, J. (1997). Introduction to Multi-Layer Feed-Forward Neural Networks. *Chemometrics Intell. Lab. Syst.* 39 (1), 43–62. doi:10.1016/s0169-7439(97)00061-0

Tang, Z. Q., Heung, H. L., Tong, K. Y., and Li, Z. (2019a). "A Novel Iterative Learning Model Predictive Control Method for Soft Bending Actuators," in 2019 International Conference on Robotics and Automation (ICRA), Montreal, Canada, May 20–24, 2019, 4004–4010. doi:10.1109/icra.2019.8793871

Tang, Z. Q., Heung, H. L., Tong, K. Y., and Li, Z. (2019b). Model-Based Online Learning and Adaptive Control for a "Human-Wearable Soft Robot" Integrated System. *Int. J. Robotics Res.* 40 (1), 256–276. doi:10.1177/0278364919873379

Thuruthel, T. G., Falotico, E., Cianchetti, M., and Laschi, C. (2016a). "Learning Global Inverse Kinematics Solutions for a Continuum Robot," in Symposium on Robot Design, Lisbon, Portugal, July 29–31, 2016, (Dynamics and Control), 47–54. doi:10.1007/978-3-319-33714-2_6

Thuruthel, T., Falotico, E., Cianchetti, M., Renda, F., and Laschi, C. (2016b). "Learning Global Inverse Statics Solution for a Redundant Soft Robot," in Proceedings of the 13th International Conference on Informatics in Control, Udine, Italy, Jun 20–23, 2016 (Automation and Robotics), 303–310. doi:10.5220/0005979403030310

Thuruthel, T. G., Falotico, E., Renda, F., and Laschi, C. (2018). Model-based Reinforcement Learning for Closed-Loop Dynamic Control of Soft Robotic Manipulators. *IEEE Trans. Robotics* 35 (1), 124–134. doi:10.1109/TRO.2018.2878318

Trivedi, D., Lotfi, A., and Rahn, C. D. (2008). Geometrically Exact Models for Soft Robotic Manipulators. *IEEE Trans. Robot.* 24 (4), 773–780. doi:10.1109/tro.2008.924923

Wang, K., Mak, C. H., Ho, J. D.-L., Liu, Z.-Y., Sze, K. Y., Wong, K. K., et al. (2021). Large-scale Surface Shape Sensing with Learning-Based Computational Mechanics. *Adv. Intell. Syst*. doi:10.1002/aisy.202100089 (Accepted)

Wang, X., Fang, G., Wang, K., Xie, X., Lee, K.-H., Ho, J. D. L., et al. (2020). Eye-in-Hand Visual Servoing Enhanced with Sparse Strain Measurement for Soft Continuum Robots. *IEEE Robot. Autom. Lett.* 5 (2), 2161–2168. doi:10.1109/lra.2020.2969953

Wang, X., Lee, K.-H., Fu, D. K. C., Dong, Z., Wang, K., Fang, G., et al. (2018). Experimental Validation of Robot-Assisted Cardiovascular Catheterization: Model-Based versus Model-free Control. *Int. J. CARS* 13 (6), 797–804. doi:10.1007/s11548-018-1757-z

Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M., and Freitas, N. (2016). "Dueling Network Architectures for Deep Reinforcement Learning," in International conference on machine learning, New York City, NY, USA, Jun 19–24, 2016, 1995–2003.

Webster, R. J., and Jones, B. A. (2010). Design and Kinematic Modeling of Constant Curvature Continuum Robots: a Review. *Int. J. Robotics Res.* 29 (13), 1661–1683. doi:10.1177/0278364910368147

Wu, Q., Gu, Y., Li, Y., Zhang, B., Chepinskiy, S. A., Wang, J., et al. (2020). Position Control of Cable-Driven Robotic Soft Arm Based on Deep Reinforcement Learning. *Information* 11 (6), 310. doi:10.3390/info11060310

Xu, W., Chen, J., Lau, H. Y. K., and Ren, H. (2017). Data-Driven Methods towards Learning the Highly Nonlinear Inverse Kinematics of Tendon-Driven Surgical Manipulators. *Int. J. Med. Robotics Comput. Assist. Surg.* 13 (3), e1774. doi:10.1002/rcs.1774

Yip, M. C., and Camarillo, D. B. (2014). Model-less Feedback Control of Continuum Manipulators in Constrained Environments. *IEEE Trans. Robot.* 30 (4), 880–889. doi:10.1109/tro.2014.2309194

Yip, M. C., and Camarillo, D. B. (2016). Model-less Hybrid Position/force Control: a Minimalist Approach for Continuum Manipulators in Unknown, Constrained Environments. *IEEE Robot. Autom. Lett.* 1 (2), 844–851. doi:10.1109/lra.2016.2526062

Yip, M. C., Sganga, J. A., and Camarillo, D. B. (2017). Autonomous Control of Continuum Robot Manipulators for Complex Cardiac Ablation Tasks. *J. Med. Robot. Res.* 02 (01), 1750002. doi:10.1142/s2424905x17500027

You, H., Bae, E., Moon, Y., Kweon, J., and Choi, J. (2019). Automatic Control of Cardiac Ablation Catheter with Deep Reinforcement Learning Method. *J. Mech. Sci. Technol.* 33 (11), 5415–5423. doi:10.1007/s12206-019-1036-0

You, X., Zhang, Y., Chen, X., Liu, X., Wang, Z., Jiang, H., and Chen, X. (2017). "Model-free Control for Soft Manipulators Based on Reinforcement Learning," in 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, Canada, Sep 24–28, 2017, 2909–2915. doi:10.1109/iros.2017.8206123

Zhao, W, Queralta, J P, and Westerlund, T (2020). "Sim-to-real transfer in deep reinforcement learning for robotics: a survey," in in IEEE Symposium Series on Computational Intelligence (SSCI), Canberra, Australia, Vancouver, Canada, December 1–4, 2020, 1737–744. doi:10.1109/SSCI47803.2020.9308468

Conflict of Interest: Author XW was employed by the company Multi-Scale Medical Robotics Center Limited.

The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.