actuators

MDPI

Article

Hybrid Control of Soft Robotic Manipulator

Arnau Garriga-Casanovas [1], Fahim Shakib [2,*], Varell Ferrandy [1] and Enrico Franco [1,*]

[1] Mechatronics in Medicine Lab, Hamlyn Centre, Imperial College London, South Kensington, London SW7 2AZ, UK; a.garriga-casanovas14@imperial.ac.uk (A.G.-C.); v.ferrandy22@imperial.ac.uk (V.F.)
[2] Department of Electrical and Electronic Engineering, Imperial College London, South Kensington, London SW7 2AZ, UK
* Correspondence: m.shakib@imperial.ac.uk (F.S.); e.franco11@imperial.ac.uk (E.F.)

Abstract: Soft robotic manipulators consisting of serially stacked segments combine actuation and structure in an integrated design. This design can be miniaturised while providing suitable actuation for potential applications that may include endoluminal surgery and inspections in confined environments. The control of these robots, however, remains challenging, due to the difficulty in accurately modelling the robots, in coping with their redundancies, and in solving their full inverse kinematics. In this work, we explore a hybrid approach to control serial soft robotic manipulators that combines machine learning (ML) to estimate the inverse kinematics with closed-loop control to compensate for the remaining errors. For the ML part, we compare various approaches, including both kernel-based learning and more general neural networks. We validate the selected ML model experimentally. For the closed-loop control part, we first explore Jacobian formulations using both synthetic models and numerical approximations from experimental data. We then implement integral control actions using both these Jacobians, and evaluate them experimentally. In an experimental validation, we demonstrate that the hybrid control approach achieves setpoint regulation in a robot with six inputs and four outputs.

Keywords: soft robotics; machine learning; closed-loop control

check for updates

Citation: Garriga-Casanovas, A.; Shakib, F.; Ferrandy, V.; Franco, E. Hybrid Control of Soft Robotic Manipulator. *Actuators* **2024**, *13*, 242. https://doi.org/10.3390/act13070242

Academic Editor: Han Yuan

Received: 27 May 2024
Revised: 19 June 2024
Accepted: 25 June 2024
Published: 27 June 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Soft robotic manipulators are attractive since they offer multiple degrees of freedom (DOFs), a compliant structure, and the potential for miniaturisation [1,2]. This makes them potential candidates for applications that may include endoluminal surgery [3–5] and industrial inspections [6–8]. An important part of these soft robots is actuated by pressurised fluids, typically air, and consists of a set of segments with 2–3 DOFs of actuation, each stacked serially. One such example is [9,10], which we use as reference in this work to explore control methods since it is a versatile soft robot that can be applied to a range of applications while offering higher force for a given diameter than similar designs.

The control of soft robots is a challenge due to their significant nonlinearity, continuous deformation along the arc length, and variability between prototypes. This last factor makes it difficult to create an accurate robot model and typical models contain significant uncertainty. Accurate finite element models have been developed [11,12], but even small variations in the manufacturing process due to practical tolerances lead to significant variations in robot behaviour. Despite these uncertainties, model-based closed-loop dynamic control has been investigated by these authors [13]. However, it has so far only been applied to control the robot in a predefined sub-region of the workspace, and while current models capture the robot behaviour locally, they fail to generalise to the full workspace. More generally, models have been developed for soft robots [14], but they typically need to be tuned to the specific properties of each prototype, which vary due to manufacturing tolerances. Open-loop control has also been developed [15], but it leads to average errors of 10 mm and it cannot cope with disturbances. Stiffness control using impedance control

has been developed for continuum robots [16], which can also include a gripper [17], but it is applied to a one-segment robot that is tendon-driven rather than pneumatic.

Machine learning (ML) approaches for modelling or control have also been developed for soft robot control [18–20] with approaches that can cope with hysteresis and nonstationary behaviour [21], and approaches that perform model predictive control based on ML for a fundamental model [22]. The main drawback of these approaches is that these tend to require significant amounts of training data, typically in the order of ten thousand or more datapoints. Reinforcement learning (RL) has also been explored for the control of soft robots [23–25], but it also requires significant amounts of data. Soft robots, and in particular the robot used in this work [9], have a limited life and noticeably degrade with use after a few thousand cycles. Therefore, only a fraction of those cycles can be used to collect experimental data for training. Hybrid approaches combining model predictive control with iterative learning have also been proposed [26] although they are applied to a single-segment robot. Similarly, nonparametric and online learning for a continuum robot [19] and AI-based navigation of a bronchoscope [27] are applied to a single robot segment.

Potential applications such as surgery and confined space inspections typically require precise robot control using the full workspace and multiple segments, and these operations can last a few hours, which can be equivalent to a few thousand pressurisation cycles. Thus, existing model-based control approaches [13] are generally not suitable given the full workspace requirements. Pure ML approaches [18] and RL solutions are also not suitable given the significant amounts of training data they require to achieve precise operation in the full workspace, which deplete most of the life of the soft robot used in this work. It should be noted that the specific requirements can vary, and in some applications that require operating in a subregion of the workspace for hundreds of cycles, the existing approaches can be appropriate. However, in general, a more robust and broadly applicable control approach is desirable.

In this paper, we propose a hybrid approach to the setpoint control of the soft robot, which combines machine learning and integral feedback control. Using machine learning, we train an inverse kinematics model on experimentally obtained data to provide an estimate of the required inputs to reach the desired setpoint. By integral feedback control, we then regulate the pose of the soft robot resulting from the input estimated by the inverse kinematics ML model to the setpoint. The work focuses on a robot actuated in two segments, which has six pressure inputs and four pose coordinates as outputs. The inverse kinematics is a mapping from the pose coordinates to the pressure inputs. We learn this mapping using feedforward neural networks and kernel-based machine learning on a limited-sized experimentally obtained dataset. Both neural network and kernel-based models can effectively learn from limited-sized and sparse datasets, with minimal user expertise and robust regularisation techniques to prevent overfitting [28,29].

We then validate these models experimentally. After that, we use these models to generate an initial estimate of the pressure required to reach a desired pose. We then show that with integral feedback control, we can reach the desired pose with satisfactory accuracy. This work is essential for the soft robot to be used as a reliable actuator, and in the future carry a surgical payload.

This work builds on our previous work [30], where a neural network is used for initial pressure estimation followed by an integral control. However, our previous work [30] only allows for robot operation in a small region of the workspace, where there is a high density of training datapoints, and uses only four pressure inputs and three position outputs. Moreover, our previous work relies on a model-free approach, where the relation between pressure inputs and outputs is simply based on the sign of such relation, where an increase in an input is directly mapped to an increase/decrease in an output based only on the sign. This sign-based approach restricts its application to limited regions in the workspace.

The contributions of this work are the following: (i) a comprehensive exploration and comparison of machine learning approaches (both feedforward neural networks and kernel-based models) to learn the full direct and inverse kinematics of a full two-segment

robot with six pressure inputs and four pose outputs covering the full workspace and using limited training data; (ii) an experimental validation of the selected machine learning model, which can cope with redundancies in the kinematics; (iii) a hybrid control approach combining machine learning with closed-loop control that can operate in the entire robot workspace given that the control is based on the robot Jacobian; (iv) an exploration and experimental comparison of both synthetic and data-based Jacobian approximations used for control.

The hybrid approach is well suited to cope with the nonlinearities and redundancies of the soft robot. The ML part learns the kinematics of each specific prototype using a viable number of training datapoints, and serves as a first estimate of the required pressure inputs accounting for the nonlinearities. The ML part also copes with redundancies, such as six inputs and four pose outputs, and steers the inputs and outputs towards the desired setpoint. The ML estimate does not lead to high precision since this would require an excessively high number of datapoints, but the remaining errors are then compensated by the integral feedback control. Moreover, the same training datapoints are used to estimate the data-based Jacobian for feedback control, accounting for nonlinearities and behaviour that can be unique to each prototype.

The paper is structured as follows. The control problem is formulated in Section 2. Machine learning approaches for both direct and inverse kinematics are proposed, validated, and compared in Section 3. The hybrid closed-loop control is presented in Section 4 together with experiments to validate the work. Finally, concluding remarks are summarised in Section 5.

2. Problem Formulation

The aim of this work is to control a soft continuum robot consisting of two segments. Each segment is designed following [31] with the robot design reported in [9]. In summary, each segment has a tubular design with three internal chambers equally spaced at 120 degrees and partition walls with a zig-zag configuration. The robot has an inextensible central tube running along its centre, to which discs at the ends of the segments are glued. The chambers in each segment are designed to expand when pressurised, pushing the central tube to one side, and creating a morphing design that maximises the bending force [10].

Each segment bends approximately as a constant curvature arc. The two-segment robot can thus be considered to bend in a piecewise constant curvature fashion when no external loads are applied. Furthermore, each segment has three pressure inputs to each of its chambers, and it offers two DOFs, which can be viewed as a bending direction and a bending angle. Therefore, the two-segment robot has a total of four DOF outputs. Consequently, there are two degrees of redundancy a priori between the six actuation inputs and the four DOF outputs.

In practice, at least one chamber in each segment is always set to have zero pressure since this is the chamber at the side towards which the robot bends. Applying pressure in all three chambers of a segment simultaneously simply increases stiffness but does not increase kinematic DOFs. In this work, we are not concerned with the robot stiffness, only with the control of its position in free space.

The objective of this work is to perform setpoint regulation of the two-segment robot, which has four DOFs as regulation outputs, with six pressure inputs and the condition that at least one chamber in each segment should have approximately zero pressure. We set the four DOFs as three position coordinates in Euclidean space X, Y, Z, and tip bending angle θ. These position coordinates are collected in the vector $x^\top := \begin{bmatrix} X, Y, Z, \theta \end{bmatrix}$. The pressure inputs are collected in a vector $p^\top := \begin{bmatrix} p_1, \ldots, p_6 \end{bmatrix}$.

The robot has a limited life of approximately 10,000 cycles. It also degrades with time, leading to noticeable behaviour changes at roughly 5000 cycles (the specific value of cycles changes significantly between prototypes, and this is an empirical approximation based on four prototypes tested). The robot is also difficult to model due to manufacturing tolerances,

nonlinear material and deformation properties, internal deformation and buckling of its internal partition walls, and hysteresis.

The approach proposed in this work is to use ML with a sparse dataset to learn the robot kinematics in order to estimate the pressures required to reach the target, and combine it with a closed-loop control based on an integral formulation on top. This approach suits our robot since it requires a low number of training datapoints (fewer than 2000, which ensures the robot has sufficient life for operation afterwards), and it is adaptive to the particular behaviour of each robot prototype, including imperfections, nonlinearities, and internal deformations, without requiring a detailed model of it. This work is focused on the control of two segments to illustrate the approach, which is, however, generalizable to three segments. The limiting factor is that training ML for a three-segment robot requires additional datapoints, with the number of data growing exponentially with each segment. Transfer learning will be explored in future work to reduce the amount of experimental data required by relying on simulations.

3. Machine Learning for Direct and Inverse Kinematics

3.1. Pressure–Pose Dataset

We acquired a dataset that consists of $N = 1369$ datapoints of input pressures $p \in \mathbb{R}^6$ and realised poses $x \in \mathbb{R}^4$:

$$\mathcal{D} = \{x_k, p_k\}_{i=1}^{N}. \quad (1)$$

The dataset was acquired using structured pressure inputs at 0, 0.5, 1, 1.5, 1.8 bar, applied to all six chambers using all relevant combinations of pressure inputs. The dataset respected the condition that at least one chamber in each segment must always have zero pressure. This dataset structure was selected given that its maximum size, i.e., N, is limited by the robot life, and a well-structured data representation of all combinations of pressures was needed to cover the full workspace.

The acquisition was conducted using the experimental setup shown in Figure 1. The robot is 12 mm in its outer diameter, and it has a central catheter with a 2 mm working channel [9]. The robot is made of three segments, each of them 26 mm long, and made of Elastosil 4601M. Each segment has three equal chambers and a constant cross section as reported in [9]. Each segment can bend up to 180 degrees and can apply lateral forces up to 2.9 N when tested individually and held at the base [10]. However, in this work, the bending is limited to approximately 90 degrees, and maximum force is approximately 1.5 N due to a limit on the maximum pressure to extend the robot lifespan to at least a few thousand cycles. The supply tubes for distal segments pass through proximal segments.

The robot was hung upside down to eliminate gravity biases and it was pressurised using proportional pressure regulators (Tecno Basic, Hoerbiger, Schongau, Germany) in each chamber. These were controlled using a digital microcontroller (mbed NXP LPC1768, Arm, Cambdridge, UK), which received the pressure commands via a serial interface from a PC. A sensor was placed inside the central tube at the robot tip, and its pose was measured using an electromagnetic tracker (Aurora, NDI, Waterloo, ON, Canada). The system has an RMS of 0.7 mm, quoted by the manufacturer, NDI (Waterloo, ON, Canada). The electromagnetic sensor was secured at the tip of the robot using blu tack, and a calibration procedure was conducted at the beginning of each acquisition to ensure that the origin of the coordinate frame is at the tip of the sensor when the robot is unpressurised and the Y direction corresponds to the axis running along the centre of the robot at rest. A Matlabscript (version R2017b) was used to set the robot pressure inputs and record the robot pose for all the datapoints.

The resulting dataset is structured by three evenly spaced regions that reflect the design with three chambers. The distribution of points in the three regions has similarities but is not equal due to manufacturing tolerances and hysteresis in the robot.

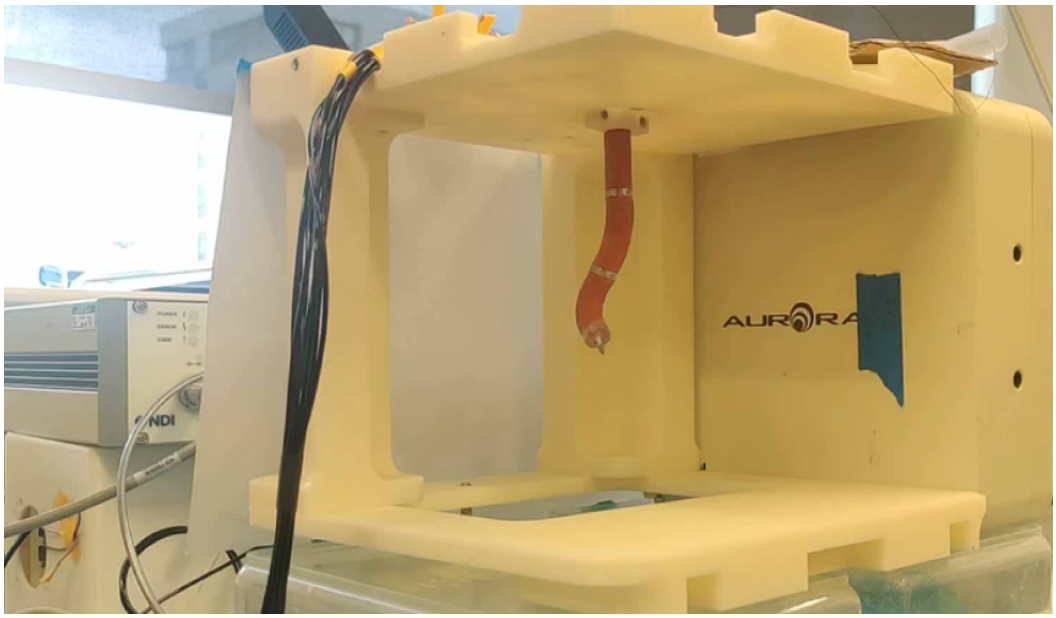


Figure 1. Setup used for dataset acquisition and for testing. The depicted robot, used for experiments, is fabricated with three segments, but only two segments were used in this work to investigate control, which are the middle and distal segments.

3.2. Neural Network Architectures for Regression

We explored a range of neural networks (NN) to learn the direct kinematics (DK) and inverse kinematics (IK) of the robot. We primarily focused on deep neural networks with a fully connected architecture since we have limited a priori knowledge about the structure of the DK and IK mappings.

The potential symmetries associated to the three equal chambers in each segment are obscured by the manufacturing imperfections, internal deformation, and nonlinearities, and thus the resulting DK and IK mappings cannot be assumed to have symmetries. This means that convolutional neural network (CNN) architectures, which could exploit symmetries, are not considered to provide additional benefits for this dataset. The limited number of datapoints also means that residual network architectures are not appropriate since deep networks with a high number of hidden layers in the tens would quickly overfit the data. Residual networks improve the learning of identity mappings. However, given the nonlinearity of the unknown function, we do not expect that residual networks will provide a significant advantage over fully connected deep networks. Thus, we focused on fully connected neural networks, using up to seven layers (five hidden layers in addition to the input and output layers) and from 16 to 1024 neurons per layer.

We explored both small and large numbers of neurons relative to the dataset size to reach the point of overfitting and exceed it. This was because of the sparse nature of our dataset, which meant that some points in the dataset are likely to be unique in representing features of the kinematics mappings. In the NNs, we used a rectified linear unit as an activation function, and a Ridge regulariser with $L2 = 0.01$. In addition, we also added a dropout of 10% in the training to prevent undesired overfitting.

All training was performed using a stochastic adaptive moment estimation (Adam) optimiser 1000 epochs, and the mean squared error as the loss function. We also used a 10% split between training and validation datasets in all the work, unless otherwise specified. The data were shuffled at the beginning of every training, and 10% was reserved for testing.

3.3. Kernel-Based Learning for Function Estimation

In addition to neural networks, we also explored kernel-based learning for the learning of the DK and IK mappings. Kernel-based learning has only a regularisation parameter and typically only a few hyperparameters that need to be tuned, which enables an efficient

learning process. Furthermore, this regularisation parameter enables control over the smoothness of the found mapping, which can enhance the generalisation capabilities of the found model [28].

We first briefly recall the main principles behind kernel-based learning. Consider the dataset $\{z_k, w_k\}_{i=1}^N$, where $w \in \mathbb{R}$ is the set of observed outputs and $z \in \mathcal{Z} \subseteq \mathbb{R}^{n_z}$ the set of (observed) inputs. Furthermore, consider a Hilbert space $\mathcal{H}$ characterised by a symmetric positive semidefinite kernel function K. The space $\mathcal{H}$ defined by the kernel function K and its hyperparameters has desirable properties such as completeness and a reproducing property; see [28] for details. Kernel-based learning searches for the function $g : \mathbb{R}^{n_z} \to \mathbb{R}$ inside the space $\mathcal{H}$ that minimises the following cost functional:

$$g = \min_{\hat{g} \in \mathcal{H}} \left[\frac{\gamma}{2N} \sum_{i=1}^{N} (w_k - \hat{g}(z_k))^2 + \frac{1}{2} \|\hat{g}\|_{\mathcal{H}}^2 \right]. \tag{2}$$

In (2), the first term penalises mismatch in the data fit and the second term regularises on the function complexity or smoothness. The regularisation constant $\gamma \geq 0$ controls the trade-off between these two objectives.

The minimiser g of (2) has a closed-form solution and is given by

$$g(\cdot) = \sum_{i=1}^{N} \alpha_i K_{z_i}(\cdot) \tag{3}$$

with $K_{z_i}(\cdot) := K(\cdot, z_i)$ and $\alpha = \begin{bmatrix} \alpha_1 & \dots & \alpha_N \end{bmatrix} \in \mathbb{R}^N$ being given by

$$\alpha = \left(\frac{1}{N} \mathcal{K}_{ZZ} + \gamma^{-1} I_N \right)^{-1} \frac{1}{N} W, \tag{4}$$

where $W = \begin{bmatrix} w_1 & \dots & w_N \end{bmatrix}^\top$, $I_N \in \mathbb{R}^{N \times N}$ is an identity matrix, and $\mathcal{K}_{ZZ} \in \mathbb{R}^{N \times N}$ is the Gram matrix defined as $\mathcal{K}_{ZZ}(i, j) := K(z_i, z_j)$. If the output space w is multidimensional, i.e., $w \in \mathbb{R}^{n_w}$, where $n_w > 1$, we learn n_w individual functions and concatenate them. The function g in (3) is called the *kernel-based* model.

The quality of the data fit depends on the kernel, the kernel hyperparameters, and the regularisation parameter γ. In the literature, a wide variety of kernels are proposed, among others, linear, polynomial, rational, spline, and wavelet kernel functions; see [28]. Their hyperparameters and the regularisation parameter γ can be tuned in various ways, e.g., by maximizing the so-called log marginal likelihood function; see [32] for details. In this study, we focus on the squared-exponential (SE) kernel with automatic relevance detection, which is defined as follows:

$$K(w_1, w_2) := \sigma_0^2 \exp \left[-\frac{1}{2} \sum_{i=1}^{n_z} \frac{(w_1(i) - w_2(i))^2}{\sigma_i^2} \right], \tag{5}$$

where $\sigma_0, \dots \sigma_{n_z}$ are hyperparameters that need to be tuned. This kernel has the universal approximation property [28]. The regularisation and hyperparameters are tuned using the GPML toolbox [33], which maximises the log marginal likelihood function.

In the DK problem, the mapping $g : \mathbb{R}^6 \to \mathbb{R}^4$ maps the pressure $p \in \mathbb{R}^6$ to the pose $x \in \mathbb{R}^4$ in the dataset $\mathcal{D}$ in (1). Thus, in this problem, $z = p, n_z = 6, w = x$, and $n_w = 4$. In the IK problem, the mapping $g : \mathbb{R}^4 \to \mathbb{R}^6$ maps the pose x to the pressure p. Therefore, $z = x, n_z = 4, w = p$, and $n_w = 6$.

3.4. Machine Learning for Direct Kinematics

We explored a range of fully connected NNs with a six-element input layer, a number of hidden layers that ranged from one to three, and a four-element output layer. We used 2^n neurons per layer, with a range of $n = 4$ to $n = 9$, and the regularisers described above.

Furthermore, we applied kernel-based learning as described in Section 3.3 using the SE kernel in (5). The training of both the NN models as well as the kernel-based model was performed on the same split in the dataset.

We found that the best NN model for the DK is composed of an input layer, three hidden layers with 256, 128, 64 neurons, respectively, and an output layer, with an $L2 = 0.01$ regulariser applied to all weights in the network. The results of the evaluation of the DK NN model on an unseen testing dataset are shown in Figure 2 and are represented by the blue circles. In the same figure, we also depict the results of the evaluation of the kernel-based DK model, represented by the red crosses, on the same unseen testing data. The testing data are independent of the training data, but both of them are subsets of the original dataset acquired, and are therefore obtained under similar operation conditions. If the operation conditions of the robot change with respect to those of the training dataset, for example, due to the addition of a constant load, we expect the performance of both the NN model and kernel-based model to degrade in a similar manner since they both make similar predictions on the kinematics and therefore they would generate similar errors.

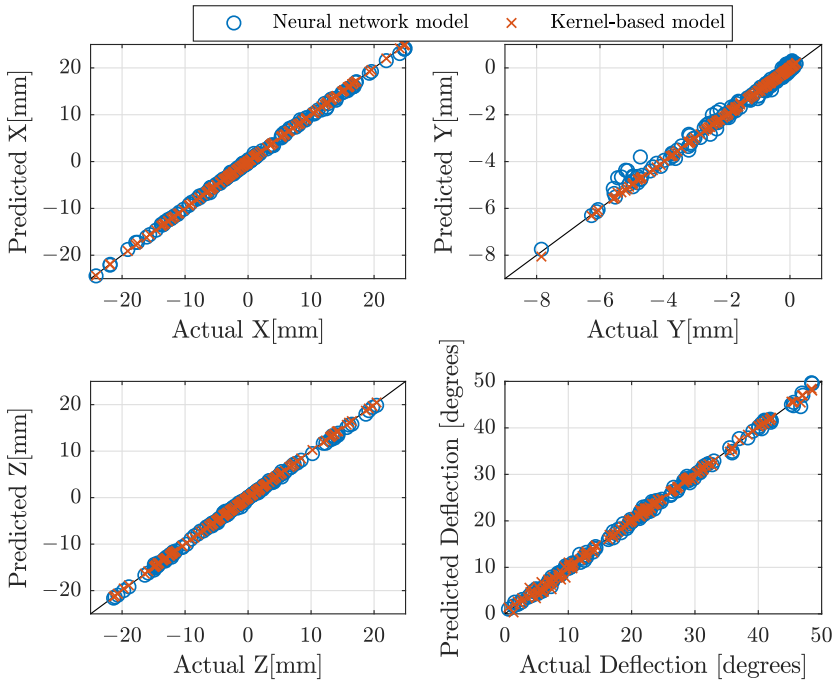


Figure 2. Neural network (blue circles) and kernel-based (red crosses) models for direct kinematics prediction on unseen testing dataset. The results show the actual pose versus the predicted pose. The testing dataset is independent from the training dataset, but both datasets are acquired under similar operation conditions as part of the original dataset.

To quantify the fitting, we define the coefficient of determination as

$$R^2 := \left(1 - \frac{\sum_{i=1}^{N} |x_i - \xi_i|^2}{\sum_{i=1}^{N} \left| x_i - \frac{1}{N}\sum_{i=1}^{N} x_i \right|^2}\right) \cdot 100\%, \tag{6}$$

where x_i denotes each actual value of the pose variables concatenated for the i-th datapoint, N is the total number of points in the testing set concatenating the four variables, ξ_i are the predicted pose variables for the i-th datapoint, and $|x|^2 := x^\top x$ for $x \in \mathbb{R}^4$. The R^2 value for the neural network model is $R^2 = 99.8\%$ and for the kernel-based model is $R^2 = 99.9\%$. Therefore, both machine learning methods successfully learned the DK mapping for all four output pose variables.

3.5. Machine Learning for Inverse Kinematics

Learning the IK mapping is a more complex problem. The IK is a mapping from $\mathbb{R}^4$ to $\mathbb{R}^6$ with the conditions described in Section 2. In particular, the machine learning models for IK need to learn that at least one chamber in each segment must have approximately zero pressure for each given pose. We hypothesised that the machine learning models should be capable of learning this given that each region of the dataset is associated with zero pressures in specific inputs, which is the result of the geometric bending of the robot to reach each region. Thus, the IK machine learning models would map each region in $\mathbb{R}^4$ to a hyperplane in $\mathbb{R}^6$ that corresponds to zeros in specific pressure inputs, if trained appropriately.

A potential issue arises at the boundaries between regions that have zero pressure in different inputs. However, knowledge of the behaviour of the robot, described in Section 2, indicates that boundaries are not expected to be a problem. For a given segment, the transition from bending in a direction that corresponds to zero pressure in one chamber to another direction that corresponds to zero pressure in the adjacent chamber is to pass through a configuration where both chambers have zero pressure simultaneously. Thus, a continuous transition between regions is expected, which should be reflected in the data and thus can be learned.

Given these considerations, we explored a range of fully connected NN models to learn the IK. All NN models explored consisted of an input layer with four elements, a set of hidden layers that ranged from two to five, and an output layer with the six predicted pressures. We used 2^n neurons per layer with a range of $n = 4$ to $n = 10$, and the regularisers described above. Furthermore, we also learned a kernel-based IK model using the approach in Section 3.3.

The best IK NN model was found to consist of an input layer, three hidden layers with 64 neurons each, and an output layer, with a $L2 = 0.01$ regulariser. The corresponding prediction of this model on unseen testing data is shown in Figure 3 (blue circles). This figure also depicts the predictions of the kernel-based IK model (red crosses) on the same testing dataset. Analogous to (6), the coefficient of determination R^2 quantifies the performance of the IK models. When evaluating the unseen testing dataset, the IK NN model achieved $R^2 = 79.8\%$, whereas the kernel-based IK model achieved $R^2 = 78.0\%$.

These results confirm that the explored machine learning models can learn the IK mapping and generally predict which pressure inputs should have approximately zero pressure. However, these models do not exactly predict zero pressure in all occasions, and in some cases, these models predict a negative pressure instead of zero. This is due to the fact that the data-driven models estimate the inverse kinematics function from a limited-sized and sparse dataset in an unconstrained fashion. In the regions where negative pressures are predicted, the inverse kinematics mapping is not accurate and provides, on some occasions, negative pressure values. By saturating negative pressures to zero and considering that the robot bending is nearly negligible at pressures below 0.5 bar, the IK NN model correctly predicts nearly zero pressure (below 0.5 bar) in 92.6% of all cases in the test dataset, whereas this score is 91.9% for the kernel-based IK model.

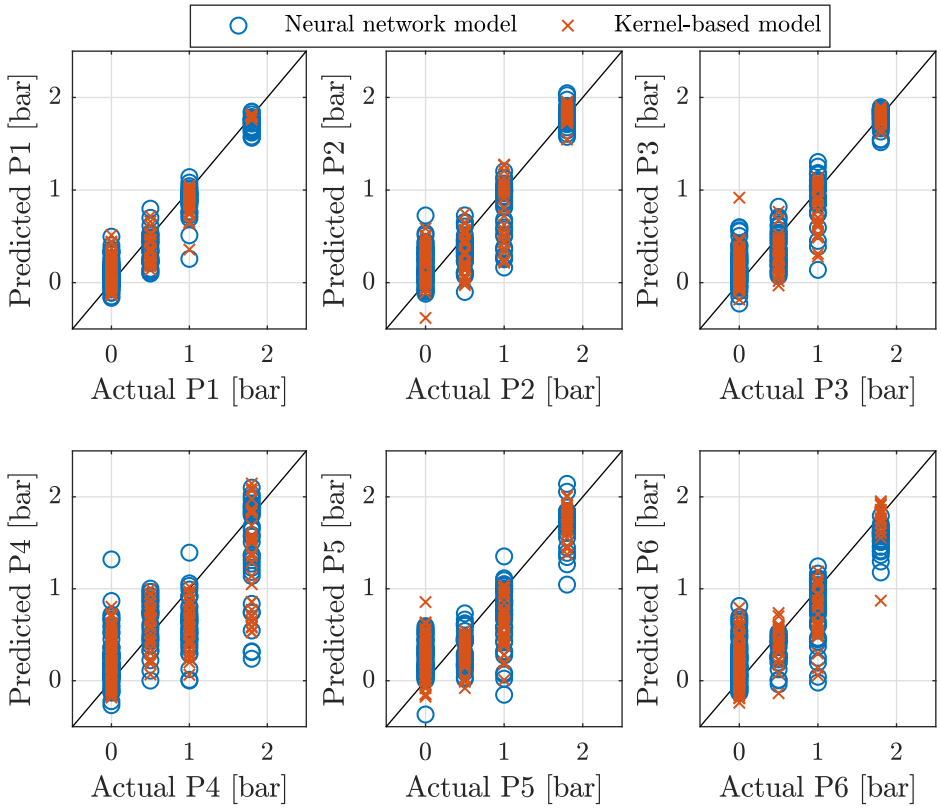


Figure 3. Neural network (blue circles) and kernel-based (red crosses) models for inverse kinematics prediction on unseen testing dataset. The results show the actual pressures versus the predicted pressures.

3.6. Inverse-Direct Machine Learning Models

Machine learning for the IK mappings shows a significantly lower coefficient of determination than machine learning for the DK mapping. We developed a k-nearest neighbours (k-NN) algorithm to investigate the potential source. For a given pose, the algorithm finds the three nearest points in the training dataset and the corresponding pressures. This reveals that in over 25% of instances, for a given pose, there are two or more points in the training data within 1 mm Euclidean distance. However, the pressures used to reach those points present significant differences, both in terms of values, which can vary by over 1 bar in the chambers, and in terms of the inputs that are set to zero pressure. This indicates that pressure inputs that are significantly different can lead to very similar given poses. Consequently, despite the spread in the predicted IK pressures, the values provided by the machine learning models are expected to lead the robot near the desired pose.

To evaluate the spread in the predictions of the learned IK models, we combine the IK models together with the DK model to create an IK-DK model. Thus, for a given pose, we run the IK model to find the pressures required to reach the desired pose, and we then input the pressures to the DK model to evaluate the resulting pose. If the IK models and the DK models are accurate, then the IK-DK model should be an identity mapping.

The IK and DK models are trained independently. Therefore, the IK-DK will return a point close to the original pose if both the IK and DK models are accurate representations of the true IK and DK mappings, but it will also highlight errors in each model if these individual mappings are not learned accurately.

The results of the IK-DK models are shown in Figure 4 for the NN models (blue circles) and the kernel-based models (red crosses). For the NN models, the combined coefficient of determination is $R^2 = 98.1\%$, whereas this score is 97.3% for the kernel-based IK-DK model. These results indicates that the IK NN model and kernel-based IK model are a good approximation of the true IK mapping. Furthermore, we can conclude that the deviations shown in Figure 3 in terms of pressures, in fact, do not affect the resulting pose significantly

since they converge to a resulting pose that is close to the desired one. It should be noted, however, that the coefficients of determination of the IK-DK are slightly lower than those of the DK in Section 3.4 ($R^2 = 99.9\%$ and $R^2 = 99.8\%$, respectively). This is due to the fact that, even though poses of points nearby converge, a slight error is introduced by the IK since the significant differences in pressures to reach two nearby points mean that the IK mapping has to fit abrupt changes that create a degree of error. This results in a slight drop in the coefficients of determination for the IK-DK.

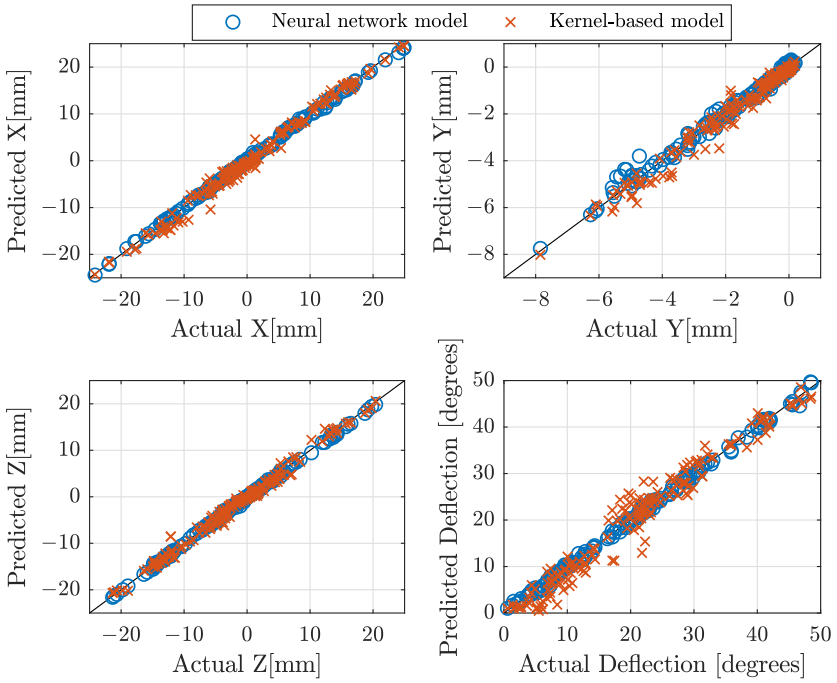


Figure 4. Neural network (blue circles) and kernel-based (red crosses) models for the composed mapping of direct kinematics prediction evaluated on the inverse kinematics predictions on unseen testing dataset. The results show the desired pose versus the predicted pose.

3.7. Experimental Validation of the IK Neural Network Model

The DK and IK kernel-based models show similar performance to the NN models. Therefore, only the IK NN model selected in Section 3.5 was validated experimentally. The testing dataset poses were input into the IK NN model, which predicted the pressures to reach them. The pressures were then applied to the robot using the same setup used for data acquisition. The robot pose was measured and recorded using the EM sensor.

The results of the measured poses reached by the robot versus the desired poses are shown in Figure 5. The results validate the IK NN model, but they also show that there is a degree of error in the experimental pose reached that is greater than in the IK-DK evaluated numerically in Section 3.6. The experimental coefficient of determination is 94.87%. We attribute this to the fact that the robot exhibits a memory effect, which is the combination of hysteresis, and the fact that deformation in the chambers does not flow easily from one pose to the next due to internal friction and buckling. The training data and the machine learning models capture this memory effect. When performing IK-DK numerically on the testing dataset, which is a subset of the full dataset gathered at once, both the DK model and the IK model contain a similar effect of memory in their mappings that they learned from the same training dataset. However, when experimentally testing the IK NN model, the robot is commanded to reach points in a different order than in the training dataset, and that means that the memory effect creates deviations that are different from those captured by the NN models. Thus, the deviations due to memory effect increase the errors when comparing the experimental validation of the IK NN model with $R^2 = 94.87\%$ and the

numerical IK-DK machine learning models with $R^2 = 98.1\%$ and 97.3% respectively. It should be noted that the scaling in the y-axis plot in Figure 5 is different from the other axes since the total range of motion is lower in this axis. This makes the errors more apparent in the y-axis, but the absolute errors are similar to those in the other axes.

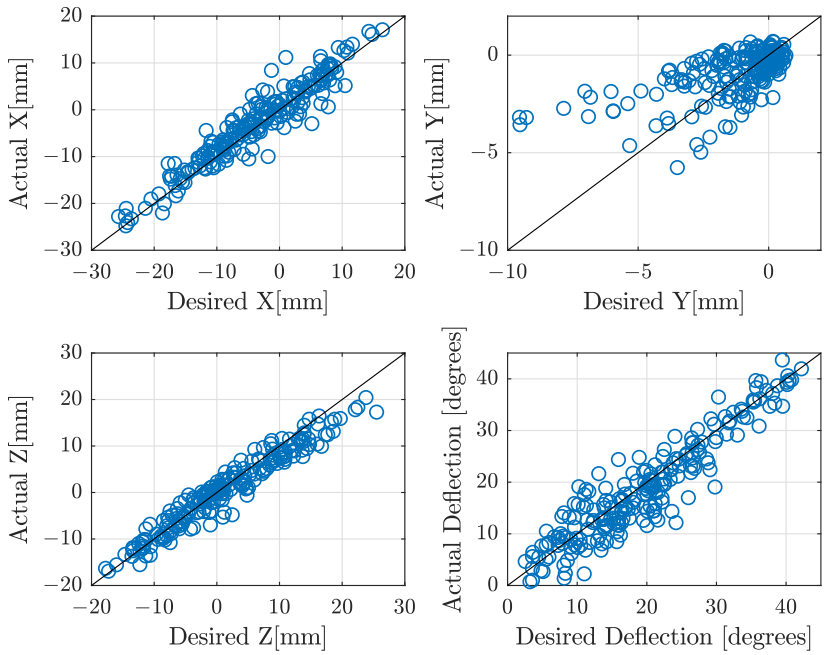


Figure 5. Experimental validation of the IK NN model when tested on the robot.

4. Hybrid Closed-Loop Control

The results in the previous section show that the IK NN is capable of approximating the robot to a pose within a few millimetres of the desired setpoint. A closed-loop formulation based on an integral is proposed in this section to compensate for the remaining error and reach the desired setpoint.

A key question is how to map the error in the four pose outputs to the six pressure inputs. In the following subsections, we explore two approaches based on either an experimental or synthetic Jacobian.

In both approaches, a hybrid closed-loop control is proposed based on the IK NN model and an integral controller that is formulated for setpoint regulation of the robot using the Jacobian pseudo-inverse. The proposed control is

$$U = IK(x_t) + k_i \mathbf{J}^{-1}(x_c) \int e dt + \mathbf{J}^{-1}(x_c) k_p e \tag{7}$$

where U is the control input consisting of six pressures (p_1, p_2, p_3, p_4, p_5, p_6), e is the tip pose error (in x, y, z, θ), k_i is the integral parameter, k_p is the proportional term parameter, and $\mathbf{J}^{-1}(x_c)$ denotes the Jacobian pseudo-inverse using the (4×6) matrix terms corresponding to the four tip pose variables x, y, z, θ differentiated with respect to all six pressures, and evaluated at the current robot pose x_c. The prediction by the IK NN model is denoted by $IK(x_t)$ and provides approximate pressures to reach the desired pose. $IK(x_t)$ plays its main role at the beginning of setpoint regulation as a first estimate of the pressures, after which the integral term becomes increasingly prominent to compensate for the remaining error.

The control is suitable as long as there is no saturation, and the dataset is sufficiently dense. We use unwinding of the integral to mitigate the effect of saturation [34].

The various approaches to estimate $\mathbf{J}^{-1}$ are described in the following subsections.

4.1. Experimental Jacobian Control

The first method explored to estimate the Jacobian is based on an experimental approximation of the Jacobian. This is extracted from the full dataset gathered experimentally, and thus is intended to capture the behaviour of the specific prototype.

To estimate the Jacobian at a given point, first a k-Nearest Neighbours (kNN) algorithm is used to find the nearest datapoint in the full dataset. Then, the partial derivatives of the six pressures with respect to pose, i.e., $\Delta p_i / \Delta X_i$, are estimated for that point found. To estimate the partial derivatives, we search again the full dataset to find the datapoints corresponding to an increment in each of the pressures p_1 to p_6. Since the dataset is structured, there is always a datapoint corresponding to either an increment of 0.5 bar or 0.8 bar. For datapoints at the edge of the dataset, which are obtained at a 1.8 bar pressure, the partial derivative is estimated based on the datapoints corresponding to a decrement of pressure. The partial derivatives are finally assembled in matrix form and normalised, yielding the desired estimate of the Jacobian inverse $\mathbf{J_e^{-1}}$.

In order to use $\mathbf{J_e^{-1}}$ in (7), it needs to be evaluated near x_c. In this approach, $\mathbf{J}_e^{-1}$ is evaluated at the target x_t to ensure that it remains stable. This assumes that $\mathbf{J}_e^{-1}(x_t)$ is representative of the current Jacobian, and is valid as long as the IK NN brings the robot sufficiently close to the target. Thus, in this approach, the controller (7) is implemented using $\mathbf{J}^{-1}(x_c) = \mathbf{J}_e^{-1}(x_t)$.

The controller (7) with $\mathbf{J}^{-1}(x_c) = \mathbf{J}_e^{-1}(x_t)$ was experimentally tested using the setup described in Section 3.1 but using a Python script in an environment with Tensorflow to implement the IK NN and the closed-loop control. The setpoint was selected to be $x = -4$ mm, $y = -0.5$ mm, $z = 1$ mm, $\theta = 6^\circ$, which is a representative point inside the workspace and not close to saturation. A value of $k_i = 0.2$ was used, and the proportional term was not used, i.e., $k_p = 0$. The value of k_i was selected by experimentally exploring a range of values (from $k_i = 2$ based on previous works [13] to $k_i = 0.1$) and choosing the one that led to better performance. Higher k_i leads to faster response and adaptation to disturbances but it can also result in overshoot and larger oscillations. There is also a risk that the robot reaches saturation during overshoot and oscillations, making higher k_i less desirable. Lower k_i offer slower response times, but they reduce overshoot and oscillations. In this work, a compromise value of $k_i = 0.2$ was selected since it reaches the target in approximately 2 s while avoiding oscillations or overshoot.

The experimental results with controller (7) are shown in Figure 6. As can be seen, the robot tends to the desired pose. There are low oscillations caused by the large value of k_i. This value can be reduced to mitigate oscillations, but this results in a slow response, creating a trade-off between oscillatory behaviour and settling time. Another potential solution to reduce oscillations is to add a dissipative term to the control law (7), such as a derivative term proportional to the velocity of the robot. However, velocity estimation with our experimental setup can have limited precision and lead to local errors. Future work will explore methods such as filtering to reliably add dissipative terms to mitigate oscillations. Overall, the experimental approximation of the Jacobian yields a controller that converges as long as the IK NN provides a good first approximation of the pressures required to reach the target.

A potential drawback of the experimental Jacobian is that it can be inaccurate at poses far away from the target pose x_t (typically a few millimetres). As can be seen in Figure 5, the error of the IK NN can be significant for some points, in which case controller (7) does not converge. This is particularly an issue when the IK NN model initially sets pressures that move the robot to a pose distant from the target. Furthermore, the Jacobian can be inaccurate in regions of the dataset where the Jacobian is distorted due to the robot's memory effect.

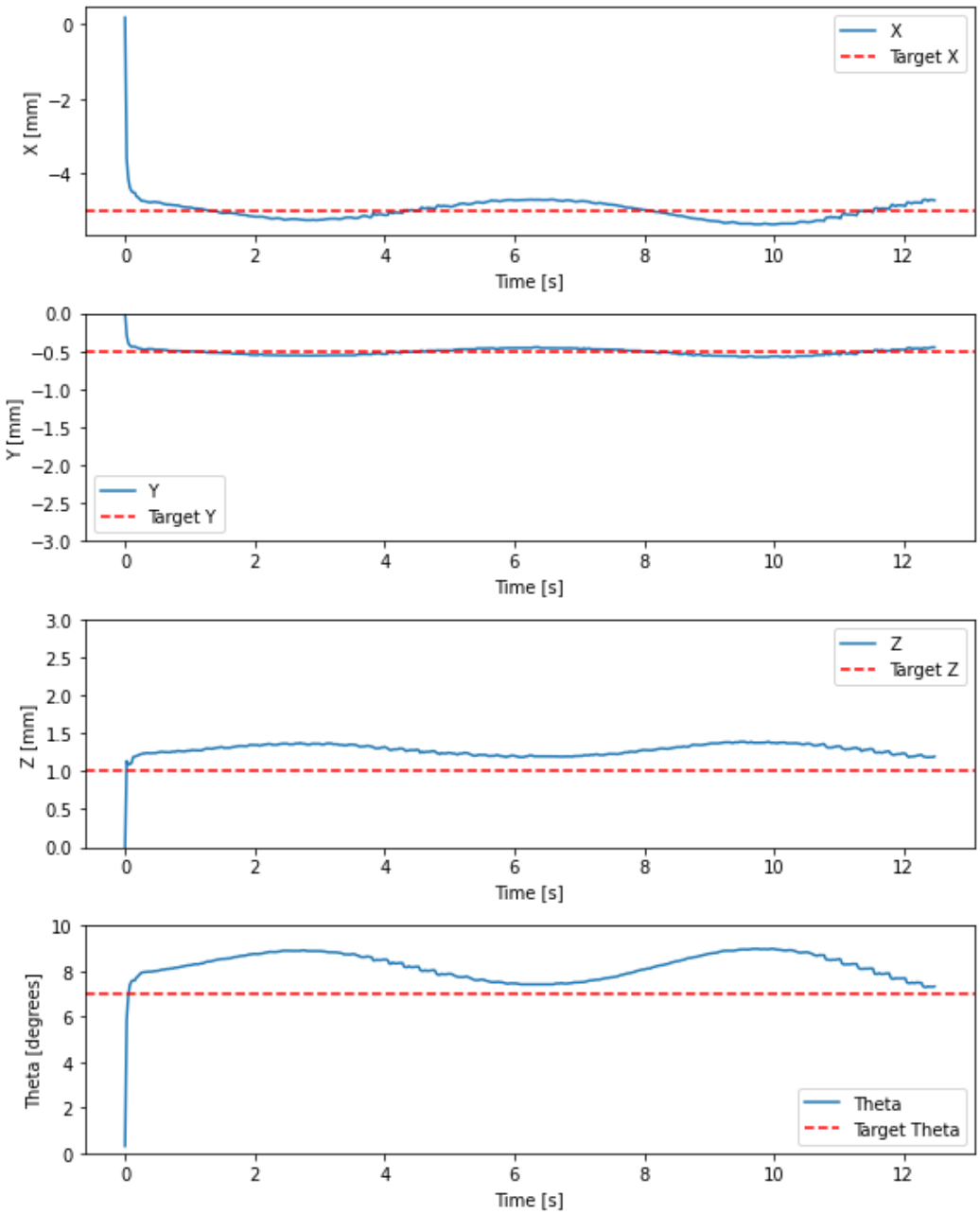


Figure 6. Experimental results of setpoint regulation using a hybrid control combining IK NN and an integral action based on an experimental estimate of the Jacobian.

4.2. Synthetic Jacobian

A synthetic Jacobian is explored in this section to avoid the issues of distorted and abruptly changing Jacobian caused by the experimental estimation in the previous section. The synthetic Jacobian is derived by assuming piecewise constant curvature (PCC) bending of the robot segments. We use a model relating the bending angle and bending plane of each segment to the pressure applied to its chambers, and combine it with robot kinematics of the two segments stacked.

The segment model used is based on [13], and is

$$\begin{aligned} \tan(\phi) &= \frac{\sqrt{3}(p_1 - p_3)}{2p_2 - p_3 - p_1} \\ \theta &= \frac{1}{k}\sqrt{p_1^2 + p_2^2 + p_3^2 - p_1p_3 - p_2p_3 - p_1p_2} \end{aligned} \tag{8}$$

where θ is the segment bending angle, ϕ is the bending plane of the segment, p_1, p_2, p_3 are the pressures applied to the three chambers of the segment, and k is a parameter representing the segment structural stiffness. The same model is used for both robot segments, with $k = 5$ bar/rad.

The robot kinematics are based on [35]. They are used to find the direct kinematics relating the tip pose to the arc parameters of the robot segments as

$$x, y, z, \alpha, \theta, \gamma = f(\phi_1, \theta_1, \phi_2, \theta_2, l_1, l_2) \tag{9}$$

Here x, y, z represent the robot tip position, α, θ, γ are the ZYZ Euler angles of the robot tip orientation, all relative to a reference frame positioned at the robot base with the Z vector co-axial with the robot centreline when not pressurised, ϕ_i, θ_i are the bending plan and angle of segment i, and l_1 and l_2 are parameters representing the lengths of the segments, respectively. In this work, $l_1 = l_2 = 25$ mm. The specific function f is not reproduced here for length reasons, but it can be obtained from [35] using the homogeneous transformation of each segment

$$\mathbf{T}_i = \begin{bmatrix} s_{\phi_i}^2(1 - c_{\theta_i}) + c_{\theta_i} & s_{\phi_i} c_{\phi_i}(c_{\theta_i} - 1) & c_{\phi_i} s_{\theta_i} & \sigma_i s_{\theta_i/2} c_{\phi_i} \\ s_{\phi_i} c_{\phi_i}(c_{\theta_i} - 1) & c_{\phi_i}^2(1 - c_{\theta_i}) + c_{\theta_i} & s_{\phi_i} s_{\theta_i} & \sigma_i s_{\theta_i/2} s_{\phi_i} \\ -c_{\phi_i} s_{\theta_i} & -s_{\phi_i} s_{\theta_i} & c_{\theta_i} & \sigma_i c_{\theta_i/2} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{10}$$

where $\sigma_i = \frac{2 l_i \sin(\theta_i/2)}{\theta_i}$, and s_w and c_w denote the sine and cosine of w, respectively.

The total transformation for the robot is

$$\mathbf{T}_t = \mathbf{T}_1 \mathbf{T}_2 \tag{11}$$

Then, the tip x, y, z can be directly obtained from (11), and the ZYZ Euler tip angles can be obtained, e.g., see [36], as

$$\begin{aligned} \alpha &= \text{atan2}\left(\tfrac{T_{t23}}{\sin\theta}, \tfrac{T_{t13}}{\sin\theta}\right) \\ \theta &= \text{atan2}\left(\sqrt{T_{t_{31}}^2 + T_{t_{32}}^2}, T_{t_{33}}\right) \\ \gamma &= \text{atan2}\left(\tfrac{T_{t32}}{\sin\theta}, -\tfrac{T_{t31}}{\sin\theta}\right) \end{aligned} \tag{12}$$

Finally, substituting the segment model (8) into the kinematics (9) yields

$$x, y, z, \alpha, \theta, \gamma = f(p_1, p_2, p_3, p_4, p_5, p_6, l_1, l_2, k) \tag{13}$$

Differentiating (13), we obtain the robot Jacobian $\mathbf{J}(p_1, p_2, p_3, p_4, p_5, p_6, l_1, l_2, k)$, where each element is

$$J_{ij} = \frac{\partial x_i}{\partial p_j} \tag{14}$$

4.3. Integral Control Using Synthetic Jacobian

The hybrid controller (7) was initially implemented using the synthetic Jacobian and no proportional term ($k_p = 0$). The implementation was equivalent to that in Section 4.1 with the difference being that the Jacobian is synthetic and continuously evaluated at the current robot state x_c. The setpoint was selected to be $x = -5$ mm, $y = -0.5$ mm, $z = 0$ mm, $\theta = 6°$, which is a representative point inside the workspace not close to saturation. A value of $k_i = 0.3$ was used. As in the previous Section 4.1, the value of k_i was selected experimentally using a heuristic process. Larger values of k_i improve performance but can lead to overshoot and oscillations, with the risk of reaching saturation, whereas smaller values of k_i yield a smoother response but reduce performance in terms of response time. A compromise value of $k_i = 0.3$ was used, which is higher than in Section 4.1 since the control using the synthetic Jacobian showed lower oscillations.

The results are shown in Figure 7. The robot pose tends to the setpoint. The integral action creates minor fluctuations in the robot, and the robot does not exactly reach the target, both of which are due to imperfections in the synthetic kinematics model. Nonetheless, the robot is within 0.5 mm of the target. It should be noted that the position error of the electromagnetic tracker used has a standard deviation of 0.7 mm. The error is thus below one standard deviation of the equipment error.

The hybrid control using a synthetic Jacobian performs well as long as there is no saturation as illustrated in Figure 7.

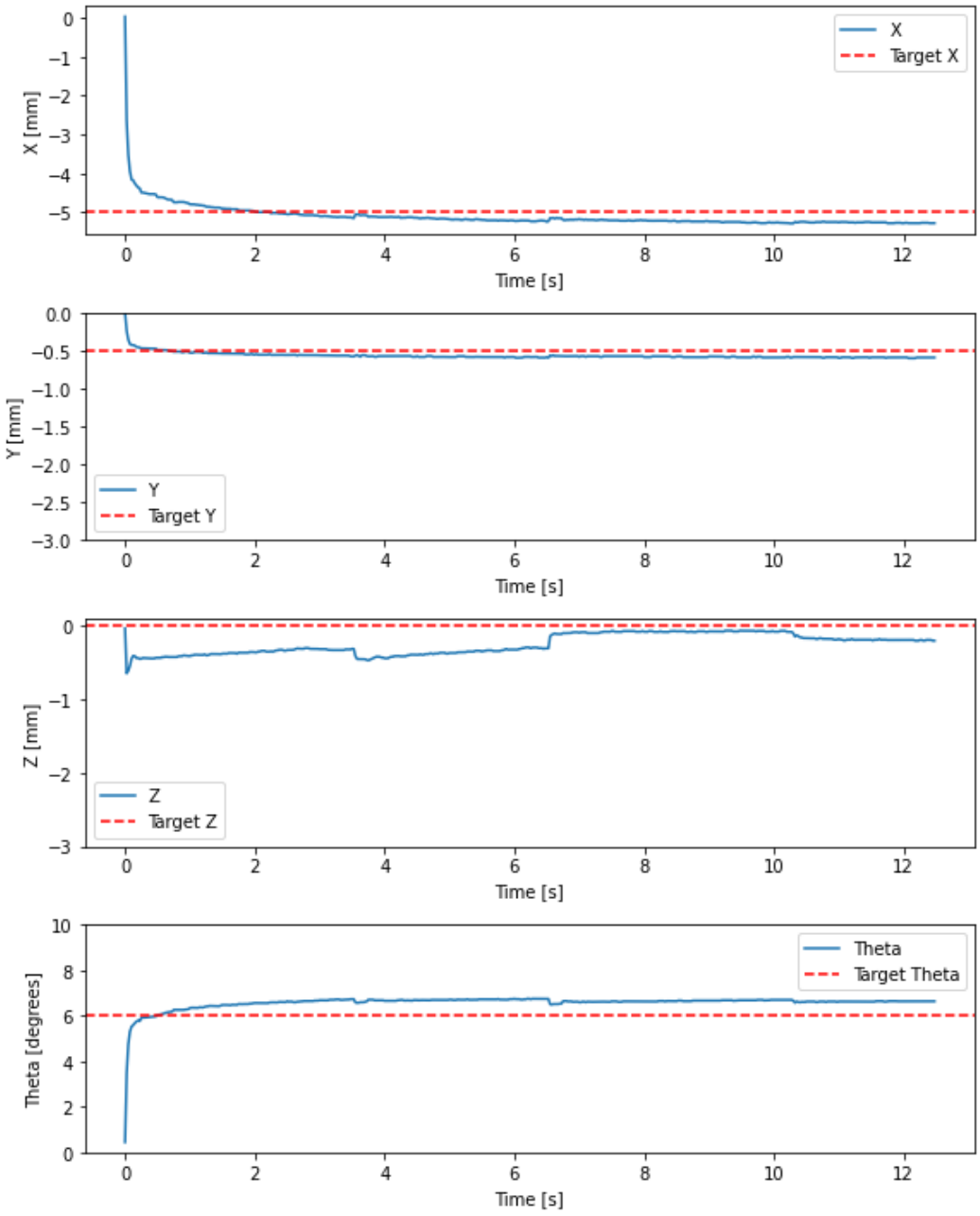


Figure 7. Experimental results of setpoint regulation using a hybrid control combining IK NN and an integral action based on a synthetic Jacobian.

4.4. Proportional Integral Control Using Synthetic Jacobian

The addition of the proportional term in the hybrid control is explored to mitigate the issues described in the previous paragraph. The proportional term is intended to improve the convergence speed and help prevent the controller from getting stuck when reaching transient saturation states. It should be noted that there are two types of saturation: transient and permanent. Transient saturation occurs during oscillations or transient movements, where the robot unintendedly crosses into a configuration with saturation in the pressure inputs. In these cases, the integral term stops updating, and the proportional term can help the control recover. The aim for the proportional term is to provide a contribution to the control that is continuously updated in a way that is proportional to the error while the integral term is not updating, bringing the robot out of transient saturation. Once saturation is no longer present, the integral term can be updated again.

The controller (7) was implemented in an equivalent manner as in the previous subsection. The setpoint was selected to be $x = -3$ mm, $y = -0.3$ mm, $z = 2$ mm, $\theta = 6°$, which is also a point near saturation. The controller gains were selected as $k_i = 0.3$ and $k_p = 0.02$. The value of k_i was maintained from the previous subsection given the similarities in the control law. As in Section 4.3, k_p was selected experimentally, exploring a range values that kept the ratio f k_p/k_i similar to our previous work [13]. Larger values of k_p lead to faster response but also yield oscillations and overshoot. These can lead the robot to a configuration with saturation, where it cannot recover. As a compromise, $k_p = 0.02$ was selected.

The results are shown in Figure 8. As can be seen, the robot tends to the desired setpoint. The robot displays a trade-off in the four tracked variables, which is due to model inaccuracies, which means that even though the robot has four inputs, it cannot exactly converge in all four outputs. Nonetheless, the robot reaches a submillimetre error. In addition, the robot is capable of avoiding a region near saturation at approximately 1.8 s, and recovers to tend to the setpoint.

Permanent saturation is a more complex problem that is not addressed in this work. It occurs when the robot reaches a steady configuration with saturation, e.g., because to reach the target, the robot needs to pass through an intermediate configuration where saturation occurs for an extended period of time. The control methods proposed in this work fail in those cases, leaving the robot stuck. Motion planning can be used to avoid regions with permanent saturation, and this will be explored in future work.

4.5. Hybrid Control Discussion

Overall, we found the best hybrid control to consist of the IK NN for a first approximation of the required pressures combined with a PI control using a synthetic Jacobian to converge to the desired setpoint. This control performs well as long as there is no saturation.

The robustness of this control, as well as that of the control formulation using an experimental Jacobian, depends on the operation conditions. The control formulation using the synthetic Jacobian is particularly vulnerable to model inaccuracies such as those caused by changes in robot behaviour due to significant manufacturing tolerances, as well as hysteresis and nonlinearities. If inaccuracies are very significant, which can occur in extreme cases where manufacturing variations are coupled with robot nonlinearities, the control can diverge because the Jacobian based on the model is not representative of the robot behaviour. Even in cases where inaccuracies are smaller, they can reduce performance since the synthetic model that underpins the control will not be accurate. As such, if a high-density dataset is available, the hybrid control based on an experimental Jacobian estimate (7) can perform better.

The experimental Jacobian-based control, however, has significant drawbacks as noted in Section 4.1. The Jacobian can be distorted, and it changes abruptly between points, particularly when datapoints are sparse, which means that it needs to be evaluated at the target pose, as implemented in this work. The consequence is that, if the robot pose moves far from the target, the Jacobian can cease to be representative of robot behaviour

and the control can fail. Thus, if the dataset is sparse, as in the case of our robot given its limited lifespan, the synthetic Jacobian formulation is more generally suitable. The synthetic Jacobian provides satisfactory experimental results.

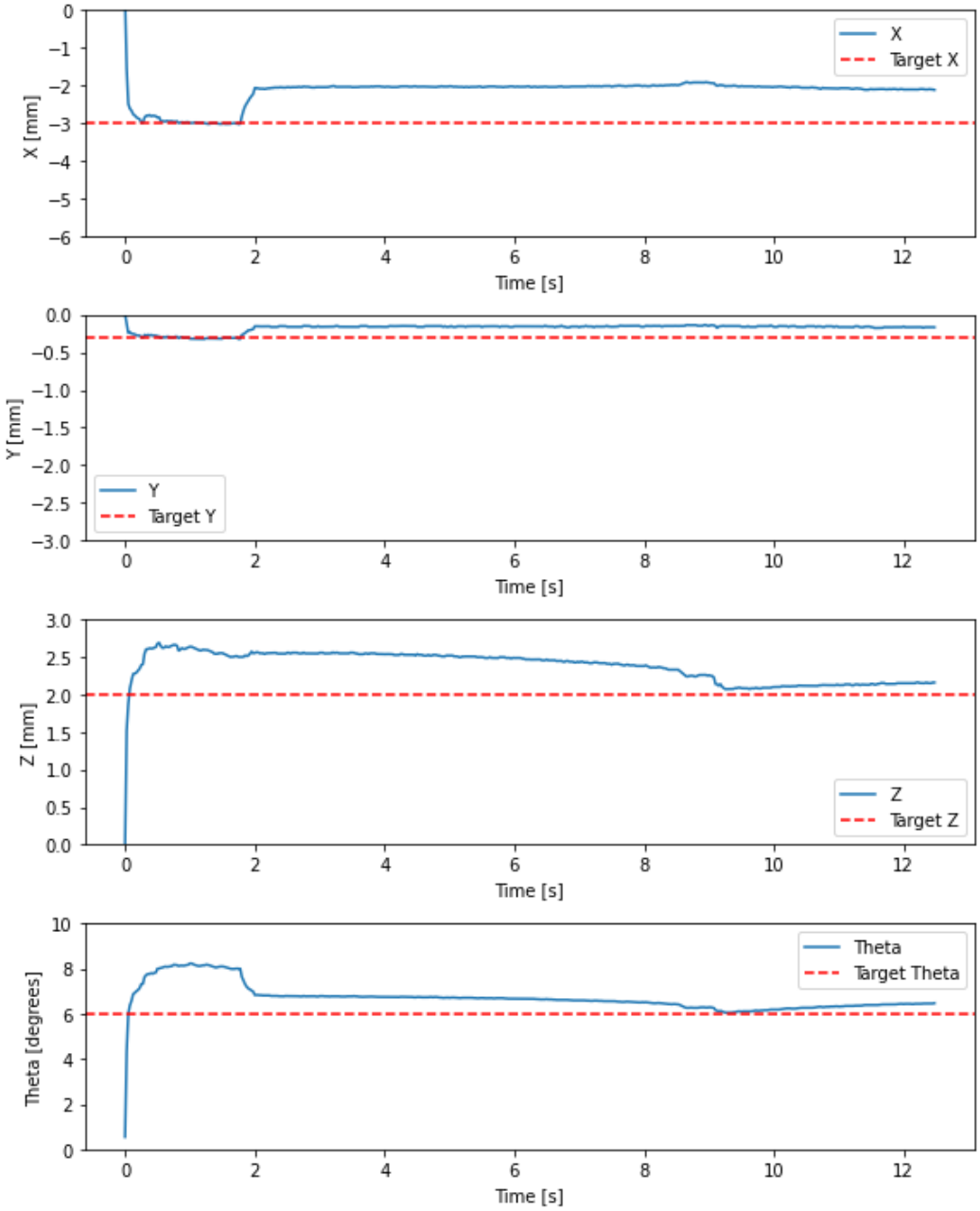


Figure 8. Experimental results of setpoint regulation using a hybrid control combining IK NN and a proportional–integral action based on a synthetic Jacobian.

The combination of both experimental and synthetic Jacobian-based control can be advantageous by retaining the advantages of both. This could be achieved using transfer learning to merge the experimental data of robot behaviour with the synthetic model. However, this requires further research, and it will be addressed in future work.

Both synthetic and experimental Jacobian formulations offer some robustness to unforeseen external disturbances, e.g., constant external loads. The integral part of the control is designed to compensate for the effect of disturbances, and the proportional yields faster adjustments. However, both formulations are vulnerable to disturbances that take the robot to a configuration where the Jacobian is no longer an accurate representation of the

robot behaviour, e.g., in case the robot bends due to an interaction with a hard constraint in the environment such as a wall. In those cases, the derivatives of pose with respect to pressures in the Jacobian do not match reality. In general, the experimental Jacobian is expected to be less robust to those unforeseen disturbances, given that it interpolates between points in the training dataset. If the robot reaches a configuration outside the training dataset, the experimental Jacobian is prone to failing. The synthetic Jacobian instead can maintain a better representation of the robot behaviour even outside the training dataset since it relies on the analytical model for extrapolation. Even so, if the robot is bent to a great extent due to interaction with a hard constraint, the evaluation of the Jacobian based on the current configuration relying on the model may not be representative of the robot behaviour in practice, leading to large errors. The robustness of both control formulations to changes in robot dynamics, e.g., those caused by a change in robot orientation relative to gravity, is also expected to differ if the changes in dynamics lead to a configuration where the Jacobian is less representative. For example, if the robot experiences a significant change in bending angle due to gravity, neither of the Jacobian estimates may be accurate, and the control may yield large errors. As in the previous case, the experimental Jacobian is expected to be less robust in robot configurations outside the training dataset, while the synthetic Jacobian is expected to extrapolate better, provided that the changes in robot dynamics are relatively small.

The hybrid control with the synthetic Jacobian proposed in this work is an improvement with respect to the control in [30]. The control in this work can operate in the entire workspace rather than only in one quadrant. Moreover, this work is applicable to robots with six inputs and four pose outputs, while [30] is limited to four inputs and three outputs. In addition, the use of the Jacobian in this work is an advancement with respect to only using the sign in the relation between inputs and outputs, which leads to better performance. The hybrid control is also an advancement with respect to the hybrid control in [26] since it applies to a two-segment robot. The hybrid control is also advantageous relative to [13] since it works in the full workspace of a two-segment robot, and the NN captures the behaviour of each prototype. The disadvantage relative to [13] is that the hybrid control requires over 1000 datapoints for training, it is slower to converge, and it is less able to compensate for disturbances.

Comparing the present hybrid control with ML and RL solutions in the literature, the present control requires significantly less training data. Leading works for soft manipulators such as [23] for RL and [20] for ML used in the order of 10,000 or more datapoints, whereas in this work, we use 1369. The absolute errors in this work are also lower than in [23], and the relative errors are comparable or lower. However, it should be noted that this present work only focuses on setpoint regulation, whereas [23] addressed trajectory tracking. The errors in [20,24] are lower than in this paper, but it should be noted that those works were conducted on tendon-driven soft robots, which are typically simpler to model, more repeatable, and with lower nonlinearities and hysteresis. The RL and ML approaches in [20,24] have not yet been applied to the soft robot used in this paper, as they require a robot with a significantly longer lifespan.

5. Conclusions

The control of a soft robotic manipulator was explored in this work using a hybrid approach that combines ML to learn the robot kinematics and closed-loop control to compensate for errors. Various ML approaches were explored to solve the DK and IK problems for a manipulator with six pressure inputs and four robot pose outputs with conditions on the combinations of pressure inputs. The most suitable DK NN and IK NN were selected, and this was validated experimentally. A hybrid control formulation based on an integral action was proposed, and two main approaches were explored to map the pressure inputs to the pose output errors using the robot Jacobian. A Jacobian estimate built using an experimental dataset was first developed, and the corresponding control was tested. This showed correct performance as long as the IK NN bring the robot close to

the desired setpoint, but it showed the Jacobian estimate to be unreliable when deviations from the setpoint are significant and when the Jacobain is distorted, typically near the workspace edge. A second approach using a synthetic Jacobian developed assuming piecewise constant curvature bending and using a robot model was then investigated. Both integral control and PI control were built based on the synthetic Jacobian. This was experimentally tested to show correct performance provided that there is no saturation. The tests, however, highlighted the need for an improved control formulation when the control saturates, and when the robot behaviour deviates from the model. Future work will investigate improved models combining the theoretical model with experimental data through physics-informed and transfer learning approaches.

Author Contributions: Conceptualisation, A.G.-C., F.S., V.F., E.F.; methodology, A.G.-C., F.S., V.F.; software, A.G.-C., F.S., V.F.; validation, A.G.-C., F.S., V.F.; formal analysis, A.G.-C., F.S.; investigation, A.G.-C., F.S., V.F.; resources, A.G.-C., F.S., V.F.; data curation, A.G.-C., F.S.; writing—original draft preparation, A.G.-C., F.S.; writing—review and editing, V.F., E.F.; visualisation, A.G.-C., F.S., V.F.; supervision, E.F.; project administration, A.G.-C., F.S.; funding acquisition, E.F. All authors have read and agreed to the published version of the manuscript.

Funding: This work was in part supported by a collaboration with the Multi-scale Medical Robotics Centre, Chinese University of Hong Kong. The funder was not involved in the study design, collection, analysis, interpretation of data, the writing of this article or the decision to submit it for publication. Fahim Shakib has been partially funded by the EPSRC grant Model Reduction from Data, Grant No. EP/W005557.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Suzumori, K.; Iikura, S.; Tanaka, H. Development of flexible microactuator and its applications to robotic mechanisms. In Proceedings of the 1991 IEEE International Conference on Robotics and Automation, Sacramento, CA, USA, 9–11 April 1991; IEEE Computer Society: Washington, DC, USA, 1991; pp. 1622–1623.
2. Suzumori, K.; Iikura, S.; Tanaka, H. Applying a flexible microactuator to robotic mechanisms. *IEEE Control Syst. Mag.* **1992**, *12*, 21–27.
3. Cianchetti, M.; Nanayakkara, T.; Ranzani, T.; Gerboni, G.; Althoefer, K.; Dasgupta, P.; Menciassi, A. Soft Robotics Technologies to Address Shortcomings in Today's Minimally Invasive Surgery: The STIFF-FLOP Approach. *Soft Robot.* **2014**, *1*, 122–131. [CrossRef]
4. Pagliarani, N.; Arleo, L.; Albini, S.; Cianchetti, M. Variable Stiffness Technologies for Soft Robotics: A Comparative Approach for the STIFF-FLOP Manipulator. *Actuators* **2023**, *12*, 96. [CrossRef]
5. Chen, G.; Pham, M.T.; Maalej, T.; Fourati, H.; Moreau, R.; Sesmat, S. A Biomimetic steering robot for Minimally invasive surgery application. In *Advances in Robot Manipulators*; IN-TECH: London, UK, 2009.
6. Dong, X.; Axinte, D.; Palmer, D.; Cobos, S.H.; Raffles, M.; Rabani, A.; Kell, J. Development of a slender continuum robotic system for on-wing inspection/repair of gas turbine engines. *Robot.-Comput.-Integr. Manuf.* **2017**, *44*, 218–229. [CrossRef]
7. Zhang, J.; Li, Y.; Kan, Z.; Yuan, Q.; Rajabi, H.; Wu, Z.; Peng, H.; Wu, J. A Preprogrammable Continuum Robot Inspired by Elephant Trunk for Dexterous Manipulation. *Soft Robot.* **2023**, *10*, 636–646. [CrossRef] [PubMed]
8. Kolachalama, S.; Lakshmanan, S. Continuum Robots for Manipulation Applications: A Survey. *J. Robot.* **2020**, *2020*, 4187048. [CrossRef]
9. Treratanakulchai, S.; Franco, E.; Garriga-Casanovas, A.; Minghao, H.; Kassanos, P.; y Baena, F.R. Development of a 6 dof soft robotic manipulator with integrated sensing skin. In Proceedings of the 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Kyoto, Japan, 23–27 October 2022; pp. 6944–6951.
10. Garriga-Casanovas, A.; Treratanakulchai, S.; Franco, E.; Zari, E.; Ferrandy, V.; Virdyawan, V.; y Baena, F.R. Optimised Design and Performance Comparison of Soft Robotic Manipulators. In Proceedings of the 2022 7th International Conference on Mechanical Engineering and Robotics Research (ICMERR), Krakow, Poland, 9–11 December 2022; pp. 129–136.

11. Ferrandy, V.; Ferryanto, F.; Sugiharto, A.; Franco, E.; Garriga-Casanovas, A.; Mahyuddin, A.I.; Rodriguez y Baena, F.; Mihradi, S.; Virdyawan, V. Modeling of a two-degree-of-freedom fiber-reinforced soft pneumatic actuator. *Robotica* **2023**, *41*, 3608–3626. [CrossRef]
12. Runge, G.; Wiese, M.; Raatz, A. FEM-based training of artificial neural networks for modular soft robots. In Proceedings of the 2017 IEEE International Conference on Robotics and Biomimetics (ROBIO), Macau, Macao, 5–8 December 2017; pp. 385–392.
13. Franco, E.; Garriga-Casanovas, A. Energy-shaping control of soft continuum manipulators with in-plane disturbances. *Int. J. Robot. Res.* **2021**, *40*, 236–255. [CrossRef]
14. Roshanfar, M.; Dargahi, J.; Hooshiar, A. Cosserat Rod-Based Dynamic Modeling of a Hybrid-Actuated Soft Robot for Robot-Assisted Cardiac Ablation. *Actuators* **2023**, *13*, 8. [CrossRef]
15. García-Samartín, J.F.; Rieker, A.; Barrientos, A. Design, Manufacturing, and Open-Loop Control of a Soft Pneumatic Arm. *Actuators* **2024**, *13*, 36. [CrossRef]
16. Duan, J.; Zhang, K.; Qian, K.; Hao, J.; Zhang, Z.; Shi, C. An Operating Stiffness Controller for the Medical Continuum Robot Based on Impedance Control. *Cyborg Bionic Syst.* **2024**, *5*, 0110. [CrossRef] [PubMed]
17. Zhang, S.; Li, F.; Fu, R.; Li, H.; Zou, S.; Ma, N.; Qu, S.; Li, J. A Versatile Continuum Gripping Robot with a Concealable Gripper. *Cyborg Bionic Syst.* **2023**, *4*, 0003. [CrossRef] [PubMed]
18. Kim, D.; Kim, S.H.; Kim, T.; Kang, B.B.; Lee, M.; Park, W.; Ku, S.; Kim, D.; Kwon, J.; Lee, H.; et al. Review of machine learning methods in soft robotics. *PLoS ONE* **2021**, *16*, e0246102. [CrossRef] [PubMed]
19. Lee, K.H.; Fu, D.K.; Leong, M.C.; Chow, M.; Fu, H.C.; Althoefer, K.; Sze, K.Y.; Yeung, C.K.; Kwok, K.W. Nonparametric online learning control for soft continuum robot: An enabling technique for effective endoscopic navigation. *Soft Robot.* **2017**, *4*, 324–337. [CrossRef] [PubMed]
20. Thuruthel, T.G.; Falotico, E.; Renda, F.; Laschi, C. Learning dynamic models for open loop predictive control of soft robotic manipulators. *Bioinspiration Biomimetics* **2017**, *12*, 066003. [CrossRef]
21. Chin, K.; Hellebrekers, T.; Majidi, C. Machine Learning for Soft Robotic Sensing and Control. *Adv. Intell. Syst.* **2020**, *2*, 1900171. [CrossRef]
22. Hyatt, P.; Wingate, D.; Killpack, M.D. Model-based control of soft actuators using learned non-linear discrete-time models. *Front. Robot. AI* **2019**, *6*, 22. [CrossRef] [PubMed]
23. Centurelli, A.; Arleo, L.; Rizzo, A.; Tolu, S.; Laschi, C.; Falotico, E. Closed-loop Dynamic Control of a Soft Manipulator using Deep Reinforcement Learning. *IEEE Robot. Autom. Lett.* **2022**, *7*, 4741–4748. [CrossRef]
24. Thuruthel, T.G.; Falotico, E.; Renda, F.; Laschi, C. Model-Based Reinforcement Learning for Closed-Loop Dynamic Control of Soft Robotic Manipulators. *IEEE Trans. Robot.* **2019**, *35*, 124–134. [CrossRef]
25. Zhang, H.; Cao, R.; Zilberstein, S.; Wu, F.; Chen, X. Toward Effective Soft Robot Control via Reinforcement Learning. In *Intelligent Robotics and Applications, Proceedings of the 10th International Conference, ICIRA 2017, Wuhan, China, 16–18 August 2017*; Springer International Publishing: Heidelberg, Germany, 2017; pp. 173–184. [CrossRef]
26. Dai, Y.; Deng, Z.; Wang, X.; Yuan, H. A Hybrid Controller for a Soft Pneumatic Manipulator Based on Model Predictive Control and Iterative Learning Control. *Sensors* **2023**, *23*, 1272. [CrossRef]
27. Zhang, J.; Liu, L.; Xiang, P.; Fang, Q.; Nie, X.; Ma, H.; Hu, J.; Xiong, R.; Wang, Y.; Lu, H. AI co-pilot bronchoscope robot. *Nat. Commun.* **2024**, *15*, 241. [CrossRef]
28. Schölkopf, B.; Smola, A.J. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*; MIT Press: Cambridge, MA, USA, 2001.
29. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
30. Treratanakulchai, S.; Franco, E.; y Baena, F.R. Model-free Position Control of a Soft Continuum Manipulator in Cartesian Space. In Proceedings of the 2023 International Conference on Control, Automation and Diagnosis (ICCAD), Rome, Italy, 10–12 May 2023; pp. 1–6.
31. Garriga-Casanovas, A.; Collison, I.; Rodriguez y Baena, F. Toward a common framework for the design of soft robotic manipulators with fluidic actuation. *Soft Robot.* **2018**, *5*, 622–649. [CrossRef]
32. Williams, C.K.; Rasmussen, C.E. *Gaussian Processes for Machine Learning*; MIT Press: Cambridge, MA, USA, 2006; Volume 2.
33. Rasmussen, C.E.; Nickisch, H. Gaussian processes for machine learning (GPML) toolbox. *J. Mach. Learn. Res.* **2010**, *11*, 3011–3015.
34. Astrom, K.J.; Rundqwist, L. Integrator windup and how to avoid it. In Proceedings of the 1989 American Control Conference, Pittsburgh, PA, USA, 21–23 June 1989; pp. 1693–1698.
35. Garriga-Casanovas, A.; Rodriguez y Baena, F. Kinematics of continuum robots with constant curvature bending and extension capabilities. *J. Mech. Robot.* **2019**, *11*, 011010. [CrossRef]
36. Murray, R.M.; Li, Z.; Sastry, S.S. *A Mathematical Introduction to Robotic Manipulation*; CRC Press: Boca Raton, FL, USA, 1994. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.