



# Fault-Tolerant Six-DoF Pose Estimation for Tendon-Driven Continuum Mechanisms

Antonin Raffin\*, Bastian Deutschmann and Freck Stulp

German Aerospace Center (DLR), Robotics and Mechatronics Center, Weßling, Germany

We propose a fault-tolerant estimation technique for the six-DoF pose of a tendon-driven continuum mechanisms using machine learning. In contrast to previous estimation techniques, no deformation model is required, and the pose prediction is rather performed with polynomial regression. As only a few datapoints are required for the regression, several estimators are trained with structured occlusions of the available sensor information, and clustered into ensembles based on the available sensors. By computing the variance of one ensemble, the uncertainty in the prediction is monitored and, if the variance is above a threshold, sensor loss is detected and handled. Experiments on the humanoid neck of the DLR robot DAVID, demonstrate that the accuracy of the predicted pose is significantly improved, and a reliable prediction can still be performed using only 3 out of 8 sensors.

**Keywords:** pose estimation, fault-tolerant, data-driven, machine learning, continuum mechanism

## OPEN ACCESS

### Edited by:

Thomas George Thuruthel,  
University of Cambridge,  
United Kingdom

### Reviewed by:

Yasmin Ansari,  
Scuola Sant'Anna di Studi Avanzati,  
Italy  
Chaoyang Song,  
Southern University of Science and  
Technology, China

### \*Correspondence:

Antonin Raffin  
antonin.raffin@dlr.de

### Specialty section:

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

**Received:** 19 October 2020

**Accepted:** 12 January 2021

**Published:** 30 April 2021

### Citation:

Raffin A, Deutschmann B and Stulp F  
(2021) Fault-Tolerant Six-DoF Pose  
Estimation for Tendon-Driven  
Continuum Mechanisms.  
Front. Robot. AI 8:619238.  
doi: 10.3389/frobt.2021.619238

## 1 INTRODUCTION

Robotic systems with deliberately introduced elasticity have become a promising alternative to classical rigid robots, be it in manipulation or locomotion tasks, in humanoid or animoid robots. Apart from their favorable dynamic properties, passive compliance protects the actuators from peak forces or torques that arise from collisions, be they unexpected or intentional. In this context, the development of distinct joints arose termed continuum mechanism. They extend the current portfolio of joints with passive compliance and use a spring element made out of a continuously deformable soft material with a beam-like geometry. As an example, the neck joint of the humanoid robot David (Reinecke et al., 2016) is shown in **Figure 1**.

The application requires to control the pose of the top end of the continuum mechanism, termed end effector pose, along trajectories or toward equilibrium positions in its workspace. Rigorous models which relate encoder values of the actuation to end effector poses are computationally expensive and prone to model parameter uncertainties. As we are aiming for mobile systems, the need of external tracking system must be limited to calibration and evaluation of a pose estimation method. At deployment time, such pose estimation method must use onboard sensors only.

In this paper, we propose a data-driven approach to pose estimation for elastic structures, which also features uncertainty estimation and error-handling. The specific contributions, which also define the structure of the paper, are:

- A data-driven approach for real time six-DoF pose of tendon-driven continuum mechanism using few data points. The model-free estimator can be trained in the experiment and requires only a small number of measurements (**Section 4.1**).

- Uncertainty estimation to detect sensor failure (e.g., slack tendons). This is done by creating an ensemble of estimators, where each estimator takes only a subset of sensor information as an input, and monitoring the uncertainty (**Section 4.3**).
- A strategy to handle sensor by adapting the pose estimation. As soon as an anomaly is detected, we select the estimators not using the faulty sensors and continue predicting accurately the pose. To the best of our knowledge, this is the first work dealing with failure detection and handling in the context of continuum mechanisms (**Section 4.3**).
- Demonstrating the effectiveness of these methods on the elastic neck of the DLR robot DAVID. In particular, we demonstrate that the accuracy of the pose estimation is significantly improved, and that reliable predictions can still be made with only 3 out of 8 sensors (**Section 5**).

Before following this structure, we first formalize the problem statement in the next section.

## 2 PROBLEM STATEMENT

This work treats the estimation problem of the position and orientation (pose) of the upper platform of a tendon-driven continuum mechanism. The considered mechanism is depicted, in its current application in **Figure 1**, and a schematic drawing to illustrate the kinematics in **Figure 2**. It consists of a inertial fixed lower platform, a moving upper platform and a continuum structure in between. The inertial frame of reference is denoted  $\mathcal{B}$  and is attached to the base of the neck. The position of the upper platform is described by the origin of frame  $\mathcal{H}$ , expressed in  $\mathcal{B}$ , and is denoted as  ${}_{\mathcal{B}}\mathbf{r} = (x, y, z)^T$ . The orientation of frame  $\mathcal{H}$ , expressed in  $\mathcal{B}$ , is represented by the three Euler angles, denoted as  $\boldsymbol{\theta} = (\theta_x, \theta_y, \theta_z)^T$ , and combined with the transformation order

$$A_{\mathcal{B}\mathcal{H}} = \mathbf{R}(\theta_z)\mathbf{R}(\theta_y)\mathbf{R}(\theta_x) \quad (1)$$

where  $A_{\mathcal{B}\mathcal{H}} \in \mathbb{R}^{3 \times 3}$  incorporates the base vectors of frame  $\mathcal{H}$ . As the workspace of the system does not exceed  $\pm 90^\circ$  in any direction, no singular configurations of the Euler angles appear. The pose is summarized in the vector  $\mathbf{q} \in \mathbb{R}^6$ ,

$$\mathbf{q} = (x, y, z, \theta_x, \theta_y, \theta_z)^T \quad (2)$$

For actuation, tendons are connected to the upper platform. By putting tension on them, a loading is introduced onto the upper platform which deforms the continuum and initiates a motion. The tendons are routed alongside the continuum to the lower platform without touching, **Figure 1** indicated with red. In the lower platform, the tendons actuators are located. The incorporated position measurement  $\Delta \tilde{\mathbf{l}} \in \mathbb{R}^4$  allows the free length of each of the tendon to be measured, assuming a known initial length  $\mathbf{l}_{i,0} \in \mathbb{R}^4$ ,

$$\mathbf{l}_i = \mathbf{l}_{i,0} - \Delta \tilde{\mathbf{l}} \quad (3)$$

Furthermore, four additional length sensors are placed on the system indicated in green in **Figure 1**. They are also spanned from

the upper to the lower platform and provide sensor values  $\tilde{\mathbf{s}} \in \mathbb{R}^4$  which correspond linearly to their length,

$$\mathbf{l}_s = \mathbf{K}_s \tilde{\mathbf{s}} \quad (4)$$

with the constant calibration matrix  $\mathbf{K}_s \in \mathbb{R}^{4 \times 4}$ . In summary, the sensor information are included in  $\tilde{\mathbf{l}} = (\tilde{\mathbf{l}}_t^T, \tilde{\mathbf{l}}_s^T)^T \in \mathbb{R}^8$  and as the attachment and routing points of additional length sensors are known, a geometric model for the measured length is available,

$$l_i(\mathbf{q}) = \|\mathbf{r}_{p,i} - {}_{\mathcal{B}}\mathbf{r} - A_{\mathcal{B}\mathcal{H}}\mathbf{r}_i\|_2, \text{ for } i = 1 \dots 8 \quad (5)$$

In a prior publication (Deutschmann et al., 2019), the estimation problem was solved using the geometric model for the length sensors of **Eq. 5**. An online minimization of the error  $\mathbf{e}_l \in \mathbb{R}^8$  between the length measurements  $\tilde{\mathbf{l}}$  and the model  $\mathbf{l}(\mathbf{q})$  is carried out to find an estimate of the pose  $\hat{\mathbf{q}} \in \mathbb{R}^6$ ,

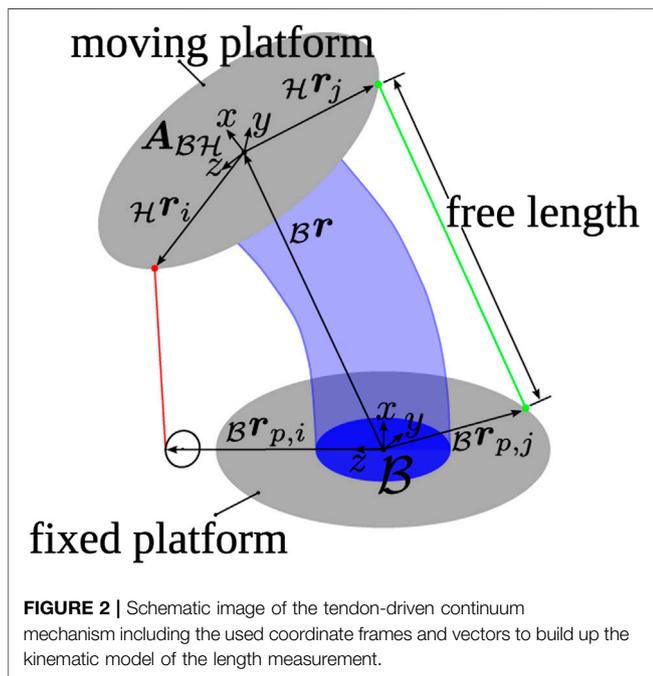
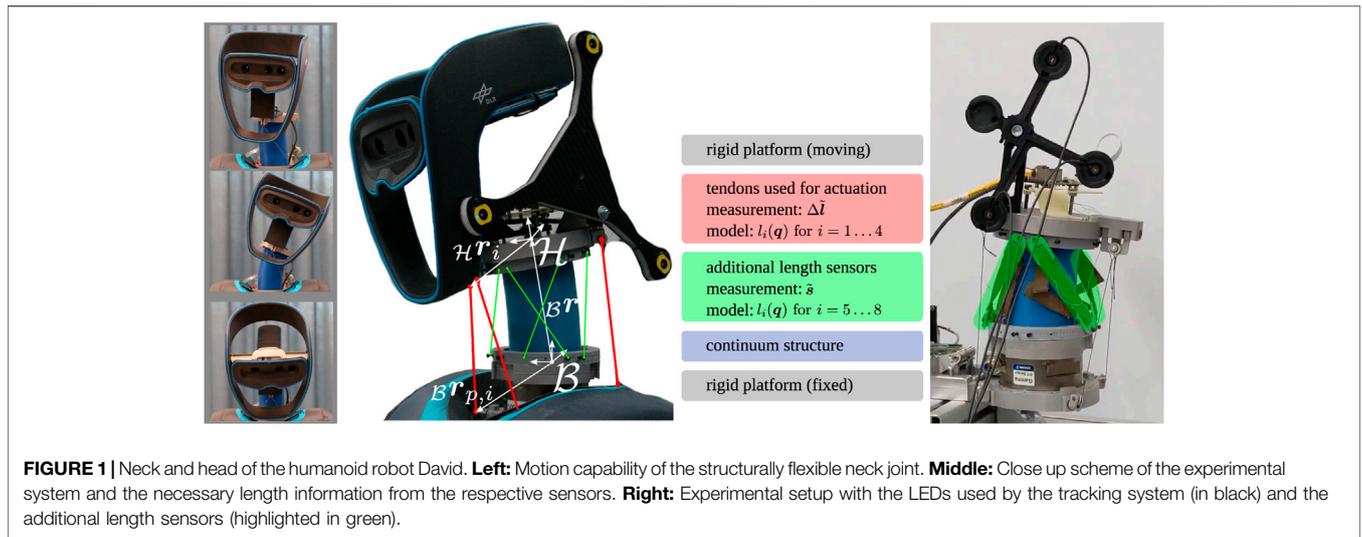
$$\min_{\hat{\mathbf{q}}} \|\tilde{\mathbf{l}} - \mathbf{l}(\hat{\mathbf{q}})\|_2^2 = \min_{\hat{\mathbf{q}}} \|\mathbf{e}_l\|_2^2 \quad (6)$$

Sources of error using this pose estimation technique are related to the following assumptions. First, it is assumed that the sensor readings in **Eqs 3, 4** correspond linearly to a length of the tendons or sensors. Second, the geometric model in **Eq. 5** relies on the accurate knowledge about the pulley kinematics of the tendons, and perfectly known attachment points and straight sensors. If these assumptions do not hold, the accuracy of the resulting pose which minimizes **Eq. 6** might be affected.

In tests on the real hardware, the pose could be predicted with a maximum estimation error of  $\pm 4.5$  mm for position and  $\pm 4^\circ$  orientation (Deutschmann et al., 2019). The drawbacks of this method are that a good initial estimate is required to converge, the presence of local minima, and the assumption of perfectly known dimensions, e.g., the location of the pulleys and hinges w.r.t.  $\mathcal{B}$ .

## 3 RELATED WORK

There are two main approaches to pose estimation for elastic structures. If the geometric and material parameters are known with high accuracy, geometric (Jones and Walker, 2006) or static (Camarillo et al., 2009; Rucker and Webster, 2011) deformation models are employed. As these models require information about actuator positions or forces/torques, a common strategy is to use actuation sensors or additional sensors such as passive cables (Rolf and Steil, 2012) or fiber-bragg sensors (Roesthuis and Misra, 2016). More recently, a pose estimation technique without a deformation model for tendon-driven continuum mechanism was proposed (Deutschmann et al., 2019). It uses encoder values of the tendon actuation, additional deformation based length sensors and a model for the pose-dependent length measurements to extract the pose by nonlinear optimization or an Extended Kalman Filter. Two disadvantages of model-based approaches is that they are prone to parameter uncertainty, and also rely on the assumption of taut tendons (and length sensors) which might be jeopardized in fast motions or external contacts.



The second approach is to use data-driven methods, which learn a direct input-output behavior from measured data. The inputs are commonly actuation torques or actuator positions, and the corresponding output is the end effector pose. Popular models are based on neural networks to approximate the static characteristics from actuation forces (Braganza et al., 2007; Giorelli et al., 2015), or a Gaussian mixture model to relate actuator lengths to end effector poses (Malekzadeh et al., 2016).

Fault detection in robotic systems (Khalastchi and Kalech, 2018) is a particular instance of the wide field of anomaly detection (Chandola et al., 2009). Three main approaches to

tackle the issue are distinguished (Khalastchi and Kalech, 2018): 1) knowledge-based, 2) data-driven-based and 3) model-based approaches.

The fundamental assumption in knowledge-based approaches are that all faults and their corresponding symptoms are known. Then, casual analysis is used based on the fault-symptom relationship to find to fault which is occurring. In (Hamilton et al., 2001) this approach is used to for autonomous underwater vehicles.

Similarly, a common technique in machine-learning is train a model to classify normal and abnormal behaviors in a supervised fashion (Hornung et al., 2014). This requires a labeled dataset, and is unlikely to generalize to unseen faults. When no labels are available, unsupervised methods such as distance to a nearest neighbor (He and Wang, 2007) or data clustering (e.g., fitting a Gaussian mixture model to the data (Yu et al., 2010)) can be used instead.

Machine learning techniques belong to the second category, i.e., data-driven approaches and represent a large source for possible fault detection techniques. Other approaches covers the generation of data for nominal behavior by a physical simulator (Haidu et al., 2015). Based on the data, a failure envelope is learned for each datatype and failures are distinguished if the envelope is violated during task execution.

Other data-driven approaches make use of statistical filters. A common approach in mobile systems is the usage of Kalman-filters. A fast approach which utilizes one Kalman filter can be found in (Schmid et al., 2012; Steidle et al., 2016). Faulty sensors used in the process update are handled based on a confidence measure of the sensor data. Depending on that confidence, sensor information is added or removed in the process update. Also, a bank of Kalman-filters is utilized where each Kalman-filter predicts the nominal state of a systems assuming a specific failure has occurred. For a mobile autonomous robot in (Goel et al., 2000), a bank of eight filters are utilized (implying eight possible failure sources) and the most reliable filter, used for state

estimation, is chosen based on pre-trained neural-network using the filter residuals. As every filter is related to a specific failure, diagnosis is already incorporated.

The last category are model-based approaches. Commonly, they utilize a model of the system for a nominal behavior and fault are correspondingly identified if the real behavior, measured by some signals, is not coherent with the model of this signal. For hardware failures, physically inspired models are commonly used and a comprehensive treatment can be found in (Chen and Patton, 2012).

Our proposed method is a data-driven approach which utilizes machine learning with unsupervised training as it does not require labels. Unlike the previous approaches, that only detect a potential failure, we leverage minimal knowledge about the task to detect and handle the fault.

## 4 METHOD

### 4.1 Pose Estimation as a Regression Problem

The problem of predicting the six-DoF pose  $\mathbf{q}$  given measurements  $\tilde{\mathbf{I}}$  can be formulated as a supervised learning problem. For a given estimator  $f_{\Theta}$ , parametrized by the vector  $\Theta \in \mathbb{R}^p$  ( $p$  is the number of parameters), the objective is to find the parameters  $\Theta$  that minimize the error between its prediction  $\hat{\mathbf{q}} = f_{\Theta}(\tilde{\mathbf{I}}) \in \mathbb{R}^6$  and the true pose  $\mathbf{q}$ :

$$\mathcal{L}(\mathbf{q}, \hat{\mathbf{q}}) = \|\mathbf{q} - \hat{\mathbf{q}}\|_2^2 \quad (7)$$

where  $\mathcal{L}$  is the loss function or objective function.

This is a classic regression problem that can be solved using various techniques (Stulp and Sigaud, 2015). In this paper, we use linear models  $f_{\Theta}(\tilde{\mathbf{I}}) = \Theta^T \tilde{\mathbf{I}}$  and second-order polynomial models  $f_{\Theta}(\tilde{\mathbf{I}}) = \Theta^T \phi(\tilde{\mathbf{I}})$ , where  $\phi$  extracts polynomial features. We estimate the parameters with least squares. We select those models and method because they are fast to compute, do not require hyperparameter tuning or a large amount of data, and the learned model is easily interpretable. In our case, we also found that adding more complexity (e.g., using a neural network or higher order polynomial) did not improve accuracy.

### 4.2 Uncertainty Estimation Using Bootstrapped Ensemble

Querying only one predictor  $f_{\Theta}$  may be fastest, but one predictor alone does not provide uncertainty estimation, which is crucial to detect and handle failures. Here, we distinguish between aleatoric and epistemic uncertainty (Gal, 2016). Aleatoric uncertainty is due to the sensor noise and is irreducible (unless we improve the sensor precision). Epistemic uncertainty corresponds to the uncertainty in the model. This can be reduced by providing more training data, and increases with input samples out of the training distribution (Kendall and Gal, 2017). This is the uncertainty that allows failure detection.

Uncertainty estimation can be achieved by training an ensemble of  $n$  models  $\varepsilon = \{f_{\Theta_1}, f_{\Theta_2}, \dots, f_{\Theta_n}\}$ , each model having

different parameters  $\Theta_i$  (e. g., by using different parameter initialization for each model). A measure of the uncertainty would be the variance of the predictions. That is to say, the uncertainty for dimension  $k$  of the pose  $\mathbf{q} \in \mathbb{R}^6$  can be estimated with:

$$\sigma_k^2 = \frac{1}{n} \sum_{i=1}^n \left[ f_{\Theta_i}(\tilde{\mathbf{I}})_k - \overline{f_{\Theta}(\tilde{\mathbf{I}})}_k \right]^2 \quad (8)$$

where  $\overline{f_{\Theta}(\tilde{\mathbf{I}})}$  is the mean of the predictions and the subscript  $k$  denotes the  $k$ -th element of the vector.

Unfortunately, such an ensemble underestimates the epistemic uncertainty, as all the models are trained on the same dataset  $D$ . To address this issue, we can train each model  $f_{\Theta_i}$  only on a subset of the training set  $D_i \subset D$ . That way, each model makes different errors and it reduces the overconfidence of the ensemble. This technique is called bootstrapping (Efron, 1992; Breiman, 1996) and the resulting ensemble is a bootstrapped ensemble. The interesting bit of bootstrapped ensemble is that we can use any predictor  $f_{\Theta_i}$ . **Figure 3** summarizes the creation of ensembles and sub-ensembles of models.

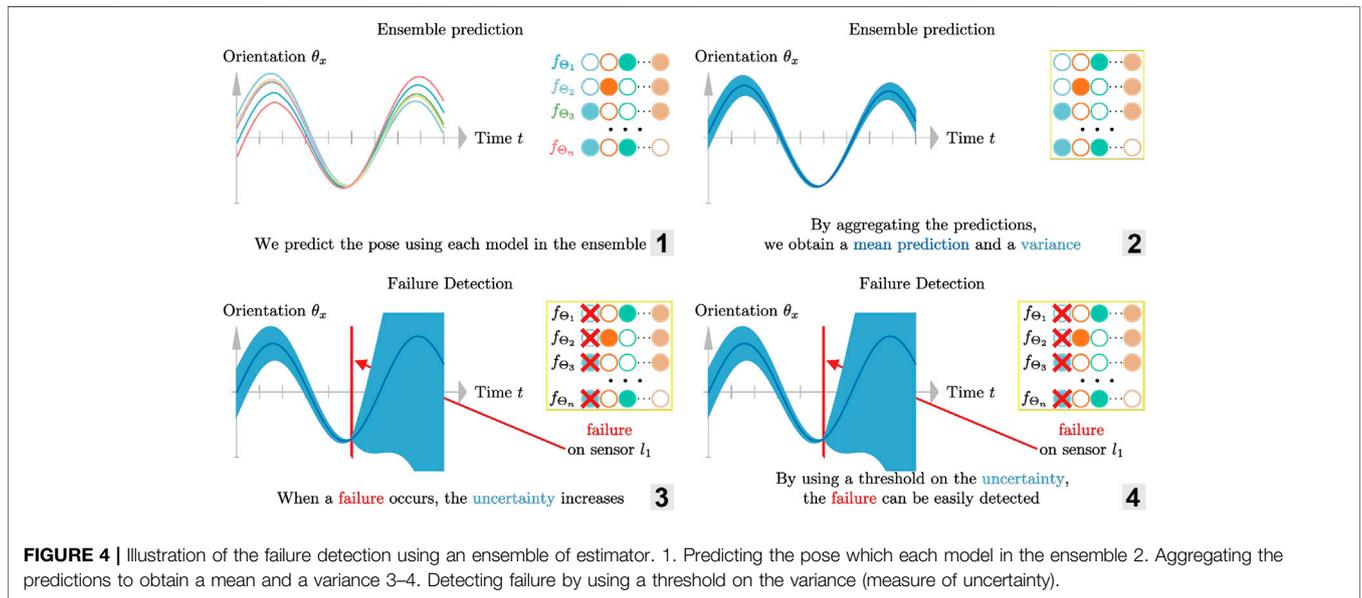
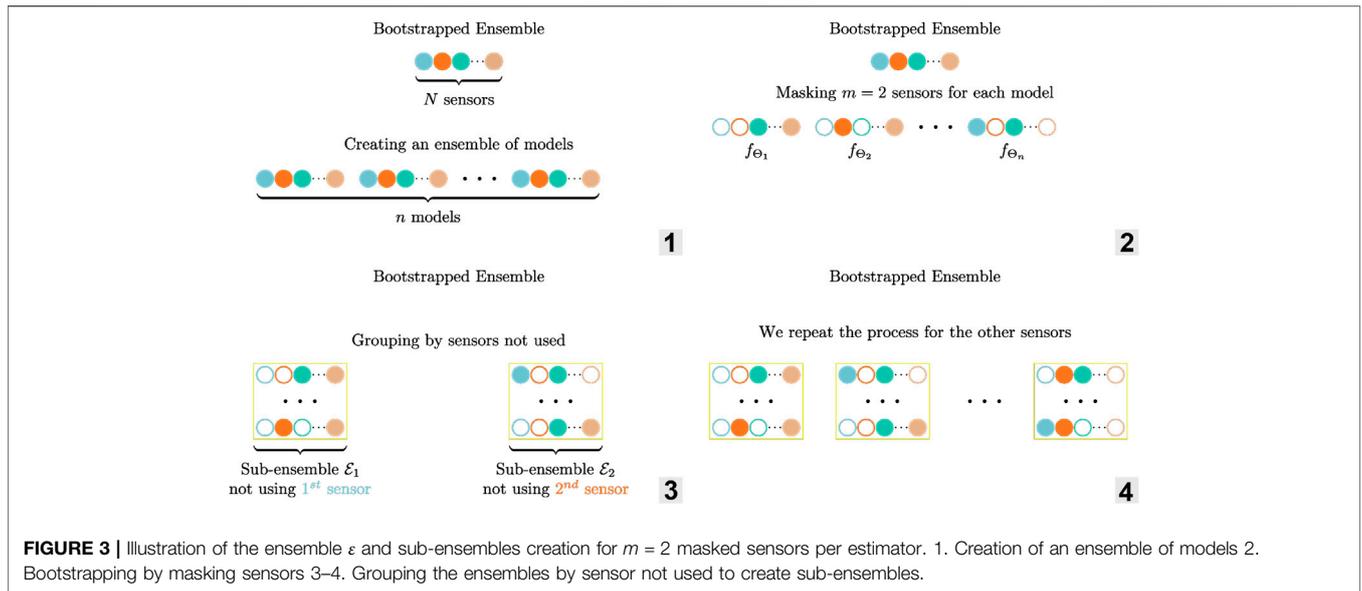
### 4.3 Failure Detection and Handling

The ensemble of estimators presented in the previous section gives us an uncertainty measure: the variance  $\sigma_k^2$  of the estimators. With this uncertainty measure, we can simply use a threshold  $\lambda$  to detect a failure. That is to say, we consider that there is a failure if  $\sigma_k^2 > \lambda$ . **Figure 4** illustrates how such ensemble can be used to detect a failed sensor.

For our six-DoF pose estimation problem, we have redundant sensor information: there are more sensors than needed and there is coupling in the system. Because of the extra sensors, we can detect which one(s) failed, and react to it by grouping the predictions. This allows to estimate the pose  $\mathbf{q}$  accurately even with multiple failures. The key insight is how we define the subsets  $D_i \subset D$  to create the ensemble (cf. **Figure 3**): we use only part of the sensors measurements  $\tilde{\mathbf{I}}$  to construct the subsets and then group the estimators in sub-ensembles  $\varepsilon_i \subset \varepsilon$  by sensor(s) not used for prediction. As an example, masking the first sensor results in the input vector  $\tilde{\mathbf{I}} = (0 \ \tilde{I}_2 \ \tilde{I}_3 \ \dots \ \tilde{I}_8)$ .

To illustrate the idea, we now consider the case where we mask  $m = 2$  sensors (out of  $N = 8$ ) as shown in **Figures 3, 5**. Among the ensemble of trained models  $\varepsilon = (f_{\Theta_i}, i = 1 \dots n)$ , one is not using sensors  $\tilde{I}_1$  and  $\tilde{I}_2$  (masked sensors are filled with black in **Figure 3**), another not using  $\tilde{I}_1$  and  $\tilde{I}_3, \dots$  in a way that there is a sub-ensemble  $\varepsilon_{i_1} \subset \varepsilon$  composed of estimators that don't use the first sensor  $\tilde{I}_1$  for predicting the pose (first sub-ensemble in **Figure 3**), a second one  $\varepsilon_{i_2}$  with estimators that don't use the second sensor  $\tilde{I}_2, \dots$  as a result, if one sensor breaks down as in **Figure 5**, we can still use the predictions of one sub-ensemble: the one that does not use the failed sensor.

In order to check if a sub-ensemble  $\varepsilon_i$  is usable, we compute the sub-ensemble variance  $\sigma_k^2$  and compare it to a threshold. When there is no error, all sub-ensembles should have a variance below the threshold. If one failure occurs, e. g. on the sensor  $j$ , then only one sub-ensemble should pass the test:  $\varepsilon_{i_j}$  which is composed of



estimators that do not use sensor  $j$ . The appropriate threshold  $\lambda$  is determined empirically.

The previous example was for one failed sensor only. To detect and handle more than one, we repeat three steps:

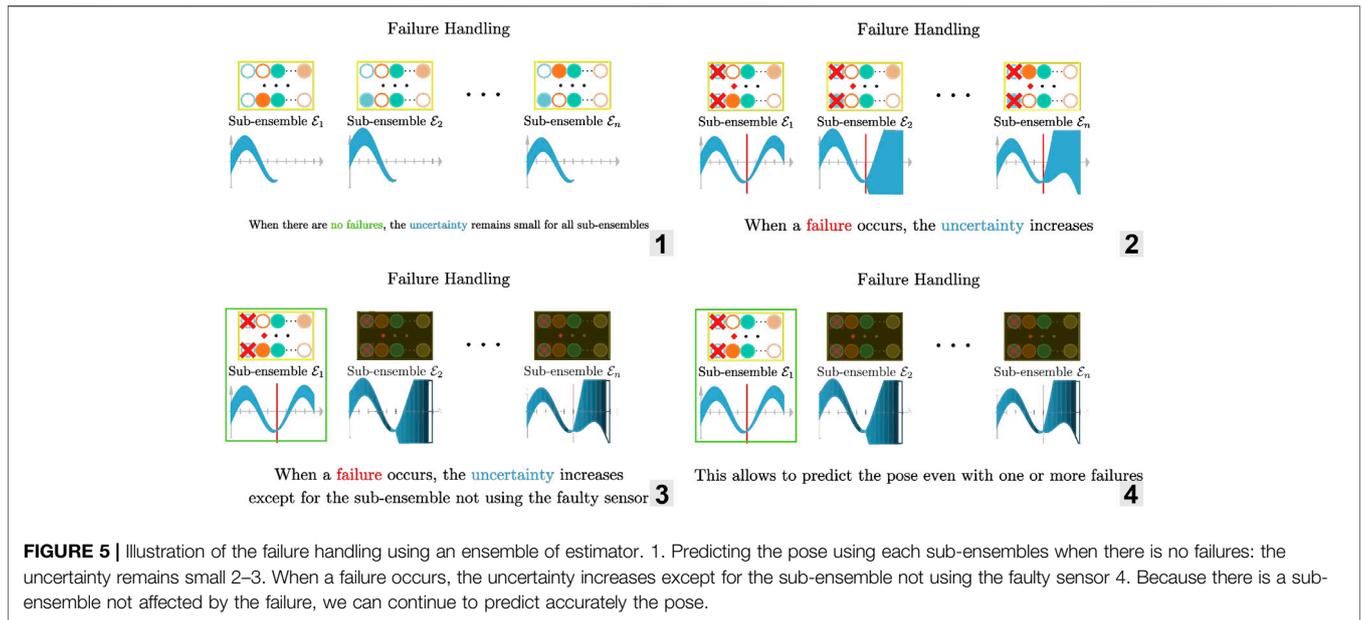
1. Ensemble prediction: predict the pose using each model from the ensemble  $\varepsilon$
2. Group predictions by sensors not used to create the sub-ensembles  $\varepsilon_l \in \varepsilon$
3. Check the variance of each sub-ensemble to detect failure

We start with  $m = 2$  masked inputs to detect and handle one failure and increment that number if needed. We provide the

reader with a visual explanation of the method in the **Supplementary Video**.

To detect  $n_{failures}$  failures, we need to create sub-ensembles  $\varepsilon_l$  with  $m = n_{failures}$  sensors masked. To identify which sensor(s) failed, we need to mask one additional sensor. As a first approach, we create the subsets  $\mathcal{D}_i \subset \mathcal{D}$  corresponding to all the possible  $\binom{N}{m}$  mask configurations, and train one model for each subset. That is to say, we train an ensemble  $\varepsilon$  of  $n = \binom{N}{m}$  models.

We then repeat that process with additional sensors masked, until there are not enough sensors left to have a reliable prediction. For the experimental platform used in this paper, the experiments reveal that



relying on 3 (out of  $N = 8$ ) sensors is enough to have an acceptable precision, which means we can detect and handle up to 4 failures.

Concretely, to be robust up to 4 failures, we need to cover the cases where:

- there is one failure, therefore mask 2 sensors and train  $\binom{8}{2} = 28$  polynomial models.
- there are two failures, therefore mask 3 sensors and train  $\binom{8}{3} = 56$  polynomial models.
- there are three failures, therefore mask 4 sensors and train  $\binom{8}{4} = 70$  polynomial models.
- there are four failures, therefore mask 5 sensors and train  $\binom{8}{5} = 56$  polynomial models.

Which means we have to train  $\sum_{m=2}^5 \binom{N}{m} = 210$  polynomial models in total to handle all possible cases. We start with  $m = 2$  masked sensors and the first 28 polynomial models (nominal case, no failure) and then increment the number of masked sensors and use additional polynomial models only if needed. The failure detection and handling process remains the exact same at each stage: they are the three steps presented above.

Although simple, this naive way of creating subsets does not scale well if the number of sensors or failures handled increases. We discuss in **Section 5.6** how to optimize that process.

## 5 EXPERIMENTS

The goal of this section is to evaluate the performance of the proposed method in terms of speed and accuracy, and investigate its robustness against one or more sensor failures.

### 5.1 Experimental Setup

The experimental setup consists of the tendon-driven continuum mechanisms used as the neck of the humanoid robot DAVID, **Figure 1**. The upper platform is equipped with a marker target of an external camera tracking system, which serves as the ground truth data of the pose. The tracking system is only used for training, evaluation and is not needed afterward. The pose dependent length information that is retrieved at a frequency of 300 Hz comes from two sources. The tendon lengths are given by the tendon actuators from Robodrive and the four additional length sensors are provided by Kinfinity UG.

The placement of the two type of sensors is not arbitrary. The tendons, i.e., the power transmission element of the actuation, decide the reachable workspace of the mechanism and therefore their routing cannot be changed. In contrast the placement of length sensors can be chosen almost freely. In (Deutschmann et al., 2019), different placements of the sensors were experimentally investigated regarding their accuracy. To provide a fair comparison, the configuration yielding best results was also chosen for the present paper. The system is driven to different poses or along trajectories by commanding different sets of tendon-tension, which are realized by a local tendon-tension controller in each of the actuators (Chalon et al., 2011).

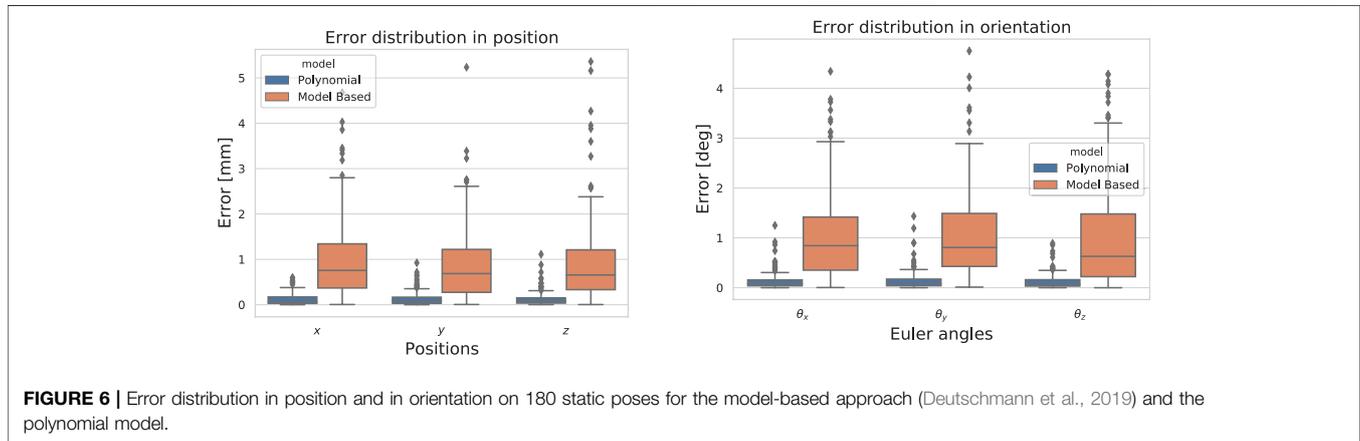
### 5.2 Data-Driven Pose Estimation

#### 5.2.1 Static and Dynamic Estimation Error

*Static Pose Estimation.* To assess the performance of our six-DoF estimator and compare it to previous work, we first command the neck to reach 200 static poses. For each pose, we retrieve the ground truth  $q$  using an external camera tracking system from Nikon (Metris, 2016) and compute the prediction error in position and orientation. To train the models, we sample randomly 20 poses and use the rest of the datapoints for testing. Each model is therefore trained with the same 20

**TABLE 1** | Comparison of mean runtime and error (both in position and orientation) for each method. The data-driven approaches are fast and also more accurate than the model-based approach. For each metric, we bolded the best mean. "N/A" means that the data is not available.

	Model-Based Deutschmann et al. 2019	Linear	Polynomial
Runtime (ms)	N/A ± N/A	<b>0.1 ± 0.0</b>	0.2 ± 0.0
Static Pose Estimation			
Error in position (mm)	1.1 ± 1.0	0.3 ± 0.3	<b>0.2 ± 0.2</b>
Error in orientation (deg)	0.9 ± 0.8	0.3 ± 0.3	0.1 ± 0.1
Dynamic Pose Estimation			
Error in position (mm)	5.1 ± 2.3	1.7 ± 1.4	<b>1.6 ± 1.4</b>
Error in orientation (deg)	3.8 ± 3.0	0.8 ± 0.6	<b>0.5 ± 0.4</b>



**FIGURE 6** | Error distribution in position and in orientation on 180 static poses for the model-based approach (Deutschmann et al., 2019) and the polynomial model.

**TABLE 2** | Ablation study: influence of the amount of training data, number of sensors and type of model on the performance. We bolded results with the best mean error. Baseline models are trained using 20 datapoints and 8 sensors.

	Position error (mm)	Orientation error (deg)
Model-Based	1.1 ± 1.0	0.9 ± 0.8
Linear baseline	0.3 ± 0.3	0.3 ± 0.3
Polynomial baseline	0.2 ± 0.2	<b>0.1 ± 0.1</b>
Neural network (2 layers)	0.8 ± 0.6	0.8 ± 0.7
Linear (3 sensors)	1.0 ± 0.7	1.2 ± 1.0
Linear (4 sensors)	0.4 ± 0.3	0.5 ± 0.6
Polynomial (3 sensors)	0.9 ± 0.6	1.2 ± 0.9
Polynomial (4 sensors)	<b>0.1 ± 0.1</b>	<b>0.1 ± 0.1</b>
Polynomial (5 datapoints)	0.9 ± 0.9	1.1 ± 0.9
Polynomial (10 datapoints)	0.5 ± 0.5	0.6 ± 0.5
Polynomial (40 datapoints)	0.2 ± 0.3	0.3 ± 0.2

datapoints, but each one with different sensors masked, so each model uses a different set of input features. We repeat that process using 10 different random seeds. To estimate the runtime of each method, we perform 1,000 single predictions and average the time taken. The test is done on a computer equipped with 8 Intel i7-8550U CPUs at 1.80 GHz.

**Dynamic Pose Estimation.** We then evaluate the pose estimation method trained with static poses only on dynamic motions. For that, we record the ground-truth and estimated pose on a pre-defined trajectory. We provide the reader with a **Supplementary Video** showing the method in action.

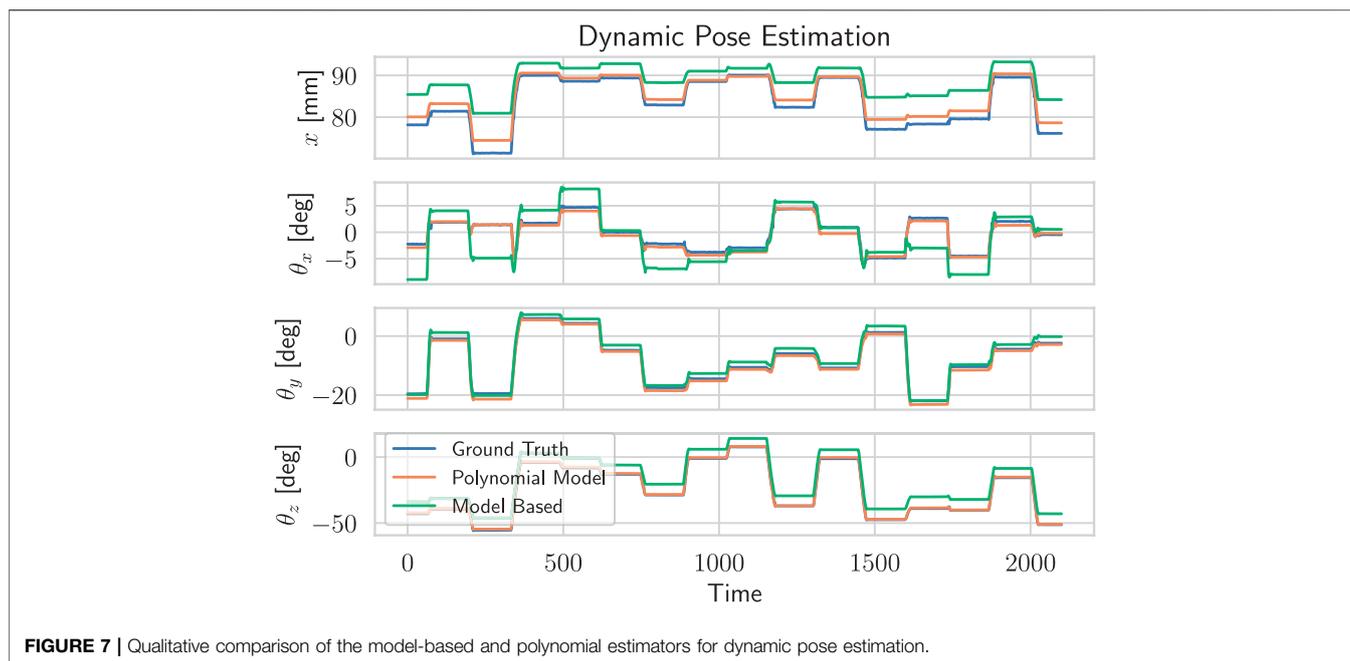
## 5.2.2 Results

**Static Pose Estimation.** The results are summarized using mean error and standard deviation over the test poses in **Table 1**, runtimes are included when available. We also show the error distribution in position and orientation in **Figure 6**.

Overall, the data-driven approaches are fast (they run at ~ 5000 Hz) and more accurate than the model-based one: the mean error is reduced up to 5 times (cf. **Table 2**). As expected, the linear model runs faster at a cost of some accuracy compared to the polynomial model. It is worth mentioning that although the model-based approach appears inaccurate in the present comparison, it is fairly accurate and fast to compute when compared to other model-based pose estimation techniques as reported in (Deutschmann et al., 2019).

**Dynamic Pose Estimation.** To report also on the dynamic estimation behavior, the experimental platform is driven to several static poses subsequently and the estimated pose for the model-based and the present approach are recorded. The mean-error along the trajectory of 200 subsequent poses is given in **Table 1** and the corresponding trajectories of four estimated coordinates ( $x, \theta_x, \theta_y, \theta_z$ ) can be seen in **Figure 7**.

As for static poses, the data-driven approaches perform best, with the polynomial model being more accurate than the linear one. Mean error reported in **Table 1** is higher than for static poses for two reasons. First, it now accounts for the transition error between two static poses, that are not covered at all during training. Then, as we are considering trajectories, static errors



at fixed poses accumulate yielding a higher mean value. It is worth noting that those models were only trained on static poses and therefore the results could be improved if part of the dynamic poses were included in the training set.

### 5.3 Hyperparameters Study

To study the effect of the different hyperparameters (type of model, training set size, number of sensors) on the performance, we compare the baseline regressors (20 datapoints and 8 sensors) to several variants. One datapoint corresponds to training data at a specific pose. We present the results in **Table 2**.

*Effect of the training set size.* We vary the training set size from 5 datapoints to 40 datapoints for the polynomial model. Overall, with more training data, the performance improves. However, after a certain amount (here 30 datapoints), adding more data does not improve the results anymore. This is due both to the irreducible error (sensor noise) and to the limited capacity of the polynomial model. Although the results are slightly worse with 40 datapoints, the difference is not significant.

*Effect of the type of model.* We compare linear, polynomial and neural network<sup>1</sup> models. Although the neural network yields good performance, it is less sample efficient than the two others, i. e., with more training data it would reach the same accuracy. It also requires hyperparameter tuning (learning rate, mini-batch size, ...) and has more parameters ( $\sim 3e6$  parameters vs. 276 for the polynomial model). For those reasons, using a polynomial model of order 2 is enough in our case.

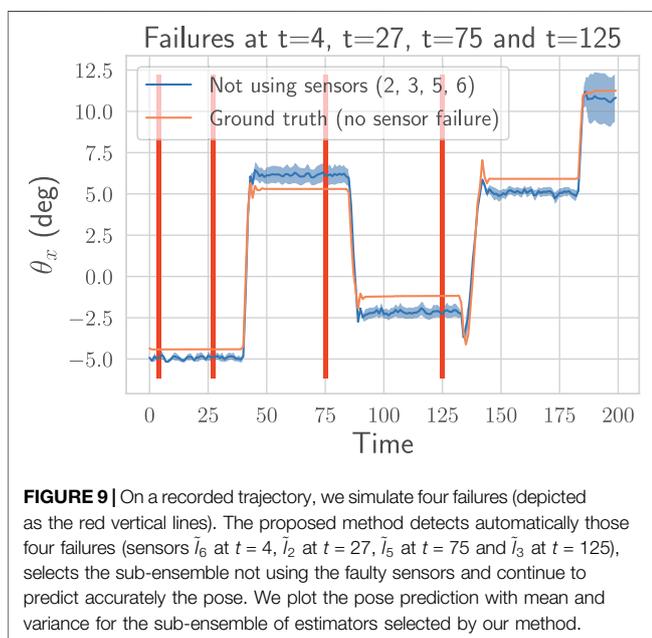
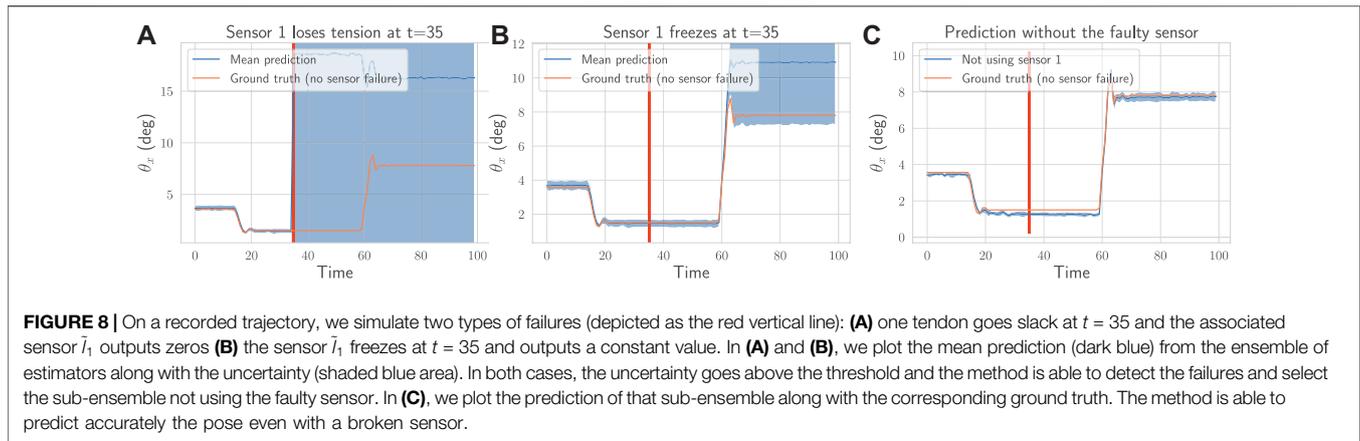
<sup>1</sup>2 fully-connected layers of 256 units each, trained with Adam optimizer until convergence using a learning rate of  $1e-3$  and a batch size of 20 (size of the training set). Different hyperparameters were tested, and these performed best.

*Effect of the number of sensors.* We compare the baseline linear and polynomial models (8 sensors) to models using less sensors (3 and 4 sensors). As expected, adding more sensors reduces the error for the linear model. Almost no changes can be observed for the polynomial model with more than 4 sensors: the results are slightly worse but the difference is not significant. As discussed in **Section 4.3**, having redundant sensor information is key to detect and handle potential failures. The more sensors we have, the more failure we can handle. Therefore having 8 sensors is preferable.

*Effect of the placement of sensors.* Because the polynomial model performs well with only 4 sensors (for instance when using only the tendon length sensors), the placement of the 4 additional length sensors will not affect much the accuracy. However, the positioning would affect the ability to detect failures: if the information that a sensor provides is not useful for the prediction, then it will not be used by the polynomial model, the weight for this input feature will be close to zero. As a result, if such sensor fails, as it does not affect the variance of the predictions, the failure will not be detected but the pose will still be predicted accurately.

### 5.4 Failure Detection and Handling

To evaluate the effectiveness of the proposed approach for detecting and handling failures, we simulate the loss of sensors while the neck is moving. A common failure is when a length sensor outputs wrong values because a tendon goes slack. Instead of a correct measurement, it outputs zeros. Another type of failure, harder to detect, is when the sensor freezes and outputs a constant value. In **Figures 8A,B**, we show the effect of both failures on the uncertainty: a jump can be observed right after the loss of tension in **Figure 8A**. When the sensor freezes (cf. **Figure 8B**), as expected, the failure is detected only when the neck position changes. Because in both cases the variance increases by a large amount, no careful tuning of the detection threshold is



required. To show that the method can handle more than one failure, we simulate in **Figure 9** the loss of four sensors.

In **Figures 8C, 9**, we display the prediction over time for the sub-ensemble not using the broken sensors. We use the ground-truth pose as a reference. The approach successfully detect all the failures and handle them by using the sub-ensemble not affected by the loss of the sensors. As a result, the proposed method can robustly and still accurately (cf. **Table 2**) predict the pose even with multiple losses.

## 5.5 Comparison With the Linearized Model

The geometry of the deformation of the neck system and the coupling of the tendon motion is highly nonlinear (Deutschmann et al., 2018). The geometrical exact nonlinear mapping can be found in (Deutschmann et al., 2019). This work confirms the non-linearity as a polynomial model of order two reveals the best prediction results. However, as stated in **Table 2**, the linear models already yield results comparable to the model-based

approach. We therefore compare the first-order term of the trained linear coefficients for four sensors used with the partial derivative of the  $q(I)$  i. e., a local linearization about the initial straight configuration in the following equation.

$$f_{lin} = \begin{pmatrix} 0.41 & 0.35 & 0.37 & 0.41 \\ -0.25 & 0.30 & 0.25 & -0.26 \\ 0.38 & 0.56 & -0.56 & -0.38 \\ 7.46 & -16.61 & 15.76 & -7.51 \\ -9.94 & -5.17 & 5.05 & 9.63 \\ -4.13 & 7.75 & 5.45 & -3.45 \end{pmatrix} \quad (9)$$

$$\left(\frac{\partial q(I)}{\partial I}\right)^T = \begin{pmatrix} 0.44 & 0.06 & 0.06 & 0.44 \\ -2.33 & 2.5 & 2.50 & -2.34 \\ 0.01 & 1.78 & -1.73 & -0.01 \\ 0.43 & -0.15 & 0.15 & -0.43 \\ -11.79 & -1.0 & 0.99 & 11.76 \\ -2.69 & 2.78 & 2.77 & -2.63 \end{pmatrix} \quad (10)$$

The top three rows correspond to positions and the bottom three rows to orientations. The trained coefficients (9) and the model-based coefficients (10) have the same signs and symmetries, indicating that the sensor information is used in a similar fashion to predict the direction of the pose. However, the magnitude of the values differs largely: simply linearizing the model would not be as accurate as the trained linear model.

## 5.6 Limitations

We have shown in the previous sections that the proposed approach yields a good estimation of the pose and handle failures while keeping a low runtime. However, we have to train  $\sum_{m=2}^5 \binom{N}{m} = 210$  models to handle all possible failures. We can optimize our approach to make it scalable both in term of number of sensors and failures handled. A first optimization is to restrict the sub-ensembles  $e_i$  to a maximum of two models. Then, to handle more sensors, we can cluster them (e.g., grouping them in a hierarchical way) or compress the information to reduce the input dimension (e.g., using an auto-encoder). Doing so, our approach would identify a failure from a group of sensors and no more from a single sensor.

## 6 DISCUSSION AND CONCLUSION

In this work, we show that data-driven approaches are competitive alternatives to estimate the pose of a tendon-driven continuum mechanism. The linear and polynomial models are fast to train, require only a small amount of data and prove to be more accurate than the model-based approach. As mentioned in the end of **Section 2**, assumptions about the sensor linearity and perfectly known kinematics are made which, in cases where the lose their validity, may result in larger errors in the pose estimation. In other words, small deviations or modeling errors in the kinematics of the tendon pulleys or the attachment points of the sensors will cost accuracy in the estimated pose. In contrast, the polynomial model learns a direct mapping from sensor readings to the measured pose and does not make these assumptions, allowing an improved accuracy.

To detect and handle sensor failures, we make use of ensembling technique and minimal knowledge about the system. By clustering the different models, our method predicts the pose accurately even with 4 out of 8 faulty sensors.

The presented method embodies a computationally fast and accurate pose estimation method. This method was already employed to train a reinforcement learning controller (Raffin and Stulp, 2020) and could also replace the model-based approach, currently used in the model-based control approach implemented on the neck system (Deutschmann et al., 2017). This would result in a more accurate positioning given the more accurate estimated pose.

One limitation is the scalability of the method, as discussed in the previous section, which should be addressed in the future. Apart from that, our method is in fact not specific to tendon-

driven continuum mechanism: it only requires sensor redundancy and learn to predict the pose directly from data.

## DATA AVAILABILITY STATEMENT

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

## AUTHOR CONTRIBUTIONS

AR conceived of the presented idea and carried out the experiment with the help of BD, AR and BD wrote the manuscript with support from FS.

## FUNDING

This work was partially funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation)—Grant Number 405032572, the European Union's Horizon 2020 Research and Innovation Program under Grant Number 951992 (project VeriDream) and by the Hermann von Helmholtz-Gemeinschaft Deutscher Forschungszentren e. V. under Grant ZT-I-0010 (RedMod) and ZT-I-PF-5-20 (LearnGraspPhases).

## SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: <https://www.frontiersin.org/articles/10.3389/frobt.2021.619238/full#supplementary-material>.

## REFERENCES

- Braganza, D., Dawson, D. M., Walker, I. D., and Nath, N. (2007). A neural network controller for continuum robots. *IEEE Trans. Robot.* 23, 1270–1277. doi:10.1109/tro.2007.906248
- Breiman, L. (1996). Bagging predictors. *Mach. Learn.* 24, 123–140. doi:10.1007/bf00058655
- Camarillo, D. B., Carlson, C. R., and Salisbury, J. K. (2009). Configuration tracking for continuum manipulators with coupled tendon drive. *IEEE Trans. Robot.* 25, 798–808. doi:10.1109/tro.2009.2022426
- Chalon, M., Friedl, W., Reinecke, J., Wimboeck, T., and Albu-Schaeffer, A. (2011). “Impedance control of a non-linearly coupled tendon driven thumb,” in IEEE/RSJ International Conference on Intelligent Robots and Systems, San Francisco, CA, September 25–30, 2011 (Piscataway, NJ: IEEE), 4215–4221.
- Chandola, V., Banerjee, A., and Kumar, V. (2009). Anomaly detection: a survey. *ACM Comput. Surv.* 41, 1–58. doi:10.1145/1541880.1541882
- Chen, J., and Patton, R. J. (2012). *Robust model-based fault diagnosis for dynamic systems*, Heidelberg, Germany: Springer Science and Business Media.
- Deutschmann, B., Chalon, M., Reinecke, J., Maier, M., and Ott, C. (2019). Six-dof pose estimation for a tendon-driven continuum mechanism without a deformation model. *IEEE Robot. Autom. Lett.* 4, 3425–3432. doi:10.1109/lra.2019.2927943
- Deutschmann, B., Dietrich, A., and Ott, C. (2017). “Position control of an underactuated continuum mechanism using a reduced nonlinear model,” in IEEE 56th annual conference on decision and control, Melbourne, VIC, December 12–15, 2017 (Piscataway, NJ: IEEE), 5223–5230.
- Deutschmann, B., Eugster, S. R., and Ott, C. (2018). Reduced models for the static simulation of an elastic continuum mechanism. *IFAC-PapersOnLine* 51, 403–408. doi:10.1016/j.ifacol.2018.03.069
- Efron, B. (1992). Bootstrap methods: another look at the jackknife. *Breakthroughs Stat.* 569–593. doi:10.1007/978-1-4612-4380-9\_41
- Gal, Y. (2016). *Uncertainty in deep learning*, Cambridge, United Kingdom: University of Cambridge.
- Giorelli, M., Renda, F., Calisti, M., Arienti, A., Ferri, G., and Laschi, C. (2015). Neural network and jacobian method for solving the inverse statics of a cable-driven soft arm with nonconstant curvature. *IEEE Trans. Robot.* 31, 823–834. doi:10.1109/tro.2015.2428511
- Goel, P., Dedeoglu, G., Roumeliotis, S. I., and Sukhatme, G. S. (2000). “Fault detection and identification in a mobile robot using multiple model estimation and neural network,” in IEEE International Conference on Robotics and Automation, San Francisco, CA, April 24–28, 2000 (Piscataway, NJ: IEEE), 2302–2309.
- Haidu, A., Kohlsdorf, D., and Beetz, M. (2015). “Learning action failure models from interactive physics-based simulations,” in IEEE/RSJ international conference on intelligent robots and systems, Hamburg, Germany, 28 September–2 October, 2015 (Piscataway, NJ: IEEE), 5370–5375.
- Hamilton, K., Lane, D., Taylor, N., and Brown, K. (2001). “Fault diagnosis on autonomous robotic vehicles with recovery: an integrated heterogeneous-knowledge approach,” in Proceedings 2001 ICRA. IEEE International

- Conference on Robotics and Automation, Seoul, South Korea, May 21–26, 2001 (Piscataway, NJ: IEEE), 3232–3237.
- He, P., and Wang, J. (2007). Fault detection using the k-nearest neighbor rule for semiconductor manufacturing processes. *IEEE Trans. Semicond. Manuf.* 20, 345–354. doi:10.1109/tsm.2007.907607
- Hornung, R., Urbanek, H., Klodmann, J., Osendorfer, C., and Van Der Smagt, P. (2014). “Model-free robot anomaly detection,” in IEEE/RSJ international conference on intelligent robots and systems, Chicago, IL, September 14–18, 2014 (Piscataway, NJ: IEEE), 3676–3683.
- Jones, B. A., and Walker, I. D. (2006). Practical kinematics for real-time implementation of continuum robots. *IEEE Trans. Robot.* 22, 1087–1099. doi:10.1109/tro.2006.886268
- Kendall, A., and Gal, Y. (2017). What uncertainties do we need in bayesian deep learning for computer vision?. *Adv. Neural Inf. Process. Syst.*, 5574–5584. doi:10.5555/3295222.3295309
- Khalastchi, E., and Kalech, M. (2018). On fault detection and diagnosis in robotic systems. *ACM Comput. Surv.* 51, 1–24. doi:10.1145/3146389
- Malekzadeh, M. S., Queißer, J. F., and Steil, J. J. (2016). “Learning the end-effector pose from demonstration for the bionic handling assistant robot,” in Proceedings of the 9th international workshop on human human friendly robotics, September 29 and 30, 2016, Genoa, Italy.
- Metris, k. C. (2016). Available at: <http://www.metris3d.huw> (Accessed August 31, 2020).
- Raffin, A., and Stulp, F. (2020). *Generalized state-dependent exploration for deep reinforcement learning in robotics*,
- Reinecke, J., Deutschmann, B., and Fehrenbach, D. (2016). “A structurally flexible humanoid spine based on a tendon-driven elastic continuum,” in International conference on robotics and automation, Stockholm, Sweden, May 16–21, 2016 (Piscataway, NJ: IEEE), 4714–4721.
- Roesthuis, R., and Misra, S. (2016). Steering of multisegment continuum manipulators using rigid-link modeling and FBG-based shape sensing. *IEEE Trans. Robot.* 32, 372–382. doi:10.1109/tro.2016.2527047
- Rolf, M., and Steil, J. J. (2012). Constant curvature continuum kinematics as fast approximate model for the bionic handling assistant,” in International conference on intelligent robots and systems, Vilamoura-Algarve, Portugal, October 7–12, 2012 (Piscataway, NJ: IEEE), 3440–3446.
- Rucker, D. C., and Webster, R. J. (2011). Statics and dynamics of continuum robots with general tendon routing and external loading. *IEEE Trans. Robot.* 27, 1033–1044. doi:10.1109/tro.2011.2160469
- Schmid, K., Ruess, F., Suppa, M., and Burschka, D. (2012). “State estimation for highly dynamic flying systems using key frame odometry with varying time delays,” in International conference on intelligent robots and systems, Vilamoura-Algarve, Portugal, October 7–12, 2012 (Piscataway, NJ: IEEE), 2997–3004.
- Steidle, F., Tobergte, A., and Albu-Schäffer, A. (2016). “Optical-inertial tracking of an input device for real-time robot control,” in International conference on robotics and automation, Stockholm, Sweden, May 16–21, 2016 (Piscataway, NJ: IEEE), 742–749.
- Stulp, F., and Sigaud, O. (2015). Many regression algorithms, one unified model: a review. *Neural Netw.* 69, 60–79. doi:10.1016/j.neunet.2015.05.005
- Yu, G., Li, C., and Sun, J. (2010). Machine fault diagnosis based on Gaussian mixture model and its application. *Int. J. Adv. Manuf Techn.* 48, 205–212. doi:10.1007/s00170-009-2283-5

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

Copyright © 2021 Raffin, Deutschmann and Stulp. This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY). The use, distribution or reproduction in other forums is permitted, provided the original author(s) and the copyright owner(s) are credited and that the original publication in this journal is cited, in accordance with accepted academic practice. No use, distribution or reproduction is permitted which does not comply with these terms.