ECCOMAS Thematic Conference on Multibody Dynamics
July 24 - 28, 2023, Lisbon, Portugal

Modeling and Advanced Control for Designing a Soft Material Robot

M. Grube[1], **T. Bekman**[1], **R. Seifried**[1]

[1] Institute of Mechanics and Ocean Engineering
Hamburg University of Technology
Eissendorfer Strasse 42, 21073 Hamburg, Germany
[malte.grube, robert.seifried]@tuhh.de

ABSTRACT

With the rise of soft robotic applications modeling and control of soft material robots becomes more and more important. Due to their softness and the resulting large elastic deformations conventional modeling and control approaches usually cannot be applied and new solutions are currently being developed. This paper is a contribution to solving this problem. In a first step it is shown that the full dynamic modeling of a tendon-actuated beam-shaped soft robot based on the Cosserat rod theory. In a second step an open-loop kinematic controller for trajectory tracking control is presented. The controller is based on a neural-network approximation of the forward kinematics of the soft robot. Finally, the controller is tested using the simulation model set up in the first step, and then evaluated experimentally.

Keywords: Soft Robotics, Control, Cosserat Rod.

1 INTRODUCTION

Soft material robots are an emerging and fast-growing field of research with potential application in various fields. In contrast to conventional robots, which are usually fabricated out of high-stiffness materials such as steel, soft material robots are mostly fabricated out of soft materials like, silicone or foam with a stiffness of only $10^4 \dots 10^9$ Pa. This usually results in large deformations such that conventional components, designs and control methodologies are not applicable. Therefore, new actuators, sensors, modeling and control concepts are currently developed.

In this paper modeling and trajectory tracking control of a simple tendon-actuated beam-shaped soft robot is presented. The soft robot is simulated with a simulation model based on the Cosserat rod theory. For control an open-loop model-free kinematic controller is used. The forward kinematics are thereby approximated with a small neural network. This saves computational costs and allows to compensate modeling inaccuracies of the physical robot. The controller is first examined in simulations and then in experiments.

1.1 Control approaches for soft robots

There is a large amount of work related to the control of soft material robots. In the following, a short overview is given. A more detailed overview can e.g. be found in [1, 2, 3]. Soft robot control methods can be categorized into two groups: model-based and model-free control. Additionally, the control approaches can be divided into kinematic control, where the dynamics of the soft robot are neglected, and dynamic control, where the dynamics of the soft robot are considered [2]. Thereby, in the soft robotics literature, the terms "static controller" and "kinematic controller" are used interchangeably. Furthermore, control methods can be divided into open-loop and closed-loop control [3]. As soft robots can usually deform continuously they often have a very large number of degrees of freedom and are therefore underactuated. Additionally, they are often redundantly actuated. This makes control much more challenging than for most rigid robots.

In soft robotics kinematic controllers are the most widely used and also usually most simple controllers. As here the dynamics are neglected, only comparatively slow movements are possible and the trajectory tracking control error is usually larger than for dynamic controllers. However, this can be accepted for many soft robotic applications. Model-based kinematic controllers for soft robots mostly rely on the direct inversion of the kinematics [4] or, if this is not possible, differential inverse kinematics [5, 6, 7] are used. Model-free kinematic controllers are especially popular for highly nonlinear and nonuniform systems which are difficult to model. In model-free kinematic control, the mapping of actuation variables to control variables is usually learned by a neural network, see e.g. [8, 9]. For redundant soft robots, the relationship between actuation variables and control variables is not unique. In this case, the current configuration of the soft robot has to be considered for the determination of the control variables to achieve a smooth movement [10, 11].

For faster movements and higher accuracy requirements, kinematic control is often insufficient and dynamic controllers are used instead. Compared to kinematic control, dynamic control is usually much more computationally expensive. A popular control approach is model predictive control in combination with mechanical models [12, 13] and data driven models [14, 15, 16]. But also, PD-controllers [17, 18], sliding mode controllers [19], and LQR-controllers [1], as well as reinforcement learning approaches [20, 21, 22] are used.

2 TRAJECTORY TRACKING CONTROL PROBLEM

In this contribution the trajectory tracking of a simple beam shaped soft robot as shown in fig. 1 is considered in simulation and experiment. Thereby a model-free kinematic control approach is chosen. The soft robot is actuated by three tendons which allows a movement of the robot's tip on a semi-sphere. In the following, the experimental setup and the used simulation model are described.

2.1 Experimental setup

In this contribution a beam-shaped soft robot with a length of $L = 120\,\mathrm{mm}$ and a radius of $r = 20\,\mathrm{mm}$ is considered. The soft robot is shown in fig. 1. The most important geometric and material parameters are listed in tab. 1. The soft robot is made out of silicone of type "HT45". It is actuated by three servos via tendons which are evenly distributed along the circumference of the soft robot. In the following, it is referred to the three tendons as a tendon triple. This allows to directly control the tendon length s_q for all three tendons. Here, $q = 1 \ldots 3$ is the index of the tendon.

As in this contribution kinematic control is considered, which means that the dynamics are neglected, there is a direct relationship between the tendon length s_q and the tendon forces F_q. This will be used later on. Additionally, it can be assumed for the controller design, that the soft robot performs a pure bending movement and elongation can be neglected. From this follows that only two of the three tendon lengths $s_1 \ldots s_3$ can be chosen independently of each other. The third tendon length then results from the kinematics of the soft robot. This has to be considered for the experimental setup. For reference measurements of the tip position, a cube with AprilTags [23] on its sides is mounted at the tip of the soft robot. This allows to track the tip position with a simple webcam.

2.2 Simulation model

Soft robots often have the shape of a long, slender beam with dominant bending deformation. Other forms of deformations can often be neglected. Therefore, for the modeling of soft robots mostly beam models such as the piecewise constant curvature (PCC) model or the Cosserat rod theory, also called geometrically exact rod theory, are used [3]. The PCC only models bending and torsion, while the Cosserat rod model additionally considers elongation. In this contribution a Cosserat rod model based on [24] with a discretization into 6 segments is used. In the following, the used model is briefly summarized. The derivation of the equation of motion is shown in detail in [24] and [25] and therefore omitted here.

Table 1: Geometric and material parameters of the used soft robot.

		value	unit
total length	L	120	[mm]
radius	r	20	[mm]
density	ρ	1100	$[{}^{kg}/_{\mathrm{m}^3}]$
Young's modulus	E	2.04	[MPa]
Poisson's ratio	ν	0.49	[-]

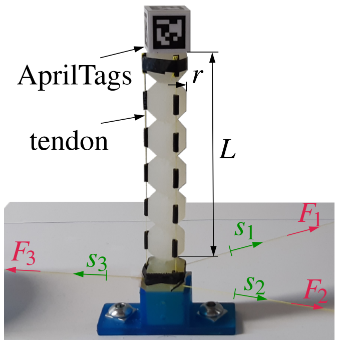


Figure 1: Soft robot.

2.2.1 Beam model

The used Cosserat rod model describes the continuous beam with

$$\boldsymbol{x}(s,t) : [0,L] \times [0,T] \to \mathbb{R}^3 \tag{1}$$
$$\boldsymbol{p}(s,t) : [0,L] \times [0,T] \to \mathbb{S}^3. \tag{2}$$

The vector $\boldsymbol{x}(s,t)$ represents the position of a point of the beam along the center line of th beam s at time t. The orientation is determined by the quaternions $\boldsymbol{p}(s,t)$. The beam is spatially discretized along the beam coordinate on a staggered grid as shown in fig. 2. The translatory degrees of freedom are located at the vertices, the rotatory degrees of freedom are located at the segment midpoints as quaternions. The model can be written as a set of ordinary differential equations of form

$$\ddot{\boldsymbol{x}}_n = \boldsymbol{g}(\dot{\boldsymbol{x}}_n, \dot{\boldsymbol{p}}_n, \boldsymbol{x}_n, \boldsymbol{p}_n, t, \boldsymbol{f}_{\text{tendon},i}, \boldsymbol{l}_{\text{tendon},i}) \tag{3}$$
$$\ddot{\boldsymbol{p}}_n = \boldsymbol{h}(\dot{\boldsymbol{x}}_n, \dot{\boldsymbol{p}}_n, \boldsymbol{x}_n, \boldsymbol{p}_n, t, \boldsymbol{f}_{\text{tendon},i}, \boldsymbol{l}_{\text{tendon},i}). \tag{4}$$

Here $\boldsymbol{f}_{\text{tendon},i}$ and $\boldsymbol{l}_{\text{tendon},i}$ are the forces and torques resulting from the tendon actuation. Their derivation is described in in sec. 2.2.2.

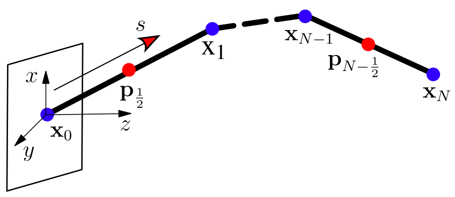


Figure 2: Schematic illustration of the discretized beam.

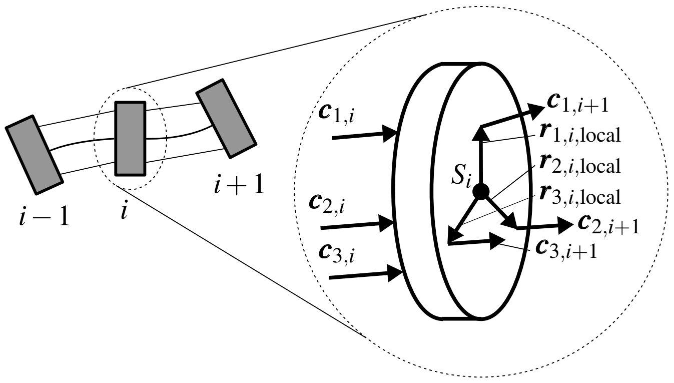


Figure 3: Position of tendon routing points and tendon force vectors on a disk.

2.2.2 Actuation forces and torques

The forces and torques resulting from the cable actuation can be derived analogously to the procedure described in [1] for the planar case. In the following, the extension to the 3D case is briefly shown. For the calculation of the actuation forces friction in the tendon guidance is neglected.

Therefore, for each tendon the tendon force F_q is constant over the whole tendon length. Here, the index $q = 1 \dots 3$ is the index of the tendon. A tendon triple can either pass through a disk, end at the disk or not reach this disk. It is assumed, that the tendon triple passes the first $k-1$ disks and ends at disk k. Note that actuation forces also act on all disks which the tendon triple only passes. For the calculation of the actuation forces the routing points of the tendons through the disks are of importance. These are shown in fig. 3 for the configuration of tendons used in this contribution. The locations of the routing points in relation to the center of gravity S_i of a disk is given by

$$\boldsymbol{r}_{1,i,\text{local}} = r_{\text{tendon}} \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^{\mathrm{T}}, \tag{5}$$

$$\boldsymbol{r}_{2,i,\text{local}} = r_{\text{tendon}} \begin{bmatrix} -1/2 & \sqrt{2}/2 & 0 \end{bmatrix}^{\mathrm{T}}. \tag{6}$$

$$\boldsymbol{r}_{3,i,\text{local}} = r_{\text{tendon}} \begin{bmatrix} -1/2 & -\sqrt{2}/2 & 0 \end{bmatrix}^{\mathrm{T}}. \tag{7}$$

In global coordinates this can be written as

$$\boldsymbol{r}_{q,i} = \boldsymbol{r}_{q,i-1} + \boldsymbol{R}_i \boldsymbol{r}_{q,i,\text{local}}. \tag{8}$$

From the position of the routing points $\boldsymbol{r}_{q,i}$ now the direction of the tendons, and therewith the direction of the tendon forces, from disk i to disk $i-1$ can be calculated by

$$\boldsymbol{c}_{q,i} = \frac{\boldsymbol{r}_{q,i-1} - \boldsymbol{r}_{q,i}}{||\boldsymbol{r}_{q,i-1} - \boldsymbol{r}_{q,i}||}. \tag{9}$$

From this, the forces $\boldsymbol{f}_{\text{tendon},i}$ and torques $\boldsymbol{l}_{\text{tendon},i}$ acting on disk i result in

$$\boldsymbol{f}_{\text{tendon},i} = \begin{cases} \sum\limits_{q=1}^{3} (\boldsymbol{c}_{q,i} F_q - \boldsymbol{c}_{q,i+1} F_q) & i < k \\ \sum\limits_{q=1}^{3} \boldsymbol{c}_{q,i} F_q & i = k \\ \boldsymbol{0} & \text{otherwise} \end{cases}, \tag{10}$$

$$\boldsymbol{l}_{\text{tendon},i} = \begin{cases} \sum\limits_{q=1}^{3} \left((\boldsymbol{R}_i \cdot \boldsymbol{r}_{i,q,\text{local}}) \times (\boldsymbol{c}_{q,i} F_q - \boldsymbol{c}_{q,i+1} F_q) \right) & i < k \\ \sum\limits_{q=1}^{3} \left((\boldsymbol{R}_i \cdot \boldsymbol{r}_{i,q,\text{local}}) \times (-\boldsymbol{c}_{q,i} F_q) \right) & i = k \\ \boldsymbol{0} & \text{otherwise} \end{cases}. \tag{11}$$

3 CONTROLLER SETUP FOR TRAJECTORY TRACKING CONTROL

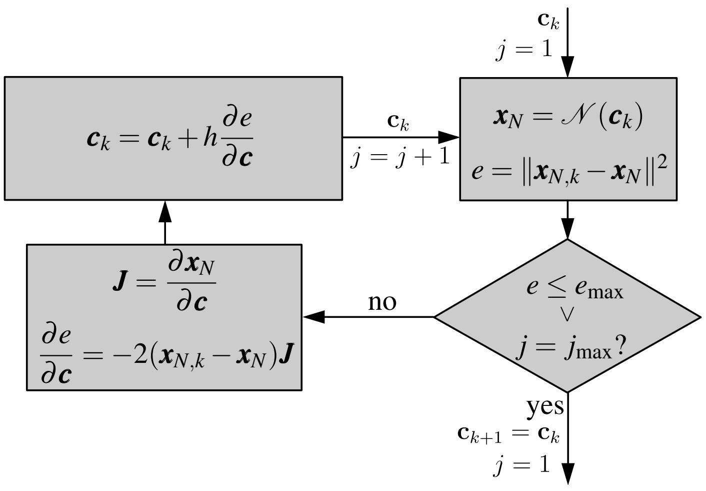
3.1 Trajectory tracking problem

The trajectory tracking problem of the tip position $\boldsymbol{x}_N$ can be formulated as a set of points $\mathscr{T} = \{\boldsymbol{x}_{N,1}, \dots, \boldsymbol{x}_{N,k-1}, \boldsymbol{x}_{N,k}, \boldsymbol{x}_{N,k+1}, \dots, \boldsymbol{x}_{N,K}\}$ which have to be reached at specified points of time $t = t_0(T)t_K$ with $k = 1(1)K$ as the number of trajectory points. For simplicity a constant sample-time T can be assumed. Using these, the control variables $\boldsymbol{c}_k \in \mathbb{R}^n$ must be found that lead to the location $\boldsymbol{x}_{N,k}$, where n is the number of independent actuators. In each timestep the control variables $\boldsymbol{c}_k$ have to be determined such that the trajectory is followed. This can be formulated as an optimization problem for the tracking error e:

$$\boldsymbol{c}_k = \arg\min_{\boldsymbol{c}} e(\boldsymbol{x}_{N,k}, \boldsymbol{c}) = \arg\min_{\boldsymbol{c}} ||\boldsymbol{x}_{N,k} - \boldsymbol{x}_N(\boldsymbol{c})||^2 \tag{12}$$

where

$$e = ||\boldsymbol{x}_{N,k} - \boldsymbol{x}_N(\boldsymbol{c})||^2 \tag{13}$$



$\mathbf{c}_k$
$j=1$
$\boldsymbol{c}_k = \boldsymbol{c}_k + h\frac{\partial e}{\partial \boldsymbol{c}}$
$\mathbf{c}_k$
$j=j+1$
$\boldsymbol{x}_N = \mathcal{N}(\boldsymbol{c}_k)$
$e = \|\boldsymbol{x}_{N,k} - \boldsymbol{x}_N\|^2$
$\boldsymbol{J} = \frac{\partial \boldsymbol{x}_N}{\partial \boldsymbol{c}}$
$\frac{\partial e}{\partial \boldsymbol{c}} = -2(\boldsymbol{x}_{N,k} - \boldsymbol{x}_N)\boldsymbol{J}$
no
$e \leq e_{\max} \vee j = j_{\max}?$
yes
$\mathbf{c}_{k+1} = \mathbf{c}_k$
$j=1$

Figure 4: Control algorithm for determining control variables $\boldsymbol{c}$.

is the trajectory tracking error. The function $\boldsymbol{x}_N(\boldsymbol{c})$ represents the forward kinematics of the system. For the system considered in this contribution the control variables $\boldsymbol{c}_k$ are the tendon forces F_i of the three tendons in simulation and the tendon length s_i in experiment. As described in sec. 2.1 these two quantities can be exchanged in a kinematic approach.

3.2 Controller

For many systems the optimization problem from eq. (12) cannot be solved analytically. Therefore, in this contribution an approach based on [9, 26] is used. The control algorithm used is visualized in fig. 4. Starting from an initial value $\boldsymbol{c}_k$ for the control variables, the control error e is determined by the forward kinematics. If the error is larger than a maximum allowable error $e_{\max}$, an update step for the control variables is performed:

$$\boldsymbol{c}_{k,\text{new}} = \boldsymbol{c}_k + h\frac{\partial e}{\partial \boldsymbol{c}}, \tag{14}$$

with

$$\frac{\partial e}{\partial \boldsymbol{c}} = -2(\boldsymbol{x}_{N,k} - \boldsymbol{x}_N(\boldsymbol{c}))\boldsymbol{J}(\boldsymbol{c}). \tag{15}$$

The central difference approximation for the Jacobian $\boldsymbol{J}_i$ of the forward kinematics $\boldsymbol{s}(\boldsymbol{c})$ is

$$\boldsymbol{J}_i = \frac{\partial \boldsymbol{x}_N(\boldsymbol{c})}{\partial c_i} \approx \frac{\boldsymbol{x}_N(\ldots, c_i + \Delta c, \ldots) - \boldsymbol{x}_N(\ldots, c_i - \Delta c, \ldots)}{2\Delta c}. \tag{16}$$

The newly calculated control variables are then fed back into the forward kinematics. This process is repeated until the error is less than the maximum allowed error $e_{\max}$ or the maximum number of iterations $j_{\max}$ is reached. This is to prevent the algorithm from remaining in an infinite loop if e.g. a point outside the workspace is part of the trajectory.

3.3 Approximation of the forward kinematics with a neural network

For real-time applications the usage of full mechanical models, which are often very computationally intensive, is often not possible. In addition, it is also very difficult to account for manufacturing inaccuracies in these models. Therefore, here the forward kinematics $\boldsymbol{x}_N(\boldsymbol{c})$ are approximated with a neural network $\mathcal{N}$ with $\boldsymbol{x}_N(\boldsymbol{c}) = \mathcal{N}(\boldsymbol{c})$. In this contribution a small and lightweight neural network with two hidden layers with a *tanh* activation function is used for the trajectory tracking problem in simulation. The hidden layers have 4 neurons each. For the experimental realization a slightly more complex neural network with three hidden layers is required. This can be explained by the fact that manufacturing inaccuracies, which should be learned as well to achieve accurate control, make the system more complex. Therefore, also a more complex neural network with more parameters is needed. Finally, it is noted that in the experiments performed in this contribution the neural network was 25 times faster than the solution of the equation of motion.

For the training of the neural network for both, in simulations and in experiments, a dataset containing the control variables $\boldsymbol{c}$ and the tip position is collected. In simulation, thereby the forces F_q of all three tendons are varied with $F_q = 0(0.01)0.4\,\mathrm{N}$. Forces are only applied to two tendons while the force for the third tendon is kept at $0\,\mathrm{N}$. In a total this results in 5043 different loading cases. In the experiment the tendon length s_q is varied between $s_q = 0(2)20\,\mathrm{mm}$. This results in 363 different combinations, since only two of the tendon lengths s_i can be chosen independently. For each load case the individual simulations or experiments are executed until the robot no longer performs any measurable motion. The position of the soft robot is then stored with the associated load cases.

4 RESULTS

In the following, the performance of the trajectory tracking controller is described in simulations and experiment. In both cases, the training results are shown first, followed by the trajectory tracking results. The trajectories examined are each tracked uniformly within $T = 50\,\mathrm{s}$.

4.1 Simulation results

Firstly, the NN is trained using the simulation results. In fig. 5 an excerpt of the projection of the tip position into the $x-y$ plane of the collected training data as well as the prediction of the trained NN is shown. In fig. 6 a comparison between the validation data and the prediction of the neural network is shown. A good agreement between the simulations and the predictions of the neural network can be seen. Only for large tendon forces there is a slight difference between the training data and the prediction of the neural network. For practical applications this error is most likely not relevant, as other errors, mainly due to unmodeled effects, typically result in much larger errors. However, with a more complex neural network with more parameters the error could be further reduced.

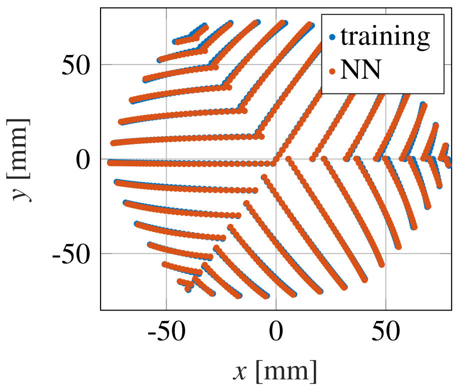


Figure 5: 2D projection of training data and NN prediction.

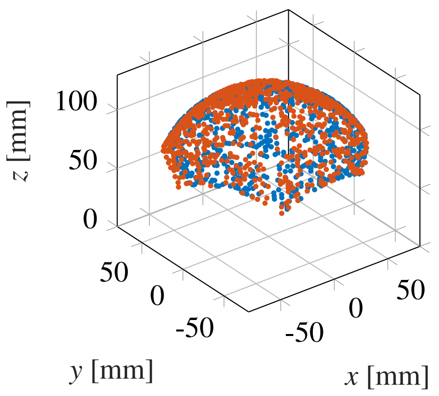


Figure 6: Results of NN on validation data.

For the investigation of the control performance, in a first step a circular trajectory $\mathscr{T}$ is tracked. This trajectory starts with a straight-line segment leading from the tip position origin to a circular path and then follows this circular path with a radius of $50\,\mathrm{mm}$. For the maximum allowed error,

$e_{\max} = (2\,\text{mm})^2$ and for the maximum number of loop repetitions $j_{\max} = 100$ is chosen. In fig. 7 the desired trajectory ($\mathscr{T}$), the trajectory calculated using the neural network (NN) and the tracked trajectory (ODE) are shown. Thereby, the output from the NN is applied to the equations of motion. A good correspondence of all three trajectories can be observed. This is confirmed by th error plot shown in fig. 8. In the plot the distance error and the maximum allowed error are shown. Both the error e_{ODE} of the solution of the ODE and the error e_{NN} of the calculated quantities by the neural network are predominantly smaller than the specified maximum allowed error. At certain points, the error e_{NN} increases abruptly. A closer look at these points shows that here a force is only applied to one tendon.

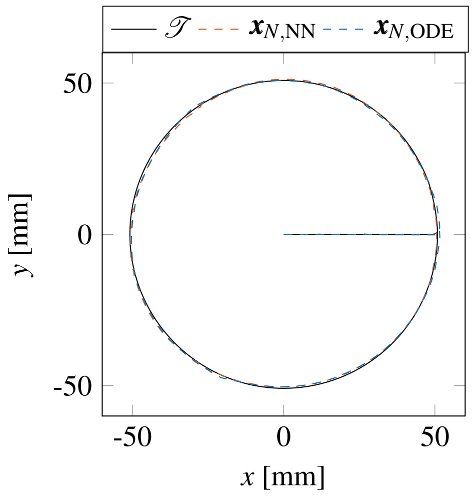


Figure 7: Tracking of circular trajectory.

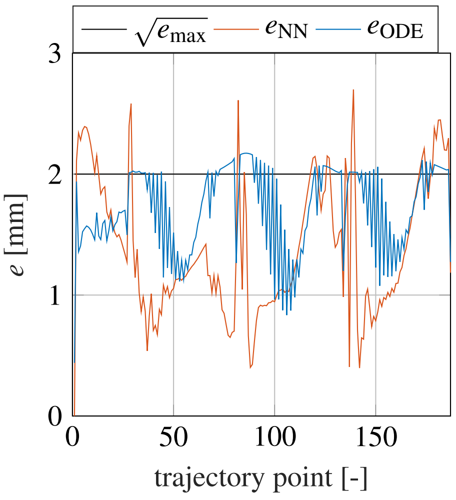


Figure 8: Absolute error of the controller and the NN.

4.2 Experimental results

The training results of the neural network in experiment are shown in fig. 9. They are comparable to the results in simulation. The main difference is that the data points are clearly divided into three sectors. This can be explained by inaccuracies in the experimental setup. The gaps correspond to load cases where only one tendon is loaded for actuation. A small number of data points is missing due to measurement errors when the camera did not correctly detect the tags. As expected, the neural network's accuracy is lower than in simulation, but overall still provides a good approximation of the forward kinematics.

As in simulation, in a first step the circular trajectory $\mathscr{T}$ is tracked. In fig. 10 the desired trajectory $\mathscr{T}$, the actually tracked trajectory $\boldsymbol{x}_{N,\text{exp}}$ in the experiment and the trajectory calculated by the neural network $\boldsymbol{x}_{N,\text{NN}}$ are shown. It can be seen that the trajectory calculated by the neural network $\boldsymbol{x}_{N,\text{NN}}$ is very similar to the desired trajectory and almost always stays within the allowed range. The actually tracked trajectory $\boldsymbol{x}_{N,\text{exp}}$, however, has much larger deviations, particularly in the upper right and lower left quadrants. In the other two quadrants the control error is much smaller. Further experiments have shown that the tracked trajectory is very accurately repeatable. This is shown in fig. 11 for 4 repetitions of the experiment. The small deviations between the four repetitions of the experiment can be explained with measurement errors of the used camera tracking system. The reproducibility indicates that there is a systematic error resulting from inaccuracies in the approximation of the forward kinematics with the neural network.

Finally, the influence of the movement speed of the soft robot on the controller performance is investigated. For this purpose, the travel time T of the trajectory is varied between $10 \ldots 50\,\text{s}$. Faster movements cannot be realized with the experimental setup used. The results are shown in fig. 12. No influence of the speed can be determined. Small differences between the trajectories can be

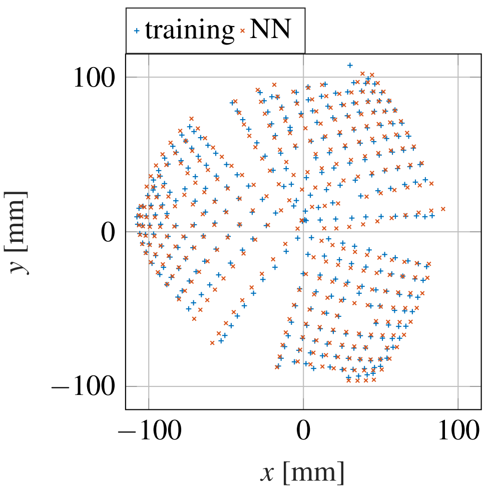


Figure 9: Training data and NN prediction.

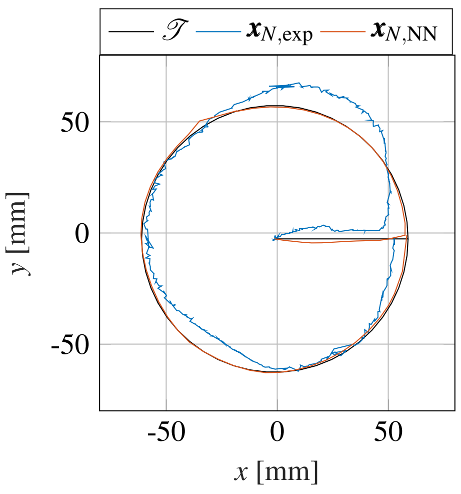


Figure 10: Tracking of circular trajectory.

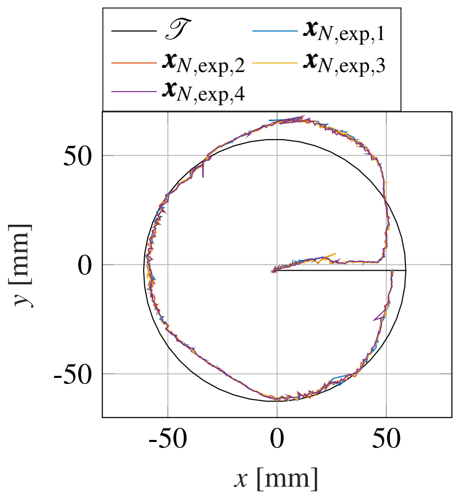


Figure 11: Reproducibility of the trajectory tracking.

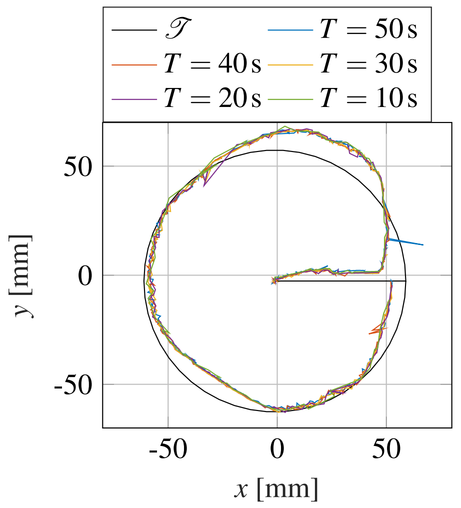


Figure 12: Trajectory tracking with different velocities.

explained by measurement errors of the camera tracking system. This confirms the assumption that the dynamics of the system can be neglected and kinematic controllers are sufficient for the control of this soft robot.

5 CONCLUSIONS

In this contribution we presented the modeling and kinematic control of a beam-shaped tendon-actuated soft robot. For reference measurements a camera tracking system based on AprilTags was used to measure the position of the tip of the soft robot. In a first step a full dynamic mechanical model of the soft robot based on the Cosserat rod theory was presented. In a second step a kinematic controller based on a neural network approximation of the forward kinematics was presented and tested in simulations and experiments. In simulations, very accurate trajectory tracking could be archived. In experiments, still good results could be archived. But, as expected, the trajectory tracking control error is much larger than in simulation. The control error can probably be further reduced by using a more complex neural network, however, this comes with additional computational costs and requires more training data. Finally, it was shown, that kinematic controllers are sufficient for the considered control task of this soft material robot and dynamic controllers most likely do not have an advantage here.

REFERENCES

[1] Grube, M., Wieck, J.C., Seifried, R.: Comparison of modern control methods for soft robots. Sensors **22**(23) (2022) 9464

[2] George Thuruthel, T., Ansari, Y., Falotico, E., Laschi, C.: Control strategies for soft robotic manipulators: A survey. Soft Robotics **5**(2) (2018) 149–163

[3] Lee, C., Kim, M., Kim, Y.J., Hong, N., Ryu, S., Kim, H.J., Kim, S.: Soft robot review. IJCAS **15**(1) (2017) 3–15

[4] Camarillo, D.B., Carlson, C.R., Salisbury, J.K.: Configuration tracking for continuum manipulators with coupled tendon drive. IEEE Transactions on Robotics **25**(4) (2009) 798–808

[5] Bailly, Y., Amirat, Y.: Modeling and control of a hybrid continuum active catheter for aortic aneurysm treatment. In: Proceedings of the 2005 IEEE International Conference on Robotics and Automation. (2005) 924–929

[6] Jones, B., Walker, I.: Kinematics for multisection continuum robots. IEEE Transactions on Robotics **22**(1) (2006) 43–55

[7] Renda, F., Armanini, C., Mathew, A., Boyer, F.: Geometrically-exact inverse kinematic control of soft manipulators with general threadlike actuators' routing. IEEE Robotics and Automation Letters **7**(3) (2022) 7311–7318

[8] Giorelli, M., Renda, F., Ferri, G., Laschi, C.: 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. (2013) 5033–5039

[9] Fang, G., Tian, Y., Yang, Z.X., Geraedts, J., Wang, C.: Jacobian-based learning for inverse kinematics of soft robots. IEEE/ASME Transactions on Mechatronics **27**(6) (2020) 5296–5306

[10] Thuruthel., T.G., Falotico., E., Cianchetti., M., Renda., F., Laschi., C.: Learning global inverse statics solution for a redundant soft robot. In: Proceedings of the 13th International Conference on Informatics in Control, Automation and Robotics - Volume 2: ICINCO,, INSTICC, SciTePress (2016) 303–310

[11] Thuruthel, T.G., Falotico, E., Manti, M., Pratesi, A., Cianchetti, M., Laschi, C.: Learning closed loop kinematic controllers for continuum manipulators in unstructured environments. Soft Robotics **4**(3) (2017) 285–296

[12] Best, C.M., Gillespie, M.T., Hyatt, P., Rupert, L., Sherrod, V., Killpack, M.D.: A new soft robot control method: Using model predictive control for a pneumatically actuated humanoid. IEEE Robotics & Automation Magazine **23**(3) (2016) 75–84

[13] Ouyang, B., Mo, H., Chen, H., Liu, Y., Sun, D.: Robust model-predictive deformation control of a soft object by using a flexible continuum robot. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE (2018) 613–618

[14] Hyatt, P., Killpack, M.D.: Real-time nonlinear model predictive control of robots using a graphics processing unit. IEEE Robotics and Automation Letters **5**(2) (2020) 1468–1475

[15] Bruder, D., Fu, X., Gillespie, R.B., Remy, C.D., Vasudevan, R.: Data-driven control of soft robots using koopman operator theory. IEEE Transactions on Robotics **37**(3) (2021) 948–961

[16] Gillespie, M.T., Best, C.M., Townsend, E.C., Wingate, D., Killpack, M.D.: Learning non-linear dynamic models of soft robots for model predictive control with neural networks. In: 2018 IEEE International Conference on Soft Robotics (RoboSoft). (2018) 39–45

[17] Kapadia, A., Walker, I.D.: Task-space control of extensible continuum manipulators. In: 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems. (2011) 1087–1092

[18] Falkenhahn, V., Hildebrandt, A., Neumann, R., Sawodny, O.: Model-based feedforward position control of constant curvature continuum robots using feedback linearization. In: 2015 IEEE International Conference on Robotics and Automation (ICRA). (2015) 762–767

[19] Kapadia, A.D., Walker, I.D., Dawson, D.M., Tatlicioglu, E.: A model-based sliding mode controller for extensible continuum robots. In: Proceedings of the 9th WSEAS international conference on Signal processing, robotics and automation. (2010) 113–120

[20] Engel, Y., Szabo, P., Volkinshtein, D.: Learning to control an octopus arm with gaussian process temporal difference methods. In Weiss, Y., Schölkopf, B., Platt, J., eds.: Advances in Neural Information Processing Systems. Volume 18., MIT Press (2005)

[21] Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., Riedmiller, M.: Deterministic policy gradient algorithms. In Xing, E.P., Jebara, T., eds.: Proceedings of the 31st International Conference on Machine Learning. Volume 32 of Proceedings of Machine Learning Research., Bejing, China, PMLR (22–24 Jun 2014) 387–395

[22] Zhang, H., Cao, R., Zilberstein, S., Wu, F., Chen, X.: Toward effective soft robot control via reinforcement learning. In: Proceedings of the 10th International Conference on Intelligent Robotics and Applications (ICIRA), Wuhan, China (August 2017) 173–184

[23] Olson, E.: Apriltag: A robust and flexible visual fiducial system. In: 2011 IEEE International Conference on Robotics and Automation. (2011) 3400–3407

[24] Lang, H., Linn, J., Arnold, M.: Multi-body dynamics simulation of geometrically exact cosserat rods. Multibody System Dynamics **25** (2011) 285–312

[25] Grube, M., Seifried, R.: Simulation of soft robots with nonlinear material behavior using the cosserat rod theory. In: 8th European Congress on Computational Methods in Applied Sciences and Engineering, ECCOMAS 2022, SCIPEDIA (2022)

[26] Fang, G., Matte, C.D., Scharff, R.B.N., Kwok, T.H., Wang, C.C.L.: Kinematics of soft robots by geometric computing. IEEE Transactions on Robotics **36**(4) (2020) 1272–1286