Robotics: Science and Systems 2019
Freiburg im Breisgau, June 22-26, 2019

On the Merits of Joint Space and Orientation Representations in Learning the Forward Kinematics in $SE(3)$

Reinhard M. Grassmann and Jessica Burgner-Kahrs
Continuum Robotics Laboratory, University of Toronto Mississauga, Canada
Email: reinhard.grassmann@utoronto.ca

***Abstract*—This paper investigates the influence of different joint space and orientation representations on the approximation of the forward kinematics. We consider all degrees of freedom in three dimensional space $SE(3)$ and in the robot's joint space $\mathcal{Q}$. In order to approximate the forward kinematics, different shallow artificial neural networks with ReLU (rectified linear unit) activation functions are designed. The amount of weights and bias' of each network are normalized. The results show that quaternion/vector-pairs outperform other $SE(3)$ representations with respect to the approximation capabilities, which is demonstrated with two robot types; a Stanford Arm and a concentric tube continuum robot. For the latter, experimental measurements from a robot prototype are used as well. Regarding measured data, if quaternion/vector-pairs are used, the approximation error with respect to translation and to rotation is found to be seven times and three times more accurate, respectively. By utilizing a four-parameter orientation representation, the position tip error is less than 0.8% with respect to the robot length on measured data showing higher accuracy compared to the state-of-the-art-modeling (1.5%) for concentric tube continuum robots. Other three-parameter representations of $SO(3)$ cannot achieve this, for instance any sets of Euler angles (in the best case 3.5% with respect to the robot length).**

I. INTRODUCTION

In general, computing the forward kinematics (FK) of a robot with rigid-links is straightforward. The kinematics of a concentric tube continuum robot (CTCR), which is composed of multiple concentric, pre-curved super-elastic tubes being rotated and translated w.r.t. each other, is characterized by highly non-linear behavior due to the elastic interactions between the tubes [5]. A CTCR is depicted in Fig. 1. Common model-based approaches [29, 9] are based on the theory of special Cosserat rods and are solved numerically. Additional factors, e.g. friction or tube tolerances, are commonly not considered due to even higher computational load and modeling effort. Thus, determining the FK of a CTCR is more challenging.

Rather than using a model-based approach to the FK, machine learning techniques can be applied in order to compute values of a function that reflects poses from specified values of the joint space. Neural networks are widely used due to their universal approximation ability. A feedforward network (FFN) can approximate a continuous function in a compact set [7, 11, 16, 17]. Therefore, neural networks can be applied in order to approximate the FK. Reinhart and Steil [28] as well as Nguyen et al. [22] consider robots with rigid-links while Phung et al. [25] utilize a robot with flexible-links.

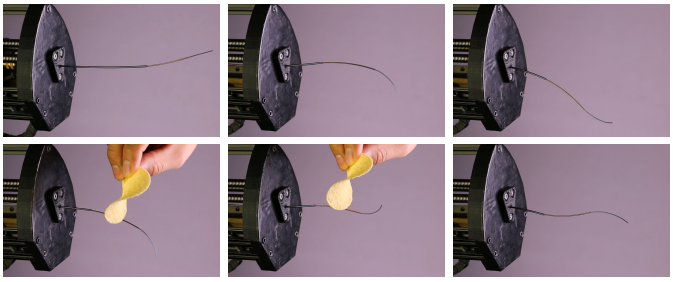

Fig. 1. Compliant reaction of a concentric tube continuum robot during a motion sequence while interacting with a potato chip. Its intrinsic compliance results from the highly non-linear characteristic of its forward kinematics.

The approximation of the FK of a continuum robot being inherently compliant and flexible such as CTCR is investigated by Bergeles et al. [3] and by Grassmann et al. [14].

Robotic applications and publications regarding neural networks are mostly focused on classification. Such tasks are not very sensitive to the actual output, since small disturbances are unlikely to cause a shift in class. In contrast, any deviation of the output increases the approximation error. Furthermore, outputs are influenced by the inputs and by the smoothness of the continuous function as well. Learning methods [12, 26, 27, 30] and robot specific FFN architectures [13, 22, 21] are investigated so far. It is our hypothesis that the joint space and orientation representation has a significant impact on input-output characteristics of the FFN being the desired approximation. To the best of our knowledge, this effect has not been investigated thus far. Exploring and finding suitable inputs and outputs can be beneficial in order to make FFN a more efficient tool in robotic applications. Lastly, studies on FK approximation are mostly restricted to positions neglecting orientations, which can be a major drawback for applications in real-world scenarios.

In this work, we investigate the impact of different joint space and orientation representations on learning the FK using FFN. For this purpose, effects and influences are investigated on a robot with rigid-links, i.e. Stanford Arm, and on a kinematically more complex robot, i.e. CTCR with three tubes. All six degrees of freedom (DOF) w.r.t. $SE(3)$ as well as all 6-DOF w.r.t. the robot joint space are considered. In particular, the following contributions are made:

- Influence of different joint space and orientation representations in $SO(3)$ on learning the FK are empirically

studied indicating that advantageous representations are mandatory even for low dimensional problems.
- The pose in $SE(3)$ consisting of tip position and orientation is learned with no simplifications.
- A simple and effective transformation leads to a fast sampling method considering inequalities and, furthermore, positively affects the approximation results due to its capability of removing linear correlations.
- As minor contribution a new representation of cylindrical joints is introduced.

II. METHODS

In this section, our utilization of artificial neural networks is briefly described. We state different representations of $SO(3)$ and provide the approximation errors used as performance criteria. Finally, we formulate three different joint space representations.

A. Artificial Neural Networks

Feedforward networks are utilized, which are a special type of artificial neural networks. Figure 2 depicts the architecture of a FFN and shows its notation. Such a FFN can approximate a smooth function in a compact set [16, 7, 11]. Therefore, it can approximate the FK. Note that a radial basis function (RBF) network could be utilized. However, Flake [10] states that RBF networks are greatly troubled by the curse of dimensionality because a single RBF unit covers only a small local region of the input space.

The ReLU (rectified linear unit) activation function $\varphi(x) = \max(0, x)$ is used in the hidden layer. It is advantageous over the commonly applied $\tanh$ activation function, in terms of computational efficiency and fast training. In order to approximate a function, linear activation functions are used for the output layer. The weights of the ReLU activation functions are initialized by HE-initialization [15], whereas the weights

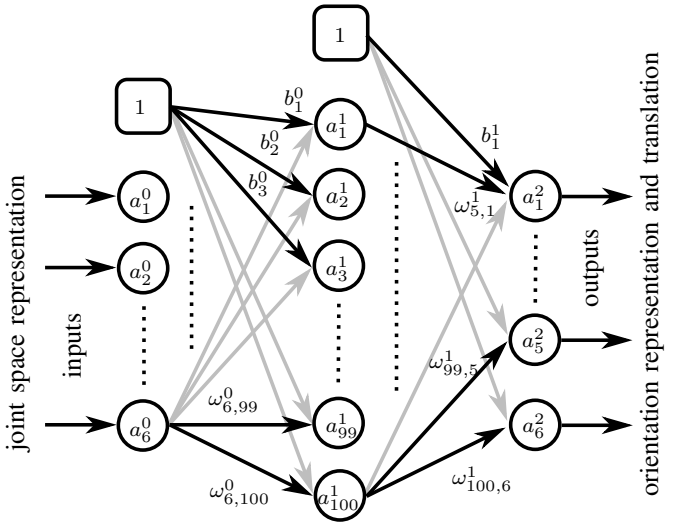

Fig. 2. Shallow fully connected feedforward network for the computation of the forward kinematics. The weight $\omega_{n,m}^l$ denotes the weight for the connection between the m^{th} artificial neuron in the $(l-1)^{\text{th}}$ layer to the n^{th} neuron in the l^{th} layer. The bias b_n^l denotes the n^{th} bias in the l^{th} layer. The activation a_n^l with $l \neq 0$ denotes the output of the n^{th} artificial neuron in the l^{th} layer, whereas a_n^0 denotes the input of the artificial neural network.

of the linear activation functions and all biases in the FFN are initialized with a uniform distribution.

The Adam optimizer [18] with mini-batch size of $N_{\text{bs}} = 128$ is utilized in order to optimize weights ω and biases b of the FFN. The batches are randomly extracted from the training set $\mathcal{S}_{\text{tra}}$ in each epoch N_{ep}. Using the Adam optimizer, we experimentally observe that neither under- nor over-fitting occurred and, furthermore, that the Adam optimizer converges much faster than a vanilla gradient-descent optimizer. Its individual adaptive learning rates and the adaptive estimation of lower-order moments of the gradients maybe cited as a cause for the described observations. It is worth mentioning, according to LeCun et al. [20] second-order optimizers are impractical in almost all useful cases.

B. Parameterization of $SO(3)$

a) Euler Angles: A robot's end-effector orientation is commonly represented in terms of a minimal number of parameters, typically three angles. A set of three valid angles with appropriate restrictions are Euler angles. The restriction is needed in order to have a unique representation [19]. There are twelve different sets of Euler angles, each having singularities depending on the sequence. Therefore, all twelve sequences of Euler angles are empirically studied in this paper. Throughout this paper, Diebels [8] compact notation for Euler angles is used in order to indicate the rotation sequences. For instance 123, which is equivalent to rotating a vector about the fixed x-axis (1^{st}-axis), followed by a rotation about the fixed y-axis (2^{nd}-axis), followed by a rotation about the fixed z-axis (3^{rd}-axis). Hence, 123 is sometimes referred as XYZ.

b) Vectorial Parameterization: For a given strictly increasing smooth scalar function $f(\vartheta)$ defined in the semi-open interval $]-\vartheta, \vartheta]$ with $f(0) = 0$ and $\vartheta > 0$, the orientation displacement can be expressed as [6]

$$\boldsymbol{r} = f(\vartheta)\,\boldsymbol{n}, \tag{1}$$

where $\boldsymbol{n} = (n_x, n_y, n_z)^T$ and ϑ are, respectively, the unit vector and the rotation of an equivalent angle/axis representation of orientation. While there exist multiple possibilities, the discussion in [2] shows that two subclasses of vectorial parameterization, namely $f(\vartheta) = \mu \tan(\vartheta/\mu)$ and

$$f(\vartheta) = \mu \sin(\vartheta/\mu) \tag{2}$$

have interesting properties, where $\mu \in \mathbb{N} \setminus \{0\}$. We are interested in the latter subclass (2). We chose the sine function due to the fact that it is bounded between ± 1. Furthermore, we set $\mu = 4$ because in this case (2) is a strictly increasing smooth scalar function for all considered angles, i.e. $0 \leq \vartheta < 2\pi$.

Using basic algebraic transformations, angle ϑ and axis $\boldsymbol{n}$ in (1) with (2) and $\mu = 4$ can be calculated. This leads to

$$\vartheta = 4 \arcsin\left(\frac{||\boldsymbol{r}||_2}{4}\right) \quad \text{and} \tag{3}$$

$$\boldsymbol{n} = \frac{\boldsymbol{r}}{||\boldsymbol{r}||_2}, \tag{4}$$

where the unit vector $\boldsymbol{n}$ can only be recovered if $\vartheta \neq 0\,\text{rad}$, which is equal to $||\boldsymbol{r}||_2 \neq 0$.

c) Quaternions: We briefly review quaternions and provide the formulae in order to compute the orientation angle ϑ and the fixed axis $\boldsymbol{n}$. In addition, we compare the rotational approximation error in terms of unit quaternions.

A unit quaternion is a hypercomplex number denoted by

$$\xi = \eta + \epsilon_1 \imath + \epsilon_2 \jmath + \epsilon_3 k \tag{5}$$

with the property $\eta^2 + \epsilon_1^2 + \epsilon_2^2 + \epsilon_3^2 = 1$. The quaternionic units $\imath$,$\jmath$, and k satisfy Hamilton's rule $\imath^2 = \jmath^2 = k^2 = -1$ and $\imath\jmath k = -1$. A unit quaternion ξ defined by

$$\xi = \cos(\vartheta/2) + (n_x \imath + n_y \jmath + n_z k) \sin(\vartheta/2) \tag{6}$$

represents an orientation. This representation of $SO(3)$ is singularity-free and global [31]. Its double coverage property of $SO(3)$, also known as antipodal property, i.e. ξ and $-\xi$ represent the same orientation, can be solved by applying $\xi' = \operatorname{sign}(\eta)\xi$, where $\operatorname{sign}(\eta)$ gives the sign of η and $\operatorname{sign}(0)$ is $+1$ by definition.

Using (6), we can easily determine the orientation angle

$$\vartheta = 2 \arccos(\operatorname{sign}(\eta)\eta) \tag{7}$$

and the fixed axis of the orientation

$$\boldsymbol{n} = \frac{\operatorname{sign}(\eta)}{\sqrt{\epsilon_1^2 + \epsilon_2^2 + \epsilon_3^2}} (\epsilon_1, \epsilon_2, \epsilon_3)^T . \tag{8}$$

Note that $\boldsymbol{n}$ can be computed if and only if $\eta^2 \neq 1$ and $\epsilon_1^2 + \epsilon_2^2 + \epsilon_3^2 \neq 0$.

d) Mapping between Quaternions and Vectorial Parameterization: The coefficients of a quaternion (5) can be expressed in terms of (1) and (2). This leads to

$$\eta = \cos\left(2 \arcsin\left(\frac{||\boldsymbol{r}||_2}{4}\right)\right) = 1 - \frac{1}{8}||\boldsymbol{r}||_2^2 \quad \text{and} \tag{9}$$

$$\epsilon_i = \frac{1}{8} r_i \sqrt{16 - ||\boldsymbol{r}||_2^2} \quad \text{for } i \in \{1,2,3\}, \tag{10}$$

where r_i are components of (1). For both (9) and (10) we take advantage of the double-angle formulae for sine and cosine functions and only consider the special case $\mu = 4$. Combining (9) with (10) and rearranging for r_i yields

$$r_i = \frac{4 \operatorname{sign}(\eta)}{\sqrt{2 + 2\eta \operatorname{sign}(\eta)}} \epsilon_i \quad \text{for } i \in \{1,2,3\}, \tag{11}$$

where $\operatorname{sign}(\eta)$ is added in order to avoid zero division.

e) Four-Parameter Angle-Axis Representation: The angle-axis representation is commonly defined as (1) with $f(\vartheta) = \vartheta$. Therefore, the commonly used angle-axis representation has three parameters. However, we construct and use a four-dimensional version including the unit vector $\boldsymbol{n}$ and the rotation ϑ. This four-parameter representation $(\vartheta, \boldsymbol{n}) = (\vartheta, n_x, n_y, n_z)$ serves as comparison to quaternions being also a four-parameter representation.

C. Approximation Error: Cartesian Space

Here, we describe the approximation error for the FK. The translational error e_t is given by

$$e_t = \sqrt{\left(t_x - \hat{t}_x\right)^2 + \left(t_y - \hat{t}_y\right)^2 + \left(t_z - \hat{t}_z\right)^2}, \tag{12}$$

where t_i and $\hat{t}_i$ are the positions along the corresponding Cartesian axis. The hat symbol is used to denote an approximated value computed by an artificial neural network. Otherwise it is the ground truth value, i.e. test set $\mathcal{S}_{\text{test}}$. The rotational error e_ϑ is considered as

$$\begin{aligned} e_\vartheta = \min \{ & 2 \arccos(\eta\hat{\eta} + \epsilon_1\hat{\epsilon}_1 + \epsilon_2\hat{\epsilon}_2 + \epsilon_3\hat{\epsilon}_3), \\ & 2 \arccos(\eta\hat{\eta} - \epsilon_1\hat{\epsilon}_1 - \epsilon_2\hat{\epsilon}_2 - \epsilon_3\hat{\epsilon}_3) \}, \end{aligned} \tag{13}$$

which respects the antipodal property of quaternions. The rotational error (13) computes the shortest distance between the quaternions ξ and $\hat{\xi}$. It can be derived from (7), where its argument is the quaternionic multiplication between ξ and the quaternionic conjugate of $\hat{\xi}$. If the vectorial parameterization is used, $\hat{\xi} = \hat{\eta} + \hat{\epsilon}_1\imath + \hat{\epsilon}_2\jmath + \hat{\epsilon}_3 k$ for (13) is determined by (9) and (10), where $\hat{\boldsymbol{r}}$ is computed by an artificial neural network. If one of the twelve Euler angles' is applied, a mapping from respective Euler angles to quaternion is used, see Diebel [8] for reference. Note that to satisfy the property $\hat{\eta}^2 + \hat{\epsilon}_1^2 + \hat{\epsilon}_2^2 + \hat{\epsilon}_3^2 = 1$, the quaternion $\hat{\xi}$ must be normalized.

D. Various Joint Space Representations of a CTCR

The 6-DOF of a CTCR are composed of translations, denoted β_i, and axial rotations, denoted α_i, of each tube. The subscript i refers to the respective tube. These 6-DOF can be seen as a composite of three lower pair connections, more specifically a composite of three 2-DOF cylindrical joint. The translations β_i are interdependent with the inequalities

$$0 \geq \beta_3 \geq \beta_2 \geq \beta_1 \quad \text{and} \tag{14}$$

$$0 \leq L_3 + \beta_3 \leq L_2 + \beta_2 \leq L_1 + \beta_1, \tag{15}$$

where L_i is the overall length of the i^{th} tube. In the following, we describe various representations of the i^{th} cylindrical joint. Figure 3 shows the notation and gives a visual aid.

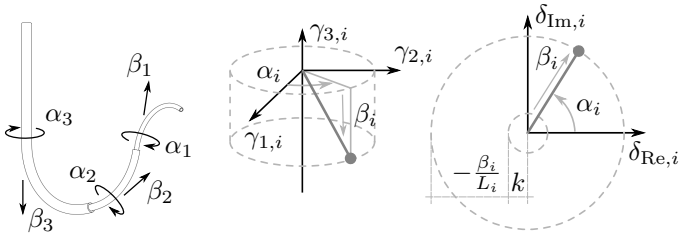

Fig. 3. Tubes of a concentric tube continuum robot can be rotated, denoted α_i, and translated, denote β_i, w.r.t. each other. The joint space representation in cylindrical form (17) and polar form (19) can be visualized as cylindrical coordinate system and Argand diagram on the complex plane, respectively.

a) Simple Form: A way to describe joints is

$$q_i = \{\alpha_i, \beta_i\}, \tag{16}$$

which is commonly used. Regarding neural network applications, Bergeles et al. [3] use this joint space representation given by (16) in order to compute the kinematics of a CTCR.

b) Cylindrical Form: In order to achieve higher accuracy, Grassmann et al. [14] subdivide the joint space $\mathcal{Q}$ into a rotational joint space $\mathcal{A}$ and a translational joint space $\mathcal{B}$. Afterwards, they take advantage of the fact that the elements of $\mathcal{A}$ denoted by $\alpha_i \in \mathbb{S}^1$ being a 1-sphere can be defined as a circle. The transformation by means of trigonometric functions leads to their proposed cylindrical joint representation

$$\gamma_i = \{\gamma_{1,i},\ \gamma_{2,i},\ \gamma_{3,i}\} = \{\cos(\alpha_i), \sin(\alpha_i), \beta_i\}, \tag{17}$$

which describes the i^{th} tube as a triplet. From (17) follows

$$\alpha_i = \operatorname{atan2}(\gamma_{2,i},\ \gamma_{1,i}), \tag{18}$$

which gives the unambiguous correct value of α_i in the respective quadrants.

c) Polar Form: A minor disadvantage of (17) is that the number of parameters increases. In order to avoid this, we propose a novel cylindrical joint representation given by

$$\begin{aligned}\delta_i &= \{\delta_{\mathrm{Re},i},\ \delta_{\mathrm{Im},i}\} \\ &= \{(k - \beta_i/L_i)\cos(\alpha_i), (k - \beta_i/L_i)\sin(\alpha_i)\},\end{aligned} \tag{19}$$

where $k > 0$ is a hyper-parameter in order to avoid zero radius if β_i is zero. Note that $\beta_i \leq 0$ and $0 \leq -\beta_i/L_i \leq 1$. Similar to (18), the correct rotary joint α_i can be obtained by

$$\alpha_i = \operatorname{atan2}(\delta_{\mathrm{Im},i},\ \delta_{\mathrm{Re},i}). \tag{20}$$

By rearranging (19) and solving for β_i, the translational joint

$$\beta_i = L_i\left(k_i - \sqrt{\delta_{\mathrm{Im},i}^2 + \delta_{\mathrm{Re},i}^2}\right) \tag{21}$$

can be recovered as well. Due to the fact that term $k_i - \beta_i/L_i$ of (19) is always positive, the solution in (21) is unique. Moreover, atan2 in (20) is influenced neither by $k_i > 0$ nor by $\beta_i \leq 0$.

III. DATA ACQUISITION

Here we give a brief overview of the testbed and the used robots. Moreover, we provide equations for time efficient sampling considering inequalities, which is also suitable for scaling.

A. Experimental Setup

In order to compare the effects of joint space representations on the approximation accuracy, we generate and acquire data for a serial kinematic manipulator and a CTCR. We selected these two robot types for our study as their joint space exhibits similar characteristics, i.e. a composition of translational and rotational joints. Furthermore, both robot types have 6-DOF in task space and 6-DOF in joint space. More importantly, the Stanford Arm has relatively simple kinematics, which can be described by the concatenation of multiple transformation matrices, while the CTCR is a more kinematically complex robot due to highly non-linear behavior between its flexible tubes. Roughly speaking, both robot types lay on the opposite side of the spectrum of the kinematic modeling complexity. In order to bound the amount of data, the joint spaces are restricted, see Table I and Table II.

Positions and quaternions are considered as the ground truth values for the comparison. The method proposed by Bar-Itzhack [1] is used in order to convert a rotation matrix to a quaternion. If one of the twelve Euler angles' is applied, a mapping from quaternion to respective Euler angles is utilized, see Diebel [8] for reference. To determine the vector parametrization $\boldsymbol{r}$ and the four-parameter angle-axis representation $(\vartheta, \boldsymbol{n})$, we apply Eq. 11 and Eq. 7-8, respectively.

TABLE I
SIMULATED STANFORD ARM.

	Restricted joint space		Denavit-Hartenberg parameters [23]			
i	min	max	α in $^\circ$	d in mm	θ in $^\circ$	a in mm
1	-90°	90°	-90	412.50	q_1	0
2	-45°	45°	90	153.67	q_2	0
3	0.5 m	0.75 m	0	q_3	-90	0
4	-90°	90°	-90	0	q_4	0
5	-25°	25°	90	0	q_5	0
6	-90°	90°	0	262.89	q_6	0

TABLE II
FULL AND CONSTRAINED JOINT SPACE OF A CTCR.

	Simulation: full joint space				Real robot: constrained joint space			
	α_i in $^\circ$		β_i in mm		α_i in $^\circ$		β_i in mm	
i	min	max	min	max	min	max	min	max
1	-180	180	-205	0	-60	60	-144	0
2	-180	180	-164	0	-60	60	-115	0
3	-180	180	-115	0	-60	60	-81	0

TABLE III
TUBE PARAMETERS OF THE CTCR.

Parameter			Set of tubes		
Term	Symbol	Unit	Tube 1 (innermost)	Tube 2 (middle)	Tube 3 (outer)
Length, overall	L	mm	370	305	170
Length, straight	L_s	mm	325	205	70
Curvature	κ_x	m^{-1}	15.8	9.27	4.37
Diameter, outer	D_o	mm	0.4	0.9	1.5
Diameter, inner	D_i	mm	0.3	0.7	1.2
Young's Modulus	E	GPa	50	50	50
Poisson's ratio	ν	1	0.3	0.3	0.3

a) Stanford Arm: It consists of two revolute joints, followed by a prismatic joint, followed by three intersecting revolute joints. Paul [23] describes its FK. The Denavit-Hartenberg parameters are listed in Table I. For this robot, we generate $\mathcal{S} = 500{,}000$ joint space/end-effector pair samples in simulation.

b) CTCR: It consists of three tubes. Mechanical properties and geometrical parameters of the CTCR are listed in Table III. The geometrically exact model proposed by Rucker et al. [29] is applied in order to compute its FK. For the CTCR in simulation $\mathcal{S} = 500{,}000$ joint-space/end-effector pose pair samples are generated.

c) Real CTCR Prototype: Similar to the CTCR in simulation, it consist of three tubes (mechanical properties and geometrical parameters are listed in Table III). The CTCR prototype is depicted and described in Fig. 4. We gathered $\mathcal{S} = 94{,}000$ measured data with the CTCR prototype. The testbed consists of a 6-DOF sensor attached at the distal tip of the CTCR prototype, and an electromagnetic tracking system (AURORA, Northern Digital Inc., ON, Canada). The tracking system provides positions and quaternions.

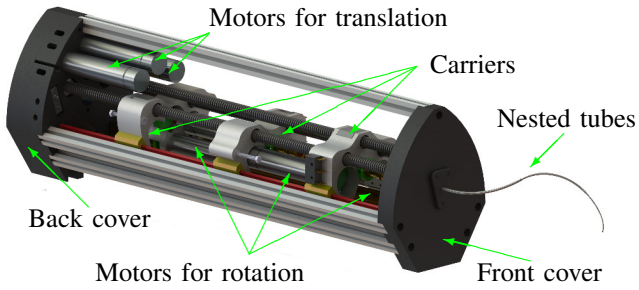

Fig. 4. CTCR prototype has three carriers each consisting a tube and a motor for rotation. Motors for translation are attached at the back cover. Six equal motors (DCX 16 L, Maxon Motor AG, OW, Switzerland) are controlled with a motion control board (DCM4163, Galil Motion Control, CA, USA).

B. Sampling for CTCR

Samples of a given robot's joint space are generated from a uniform random distribution $\mathcal{U}$. This is straightforward for the Stanford Arm as no dependencies between the joint values exist. In contrast, sampling the joint space of a CTCR requires consideration of the inequalities in (14) and (15).

$\mathcal{A}$ is a 3-torus $\mathbb{T}^3$ and can be described as a cube, where the opposite sides of this cube are pasted together. Note the property of $\mathbb{S}^1$ has to be considered, i.e. $\alpha_i = \alpha_i + 2\pi k$ for $k \in \mathbb{N}$. Taking into account the independence of the rotational parameters α_i among each other, sampling in the rotational joint space $\mathcal{A}$ is straightforward, i.e. the p^{th} value can be computed by $\alpha_i^{(p)} = \mathcal{U}\,]\alpha_{i,\min}; \alpha_{i,\max}]$.

Due to the inequalities (14) and (15), the translational parameters β_i are interdependent. Nevertheless, all valid configurations in $\mathcal{B}$ fulfilling (14)-(15) and are within a parallelepiped. Furthermore, the edges of the parallelepiped intersecting the origin define the bases of the parallelepiped. Hence, an affine transformation $M_{\mathcal{B}}$ between an unit cube and parallelepiped can be found, which is given by

$$M_{\mathcal{B}} = \begin{bmatrix} \beta_{1,\min} - \beta_{2,\min} & \beta_{2,\min} - \beta_{3,\min} & \beta_{3,\min} \\ 0 & \beta_{2,\min} - \beta_{3,\min} & \beta_{3,\min} \\ 0 & 0 & \beta_{3,\min} \end{bmatrix}, \quad (22)$$

where the columns of $M_{\mathcal{B}}$ are the bases of the parallelepiped and $\beta_{i,\min}$ are the minimum translation value and negative overall tube length L_i of the respective tube. Figure 5 provides additional visual aid. Drawing samples out of the unit cube via $\mathcal{U}^3\,[0;1]$ and, afterwards, applying (22) can be used for time efficient sampling in $\mathcal{B}$.

Note that (22) is not unique. A second affine transformation

$$M_{\mathcal{B},2} = \begin{bmatrix} \beta_{1,\min} & 0 & 0 \\ \beta_{1,\min} & \beta_{2,\min} - \beta_{1,\min} & 0 \\ \beta_{1,\min} & \beta_{2,\min} - \beta_{1,\min} & \beta_{3,\min} - \beta_{2,\min} \end{bmatrix} \quad (23)$$

can be inferred from the used sampling method by Burgner-Kahrs et al. [4]. The pattern for n tubes can be easily seen from (22) and (23) leading to a upper triangular matrix and lower triangular matrix, respectively.

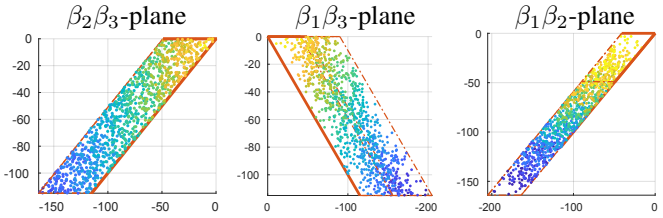

Fig. 5. Projected points of 1000 random samples in $\mathcal{B}$ onto the respective plane. All samples lie in the parallelepiped spanned by its inequalities (14) and (15) and indicated by its edges (dash-dot red lines). Column vectors (red lines) of (22) go through the origin. For the sake of visual aid, selected colormaps of the respective plane spanned by two different joints imply the joint perpendicular to it.

IV. FORWARD KINEMATICS

In the following, the approximation of the FK is evaluated. The aim is to investigate the influence of different orientation and joint space representation on the FK for the two robot types.

The measured data set $\mathcal{S} = 94{,}000$ is divided in training set $\mathcal{S}_{\text{tra}} = 90{,}000$ and test set $\mathcal{S}_{\text{test}} = 4000$. The model-based data set $\mathcal{S} = 500{,}000$ is divided in training set $\mathcal{S}_{\text{tra}} = 490{,}000$ and test set $\mathcal{S}_{\text{test}} = 10{,}000$. According to [14], no validation set $\mathcal{S}_{\text{val}}$ is needed if the Adam optimizer [18] is applied. The performance of the FFN w.r.t. the chosen representations are measured by the median error of e_ϑ given by (13) and of e_t given by (12) on the respective unseen test set $\mathcal{S}_{\text{test}}$ at epoch $N_{\text{ep}} = 500$. Respective $\mathcal{S}_{\text{tra}}$ is only utilized for training.

A. Fully Connected Feedforward Network

The input layer expects a joint description q defined by (16), (17), or (19) with $k = 1$. Moreover, we apply different scaling to the joint space representations, which is listed in Table IV. These normalizations are considered such that the inputs are defined in an interval $[-1, 1]$ or $[0, 1]$. The output layer learns Euler angles, vectorial parameterization (1) with (2) for $\mu = 4$, four-parameter angle-axis representation $(\vartheta, \boldsymbol{n})$, or quaternions ξ. Therefore, poses are represented by $[\boldsymbol{r}, \boldsymbol{t}]$, $[(\vartheta, \boldsymbol{n}), \boldsymbol{t}]$, or $[\xi, \boldsymbol{t}]$. Neither orientation nor position $\boldsymbol{t}$ of the pose are scaled. The unit of $\boldsymbol{t}$ is meter whereas the unit of rotation depending on the considered representation is one or radian. For instance, the unit of quaternion ξ is one and the unit of Euler angles are radian. In order to have almost the same amount of parameters, i.e. weights ω and biases b, along different used FFN, we compute the number of parameters, i.e. the complexity $\mathcal{C}$ of the FFN, and adjust the number of activation functions in the hidden layer. Table V lists the applied architecture and learning parameters.

TABLE IV
APPLIED SCALING FOR THE JOINT SPACE REPRESENTATION.

Legend[1]	Description of the scaling
— q_i (I)	No scaling is applied to the joints of the Stanford Arm.
— q_i (II)	Each joint is scaled via the maximum value, cf. Table I.
— γ_i (IV)	Rotary joints of the Stanford Arm are transformed by means of trigonometric functions similar to (17).
— γ_i (V)	Rotary joints are transformed by means of trigonometric functions, whereas the prismatic joint is scaled by $0.75\,\mathrm{m}$.
— α_i, β_i (I)	No scaling is applied to the joints of the CTCR.
— α_i, β_i (II)	Each joint of the CTCR is scaled by the absolute value, i.e. $\alpha_{i,\max}$ and $\beta_{i,\min}$, cf. Table II.
— α_i, β_i (III)	α_i is scaled by $\alpha_{i,\max}$ whereas β_i is transformed into an unit cube utilizing the inverse of (22).
— γ_i (IV)	The cylindrical form γ_i is applied, see (17).
— γ_i (V)	Rotary joints of the CTCR are transformed by means of trigonometric function similar to (17) while β_i is transformed by the inverse of (22).
— δ_i (VI)	The polar form δ_i is used, which is given by (19).

[1]Notation in the legend used in Fig. 6, Fig. 7, and Fig. 8.

All FFN were implemented in Tensorflow 1.9, running on a 64-bit Linux operating system, on a computer with a Xeon $3.60\,\mathrm{GHz} \times 8$ processor.

B. Results

In Fig. 6 and Fig. 7 courses of e_ϑ with model-based data are presented. Figure 8 shows courses of e_ϑ with measured robot data. For the sake of compactness, the compact notation [8] for Euler angles indicating the rotation sequences is used, e.g. 123, and the minimum errors $e_{\vartheta,\min}$ as well as $e_{t,\min}$ are reported in Table VI. The minimum errors of all different joint space representations for a particular orientation representation for the respective robot are highlighted (bold font). Regarding CTCR with $[\xi, \boldsymbol{t}]$ and γ_i, we achieve a median error of $e_t = 1.94\,\mathrm{mm}$ and $e_\vartheta = 3.48°$ on measured data. By applying $M_{\mathcal{B}}^{-1}$ onto γ_i, $e_t = 2.02\,\mathrm{mm}$ and $e_\vartheta = 1.64°$ have been determined. While $e_t = 1.79\,\mathrm{mm}$ and $e_\vartheta = 1.46°$ are achieved if α_i, β_i(III) is used. In the best case for ξ, $e_\vartheta = 1.46°$ and $e_t = 1.79\,\mathrm{mm}$ being less then $0.87\,\%$ w.r.t. the robot length of $205\,\mathrm{mm}$ are achieved. $e_\vartheta = 1.43°$ and $e_t = 1.62\,\mathrm{mm}$ have been determined w.r.t. $(\vartheta, \boldsymbol{n})$. Thus, a relative error of $0.79\,\%$ w.r.t. the robot length is achieved.

TABLE V
FEEDFORWARD NETWORKS FOR FORWARD KINEMATIC APPROXIMATION.

Architecture with ReLU			Training with Adam optimizer[2]			Weights & bias[3]
N_{ip}	N_{h}	N_{op}	N_{bs}	N_{ep}	λ	$\mathcal{C}$
6	100	6	128	500	3×10^{-5}	1306
6	93	7	128	500	1×10^{-5}	1309
9	81	6	128	500	3×10^{-5}	1302
9	77	7	128	500	1×10^{-5}	1316
11	68	7	128	500	1×10^{-5}	1299
11	72	6	128	500	1×10^{-5}	1302

[2] Beside of learning rate λ and following the notation in [18], it is further parametrized with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 1 \times 10^{-8}$.
[3] The complexity $\mathcal{C}$ is the sum of all weights and bias in the neural network. For this shallow neural network it can be computed by $\mathcal{C} = (1 + N_{\mathrm{ip}})N_{\mathrm{h}} + (1 + N_{\mathrm{h}})N_{\mathrm{op}}$.

C. Discussion

For both robot types, we can conclude that all Euler angles as well as vectorial parametrization are unfavorable orientation representations. All FFN with this orientation representation as part of their output generalize poorly and the improvement saturates quickly, which can be observed in Fig. 6, Fig. 7, and Fig. 8. e_ϑ and e_t are mainly influenced by the selected orientation representation. The performance of Euler angles highly depends on the chosen sequences. For instance most sequences with repeated axis of rotation, e.g. 121, perform poorly. The chosen sequence of elementary rotation has different singularities [8]. Poor performances resulting from the decrease in inputs cannot be cited as a reason because all complexities $\mathcal{C}$ of the FFN are normalized, cf. Table V. Remarkably, the used vectorial parametrization with (2) and $\mu = 4$ performs poorly, although it is constructed in order to avoid singularities due to the fact that the first quarter of the sine-function is bijective. According to Stuelpnagel [31], it should be noted that none of the representations of $SO(3)$ with three parameters can be both global and nonsingular. From Stuelpnagel's [31] statement as well as from the results, we hypothesize that none of the three-parameter representation of $SO(3)$ including vectorial parameterization (1) and all sets of Euler angles are suitable for machine learning applications. However, γ_i and utilizing $M_{\mathcal{B}}^{-1}$ clearly shows an increase of accuracy w.r.t. the unfavorable orientation representation.

In comparison, utilizing quaternions, errors e_ϑ and e_t are noticeably smaller regardless of the joint space representation and robot type. In some cases, the approximation is better by utilizing $(\vartheta, \boldsymbol{n})$. However, if $\vartheta = 0$, an arbitrary axis $\boldsymbol{n}$ can be chosen being undesirable in machine learning application. Note that quaternions automatically squash values of $SO(3)$ between -1 and 1, which is suitable for a FFN.

Regarding the Stanford Arm, the accuracy of conventional kinematic modeling approaches [24] are not readily achievable with neural networks. However, the results reveal that advantageous joint space and orientation representations such as quaternions are mandatory even for low dimensional applications. We are confident that with our investigations, higher dimensional machine learning applications in robotics can be improved.

In contrast to our previous work [14], here we use a 20 times smaller learning rate λ, $23\,\%$ fewer learnable parameters (previous complexity $\mathcal{C} = 1707$), and a slightly different version of e_ϑ. Moreover, the previous unit of the position $\boldsymbol{t}$ is millimeter instead of meter. We observe that approximating the FK of a CTCR with Euler angles is not achievable if the unit of $\boldsymbol{t}$ is millimeter. Furthermore, this difference in scaling may reinforce the effect observed by us [14], where γ_i outperformed untransformed α_i and β_i. By comparing the course of α_i, β_i(I) with γ_i(IV), this effect cannot be observed in Fig. 8. However, in Fig. 7 joint space representation γ_i outperforms all variants of α_i, β_i if more data is used during the training. Note that the workspace has been enlarged.

Regarding δ_i, all FFN approximate the FK worse due to the

TABLE VI
BEST ACHIEVED RESULTS ON THE RESPECTIVE TEST SET $\mathcal{S}_{\text{TEST}}$ AMONG ALL APPLIED JOINT SPACE REPRESENTATIONS FOR DIFFERENT ORIENTATION.

robot[4]	error[5]	121	123	131	132	212	213	231	232	312	313	321	323	vp[6]	ξ	$(\vartheta, \boldsymbol{n})$
sStA	$e_{\vartheta,\text{min}}$	13.6	6.6	41.5	32.6	14.8	6.6	29.6	42.4	5.5	20.7	5.5	25.1	7.1	**4.0**	5.5
sStA	$e_{t,\text{min}}$	43.3	32.7	55.0	54.8	42.3	33.0	52.7	58.8	26.9	46.4	26.9	45.8	32.8	**15.2**	22.3
sCTCR	$e_{\vartheta,\text{min}}$	18.0	15.7	29.7	30.7	16.9	15.5	23.5	23.7	15.5	20.8	15.5	21.9	15.4	**9.0**	9.1
sCTCR	$e_{t,\text{min}}$	13.4	13.8	14.6	14.3	13.3	12.9	13.4	14.0	11.7	13.3	11.8	12.9	12.5	**7.6**	7.8
rCTCR	$e_{\vartheta,\text{min}}$	50.4	11.7	10.3	10.3	21.0	10.2	10.3	13.2	10.3	10.3	11.4	32.2	10.3	**1.4**	**1.4**
rCTCR	$e_{t,\text{min}}$	7.7	7.4	7.2	7.2	7.7	7.3	7.2	7.6	7.4	7.2	7.4	7.5	7.2	1.8	**1.6**

[4] sStA: Stanford Arm, sCTCR: concentric tube continuum robot in simulation, rCTCR: real robot prototype.
[5] Approximation error at $N_{\text{ep}} = 500$ is the smallest median error among all joint space representations. $e_{t,\text{min}}$ in mm and $e_{\vartheta,\text{min}}$ in $^\circ$.
[6] vp: vector parametrization.

interconnections of δ_i, cf. (19). Hence, δ_i is a disadvantageous joint space representation concerning machine learning application due to the non-linear correlation between α_i and β_i. However, from a mathematical point of view, it can be seen that δ_i is suitable for the inverse kinematics, cf. (20) and (21). Moreover, the parameter k_i in (19) can be further improved.

Using the cylindrical form γ_i for both robot types, the accuracy and convergence can be drastically improved. Transforming the inputs can be seen as preprocessing on the inputs leading to a functional link network, see Flak [10] for reference. Including γ_i explicitly as inputs into the FFN is beneficial instead of forcing the network to learn a potentially difficult-to-model concept. This can result in simpler FFN with lower complexity $\mathcal{C}$ that can lead in an overall shorter training time and more accurate approximation.

In comparison with a state-of-the-art model-based approach for CTCR [29], we achieve almost twice better accuracy w.r.t. the relative error e_t of $0.87\,\%$ and $0.79\,\%$ by utilizing four-parameter representations, whereas our range of actuation is $50\,\%$ larger regarding the maximum translation. Even when incorporating external loading, e.g. 6-DOF sensor, a minimal tip error of $2.91\,\text{mm}$ and $1.5\,\%$ w.r.t. the robot length is reported by Rucker et al. [29]. Note that a FFN can account for effects which are challenging to model, such as friction. However, a FFN cannot model hysteresis because it is a path dependent and dynamic phenomenon.

Furthermore, the accuracy can be improved by transforming the translational joint space $\mathcal{B}$ into a unit cube via the inverse of (22), i.e. $M_{\mathcal{B}}^{-1}$. This can be clearly seen from the results w.r.t. both CTCR. The accuracy is increased and the convergence is faster, which can be seen in the course of $\alpha_i, \beta_i(\text{III})$ and $\gamma_i(\text{V})$ in Fig. 7 and in Fig. 8. Even for unfavorable orientation representations, an increase of accuracy can be observed. This is probably due to the following three reasons. First, the space spanned by the input regarding β_i may be better covered by an unit cube than by a (scaled) parallelepiped. Roughly said, the parallelepiped has unused space, cf. Fig. 5. Second, the FFN does not need to implicitly approximate the inequalities in (14) and (15), which obviously cannot be adequately modeled by the FFN. Third and more importantly, the inequalities can be modeled as a transformation given by $M_{\mathcal{B}}$, which is a linear transformation. Hence, β_i are linear correlated, which is generally an undesirable property of an input. In this case LeCun et al. [20] suggest to use principal component analysis (PCA) in order to remove linear correlations in inputs. The present results and the previous analysis, cf. Fig. 5, indicate that the translational joint space $\mathcal{B}$ can be uncorrelated if (22) is applied. As a consequence, we found a simple and effective transformation capable of decorrelating the translational joint space $\mathcal{B}$ leading to an improvement in accuracy and acceleration of the convergence. It can also be used as fast sampling method.

V. CONCLUSIONS

In this paper, we considered the problem of accurate kinematics calculation of robot types with cylindrical joints, especially the Stanford Arm and a concentric tube continuum robot. Artificial neural networks with different pose and joint space representations have been applied in order to approximate the forward kinematics with model-based and measured experimental data. By analyzing the influence of different pose and joint space representations, we observe that four-parameter orientation representations achieve at least three times higher accuracy, in comparison with other $SO(3)$ representations. Hence, we suggest the use of singularity free quaternion/vector-pairs for $SE(3)$ representation in machine learning applications. Learning the forward kinematics of the Stanford Arm reveals that advantageous joint space and orientation representations are mandatory even for low dimensional applications. Moreover, the approximation error in translation and orientation can be further reduced by utilizing the presented affine transformation $M_{\mathcal{B}}$ to the translational joint space $\mathcal{B}$ of a concentric tube continuum robot. In future work we intend to study the influence of different representations on the inverse kinematics as well.

Overall, we proved our hypothesis that the joint and orientation representation has a major impact on the effectiveness of learning the forward kinematics. It has yet to proven that this observation can be generalized to other robot types.

ACKNOWLEDGMENTS

This work was supported by the German Research Foundation (DFG) under grant no. BU 2935/1-1. Furthermore, the authors would like to thank Thien-Dang Nguyen for his valuable contribution by building the robot prototype.

REFERENCES

[1] Itzhack Y. Bar-Itzhack. New method for extracting the quaternion from a rotation matrix. *Journal of guidance, control, and dynamics*, 23(6):1085–1087, 2000.

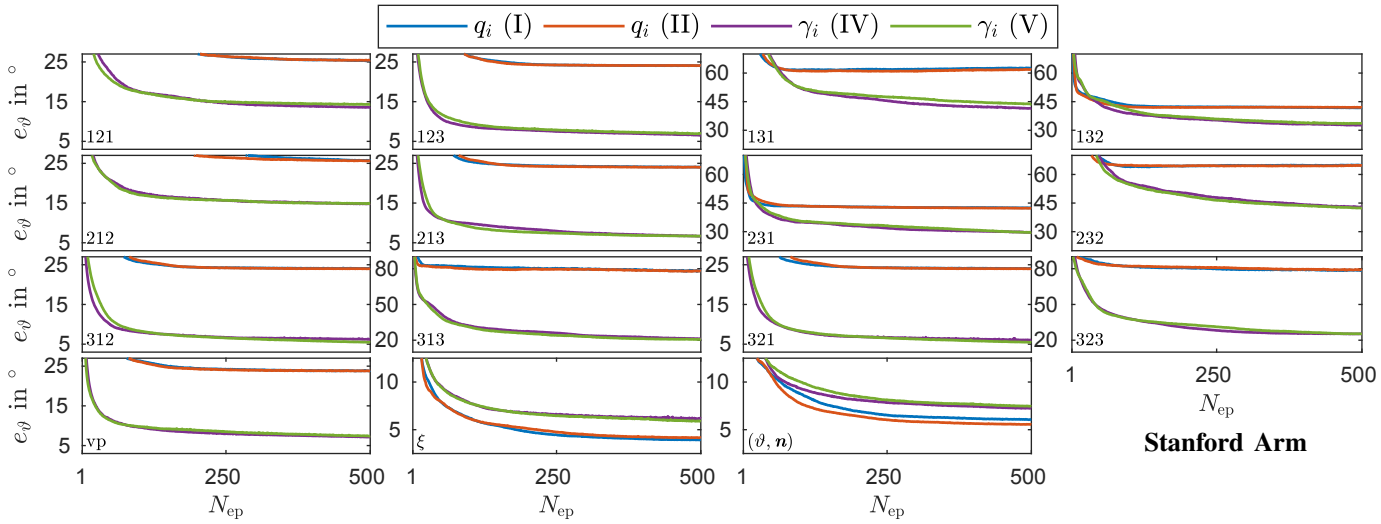

Fig. 6. Courses of approximation error e_ϑ of all twelve Euler angles (compact notation [8], e.g. 121), vectorial parameterization (vp), quaternion ξ and four-parameter angle-axis representation $(\vartheta, \boldsymbol{n})$, respectively, for different joint space representations are evaluted with the test set $\mathcal{S}_{\text{test}}$ of a simulated Stanford Arm. Please note the different scaling in y-axes. Joint representation q_i (I), q_i (II), γ_i (III), and γ_i (IV) are described in Table IV.

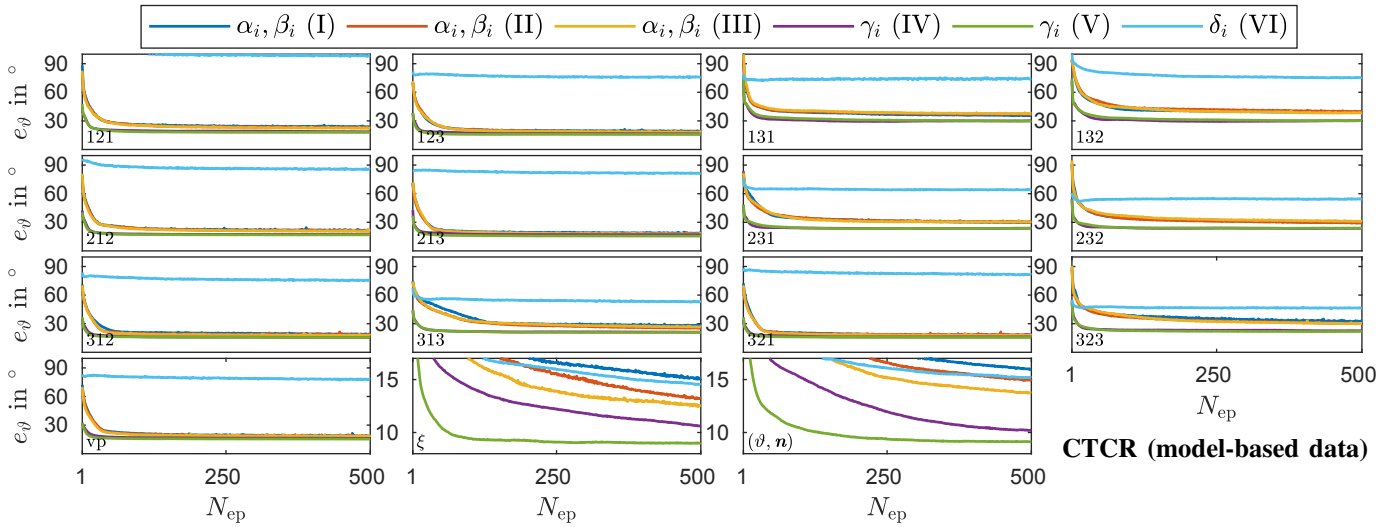

Fig. 7. Courses of approximation error e_ϑ are evaluated with the test set $\mathcal{S}_{\text{test}}$ including model-based data. Please note the different scaling. Joint space representation α_i, β_i (I), α_i, β_i (II), α_i, β_i (III), γ_i (IV), γ_i (V), and δ_i (VI) in the legend are described in Table IV.

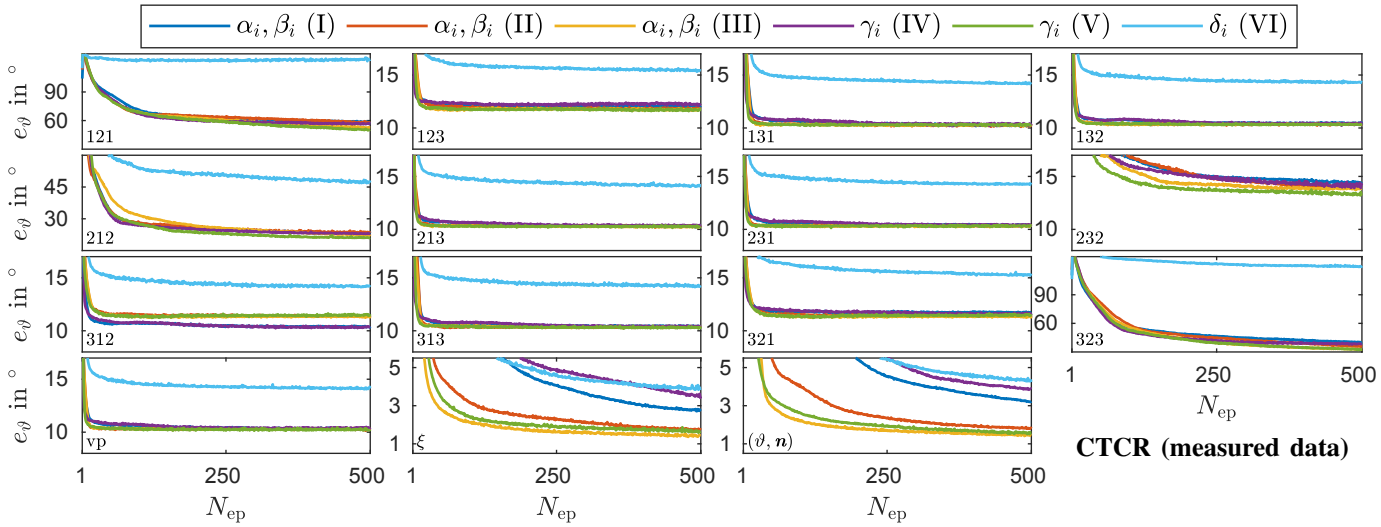

Fig. 8. Courses of approximation error e_ϑ w.r.t. the test set $\mathcal{S}_{\text{test}}$ with measured data of a CTCR prototype. Please note the different scaling in y-axes. The various joint space representations indicated in the legend are described in Table IV.

[2] Olivier A. Bauchau and Lorenzo Trainelli. The vectorial parameterization of rotation. *Nonlinear dynamics*, 32(1): 71–92, 2003.

[3] C. Bergeles, F.-Y. Lin, and G.-Z. Yang. Concentric tube robot kinematics using neural networks. In *Hamlyn Symposium on Medical Robotics*, pages 13–14, 2015.

[4] Jessica Burgner-Kahrs, Hunter B. Gilbert, Josephine Granna, Philip J. Swaney, and Robert J. Webster. Workspace characterization for concentric tube continuum robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1269–1275, 2014.

[5] Jessica Burgner-Kahrs, D. Caleb Rucker, and Howie Choset. Continuum robots for medical applications: A survey. *IEEE Transactions on Robotics*, 31(6):1261–1280, 2015.

[6] Fabrizio Caccavale, Ciro Natale, Bruno Siciliano, and Luigi Villani. Six-dof impedance control based on angle/axis representations. *IEEE Transactions on Robotics and Automation*, 15(2):289–300, 1999.

[7] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems*, 2(4):303–341, 1989.

[8] James Diebel. Representing attitude: Euler angles, unit quaternions, and rotation vectors. Technical report, Stanford University, 2006.

[9] Pierre E. Dupont, Jesse Lock, Brandon Itkowitz, and Evan Butler. Design and control of concentric-tube robots. *IEEE Transactions on Robotics*, 26(2):209–225, 2010.

[10] Gary W. Flake. Square unit augmented, radially extended, multilayer perceptrons. In Genevieve B. Orr and Klaus-Robert Müller, editors, *Neural networks: Tricks of the trade*, chapter 7, pages 143–160. Springer, 2012.

[11] Ken-Ichi Funahashi. On the approximate realization of continuous mappings by neural networks. *Neural networks*, 2(3):183–192, 1989.

[12] Ran Gazit and Bernard Widrow. Backpropagation through links: a new approach to kinematic control of serial manipulators. In *IEEE International Symposium on Intelligent Control*, pages 99–104. 1995.

[13] Zheng Geng and Laurie Haynes. Neural network solution for the forward kinematics problem of a stewart platform. In *IEEE International Conference on Robotics and Automation*, pages 2650–2655, 1991.

[14] Reinhard Grassmann, Vincent Modes, and Jessica Burgner-Kahrs. Learning the forward and inverse kinematics of a 6-dof concentric tube continuum robot in SE(3). In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5125–5132, 2018.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *IEEE International Conference on Computer Vision*, pages 1026–1034, 2015.

[16] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[17] Bunpei Irie and Sei Miyake. Capabilities of three-layered perceptrons. In *IEEE International Conference on Neural Networks*, pages 641–648. 1988.

[18] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, pages 1–15, 2015.

[19] Jean-Claude Latombe. *Robot motion planning*, volume 124. Springer Science & Business Media, 1991.

[20] Yann A. LeCun, Léon Bottou, Genevieve B. Orr, and Klaus-Robert Müller. Efficient backprop. In *Neural networks: Tricks of the trade*. Springer, 2012.

[21] Fernando Díaz Ledezma and Sami Haddadin. First-order-principles-based constructive network topologies: An application to robot inverse dynamics. In *IEEE-RAS International Conference on Humanoid Robotics*, pages 438–445, 2017.

[22] L. Nguyen, R. V. Patel, and K. Khorasani. Neural network architectures for the forward kinematics problem in robotics. In *International Joint Conference on Neural Networks*, pages 393–399, 1990.

[23] Richard Paul. Modelling, trajectory calculation and servoing of a computer controlled arm. Technical report, Stanford Artificial Intelligence Lab, Stanford Univ., Stanford, CA, 1972.

[24] Richard P. Paul and Hong Zhang. Computationally efficient kinematics for manipulators with spherical wrists based on the homogeneous transformation representation. *The International Journal of Robotics Research*, 5(2):32–44, 1986.

[25] Anh Son Phung, Jörn Malzahn, Frank Hoffmann, and Torsten Bertram. Data based kinematic model of a multi-flexible-link robot arm for varying payloads. In *IEEE International Conference on Robotics and Biomimetics*, pages 1255–1260, 2011.

[26] Amar Ramdane-Cherif, Vkronique Perdereau, and Michel Drouin. Optimization schemes for learning the forward and inverse kinematic equations with neural network. In *IEEE International Conference on Neural Networks*, pages 2732–2737, 1995.

[27] Amar Ramdane-Cherif, Vkronique Perdereau, and Michel Drouin. Acceleration of back-propagation for learning the forward and inverse kinematic equations. In *International Symposium on Neuroinformatics and Neurocomputers*, pages 261–265. 1995.

[28] R. Felix Reinhart and Jochen J. Steil. Recurrent neural associative learning of forward and inverse kinematics for movement generation of the redundant pa-10 robot. In *ECSIS Symposium on Learning and Adaptive Behaviors for Robotic Systems*, pages 35–40, 2008.

[29] D. Caleb Rucker, Bryan A. Jones, and Robert J. Webster III. A geometrically exact model for externally loaded concentric-tube continuum robots. *IEEE Transactions on Robotics*, 26(5):769–780, 2010.

[30] H. Sadjadian, Hamid D. Taghirad, and Alireza Fatehi. Neural networks approaches for computing the forward

kinematics of a redundant parallel manipulator. *International Journal of Computational Intelligence*, 2(1):40–47, 2005.

[31] John Stuelpnagel. On the parametrization of the three-dimensional rotation group. *Society for Industrial and Applied Mathematics Review*, 6(4):422–430, 1964.