Soft Matter

PAPER

View Article Online
View Journal | View Issue

Cite this: *Soft Matter*, 2023, 19, 44

Learning 3D shape proprioception for continuum soft robots with multiple magnetic sensors†

Thomas Baaij, ‡[a] Marn Klein Holkenborg, ‡[a] Maximilian Stölzle, ‡*[a] Daan van der Tuin, ‡[a] Jonatan Naaktgeboren, [a] Robert Babuška [ab] and Cosimo Della Santina [ac]

Received 6th July 2022,
Accepted 28th November 2022

DOI: 10.1039/d2sm00914e

rsc.li/soft-matter-journal

Sensing the shape of continuum soft robots without obstructing their movements and modifying their natural softness requires innovative solutions. This letter proposes to use magnetic sensors fully integrated into the robot to achieve proprioception. Magnetic sensors are compact, sensitive, and easy to integrate into a soft robot. We also propose a neural architecture to make sense of the highly nonlinear relationship between the perceived intensity of the magnetic field and the shape of the robot. By injecting *a priori* knowledge from the kinematic model, we obtain an effective yet data-efficient learning strategy. We first demonstrate in simulation the value of this kinematic prior by investigating the proprioception behavior when varying the sensor configuration, which does not require us to re-train the neural network. We validate our approach in experiments involving one soft segment containing a cylindrical magnet and three magnetoresistive sensors. During the experiments, we achieve mean relative errors of 4.5%.

1 Introduction

The past decade has seen an explosion of novel continuum soft robotic platforms.[1–3] Inspired by invertebrate animals, these robots are almost entirely composed of soft deformable materials.[4] This makes them robust and safe, but at the same time, it renders their modeling,[5] control,[6] and shape sensing[7] substantially more complex than for their rigid counterparts. Shape sensing is especially intricate because it is both a technological and algorithmic challenge. Sensors must not obstruct the natural behavior or reduce the compliance of soft robots. At the same time, non-collocated and nonlinear sensors require algorithms for the measurements to be interpreted and connected to a description of the robot's shape.

Several sensing modalities have been considered to implement shape sensing, such as resistive,[8,9] capacitive,[10,11] optical,[12] and visual.[13] Magnetic sensors[14–18] are a promising solution as they are compact, highly sensitive, and can be easily integrated into existing soft robot designs. Thus, they can provide reliable and fully proprioceptive measurements at the cost of a minimal decrease in the robot's softness. Among the above-cited papers, the only work leveraging external magnetic fields is by Song *et al.*,[14] where coils placed at a distance generate the magnetic field. Along this thought, an obvious choice would be to measure the earth's magnetic field for proprioception purposes. However, it would only allow estimating two rotational components, and any translational effects, such as an elongation of the continuum robot, could not be captured. Alternatively, some papers[15–17] use co-axial pairs of magnets and sensors embedded in the robot to estimate planar deformations. Recently, Mitchell *et al.*[18] have shown that a similar strategy can sense 3D deformations. Such simple arrangements greatly simplify the analysis, allowing for connecting sensor readings to the robot's shape through direct interpolation. Nevertheless, relying on isolated pairs of sensors and magnets also strongly limits the density and the amount of information gathered through this method.

This paper proposes to use permanent ring magnets and multiple magnetic sensors for shape sensing of continuum soft robots. Importantly, we remove the requirement of magnets and sensors being placed in co-axial pairs. We have been inspired by recent work leveraging deep learning to interpret various types of non-magnetic sensor data for proprioception purposes.[19–22] However, learning end-to-end mappings from sensors to configurations has three significant drawbacks: (i) it is data-intensive, (ii) it calls for recurrent architectures to encourage temporal consistency for the robot's configuration

[a] *Cognitive Robotics, Delft University of Technology, Mekelweg 2, 2628 CD Delft, The Netherlands. E-mail: T.A.Baaij@tudelft.nl, M.L.B.KleinHolkenborg@tudelft.nl, M.W.Stolzle@tudelft.nl, D.D.P.vanderTuin@tudelft.nl, J.Naaktgeboren-1@tudelft.nl, R.Babuska@tudelft.nl, C.DellaSantina@tudelft.nl*
[b] *Czech Institute of Informatics Robotics and Cybernetics, Czech Technical University in Prague, 160 00 Prague, Czech Republic*
[c] *Institute of Robotics and Mechatronics, German Aerospace Center (DLR), 82234 Weßling, Germany*

† Electronic supplementary information (ESI) available. See DOI: https://doi.org/10.1039/d2sm00914e

‡ These authors contributed equally to this work and are co-first authors.

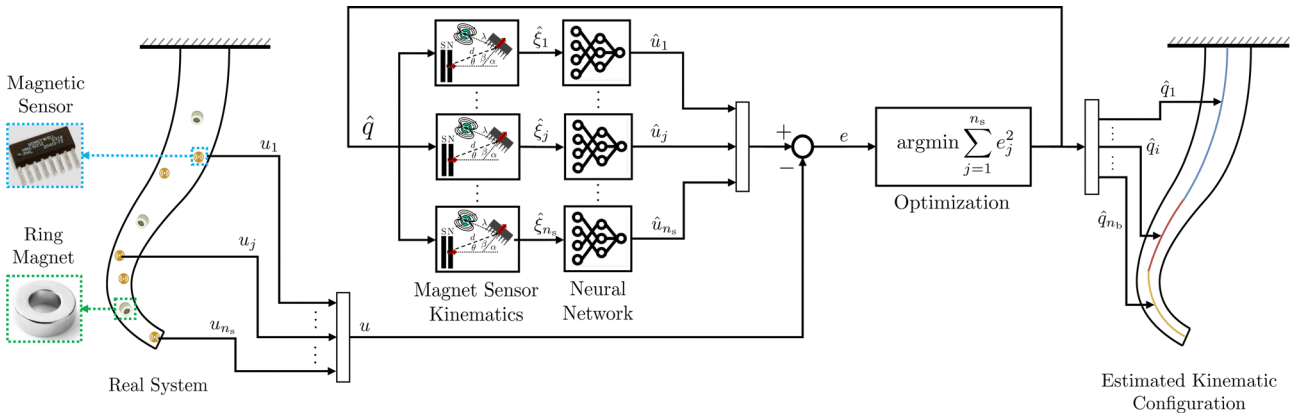


Fig. 1 Proprioception for continuum soft robots with magnetic sensors: an initial configuration estimate $\hat{q}$ is employed to derive the kinematic relationship $\hat{\xi}$ between a sensor and each magnet. Subsequently, this kinematic description is used to predict the sensor measurements $\hat{u} \in \mathbb{R}^{n_s}$ with a neural network trained in advance. A mean squared error (MSE) metric evaluates the accuracy of the predictions $\hat{u}$ compared to the actual measurements u. Finally, we optimize the configuration estimate $\hat{q}$ to achieve proprioception by minimizing the sensor measurement prediction loss.

estimates, (iii) it requires re-training when changing the kinematic model of the robot. We propose a neural architecture that circumvents all three issues (see Fig. 1). First, we train shallow neural networks to predict the measurements of the magnetic sensors from a parameterization describing their relative pose with respect to the magnets. We then optimize the configuration estimate – and thus the sensor positions – to minimize the error between the predicted and actual sensor measurements. This way, we introduce *a priori* information on the modes of deformation of a continuum soft robot, effectively removing the kinematics from the black box. The presented strategy enables us to re-arrange and remove redundant sensors during inference without requiring us to re-train the neural network on the adjusted sensor configuration.

To summarize, this paper contributes to the state of the art in soft robot sensing with:

- A proprioceptive sensing modality relying on multiple magnetic sensors in conjunction with a neural network-based architecture that learns to estimate the full 3D shape of the robot from the sensor readings.
- Injection of kinematic priors through a description to spatially relate the poses of a sensor to the magnets. This proposed parametrization serves as an input to a neural network predicting the sensor measurements.
- Experimental verification of the approach for a one-segment robot with three integrated magnetoresistive sensors and proprioception of 3D curvature.

We open-source a Python / PyTorch implementation of the proposed algorithm and the corresponding datasets on GitHub.§

2 Proprioception with magnetic sensors

This section introduces a methodology to achieve proprioception for soft robots with magnetic sensors. We consider a continuum robot with the shape of its backbone described by the configuration variables $q \in \mathbb{R}^{n_q}$. In the commonly used piecewise constant curvature (PCC) kinematic state parametrization,[23] the continuum robot is assumed to consist of n_b segments with each segment exhibiting constant curvature and elongation along its length. Therefore, the configuration of the soft robot can be described with $q \in \mathbb{R}^{3n_b}$. Please refer to Appendix A.1 for more details. We indeed use PCC for most of our simulations and experiments. Note, however, that the proposed proprioception algorithm applies to any finite-dimensional kinematic description of a soft robot.[5] In fact, we also specifically consider a robot with affine curvature[24,25] with its shape described by the configuration $q \in \mathbb{R}^4$. We document this alternative kinematic model in Appendix A.2. The proprioception methodology described in the remaining section is agnostic to the chosen kinematic model.

2.1 Proposed method at a glance

We integrate n_m axially symmetric magnets and n_s magnetic sensors into the robot. The magnets must be installed along the center line of the segment while the sensors can be arbitrarily placed. Fig. 1 concisely describes the proposed shape-sensing strategy with a pictorial example of a soft robot consisting of three segments and equipped with five magnetic sensors and three cylindrical magnets.

The goal of the algorithmic part (center of the figure) is to regress the robot shape (left) by estimating the configuration $\hat{q}$ of all segments (right), starting from the measurements of the magnetic sensors u (*e.g.* usually the magnetic flux density). We achieve this by training a sensor measurement predictor and subsequently optimizing the configuration estimate $\hat{q}$ for the prediction $\hat{u}$ to match the actual sensor measurement u. Instead of predicting the sensor measurements end-to-end, we decouple the kinematics by explicitly describing the kinematic relationship $\hat{\xi}_j = f_{\xi,j}(\hat{q})$ between sensor j and each magnet. Then, we train a neural network f_{π_j} to predict the measurement $\hat{u}_j$ based on the input $\hat{\xi}_j$. To achieve proprioception, we jointly optimize $\hat{q}$ for all sensor measurement predictions.

§ https://www.github.com/tud-cor-sr/promasens.

2.2 Magnet sensor kinematics

This subsection derives the kinematic relationship $\xi_j = f_{\xi_j}(q) \in \mathbb{R}^{1+4n_\mathrm{m}}$ between the jth sensor and all n_m magnets as we hypothesize that we can estimate the sensor measurement u_j solely based on (a) the angle $\lambda_j \in \mathbb{R}^1$ to the earth's magnetic field, (b) the distance d_j^k between the jth sensor and kth magnet, (c) the angle α_j^k between the sensor measurement direction and the cylindrical axis of the magnet, (d) the angle β_j^k between the sensor measurement direction and the vector from the magnet to the sensor, and (e) the angle θ_j^k between the cylindrical axis of the magnet and the vector from the magnet to the sensor. Accordingly, ξ_j is defined as

$$\xi_j = f_{\xi,j}(q) = \begin{pmatrix} \lambda_j & \xi_j^{1\mathrm{T}} & \cdots & \xi_j^{k\mathrm{T}} & \cdots & \xi_j^{n_\mathrm{m}\mathrm{T}} \end{pmatrix}^\mathrm{T} \in \mathbb{R}^{1+4n_\mathrm{m}}, \quad (1)$$

with $\xi_j^k \in \mathbb{R}^4$ the kinematic relationship between the jth sensor and the kth magnet: $\xi_j^k = \begin{pmatrix} d_j^k & \alpha_j^k & \beta_j^k & \theta_j^k \end{pmatrix}^\mathrm{T} \in \mathbb{R}^4$. We visualize the parameters incorporated in ξ_j^k in Fig. 2(a).

In the following, we present the derivation of all components of ξ_j^k. Please note that all kinematic frames used in the following paragraphs are visualized in Fig. 2(b).

We define that the kth magnet is integrated into the ith segment. Now, we first derive a transformation matrix $T_{i-1}^{\mathrm{m}_k}$ from the base frame $\{S_{i-1}\}$ to the magnet frame $\{S_{\mathrm{m}_k}\}$. This can be achieved by evaluating the chosen kinematic model, two of which we report in the Appendix A, at the segment coordinate $v = \dfrac{d_{\mathrm{m}_k}}{L_{0,i}}$. This means that the cylindrical magnet is integrated at a distance, which is measured along the backbone, of d_{m_k} from the base of the segment.

Subsequently, we describe the pose of the jth sensor with respect to the base of the ith segment. Denote with $d_{\mathrm{s}_j,\mathrm{r}}$ the radial distance of the sensor from the center line, with φ_j the azimuth angle of the sensor in the cylindrical plane, and with $d_{\mathrm{s}_j,\mathrm{a}}$ the axial distance along the center line from the base of the ith segment. We derive the transformation $T_{i-1}^{\mathrm{s}_j,\mathrm{r}_0}$ to the center of the cylindrical plane of the sensor analogously as for the magnets by evaluating the forward kinematics at $v = \dfrac{d_{\mathrm{s}_j,\mathrm{a}}}{L_{0,i}}$. This is followed by applying the radial offset $d_{\mathrm{s}_j,\mathrm{r}}$ in the cylindrical plane of the sensor

$$T_{i-1}^{\mathrm{s}_j} = T_{i-1}^{\mathrm{s}_j,\mathrm{r}_0} \begin{bmatrix} \cos\varphi_j & -\sin\varphi_j & 0 & d_{\mathrm{s}_j}\cos(\varphi_j) \\ \sin\varphi_j & \cos\varphi_j & 0 & d_{\mathrm{s}_j}\sin(\varphi_j) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (2)$$

Optionally, a static rotation offset can be applied to $R_{i-1}^{\mathrm{s}_j}$ such that local z-axis o_{s_j} corresponds to the sensor measurement direction.

Knowing the transformation matrices from the base frame of the respective segment to the sensor and magnet frames, we express them in the inertial frame $\{S_0\}$ by multiplying with the kinematic chain $T_0^{i-1} = \prod_{\bar{i}=1}^{i-1} T_{\bar{i}-1}^{\bar{i}}$.

Next, we need to express the sensor measurement direction o_{s_j} and the cylindrical axis of the magnet o_{m_k} in the inertial frame

$$o_{\mathrm{s}_j} = R_0^{\mathrm{s}_j}(0 \quad 0 \quad 1)^\mathrm{T}, \quad o_{\mathrm{m}_k} = R_0^{\mathrm{m}_k}(0 \quad 0 \quad 1)^\mathrm{T}. \quad (3)$$

As the sensor measures contributions of the earth's magnetic field, we need to state the angle λ_j between o_{s_j} and the earth's magnetic field unit vector n_e. Similarly, we investigate the angle α_j^k between o_{s_j} and the cylindrical axis of the magnet o_{m_k}

$$\cos(\lambda_j) = n_\mathrm{e} \cdot o_{\mathrm{s}_j}, \quad \cos(\alpha_j^k) = o_{\mathrm{m}_k} o_{\mathrm{s}_j}. \quad (4)$$

We define the translation and distance between the magnet and the sensor in the frame $\{S_0\}$ as:

$$p_j^k = p_0^{\mathrm{s}_j} - p_0^{\mathrm{m}_k}, \quad d_j^k = \left\| p_j^k \right\|_2. \quad (5)$$

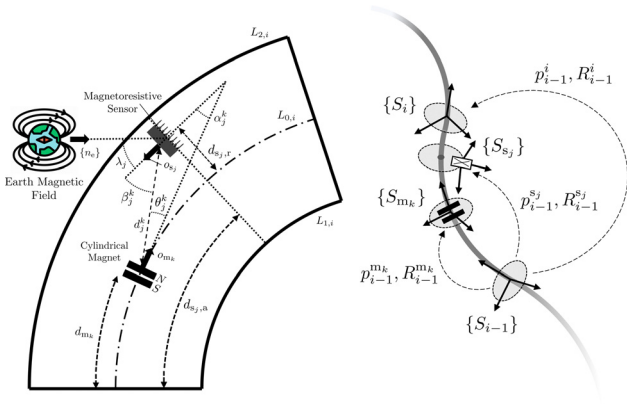
Building on the derivation in (5), we compute the angles β_j^k and θ_j^k using the dot product rule

$$\cos\left(\beta_j^k\right) = \frac{p_j^k o_{\mathrm{s}_j}}{\left\| p_j^k \right\|_2}, \quad \cos\left(\theta_j^k\right) = \frac{p_j^k o_{\mathrm{m}_k}}{\left\| p_j^k \right\|_2}. \quad (6)$$

Lastly, the kinematic descriptions for all sensors are vertically stacked as

$$\xi = \begin{pmatrix} \xi_1^\mathrm{T} \ldots \xi_j^\mathrm{T} \ldots \xi_{n_\mathrm{s}}^\mathrm{T} \end{pmatrix}^\mathrm{T} \in \mathbb{R}^{n_\mathrm{s}+4n_\mathrm{m}n_\mathrm{s}}. \quad (7)$$

We will in the following refer to the mapping $f_\xi(q): q \in \mathbb{R}^{n_\mathrm{q}} \rightarrow \xi \in \mathbb{R}^{n_\mathrm{s}+4n_\mathrm{m}n_\mathrm{s}}$ as the magnet sensor kinematics.



(a) Magnet Sensor Kinematics (b) Coordinate transformations

Fig. 2 Panel (a): Parameters used to describe the kinematics between the jth sensor and the kth magnet. To simplify the illustration, we visualize the unextended planar case with both magnet and sensor part of the ith segment. However, this is not a strict assumption as magnets and sensors can also be part of different segments. Panel (b): Coordinate frames for a soft segment containing the jth magnetoresistive sensor and the kth cylindrical magnet placed along the center line. $\{S_{i-1}\}$ and $\{S_i\}$ describe the frames of the base and tip of the ith segment, respectively.

2.3 Data-driven sensor measurement model

We use a data-driven approach to learn the forward sensor model $\hat{u} = f_\pi(\xi_j)$ for each sensor using a neural network parameterized with π. We note that the same neural network weights can be shared for all sensors, but oftentimes performance can be improved by training a specialized model with weights π_j for each sensor. During training on a dataset of length n_t, we minimize the mean squared error (MSE) error between the predicted sensor measurement $\hat{u}_j$ and the actual sensor measurement u_j:

$$\min_{\pi} \frac{1}{n_t} \sum_{t=0}^{n_t} \left(f_\pi(\xi_j(t)) - u_j(t)\right)^2, \tag{8}$$

where t denotes the current time index. Note that whenever we omit the time index in our notation, we always refer to the current time t. Finally, to simplify the notation, we combine each sensor measurement prediction $u_j \in \mathbb{R}$ into an array $u \in \mathbb{R}^{n_s}$ and stack the neural networks as $f_{II}(\xi)$:$\xi \to u$. We discuss the choice of the specific network architecture later in Section 3.

2.4 Proprioception by optimizing configuration estimate

Now, that we are able to predict the sensor measurement $\hat{u}$ using the composition of the kinematics $f_\xi(\hat{q})$ and the neural networks $f_{II}(\hat{\xi})$, we need to optimize the configuration estimate $\hat{q}$ for the predictions $\hat{u}$ to match the actual sensor measurements u as closely as possible. We capture the error between the predicted sensor measurements $\hat{u}$ and the actual sensor measurements u by the MSE loss function we strive to minimize

$$\mathscr{L}_u(\hat{q}) = \sum_{j=1}^{n_s} \frac{\left(f_\pi(\xi_j) - u_j\right)^2}{n_s} \tag{9}$$

Accordingly, the optimal configuration estimate $\hat{q}$ can be found with

$$\hat{q} = \operatorname{argmin} \mathscr{L}_u(\hat{q}_l). \tag{10}$$

We optimize the cost function (9) through iterative gradient descent, as detailed in Fig. 3. The gradient descent is initialized with the best estimate of the previous time-step $\hat{q}_0(t) = \hat{q}(t-1)$. As common in literature, we optimize the state belief $\hat{q}_l$ with a step size γ and the momentum μ using the Jacobian of the loss $\frac{\partial}{\partial \hat{q}_t}\mathscr{L}_u(\hat{q})$

$$b_{l+1} = \mu b_l + \frac{\partial}{\partial \hat{q}_t}\mathscr{L}_u(\hat{q}), \quad \hat{q}_{l+1} = \hat{q}_l - \gamma b_{l+1}. \tag{11}$$

We can use the chain rule to derive an analytical expression for the gradient of the loss incorporating the gradient of the magnet sensor kinematics $\partial_{\hat{q}} f_{\hat{\xi}}(\hat{q})$ and the gradient of the neural network $\partial_{\hat{\xi}} f_{II}(\hat{\xi})$:

$$\frac{\partial}{\partial \hat{q}}\mathscr{L}_u(\hat{q}) = \frac{2}{n_s}\left(\frac{\partial}{\partial \hat{q}} f_{\hat{\xi}}(\hat{q})\right)^T \left(\frac{\partial}{\partial \hat{\xi}} f_{II}(\hat{\xi})\right)^T (\hat{u} - u). \tag{12}$$

After executing the gradient descent for n_{it} iterations, we evaluate which iteration l^* had the lowest loss $\mathscr{L}_u$ and accordingly select $\hat{q}(t) = \hat{q}_{l^*}$ as the best configuration estimate of time-step t.

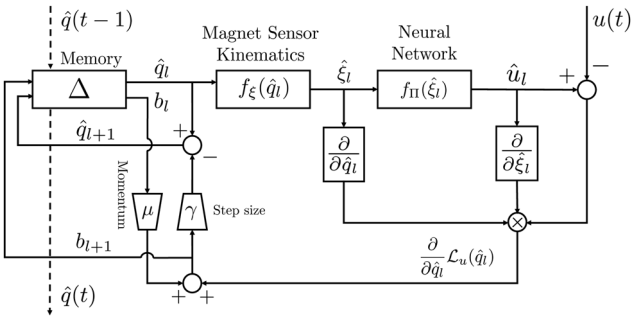


Fig. 3 A block-diagram of the gradient descent as an iterative update loop for the configuration belief $\hat{q}(t)$. The gradient descent is initialized with the optimized solution $\hat{q}_0 = \hat{q}(t-1)$ from the previous time-step. Note that during the iterative loop, $\hat{q}_{l+1}$ updates $\hat{q}_l$ in the memory block.

3 Piecewise constant curvature simulations

We evaluate the proposed methodology for estimating the PCC kinematic configuration $q \in \mathbb{R}^{3n_b}$ of soft continuum robots thoroughly in simulations. The PCC model allows for bending and elongation of each segment in 3D space. Please refer to Appendix A.1 for more details. We vary the number of robot segments n_b, remove and add sensors (*i.e.* change n_s), modify the arrangement of sensors, and the direction of the earth's magnetic field n_e. To motivate some of the unique advantages of our method, we use the same learned neural network weights for all these trials.

3.1 Simulation setup

In our simulations, we consider a robot consisting of one, two, or three segments. All cylindrical segments have an unextended length of $L_{0,i}$ = 110 mm and a radius of d_i = 22 mm. Each segment has a ring magnet of outer diameter 12 mm, inner diameter 6 mm and thickness 6 mm integrated along the backbone at a distance of d_{mk} = 55 mm from the base of the segment. In the nominal case, three sensors are placed symmetrically in the tip plane of each segment at a radius of 13 mm from the center with the sensor measurement direction pointing along the local, negative z-axis of the tip plane $\{S_i\}$. We also consider alternative placements of the sensors, which we further detail in Section 3.5.

We build on Magpylib[26] to simulate the magnetic field behavior. We model the magnets as cylindrical neodymium grade N50 rings with a magnetization of 1450 mT in the local z-direction. After simulating the magnetic field, we rotate the B-field into the local reference frame of each sensor and take the local z-component of the magnetic flux density as the sensor measurement.

3.2 Prediction network

The training set consists of 120 000 random configurations sampled from uniform distribution $\Delta_{x,i} \sim \mathcal{U}(-20.7\,\text{mm},\ 20.7\,\text{mm})$, $\Delta_{y,i} \sim \mathcal{U}(-20.7\,\text{mm}, 20.7\,\text{mm})$ and $\delta L_i \sim \mathcal{U}(0, 5.5\,\text{mm})$, where the upper bound represents a bending of the tip of 54° with respect to the

base of a segment and an elongation of 5%. We also randomize the placement of the sensors in the training set. While in the nominal case, the first of the symmetrically placed sensors is placed on the local x-axis, we randomly sample an offset angle $\varphi_{\text{off}} \sim \mathcal{U}\left(0, \frac{2\pi}{n_{s_i}}\right)$ for each training sample, where $n_{s_i} = 3$ is the number of sensors per segment. Finally, we also randomly sample the radial displacement of the sensors from the center with $d_{s,r} \sim \mathcal{U}(8.7, 17.3)$ mm and consider a tilting of the sensors (*i.e.* a rotation around the tangential axis) with $\psi_s \sim \mathcal{U}(-20^\circ, 20^\circ)$. Before training, we randomly split off 30% of the training set for validation purposes.

We conducted a selection study involving hyperparameters and feed-forward neural network architectures (number of layers, nodes, and nonlinear activation layer types) on the validation set. In particular, we aimed to generate a smooth loss landscape to improve the gradient descent convergence leading us to employ a stochastic gradient descent (SGD) optimizer in conjunction with the stochastic weight averaging (SWA)[27] strategy. The neural network itself has 18 layers in total and contains, after an initial 1D batch norm layer, four blocks and is concluded with a fully-connected layer at the end. Each block consists of a dropout with a probability of 1%, a linear layer, a ReLU, and a 1D batch norm layer. The hidden state is first increased to 96 nodes, then to 256 nodes, and finally reduced again to 64 and 24 nodes. We minimize a mean squared error (MSE) loss of the neural network prediction $\hat{u}_j(t)$ for 250 epochs with batch size 650 while setting an initial learning rate of 0.18 for the cosine annealing learning rate scheduler.[28] The SWA[27] strategy is started after 125 epochs. We train the neural network such that all sensors in the ith segment share the same weights π_i. When the training is finished, we select the model from the epoch with the lowest validation loss and save it for later testing.

3.3 Optimization

We optimize the configuration variables $\hat{q}_i = (\hat{\Delta}_{x,i}, \hat{\Delta}_{y,i}, \hat{\delta L}_i)^{\mathrm{T}}$ for each segment to minimize the sensor measurement prediction loss as defined in (9). The optimization strategy solely relies on gradient descent and uses the best solution from the last time step $\hat{q}^*(t-1)$ as an initialization $\hat{q}_0(t)$. For the first time step of the trajectory, we initialize with the ground truth. We use a step size $\gamma = 3.5 \times 10^{-4}$, 3×10^{-3}, and 2×10^{-3} during gradient descent for a one-segment, two-segment and three-segment robot respectively. For all robots with more than one segment, the step size is reduced by a factor of ten when optimizing the elongation. The momentum μ is 0.3 for all trials and we perform 20 gradient descent iterations for each time step.

3.4 Evaluation

We evaluate the performance of our method at estimating the configuration of the segment by computing a relative root mean-squared error (RMSE) metric with respect to the ground-truth configuration $q(t)$ for each configuration variable separately

$$e_{q_\star} = \frac{\sqrt{\sum_{t=0}^{n_t} (\hat{q}_\star(t) - q_\star(t))^2}}{\sqrt{n_t}\, D_{q_\star}}, \tag{13}$$

where n_t is the number of discrete time-steps and $D_{q_\star}$ is the dynamic range of each configuration variable between the maximum and minimum value in the test set. All simulations are evaluated on a full lemniscate trajectory, which is very similar to what is plotted based on experimental data in Fig. 9(d). The maximum bending angle of this trajectory is 45 and the elongation of the segment follows a cosine wave with a minimum and maximum at 1.25% and 3.75% respectively.

3.5 Results

We present all simulation results in Table 1. In the first section, robots with one to three segments, which all exhibit a nominal sensor placement, are considered separately. Neural networks are trained separately for each of these robots, as the input dimension needs to be adjusted to the number of magnets. The results show that the method works well for robots with 1–3 segments. It can be observed that the estimation error is usually lower for the distal segment(s).

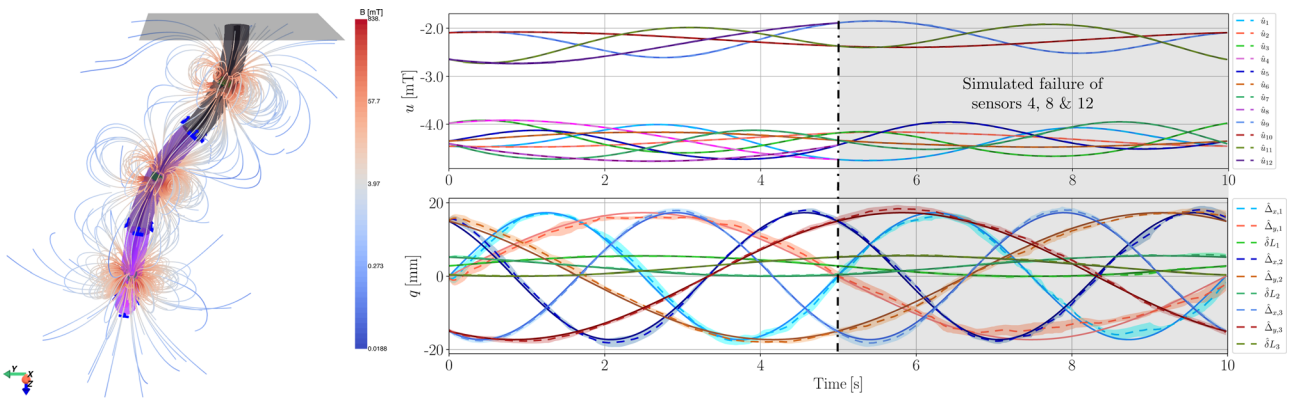
Next, the number of sensors n_s is varied for a three-segment robot. We always apply a symmetrical placement of the sensors in the tip plane of each segment. In the first trial, two sensors are mounted in the tip plane of each segment opposite to each other (6 sensors in total). While the bending along $\Delta_{x,i}$ and the elongation of the segments δL_i can still be estimated, the setup does not contain sufficient information to accurately determine the bending into $\Delta_{y,i}$. While the nominal case of nine sensors in total already achieves relative RMSEs in the range of 1.6% to 6%, the proprioception performance can be slightly improved by adding more sensors. We emphasize that the neural networks are not retrained when adding or removing sensors. In Fig. 4(b) and 5, we plot the proprioceptive performance of a three-segment robot with four sensors per segment. Then, we simulated a failure of the 4th, 8th, and 12th sensor at 5s by removing these sensor measurements from (11) (*i.e.* the gradient descent). Our method is able to adapt without re-training and leverage the nominal redundancy of sensor measurements, as three sensors per segment are sufficient for shape estimation.

Then, two adjusted sensor placements are investigated, while keeping the neural network weights constant. First, the sensors are tilted from the nominal case of pointing along the local z-axis by $\psi_s = 10^\circ$ towards the inside. In a separate simulation, the sensors are moved radially from nominally 13 mm to 16 mm. As the results show, the configuration of all three segments can still be estimated accurately with a mean error of 3.3%.

Finally, an earth's magnetic field of magnitude 0.065 mT is added. A separate neural network is trained on a training set with randomly sampled magnetic field vector directions n_e. As the last section of Table 1 demonstrates, the methodology is

Table 1 Simulation results: first, we report the absolute root mean-squared error (RMSE) e_u of sensor measurement predictions on the test set averaged across all sensors on the robot. Next, we state the relative RMSE [%] of each robot configuration estimate. All models are trained for each segment separately on a trajectory with randomly sampled configurations and sensor kinematic parameters and evaluated on a lemniscate trajectory. The first section applies our methodology to robots consisting of a different number of segments n_b with three sensors attached to the tip of each segment. The number of sensors is varied in the second section for a three-segment robot with all sensors placed symmetrically. The third set of trials then investigates how robust the method is to changing the kinematic parameters of the sensors, such as the tilting angle of the sensors ψ_s and the radial distance of the sensors $d_{s,r}$. Finally, we apply the earth's magnetic field along different cardinal directions in the inertial frame. The RMSE of the configuration estimates is normalized with the range of the dataset for each configuration variable as stated in (13). We report the error as mean ± stdev and compute the statistics over three different random seeds. The random seed determines at the start of the training the initialization of the neural network weights

Simulation	Specifications	e_u [mT]	$e_{\Delta_{x,1}}$ [%]	$e_{\Delta_{y,1}}$ [%]	$e_{\delta L_1}$ [%]	$e_{\Delta_{x,2}}$ [%]	$e_{\Delta_{y,2}}$ [%]	$e_{\delta L_2}$ [%]	$e_{\Delta_{x,3}}$ [%]	$e_{\Delta_{y,3}}$ [%]	$e_{\delta L_3}$ [%]
Variation of # of segments	$n_b = 1$, $n_s = 3$	0.015 ± 0.002	1.7 ± 1.0	1.8 ± 1.1	2.8 ± 0.6	—	—	—	—	—	—
	$n_b = 2$, $n_s = 6$	0.015 ± 0.001	4.2 ± 1.3	3.9 ± 0.6	6.3 ± 1.3	2.3 ± 0.6	2.2 ± 0.3	4.1 ± 1.3	—	—	—
	$n_b = 3$, $n_s = 9$	0.015 ± 0.002	3.7 ± 2.1	4.7 ± 1.6	6.0 ± 2.0	2.6 ± 1.6	2.5 ± 1.5	5.5 ± 1.6	1.6 ± 1.2	1.6 ± 1.1	2.7 ± 1.4
Variation of # of sensors	$n_b = 3$, $n_s = 6$	0.015 ± 0.002	3.9 ± 1.2	24.4 ± 2.7	8.4 ± 2.6	2.5 ± 1.3	53.2 ± 5.0	6.0 ± 1.7	1.5 ± 1.1	52.3 ± 7.3	3.1 ± 1.0
	$n_b = 3$, $n_s = 9$	0.015 ± 0.002	3.7 ± 2.1	4.7 ± 1.6	6.0 ± 2.0	2.6 ± 1.6	2.5 ± 1.5	5.5 ± 1.6	1.6 ± 1.2	1.6 ± 1.1	2.7 ± 1.4
	$n_b = 3$, $n_s = 12$	0.015 ± 0.001	3.7 ± 1.6	3.9 ± 1.3	4.6 ± 2.4	2.6 ± 1.3	2.6 ± 1.2	4.4 ± 1.9	1.6 ± 1.2	1.5 ± 1.3	2.5 ± 1.6
	$n_b = 3$, $n_s = 18$	0.016 ± 0.002	3.2 ± 1.5	3.5 ± 1.3	4.2 ± 1.8	2.4 ± 1.3	2.5 ± 1.2	4.4 ± 1.8	1.5 ± 1.3	1.3 ± 1.1	2.5 ± 1.5
Nominal	$n_b = 3$, $n_s = 9$	0.015 ± 0.002	3.7 ± 2.1	4.7 ± 1.6	6.0 ± 2.0	2.6 ± 1.6	2.5 ± 1.5	5.5 ± 1.6	1.6 ± 1.2	1.6 ± 1.1	2.7 ± 1.4
Sensors tilted	$\psi_s = 10^\circ$	0.015 ± 0.002	8.1 ± 4.0	6.4 ± 2.0	4.7 ± 0.7	5.1 ± 2.3	3.8 ± 0.8	5.6 ± 2.4	2.8 ± 1.2	1.9 ± 0.8	4.1 ± 0.6
Sensors shifted	$d_{s,r} = 16$ mm	0.017 ± 0.003	3.4 ± 1.5	4.0 ± 1.3	6.2 ± 2.5	2.6 ± 1.0	2.8 ± 1.8	6.7 ± 0.9	1.6 ± 1.2	1.6 ± 1.0	4.0 ± 2.0
Earth magnetic field	$n_e = (1,0,0)$	0.012 ± 0.002	2.0 ± 0.4	2.4 ± 1.1	4.3 ± 1.3	1.8 ± 0.7	1.9 ± 0.3	3.6 ± 2.0	1.5 ± 0.3	1.6 ± 0.4	3.2 ± 0.9
	$n_e = (0,1,0)$	0.012 ± 0.002	1.9 ± 0.5	2.5 ± 0.8	3.8 ± 1.1	1.9 ± 0.5	2.0 ± 0.2	3.4 ± 1.6	1.5 ± 0.3	1.6 ± 0.4	3.3 ± 0.8
	$n_e = (0,0,1)$	0.012 ± 0.001	2.0 ± 0.7	2.6 ± 0.9	4.1 ± 0.9	1.6 ± 0.3	1.7 ± 0.3	4.9 ± 1.5	1.5 ± 0.5	1.4 ± 0.1	3.9 ± 0.3



(a) Magnetic field for a three-segment robot (b) Simulated sensor failure

Fig. 4 Panel (a): Simulated magnetic field of a robot with three segments. The blue arrows mark the measurement directions of the sensors, the magnets are rendered in green and the three segments are visualized in a color sequence from black to violet. The magnetic flux density B is shown *via* streamlines and logarithmic coloring. Panel (b): Sensor measurement predictions (top) and configuration estimates (bottom) for a three-segment robot with twelve sensors nominally. We simulate a sensor failure of the 4th, 8th, and 12th sensors after 5 s of trajectory time by removing the measurements of these sensors from the gradient descent. This can be compensated automatically by the redundancy of the sensor configuration. We plot the ground-truth values u and q in solid, the estimate $\hat{u}$ and $\hat{q}$ as a mean over three random seeds with dashed lines and the standard deviation as an error band.

able to adapt to any earth's magnetic field direction by leveraging the λ_j input parameter.

4 Affine curvature simulations

To demonstrate the efficacy of the proposed method for higher-order kinematic models than PCC, we also conduct simulations of an affine curvature soft robot. The affine curvature kinematic parametrization[24] has been shown capable of representing the shape of soft tentacles[25,29] and provides a continuous function $\kappa = \kappa_0 + \kappa_1 v$ to describe the curvature of the soft robot, where κ_0, κ_1 are two configuration variables and $v \in [0,1]$ is the backbone coordinate. We allow for movement in 3D space by also specifying an azimuth bending angle ϕ and the elongation δL. Please refer to Appendix A.2 for more implementation details about the affine curvature model.

4.1 Simulation setup

We use the same simulation setup as described in Section 3.1. Therefore, we report in the following only the implemented modifications to simulate an affine curvature soft robot in Magpylib.[26] Namely, we consider one affine curvature segment

(a) $t = 0$ s (b) $t = 2$ s (c) $t = 4$ s (d) $t = 6$ s (e) $t = 8$ s (f) $t = 10$ s

Fig. 5 Sequence of stills for simulated sensor failure as shown in Fig. 4(b): the number of sensors per segment, which are rendered in blue, is reduced from four to three at t = 5 s. We visualize the ground-truth shape of the soft robot with full opacity and the estimated configuration with slight transparency. The magnets are rendered in green and the three segments are visualized in a color sequence from black to violet.

of length $L_{0,i}$ = 200 mm with in total n_s = 9 magnetic sensors. The sensors are placed on three separate cylindrical planes at distances d_{s_a} of 0 mm, 100 mm, and 200 mm from the base of the robot. In each plane, the three sensors are spaced at an angle of 120° and at a radial distance d_{s_r} = 13 mm from the backbone, as it can be seen in Fig. 7. Two ring magnets are positioned at a distance of 50 mm and 150 mm from the robot's base respectively.

4.2 Prediction network and optimization

We simulate 120 000 random configurations of the affine curvature robot to generate the training set. For this purpose, we sample the configuration variables from uniform distributions: the affine curvature parameters $\kappa_0 \in \mathcal{U}(0, 0.942\,\mathrm{rad\,m^{-1}})$, $\kappa_1 \in \mathcal{U}(0, 3.770\,\mathrm{rad\,m^{-2}})$, the azimuth angle $\phi \in \mathcal{U}(0, 2\pi\,\mathrm{rad})$, and the elongation $\delta L \in \mathcal{U}(0, 6.6\,\mathrm{mm})$. Before training, we randomly split off 30% of the training set for validation purposes. We train a specialized neural network $f_{\pi_j(\xi_j)}$ for each sensor and use the same neural network architecture as in Section 3.2 with the exception of the addition of a final layer $y(x) = \mathrm{sign}(x)e^{|x|}$. The training runs for in total 250 epochs and uses the SWA[27] strategy with a learning rate of 0.01. All other training hyperparameters are the same as in Section 3.2.

The four configuration variables are optimized to minimize the loss between the predictions and simulated measurements of the nine sensors as defined in eqn (9). For this optimization procedure, we employ gradient descent running at 40Hz with step sizes of $\gamma_{\kappa_0} = 1$, $\gamma_{\kappa_1} = 5$, $\gamma_\phi = 1$, and $\gamma_{\delta L} = 2 \times 10^{-4}$. The momentum is set to μ = 0.3 and 20 iterations are performed at each time step.

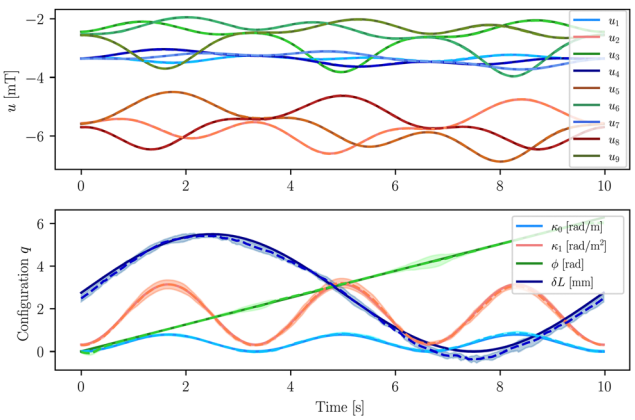
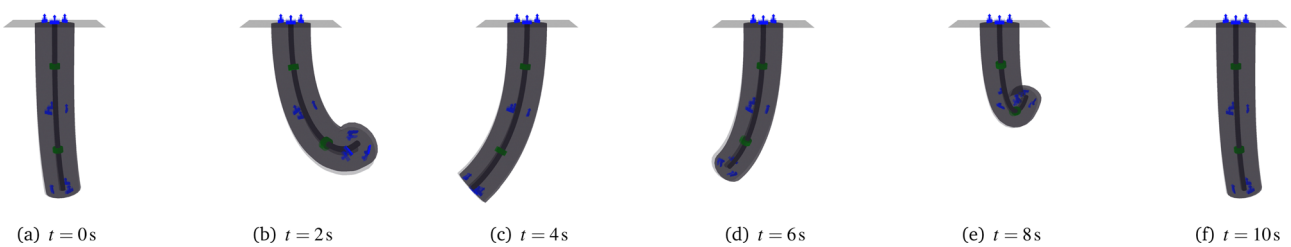


Fig. 6 Sensor measurement predictions (top) and configuration estimates (bottom) for an affine curvature segment with nine magnetic sensors. We plot the ground-truth values u and q in solid, the estimate $\hat{u}$ and $\hat{q}$ as a mean over three random seeds with dashed lines and the standard deviation as an error band. The random seed determines at the start of the training the initialization of the neural network weights.

4.3 Evaluation

We evaluate the trained model on a flower trajectory of duration 10 s and sample rate 40 Hz. The evaluation trajectory has the following characteristics: κ_0 is actuated by a sinusoidal wave of frequency 0.3Hz in the range $\left[0, \frac{\pi}{4}\,\mathrm{rad\,m^{-1}}\right]$. Similarly, κ_1 is also varied through a sinusoidal function of the same frequency and has a dynamic range of $[0.1\pi,\pi]$ rad m^{-2}. The azimuth angle ϕ is linearly scaled from 0 rad to 2π rad over the duration of the trajectory. Finally, δL follows a sinusoidal sequence of frequency 0.1 Hz in the range of [0,5.5] mm. We use the same evaluation metrics as first introduced in Section 3.4. We report the error as mean ± stdev and compute the statistics over three different random seeds. The random



(a) $t = 0$ s (b) $t = 2$ s (c) $t = 4$ s (d) $t = 6$ s (e) $t = 8$ s (f) $t = 10$ s

Fig. 7 Sequence of stills for a simulated affine curvature robot. We visualize the ground-truth shape of the soft robot with full opacity and the estimated configuration with slight transparency. The two magnets are rendered in green and the nine sensors in blue.

seed determines at the start of the training the initialization of the neural network weights.

4.4 Results

The trained neural networks achieve an RMSE error for predicting the magnetic sensor measurements of 0.025 ± 0.002 mT on the test set. When we run inference (see Fig. 6) on the flower trajectory, the configuration variables can be estimated with an absolute RMSE of $e_{\kappa_0} = 0.042 \pm 0.005$ rad m^{-1}, $e_{\kappa_1} = 0.11 \pm 0.04$ rad m^{-2}, $e_\phi = 0.08 \pm 0.02$ rad, and $e_{\delta L} = 0.001 \pm 0.001$ mm. We state the relative RMSE errors for the configuration estimates as $e_{\kappa_0} = 5.4 \pm 0.7\%$, $e_{\kappa_1} = 4.0 \pm 1.4\%$, $e_\phi = 1.3 \pm 0.4\%$, and $e_{\delta L} = 3.8 \pm 1.6\%$. In Fig. 7, we render at six different points along the trajectory the robot's shape according to the ground truth and estimated configurations respectively. The sequence qualitatively shows that our proposed method is able to estimate the affine curvature robot's shape very accurately.

5 Experiments

We verify the performance of our proposed proprioception method in experiments involving one soft robot segment with three magnetoresistive sensors attached to the tip. We aim to estimate the CC configuration $\hat{q} = (\Delta_{x,1}, \Delta_{y,1})^{\mathrm{T}} \in \mathbb{R}^2$ from the measured sensor values $u(t) \in \mathbb{R}^3$. We let the robot follow a diverse set of trajectories and evaluate the proprioception performance. After measuring the ground-truth pose of the tip of the segment with a motion capture system, we perform inverse kinematics with the closed-form solution reported by Della Santina *et al.*[30] and quantitatively compare the proprioceptive configuration estimate of the segment with the ground-truth configuration.

5.1 Robot design

We use a cylindrical, pneumatically-actuated soft robotic silicon segment of length $L_0 = 110$ mm and radius $d_1 = 22$ mm consisting of three independently inflatable cavities evenly spaced in the radial direction from the center line.[31] The proprioceptive sensing system is achieved by embedding one ring magnet in the backbone at a distance $d_{m_0} = 55$ mm from the base of the segment and three symmetrically-placed Magnetoresistive Sensors (MRSs) at the tip of the segment as visualized in Fig. 8. Although we use MRSs in our experimental setup for their high sensitivity,[32] this is not a strict condition and other sensor types measuring the magnetic field such as Hall-effect sensors can be combined with the methodology proposed in this paper too. For casting the silicone segment, we use a 3D-printed mold with a holder for the magnet which keeps it in place inside the segment. The magnet used is a neodymium ring of grade N50 with a thickness and inner diameter of 6 mm each, and an outer diameter of 12 mm. The MRSs of type Honeywell HMC1021Z are integrated into a printed circuit board (PCB) and output a voltage difference of 50 mV mT^{-1}. The three sensors are equally spaced at 120° from each other and are placed at a radial distance of $d_{s,r} = 13$ mm, and at a longitudinal distance of $d_{s,a} = 116$ mm from the base in a straight configuration. For each sensor, we implemented a Set/Reset and an amplification circuit on the PCB. The Set/Reset circuit is used for calibration of the sensor by re-aligning the magnetic domains. After amplification of the sensor output by a factor of 100, the output of the sensors is processed with a Texas Instruments ADS1115 module resulting in a digital signal of 16bit resolution. All sensor measurements u are in the range [0 mV,2048 mV], which corresponds to magnetic flux densities of [0 mT,41 mT].

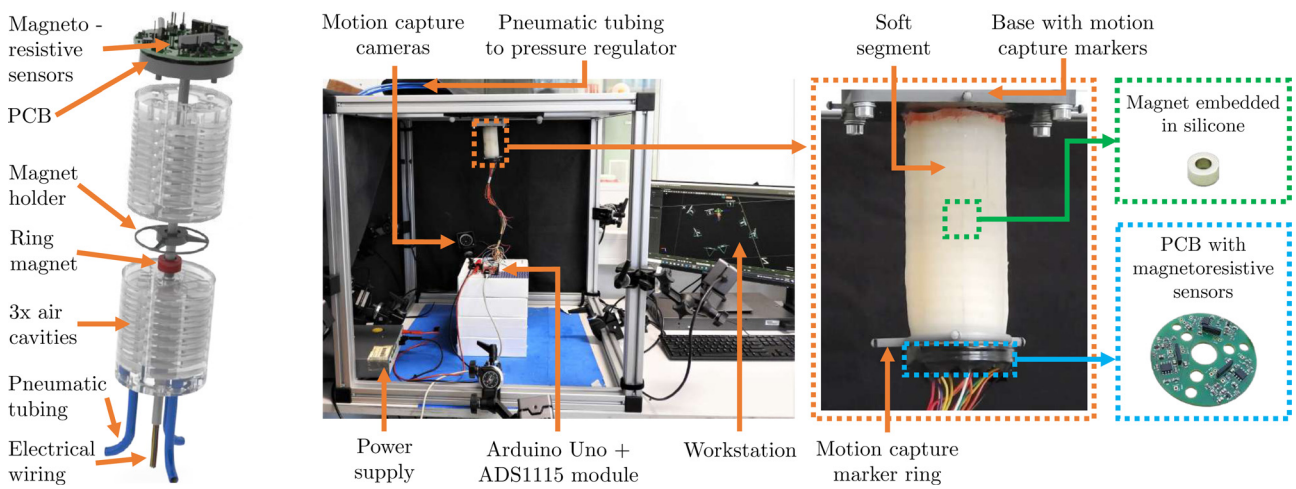


Fig. 8 Left: Exploded rendering visualizing the robot design with the three magnetoresistive sensors integrated into a PCB at the tip of the segment. The electrical wires from the PCB are passed through the backbone to the base. A ring magnet attached to a 3D-printed holder is integrated into the backbone at a half-segment-length distance from the proximal end. The air chambers of the segment are connected to a pressure regulator *via* tubing glued at the base of the segment. Right: Experimental setup with the soft robot segment attached in tip-down configuration to the motion capture cage.

5.2 Experimental setup

We conducted our experiments in a lab environment with the base of the soft robot segment mounted in a tip-down configuration to a cubical cage as shown in Fig. 8. A 3D-printed ring with four reflective markers is mounted on the tip of the segment. Eight motion capture cameras are attached to the cage tracking at 40 Hz the 3D pose of the ring. We transform the pose measurements of the tip to the base frame of the robot and compute the closed-form inverse kinematics[30] to receive a ground-truth configuration estimate $q(t) \in \mathbb{R}^2$. Each of the three pneumatic chambers of the segment is connected *via* tubing to a separate valve of a proportional pressure regulator operated at 100 Hz. We read out the analog signals of the magnetoresistive sensors with an Arduino Uno at 40 Hz and save them for later offline processing. We temporally align the motion capture and the magnetic sensor data by detecting the initial extension of the robot with a suitable threshold. The sensor noise is determined for both an unelongated straight configuration, and during fully inflated bending. Here the standard deviations of the white noise are 0.24 mV and 3.55 mV, which normalizes to 0.03% and 2% of the dynamic range respectively. Furthermore, we identify the earth's magnetic field direction in the base from as $\hat{n}_e = (-0.311, -0.234, 0.921)^T$ using a compass and the world magnetic model (WMM).[33]

5.3 Pneumatic actuation and trajectories

We consider, as visualized in Fig. 9, six continuous actuation sequences in this paper: random configuration way-points which are connected through linear interpolation (T0), planar side bending (T1), the tip following a half lemniscate (T2) and full lemniscate (T3), a spiral with constant linear velocity[34] (T4) and finally a flower-shape (T5). We define our trajectories as wrenches $\tau_{xyz} = [\tau_x \ \tau_y]^T$ on the tip of the segment in Cartesian space, where τ_x and τ_y cause bending around the local x- and y-axis of the tip respectively. The pressures we command from the pressure regulator are given by inversely evaluating the force produced at the center of pressure at the tip of the segment for each chamber for a given chamber pressure.[35] All actuation sequences are preceded by first applying an offset pressure of 225 mBar in all chambers, which causes a near-constant elongation of the segment. The peak pressure, which causes maximum bending, is set for all trajectories to 450 mBar.

The 1D bending (T1), half lemniscate (T2) and full lemniscate (T3) are all executed periodically with a period of 5 s, 10 s, and 10 s respectively. Trajectories T0 and T4 are characterized by a constant velocity in torque-space of 0.025 N m s^{-1} and 0.0125 kN m s^{-1} respectively. The flower trajectory T5 can be described as periodic 1D bending with a linearly changing azimuth angle. It exhibits an angular velocity of 0.0126 rad s^{-1} and a period of 10 s for the bending, which results in 50 bending cycles per circumnavigation. While the random configuration set points of T0 are recorded for 200 s, T1, T2 and T3 have total a duration of 90s each, and the spiral T4 and flower T5 last for 120 s and 1500 s respectively. We split off the final 20% of all datasets as a test set.

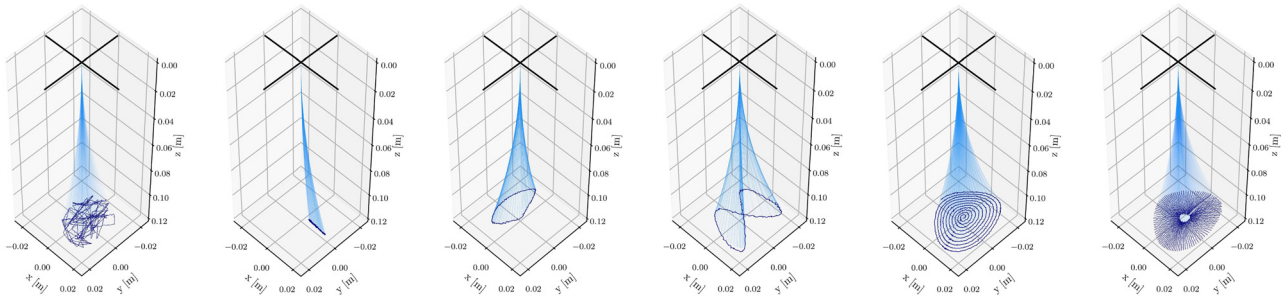
5.4 Prediction network and optimization

We use the same neural network architecture and training procedure as in Section 3.2, but with an adjusted initial learning rate of 5×10^{-5} and train the model separately for each sensor on the 1200 s long T5/flower trajectory, which results in 48000 training samples for each neural network.

We optimize the configuration variables Δ_x and Δ_y for the one segment to minimize the sensor measurement prediction loss as defined in (9). The optimization strategy solely relies on gradient descent running at 40 Hz with a step size $\gamma = 1.5 \times 10^{-8}$ and momentum $\mu = 0.2$. We visualize a sample loss landscape in Fig. 10(a).

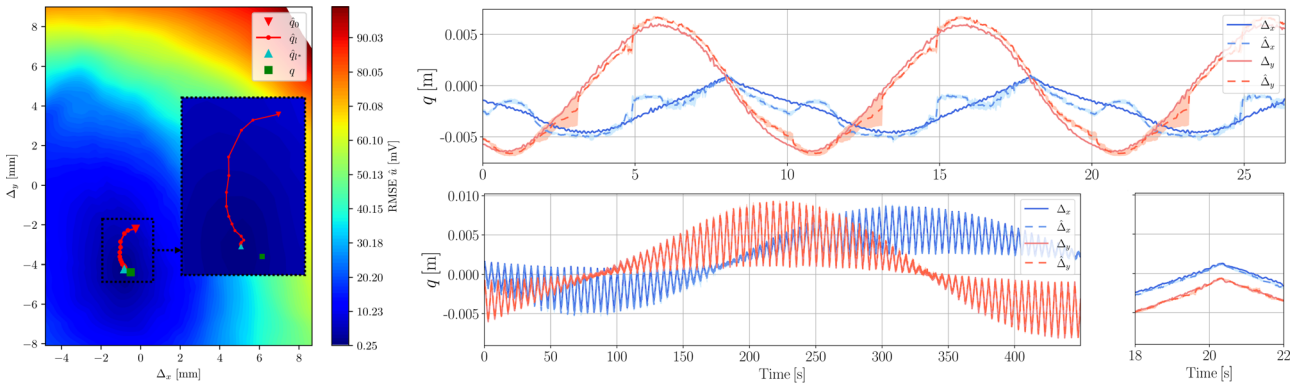
5.5 Results

First, we quantify the performance of the neural network predicting the sensor measurements $\hat{u}(t)$ for a known, ground-truth configuration $q(t)$, which we report in Table 2. We observe, that the relative RMSE lies between 0.6% and 4.6% of the range of the respective datasets with a mean of 3.1%. As expected, the predictions are generally the most accurate when evaluated on a trajectory of the same type as the neural network was trained on (T5). Next, we analyze the proprioception performance on the same trajectories. We report relative RMSEs between 1.9% and 13.6% for estimating the bending of the robot. Additionally, we visualize the configuration estimates for two



(a) **T0:** random setpoints (b) **T1:** 1D bending (c) **T2:** half lemniscate (d) **T3:** full lemniscate (e) **T4:** spiral (f) **T5:** flower

Fig. 9 Trajectories used during the experiments. We plot the shape of the segment under PCC approximation in light blue and the position of the tip of the segment in dark blue.

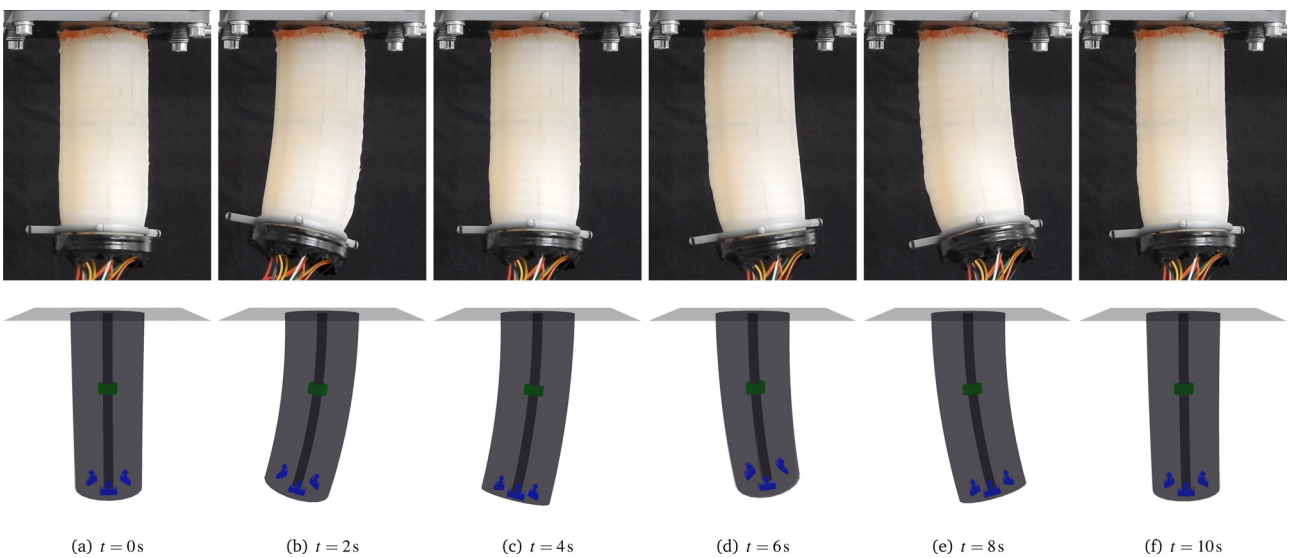


(a) Loss landscape (b) Experimental results for T2 (top) and T5 (bottom)

Fig. 10 Panel (a): Sample loss landscape for optimization of Δ_x and Δ_y on T5. With the hue, we visualize the RMSE of the sensor measurement prediction $\hat{u}$ for a given configuration $\hat{q} = (\Delta_x, \Delta_y)^\mathsf{T}$. Additionally, we denote the initial configuration estimate with $\hat{q}_0$, the trajectory of the gradient descent with $\hat{q}_l$, the optimal configuration with $\hat{q}$, and the ground truth with q. Panel (b) top: Proprioception on the test set of T2 using a model trained on T5. Panel (b) bottom: Configuration estimates for a model trained and evaluated on separated parts of trajectory 5. We plot the ground-truth configuration q in solid, the estimate $\hat{q}$ as a mean over three random seeds with dashed lines, and the standard deviation as an error band. The bottom right plot zooms onto a selected part of T5 (*e.g.* 18 s to 22 s) to more clearly visually distinguish the dashed lines from the solid lines.

Table 2 Experimental results: absolute [mm] and relative RMSE [%] of sensor measurement predictions and robot configuration estimates for various trajectories. The RMSE is normalized with the range of the dataset for u and each configuration variable respectively as stated in eqn (13). We report the error as mean ± stdev and compute the statistics over three different random seeds. The random seed determines at the start of the training the initialization of the neural network weights

Trajectory	e_u [mV]	e_u [%]	e_{Δ_x} [mm]	e_{Δ_x} [%]	e_{Δ_y} [mm]	e_{Δ_y} [%]
T5.train → T0.test	9.90 ± 0.90	3.90 ± 0.40	0.37 ± 0.06	3.70 ± 0.60	0.44 ± 0.01	3.50 ± 0.10
T5.train → T1.test	8.80 ± 0.30	4.40 ± 0.10	0.36 ± 0.08	6.50 ± 1.50	0.43 ± 0.05	—
T5.train → T2.test	11.10 ± 0.30	4.30 ± 0.10	0.78 ± 0.03	13.60 ± 0.60	0.74 ± 0.23	5.90 ± 1.80
T5.train → T3.test	12.30 ± 0.30	4.60 ± 0.10	0.52 ± 0.06	4.50 ± 0.50	0.47 ± 0.03	3.10 ± 0.20
T5.train → T4	3.01 ± 0.05	1.08 ± 0.02	0.33 ± 0.02	2.40 ± 0.10	0.52 ± 0.06	3.10 ± 0.40
T5.train → T5.test	1.60 ± 0.10	0.58 ± 0.05	0.24 ± 0.05	1.90 ± 0.30	0.24 ± 0.01	1.40 ± 0.05



(a) $t = 0$ s (b) $t = 2$ s (c) $t = 4$ s (d) $t = 6$ s (e) $t = 8$ s (f) $t = 10$ s

Fig. 11 Sequence of stills for inference on T3 (full lemniscate) of a model trained on T5. The top row shows camera recordings of an external view and the sequence in the bottom row consists of renderings of the ground-truth state (in full opacity) and the estimated segment shape (slightly transparent). The sensors are colored in blue and the ring magnet integrated into the backbone is shown in green.

trajectories: in the top-right of Fig. 10(b), we run inference for our trained model on T2. While the proprioception estimate tracks the general shape of the trajectory well, the optimization, particularly for Δ_y, gets trapped in local minima sometimes leading to periods of higher error. Next, we consider a model trained and evaluated on separated sets of the T5 trajectory. The configuration estimate tracks the ground truth very well as can be seen in the bottom right. Finally, we present a sequence of stills based on camera views and renderings of the soft segment for inference on T3 in Fig. 11.

6 Conclusion

This work proposed a sensing strategy for soft-bodied robots that relies on multiple magnetic sensors embedded directly in the robot. Thanks to a novel kinematics-aware neural architecture, we can simultaneously use information coming from all the sensors to reliably reconstruct the full robot shape. The decoupling of the kinematics from the learned sensor measurement predictor allows modifications to the placement of the sensors without requiring a re-training of the neural network. The proposed method is agnostic to the used kinematic state parametrization, which we verified in simulations using either PCC or affine curvature models. Extensive experiments with a soft segment showed that a model can be trained on one trajectory type and then be used for inference on a variety of other trajectories in the same workspace. In future work, we will validate the proposed proprioception methodology to execute closed-loop control. We also invite future research studying the optimal placement of sensors in continuum soft robots, where the optimization procedure might take advantage of the gradients provided by our algorithm.

Author contributions

T. B., M. K. H, M. S., and D. V. D. T contributed equally to this work. C. D. S., and R. B. conceived and led the project. M. S. invented the proprioception algorithm and implemented all simulations. T. B., M. K. H, J. N., and D. V. D. T designed the strategy to integrate sensors and magnets into the robot. T. B., M. K. H, J. N., and D. V. D. T. developed the PCB and sensing electronics. T. B., M. K. H, and D. V. D. T. fabricated the soft robot. T. B., M. K. H, J. N., and D. V. D. T. implemented the experimental data processing pipeline. T. B., M. K. H., M. S., and D. V. D. T. conducted experiments. T. B., M. K. H., M. S., and D. V. D. T. tuned the learning strategy and developed suitable trajectories. T. B., M. K. H., M. S., and D. V. D. T. wrote the manuscript. C. D. S., and R. B. revised the manuscript.

Conflicts of interest

There are no conflicts to declare.

Appendices

A Kinematics

A kinematic description provides us with the forward kinematic transformation $T^{s}_{i-1}(q_i,s)$ from base frame $\{S_{i-1}\}$ at the proximal end of the ith segment to the local frame $\{S_v\}$ at a coordinate $v \in [0,1]$ for a given configuration q_i. Furthermore, the tip frame of the ith segment located at the coordinate $v = 1$ is denoted as $\{S_i\}$, as shown in Fig. 2(b).

A.1 Δ-Parametrization of the piecewise constant curvature kinematics

Under the PCC hypothesis, the shape of each segment i of length L_i and radius d_i can be fully parameterized through three variables

$$q_i = [\,\Delta_{x,i} \quad \Delta_{y,i} \quad \delta L_i\,]^{\mathrm{T}} \in \mathbb{R}^3 \tag{14}$$

where $\Delta_{x,i}$ and $\Delta_{y,i}$ represent bending into the local x and y directions respectively and δL_i defines the elongation of the segment. The base frame of segment i is referred to as $\{S_{i-1}\}$ as stated in Fig. 2(b). Given q_i, a homogeneous transformation $T^{v}_{i-1}(q_i,v)$ to the point frame $\{S_v\}$ is available

$$R^{v}_{i-1}(q_i,v) = \begin{bmatrix} 1+\frac{\Delta_{x,i}^2}{\Delta_i^2}(C_v-1) & \frac{\Delta_{x,i}\Delta_{y,i}}{\Delta_i^2}(C_v-1) & \frac{\Delta_{x,i}}{\Delta_i}S_v \\ \frac{\Delta_{x,i}\Delta_{y,i}}{\Delta_i^2}(C_v-1) & 1+\frac{\Delta_{y,i}^2}{\Delta_i^2}(C_v-1) & \frac{\Delta_{y,i}}{\Delta_i}S_v \\ \frac{-\Delta_{x,i}}{\Delta_i}S_v & \frac{-\Delta_{y,i}}{\Delta_i}S_v & C_v \end{bmatrix},$$

$$p^{i}_{i-1}(q_i,v) = \frac{d_i(L_{0,i}+\delta L_i)}{v\Delta_i^2}[\,\Delta_{x,i}(1-C_v) \quad \Delta_{y,i}(1-C_v) \quad \Delta_i S_v,\,]^{\mathrm{T}} \tag{15}$$

where R^{v}_{i-1}, $p^{i}_{i-1}(q_i,v)$ denote the rotation matrix and translation vector respectively. We substituted $\Delta_i = \sqrt{\Delta_{x,i}^2 + \Delta_{y,i}^2}$, $S_v = \sin\left(\frac{v\Delta_i}{d_i}\right)$, and $C_v = \cos\left(\frac{v\Delta_i}{d_i}\right)$ for conciseness.

A.2 Affine curvature kinematics

The affine curvature hypothesis[24,25] models the bending of the soft segment to be conforming to the affine function $\kappa(t,v) = \kappa_0(t) + \kappa_1(t)v$, where κ describes the local curvature of the backbone at the coordinate $v \in [0,1]$ along the segment and $\kappa_0(t)$, $\kappa_1(t)$ are the zero-order and first-order term of the curvature polynomial respectively.[36] Specifically, we implement the recently proposed extension to 3D environments,[25] which specifies an azimuth angle of the bending direction $\phi(t)$ and additionally allows for an elongation $\delta L(t)$ of the segment. Accordingly, the configuration of the ith segment is described at any point in time by

$$q_i = [\,\kappa_{0,i} \quad \kappa_{1,i} \quad \phi_i \quad \delta L_i\,]^{\mathrm{T}} \in \mathbb{R}^4. \tag{16}$$

Now that the configuration space is defined, we aim to find a description of the forward kinematics. Firstly, the bending

angle $\theta_i(q,v)$ is found by integrating the curvature

$$\theta(q,v) = \int_{v'=0}^{v} \kappa_i(q,v')\mathrm{d}v' = \kappa_{0,i}v + \kappa_{1,i}\frac{v^2}{2}. \tag{17}$$

The rotation to the frame $\{S_v\}$ can then be easily determined with $R^i_{i-1}(q,v) = R_{\phi_i}(q,v)\,R_\theta(q,v)R^{\mathrm{T}}_{\phi_i}(q,v)$. After substituting $S_\cdot = \sin(\cdot)$, $C_\cdot = \cos(\cdot)$ for conciseness, we state the homogeneous transformation as

$$R^v_{i-1}(q,v) = \begin{bmatrix} S_{\phi_i}{}^2 C_{\theta_v} + C_{\phi_i}{}^2 & -S_{\phi_i}C_{\phi_i}C_{\theta_i} + S_{\phi_i}C_{\phi_i} & S_{\phi_i}S_{\theta_v} \\ -S_{\phi_i}C_{\phi_i}C_{\theta_i} + S_{\phi_i}C_{\phi_i} & S_{\phi_i}{}^2 + C_{\phi_i}{}^2 C_{\theta_v} & -S_{\theta_v}C_{\phi_i} \\ -S_{\phi_i}S_{\theta_v} & S_{\theta_v}C_{\phi_i} & C_{\theta_i} \end{bmatrix},$$

$$p^v_{i-1}(q,v) = (L_{0,i} + \delta L_i)\begin{pmatrix} \int_{v'=0}^{v} \sin(\theta(q,v'))\mathrm{d}v' \sin(\phi_i) \\ -\int_{v'=0}^{v} \sin(\theta(q,v'))\mathrm{d}v' \cos(\phi_i) \\ \int_{v'=0}^{v} \cos(\theta(q,v'))\mathrm{d}v' \end{pmatrix}. \tag{18}$$

We choose to integrate the translational terms in (18) numerically with 101 sample points using the Torchquad[37] implementation of the Simpson's rule, which makes the forward kinematics fully and automatically differentiable.

Acknowledgements

This work has received funding under the European Union's Horizon Europe Programme from Project EMERGE – Grant Agreement No. 101070918. We would like to thank Ehsan Heseini, Jasper Insinger and Tom Salden from Delft University of Technology, NL for their guidance in designing the PCB.

References

1 J. Luong, P. Glick, A. Ong, M. S. DeVries, S. Sandin, E. W. Hawkes and M. T. Tolley, 2019 2nd IEEE International Conference on Soft Robotics (RoboSoft), 2019, pp. 801–807.
2 D. S. Shah, J. P. Powers, L. G. Tilton, S. Kriegman, J. Bongard and R. Kramer-Bottiglio, *Nat. Machine Intell.*, 2021, **3**, 51–59.
3 J. Hughes, C. Della Santina and D. Rus, 2020 3rd IEEE International Conference on Soft Robotics (RoboSoft), 2020, pp. 733–739.
4 C. Della Santina, M. G. Catalano and A. Bicchi, *Encyclopedia Robot.*, 2021, **489**, DOI: 10.1007/978-3-642-41610-1_146-2.
5 C. Armanini, C. Messer, A. T. Mathew, F. Boyer, C. Duriez and F. Renda, *arXiv*, 2021, preprint, arXiv:2112.03645, DOI: 10.48550/arXiv:2112.03645.
6 C. Della Santina, C. Duriez and D. Rus, *arXiv*, 2021, preprint, arXiv:2110.01358, DOI: 10.48550/arXiv:2110.01358.
7 H. Wang, M. Totaro and L. Beccai, *Adv. Sci.*, 2018, **5**, 1800541.
8 B. Shih, C. Christianson, K. Gillespie, S. Lee, J. Mayeda, Z. Huo and M. T. Tolley, *Front. Robot. AI*, 2019, **6**, 30.
9 R. K. Kramer, C. Majidi, R. Sahai and R. J. Wood, 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2011, pp. 1919-1926.
10 L. Scimeca, J. Hughes, P. Maiolino and F. Iida, *IEEE Robot. Autom. Lett.*, 2019, **4**, 2479–2484.
11 J. Shintake, Y. Piskarev, S. H. Jeong and D. Floreano, *Adv. Mater. Technol.*, 2018, **3**, 1700284.
12 S. Li, S. A. Awale, K. E. Bacher, T. J. Buchner, C. Della Santina, R. J. Wood and D. Rus, *Soft Robot.*, 2021, **9**(2), 324–336.
13 E. R. Rosi, M. Stölzle, F. Solari and C. Della Santina, 2022 IEEE 5th International Conference on Soft Robotics (RoboSoft), 2022, pp. 795-801.
14 S. Song, Z. Li, H. Yu and H. Ren, *IEEE Sens. J.*, 2015, **15**, 4565–4575.
15 S. Ozel, N. A. Keskin, D. Khea and C. D. Onal, *Sens. Actuators, A*, 2015, **236**, 349–356.
16 M. Luo, E. H. Skorina, W. Tao, F. Chen, S. Ozel, Y. Sun and C. D. Onal, *Soft Robot.*, 2017, **4**, 117–125.
17 H. Guo, F. Ju, Y. Cao, F. Qi, D. Bai, Y. Wang and B. Chen, *Sens. Actuators, A*, 2019, **285**, 519–530.
18 M. D. Mitchell, F. E. Hurley and C. D. Onal, 2021 IEEE 17th International Conference on Automation Science and Engineering (CASE), 2021, pp. 215–220.
19 R. L. Truby, C. Della Santina and D. Rus, *IEEE Robot. Autom. Lett.*, 2020, **5**, 3299–3306.
20 Z. Y. Ding, J. Y. Loo, V. M. Baskaran, S. G. Nurzaman and C. P. Tan, *IEEE Robot. Autom. Lett.*, 2021, **6**, 951–957.
21 G. Soter, A. Conn, H. Hauser and J. Rossiter, 2018 IEEE international conference on robotics and automation (ICRA), 2018, pp. 2448–2453.
22 T. G. Thuruthel, B. Shih, C. Laschi and M. T. Tolley, *Sci. Rob.*, 2019, **4**, eaav1488.
23 R. J. Webster III and B. A. Jones, *Int. J. Robot. Res.*, 2010, **29**, 1661–1683.
24 C. Della Santina, 2020 59th IEEE Conference on Decision and Control (CDC), 2020, pp. 4135–4142.
25 F. Stella, Q. Guan, J. Leng, C. Della Santina and J. Hughes, Piecewise Affine Curvature model: a reduced-order model for soft robot-environment interaction beyond PCC, 2022, https://arxiv.org/abs/2211.10188.
26 M. Ortner and L. G. Coliado Bandeira, *SoftwareX*, 2020, **11**, 100466, DOI: 10.1016/j.softx.2020.100466.
27 P. Izmailov, D. Podoprikhin, T. Garipov, D. Vetrov and A. G. Wilson, *arXiv*, 2018, preprint, arXiv:1803.05407, DOI: 10.48550/arXiv:1803.05407.
28 I. Loshchilov and F. Hutter, *arXiv*, 2016, preprint, arXiv:1608.03983, DOI: 10.48550/arXiv:1608.0398.
29 F. Stella, N. Obayashi, C. Della Santina and J. Hughes, *IEEE Robot. Autom. Lett.*, 2022, **7**, 11410–11417.

30 C. Della Santina, A. Bicchi and D. Rus, *IEEE Robot. Autom. Lett.*, 2020, **5**, 1001–1008.
31 A. D. Marchese, R. K. Katzschmann and D. Rus, *Soft Robot.*, 2015, **2**, 7–25.
32 R. Popovic, P. Drljaca and C. Schott, 2002 23rd International Conference on Microelectronics, 2002, pp. 55–58.
33 A. Chulliat, W. Brown, P. Alken, C. Beggan, M. Nair, G. Cox, A. Woods, S. Macmillan, B. Meyer and M. Paniccia, *The US/UK world magnetic model for 2020–2025*, British Geological Survey, 2020.
34 O. M. Carrasco-Zevallos, C. Viehland, B. Keller, R. P. McNabb, A. N. Kuo and J. A. Izatt, *Biomed. Opt. Express*, 2018, **9**, 5052–5070.
35 C. Della Santina, A. Bicchi and D. Rus, 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2019, pp. 6622–6629.
36 C. Della Santina and D. Rus, *IEEE Robot. Autom. Lett.*, 2019, **5**, 290–298.
37 P. Gómez, H. H. Toftevaag and G. Meoni, *J. Open Source Software*, 2021, **6**, 3439.

