2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Active Learning for Forward/Inverse Kinematics of Redundantly-driven Flexible Tensegrity Manipulator

Yuhei Yoshimitsu[1], Takayuki Osa[2], Heni Ben Amor[3] and Shuhei Ikemoto[1]

***Abstract*— In flexible redundantly-driven multi-DOF systems, like living beings, the representation of redundant kinematics including the diversity of solutions, is crucial for leveraging its distinctive characteristics. This paper proposes an active learning framework for forward and inverse modeling of complex kinematics that improves expressions of control space, task space, and null space. It consists of a Variational Auto Encoder (VAE)-type network that internally holds expressions of control space, task space, and null space, and an algorithm for selecting new data using the cross-entropy method. The validity of the proposed system was verified using a tensegrity manipulator driven by 40 pneumatic cylinders. As a result, it was confirmed that active learning contributed to achieving the entire range of motion covered and a well-organized representation of the null space.**

I. INTRODUCTION

Softness and redundancy are features that have traditionally distinguished biological beings from robots. Modern, biologically-inspired robot systems aim to overcome this limitation and implement these traits in the body of a robot [1]. However, biologically faithful imitation often conflicts with engineering-plausible design.

In our work, we aim to implement softness and redundancy in an engineering-plausible design by leveraging "Tensegrity" structures [2]. Tensegrity refers to a structure in which the arrangement of multiple rigid bodies is stabilized by tension. The methodology has previously been used in mobile robots [3]–[5] and manipulators [6]–[9]. Fig. 1 depicts the developed continuum manipulator driven by 40 pneumatic cylinders [10]. Employing class-1 tensegrity, the simplest constitution of tensegrity, allowed the integration of many actuators without interference while ensuring durability. Due to their flexible nature, tensegrity structures allow for safe data collection over long horizons, thereby enabling forward and inverse kinematics modeling through a data-driven approach. However, despite these advantages, controlling such a deformable system can pose major challenges. More specifically, even with training data from long-term random trials, obtaining a representation of the task space and its orthogonal null space applicable to other tasks remains difficult.

*This work was supported by JSPS KAKENHI Grant Numbers 22H03671 and 22K19815.

[1]Y. Yoshimitsu and S. Ikemoto is with Graduate School of Life Science and Systems Engineering, Kyushu Institute of Technology, 2-4 Hibikino, Wakamatsu, Kitakyushu, Fukuoka, Japan. ikemoto@brain.kyutech.ac.jp

[2]T. Osa is with Graduate School of Information Science and Technology, The University of Tokyo, 7-3-1 Hongo, Bunkyo, Tokyo, Japan.

[3]H. B. Amor is with Ira A. Fulton Schools of Engineering, Arizona State University, 699 S Mill Ave #119, Tempe, AZ 8528, USA

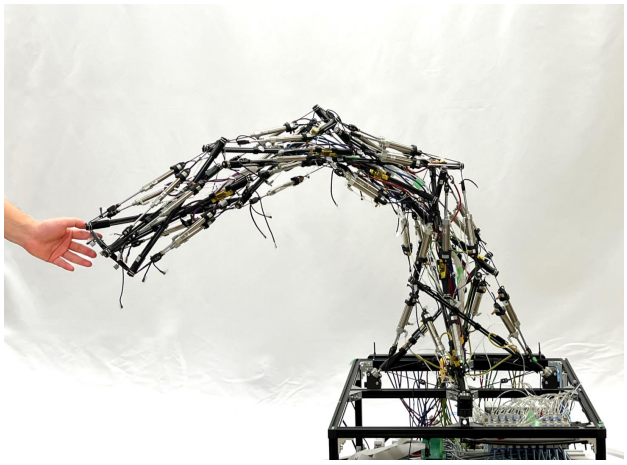

Fig. 1. The soft and redundantly-driven continuum manipulator that incorporates a class-1 tensegrity structure. It is actuated by 40 pneumatic cylinders, regulated by pressure control valves.

For example, even when the input signal is generated uniformly at random, the robot's end position is typically biased by the inherent structural characteristics. At the same time, since the robot is redundant, the shape and stiffness of manipulators that achieve the same end position will have multiple candidates, and the data will also be biased in the null space that has to organize and represent those variations. In [11], we have proposed a method for modeling the forward/inverse kinematics of tensegrity manipulators through a forward/inverse mappings conditioned on the null space expression. Training was performed on data consisting of pairs of motor commands and the corresponding end-effector positions. Although the proposed method was successfully validated, due to the bias of the data, the task space that could be represented was narrower than the original range of motion. In addition, the null space represented stiffness only around some end positions.

A promising approach to overcome this problem is Active learning [12]–[14] wherein new data is sequentially sampled and updated to meet the refined prior models. In its original formulation the learning system asked the annotator (oracle) to annotate unlabeled data thereby obtaining new data points [15]. Although imitation learning fits this description in robotics, the term "active learning" is now used to refer to a broader framework in which the robot acquires new data points through its actions and sequentially updates the dataset [16]. In this sense, the keywords "adaptive sampling" [17] and "Bayesian optimization" [18] are also used to describe

almost similar attempts. This study uses the word active learning as this broader meaning.

So far, active learning has been applied to a variety of problems, including motion planning [19]–[21], navigation [22]–[24], and dynamic modeling [25]–[27]. It has also been applied to manipulator kinematics models, such as human-independent calibration [28] and modeling the complex kinematics of redundant multi-degree-of-freedom robots [29]–[31]. The tensegrity manipulator in this study is one of the most complex manipulators, thus likely prone to biased datasets and data inefficiency. Furthermore, the learning system also seeks to acquire a representation of the null space to obtain a variety of solutions in inverse kinematics, so its impact is likely to be magnified.

This paper proposes an active learning framework for forward and inverse modeling of complex kinematics that maximizes the information gain in control, task, and null spaces. It consists of a Variational Auto Encoder (VAE)-type network that internally holds expressions of control space, task space, and null space along with an algorithm for selecting new data points using the cross-entropy method. The validity of the proposed system was verified using a tensegrity manipulator driven by 40 pneumatic cylinders shown in Fig.1. The proposed active learning strategy increased the covered range of motion of the robot while also generating a well-organized representation of the null space.

II. DEVELOPED TENSEGRITY MANIPULATOR

Fig.2 depicts the composition of the developed tensegrity manipulator. The height and mass of the manipulator are 1300 [mm] height and 3.9 [Kg], respectively. This structure complies with a class-1 tensegrity, comprising vertically stacked five 4-strut tensegrity prisms.

Out of the total 80 tensile members, 40 have been replaced with pneumatic cylinders, enabling active bending. The manipulator utilizes two types of pneumatic cylinders, each with a stroke length of 45 [mm] but varying diameters. More specifically, 24 cylinders in the lower section have a diameter of 16 [mm] (MSPCN16-45, Misumi), while the upper 16 cylinders have a diameter of 10 [mm] (MSPCN10-45, Misumi). The structure's base, comprising four strut ends, is connected to sliders and ball joints to facilitate bending. The base, which contains 40 pressure control valves (VEAB, FEST Inc.), enables independent control of internal pressures for the 40 pneumatic cylinders. The embedded AD/DA system, which is ROS2-compatible, provides target pressure values.

Employing a class-1 tensegrity design enables the incorporation of 40 pneumatic cylinders without encountering mechanical interference. Moreover, the class-1 tensegrity design results in a robot with fewer components due to the repetitive connections of compressive and tensile members. In the developed tensegrity manipulator, two types of struts 300 [mm] and 350 [mm] in length, are employed. Specifically, all struts comprise three CFRP pipes with an outer diameter of 5 [mm]. The lower 4 struts utilize 350[mm] pipes, while the upper 16 struts utilize 300[mm] pipes. Thus, despite the overall complexity of the structure, the developed tensegrity manipulator is composed of relatively few parts: two types of struts, two types of pneumatic cylinders, and the same type of stiff cables.

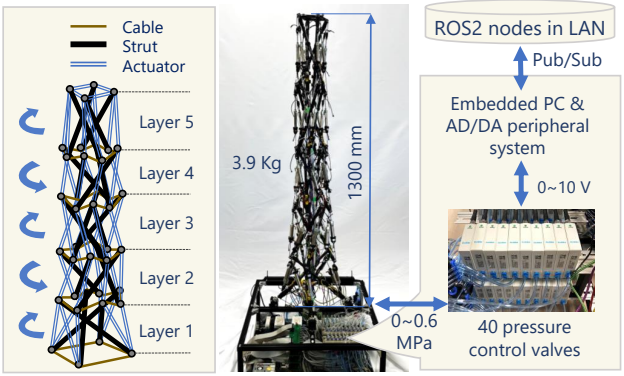

Fig. 2. The developed tensegrity manipulator used in this study. It consists of five 4-strut tensegrity prisms stacked vertically with alternating torsional directions. Out of the total 80 tensile members, 40 have been replaced with pneumatic cylinders. Their internal pressures are independently controlled to perform various bending. The system is built to be ROS2-compatible.

The features of the developed tensegrity manipulator, including the pneumatic flexibility, the abundance of actuators, minimal mechanical interference, and complex structure with limited components, pose challenges for analytical modeling. However, despite the complexity of the hardware, it effectively mitigates hardware failures. As a result, the robot can collect data through random movements without a pre-tuned kinematic model. These attributes suggest that the developed tensegrity manipulator is an ideal platform for investigating the functionalities of flexibility and redundancy from a constructivist perspective.

III. PROPOSED ACTIVE LEARNING FRAMEWORK

A. VAE-based Probabilistic Kinematics Model

Fig.3 shows the network architecture used in the proposed active learning framework. It is based on Variational Auto-Encoder (VAE) [32] and employs probabilistic outputs in both the encoder and decoder. Specifically, input/output vectors represent points in the control space, and a latent space vector means a concatenated vector of two vectors representing a point in the task space and a point in the null space, respectively. The details are described below.

Let $\boldsymbol{u} = (u_1 \cdots u_n)$, $\boldsymbol{y} = (y_1 \cdots y_m)$, and $\boldsymbol{z} = (z_1 \cdots z_d)$ be vectors in control space, task space, the null space, respectively. Here, we assume a robot where a given $\boldsymbol{u}$ uniquely determines $\boldsymbol{y}$, but a given $\boldsymbol{y}$ does not uniquely determine $\boldsymbol{u}$ without conditioning by $\boldsymbol{z}$. The network architecture gives the posterior distribution/encoder $q_\phi(\boldsymbol{y}, \boldsymbol{z}|\boldsymbol{u})$ and likelihood/decoder $p_\theta(\boldsymbol{u}|\boldsymbol{y}, \boldsymbol{z})$ following properties:

$$q_\phi(\boldsymbol{y}, \boldsymbol{z}|\boldsymbol{u}) = q_\phi(\boldsymbol{y}|\boldsymbol{u}) q_\phi(\boldsymbol{z}|\boldsymbol{u}) \quad (1)$$

$$q_\phi(\boldsymbol{y}|\boldsymbol{u}) \sim N(\boldsymbol{\mu}_y, \mathrm{diag}(\boldsymbol{\sigma}_y)) \quad (2)$$

$$q_\phi(\boldsymbol{z}|\boldsymbol{u}) \sim N(\boldsymbol{\mu}_z, \mathrm{diag}(\boldsymbol{\sigma}_z)) \quad (3)$$

$$p_\theta(\boldsymbol{u}|\boldsymbol{y}, \boldsymbol{z}) \sim N(\boldsymbol{\mu}_u, \mathrm{diag}(\boldsymbol{\sigma}_u)) \quad (4)$$

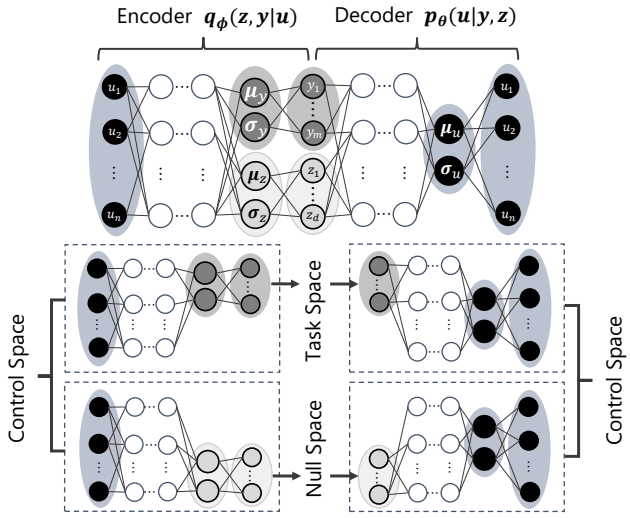

Fig. 3. The proposed VAE-based stochastic kinematics model. Not only the middle units but also the output units behave stochastically by the reparameterization trick. Each of those stochastic units is assigned to express control space $\boldsymbol{u}$, task space $\boldsymbol{y}$, and null space $\boldsymbol{z}$. To train the forward/inverse kinematics model, supervised data is given at $\boldsymbol{u}$ and $\boldsymbol{y}$, not at $\boldsymbol{z}$.

where $\{\boldsymbol{\mu}_u, \boldsymbol{\sigma}_u\}$, $\{\boldsymbol{\mu}_y, \boldsymbol{\sigma}_y\}$, and $\{\boldsymbol{\mu}_z, \boldsymbol{\sigma}_z\}$ are element-wise averages and variances of $\boldsymbol{u}$, $\boldsymbol{y}$, and $\boldsymbol{z}$, respectively. Note that ϕ and θ are network parameters of the encoder and decoder, respectively. These prior distributions are given by the reparameterization trick [32].

Let $\mathcal{D}_{\text{train}} = \{\boldsymbol{u}_i^*, \boldsymbol{y}_i^*\}_{i=1}^N$ be a dataset obtained from N trials in which a generated control input is given to a robot and the corresponding result in the task space is observed. The optimization problem for ϕ and θ can be posed as follows:

$$\phi^*, \theta^* = \underset{\phi,\theta}{\text{argmax}} \ \mathcal{L}(\phi, \theta) \tag{5}$$

$$\mathcal{L}(\phi, \theta) = \mathcal{L}_{\text{vae}}(\phi, \theta) + \mathcal{L}_{\text{task}}(\phi) \tag{6}$$

$$\mathcal{L}_{\text{vae}}(\phi, \theta) = \mathbb{E}_{q_\phi(\boldsymbol{y},\boldsymbol{z}|\boldsymbol{u}^*)}\left[\log p_\theta(\boldsymbol{u}^*|\boldsymbol{y}, \boldsymbol{z})\right] - \gamma |D_{\text{KL}}(q_\phi(\boldsymbol{z}|\boldsymbol{u}^*)||p(\boldsymbol{z}))| \tag{7}$$

$$\mathcal{L}_{\text{task}}(\phi) = \mathbb{E}_{\boldsymbol{u}^*,\boldsymbol{y}^* \sim \mathcal{D}}\left[\log q_\phi(\boldsymbol{y}^*|\boldsymbol{u}^*)\right] \tag{8}$$

where $p(\boldsymbol{z}) = N(0, 1)$ is the prior distribution of the unsupervised latent space corresponding to the null space, and γ is a constant for the regulation term.

In this training phase, $\boldsymbol{u}$ and $\boldsymbol{y}$ are supervised by $\boldsymbol{u}^*$ and $\boldsymbol{y}^*$, respectively, while $\boldsymbol{z}$ is not. Thus, in the optimization problem, $\boldsymbol{z}$ must represent information that is additionally needed to uniquely map from $\boldsymbol{y}$ to $\boldsymbol{u}$, corresponding to inverse kinematics, and that is lost in the mapping from $\boldsymbol{u}$ to $\boldsymbol{y}$, corresponding to forward kinematics. This constraint, introduced by the network architecture, results in a representation of $\boldsymbol{z}$ that corresponds to the null space.

B. Adaptive Sampling Algorithm

Assume that the initial training dataset $\mathcal{D}_{\text{train}} = \{\boldsymbol{u}_i^*, \boldsymbol{y}_i^*\}_{i=1}^N$ was obtained by randomly generating control inputs and feeding them to the robot and observing the corresponding results. Even if N is sufficiently large, training will not adequately represent the control, task, and null spaces if it is inappropriately biased. To deal with this issue, we propose an adaptive sampling method that finds locations in control space, task space, and null space that are not sufficiently represented and updates the dataset sequentially by making additional trials to obtain new data points there.

Algorithm 1 Proposed Adaptive Sampling Method

Input: ϕ^* and θ^* obtained in training based on $\mathcal{D}_{\text{train}}$, $\mathcal{C}$ as a continuation condition
Output: $\mathcal{D}'_{\text{train}}$

```
D_new ← ∅
while C holds do
    Obtain {ũ_i}_{i=1}^L that have the L largest F values by the Cross Entropy Method
    i ← 1
    if ∀u* ∈ D_new, ||ũ_i − u*|| > a then
        D_new ← D_new ∪ {ũ_i}
    else
        i ← i + 1 and go to 5
    end if
end while
S ← |D_new|
Compute F(u*), ∀u* ∈ D_train
Remove data points from D_train that have the S smallest F(u*)
D'_train ← D_train ∪ D_new
```

Algorithm 1 summarizes the proposed method. In this algorithm, we employ the Cross Entropy Method (CEM) [33] to identify new data points that will contribute to improving the model. The acquisition function $\mathcal{F}(\boldsymbol{u})$ used in CEM is defined as follows:

$$\mathcal{F}(\boldsymbol{u}) = K_y \boldsymbol{\sigma}_y + K_z \boldsymbol{\sigma}_z + K_u \boldsymbol{\sigma}_u \tag{9}$$

where K_y, K_z, and K_u are weighting constant coefficients. The $\boldsymbol{\sigma}_y$, $\boldsymbol{\sigma}_z$, and $\boldsymbol{\sigma}_u$ are obtained from the latest trained model (ϕ^*, θ^*) at the beginning of adaptive sampling phase. Since the network in Fig. 3 employs the reparameterization trick, $\boldsymbol{\sigma}_y$ and $\boldsymbol{\sigma}_z$ are obtained deterministically for a given $\boldsymbol{y}$. The $\boldsymbol{\sigma}_u$, on the other hand, is inherently stochastic since it is computed from resampled $\boldsymbol{y}$ and $\boldsymbol{z}$. To avoid computational overhead here, the $\boldsymbol{\sigma}_u$ is obtained deterministically by using $\boldsymbol{\mu}_y$ and $\boldsymbol{\mu}_y$ as $\boldsymbol{y}$ and $\boldsymbol{z}$, respectively. In addition, the constant a in Algorithm 1 is used as the threshold to accept the distanced new data point from the existing data points in $\mathcal{D}_{\text{train}}$. The continuation condition $\mathcal{C}$ is determined by $|\mathcal{D}_{\text{new}}|$ and/or the elapsed time.

C. Parallelized Active Learning Procedure

In robotic active learning, acquiring new data points by running real robot experiments is time-consuming, as is the model training process. Therefore, parallelizing these processes can improve efficiency. From another aspect, it also contributes to creating incremental behavior in which

Algorithm 2 Parallelized Active Learning Procedure

Input: ϕ^*_{t-1} and θ^*_{t-1} on $\mathcal{D}^{t-1}_{\text{train}}$, and $\mathcal{D}^{t}_{\text{train}}$ where $t > 0$

while $\mathcal{C}$: Updating ϕ^*_t and θ^*_t on $\mathcal{D}^{t}_{\text{train}}$ **do**
  $\mathcal{D}^{t+1}_{\text{train}} \leftarrow \text{AdaptiveSampling}(\phi^*_{t-1}, \theta^*_{t-1} | \mathcal{C})$
end while
$t \leftarrow t + 1$
Go to 1

the model is improved, and the sampling behavior changes without the robot stopping for a long time.

Algorithm 2 shows the parallelized procedure of the proposed active learning framework. The subscripts $t-1$, t, and $t+1$ denote the number of times adaptive sampling has been performed and indicate at which point the dataset and optimization results were obtained. The continuation condition corresponds to the termination condition of the training. In this algorithm, during training on the latest training data obtained at time t, adaptive sampling based on the evaluations of $\boldsymbol{\sigma}_y$, $\boldsymbol{\sigma}_z$, and $\boldsymbol{\sigma}_u$ is performed using the optimal parameters at time $t-1$ to build the training data at time $t+1$. This ensures that the robot does not stop during model training after completing training on the initial dataset and obtaining the next dataset.

IV. EXPERIMENTAL SETUP

We conducted experiments using the tensegrity manipulator shown in Fig.1 and 2 and their end positions as the task space. This section presents specific information on the experimental setup, including the network structure, training method, and parameters for data sampling.

A. Network Structure and Initial Dataset

Since the tensegrity manipulator is driven by 40 pressure-controlled pneumatic cylinders, the control space is 40 dimensions, and the task space is 3 dimensions to match the experimental setup. In this case, the null space is at most 37 dimensions. Accordingly, n, m, and d of the network in Fig.3 are set to 40, 3, and 37, respectively. Between the $\boldsymbol{u}$ and $(\boldsymbol{y}, \boldsymbol{z})$ layers, there are three fully-connected layers. The network was implemented using Pytorch [34] and ran on a computer that has Intel Xeon 6326 2.9 GHz 2 CPU configuration with NVIDIA RTX A5000. For training the network, the ADAM optimizer [35] was used for the optimization problem written in Eq.5. The initial dataset $\mathcal{D}^0_{\text{train}}$ consists of 40,000 data points. The control inputs, namely desired pressures [MPa], were sampled from uniform distributions of $[0.3, 0.6]$, $[0.2, 0.6]$, $[0.1, 0.6]$, $[0.1, 0.6]$, and $[0.1, 0.6]$ for pneumatic cylinders in each layer from the bottom to prevent excessive manipulator's relaxation. The end positions in the equilibrium postures corresponding to the control inputs were measured with a motion capture system (8 Optitrack Prime X13 cameras with Motive software. NatNet SDK with an add-on for ROS2-compatible data provision). Since each point took 2.5 seconds to acquire, the initial sampling process took approximately 28 hours. The number of epochs and batch size were set to 500 and 100, respectively. In these settings, the training took approximately 5 minutes.

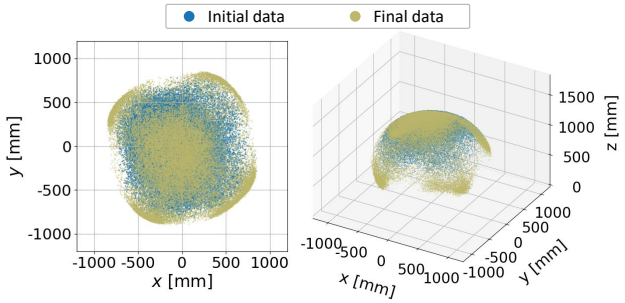

Fig. 4. The distributions of the initial dataset $\mathcal{D}^0_{\text{train}}$ and the final acquired dataset $\mathcal{D}^T_{\text{train}}$. Each dataset consists of 40,000 data points.

B. Active Learning Process

As with the initial dataset $\mathcal{D}^0_{\text{train}}$, the search by CEM on the acquisition function $\mathcal{F}$ was limited in scope to avoid excessive manipulator's relaxation. For approximately 5 minutes during the learning process on the current dataset $\mathcal{D}^t_{\text{train}}$, CEM finds control input candidates, and a data point that has better $\mathcal{F}$ value and more than the distance a form all of data points in $\mathcal{D}^t_{\text{train}}$ is fed to the manipulator to measure the corresponding end position. Approximately 170 new data points are adopted every 5 minutes. As soon as the learning process is completed, the 170 adopted data points are replaced by the data points with the lowest $\mathcal{F}$ value in $\mathcal{D}^t_{\text{train}}$ to prepare the next dataset $\mathcal{D}^{t+1}_{\text{train}}$. This process was repeated until 30,000 data points had been adopted. The entire active learning process took about 15 hours.

V. RESULTS

To validate the proposed active learning framework, we compared kinematics models trained on the initial dataset $\mathcal{D}^0_{\text{train}}$ and the sequentially updated dataset $\mathcal{D}^T_{\text{train}}$. In this section, we will explain in the following order: A) analysis of dataset updates, B) evaluation of the feedforward end position control using the obtained inverse kinematics model, and C) evaluation of the null space effects on the posture variations.

A. Analysis of the Acquired Dataset

Fig.4 shows scatter plots of the initial dataset $\mathcal{D}^0_{\text{train}}$ (blue points) and the final acquired dataset $\mathcal{D}^T_{\text{train}}$ (orange points). In the graph, the data collected are hemispherically distributed. Although the initial data points are concentrated at the center, the final acquired data points are distributed widely near the periphery. This result indicates that the proposed active learning framework can preferentially collect postures with large bending, which is difficult to obtain with the initial random sampling.

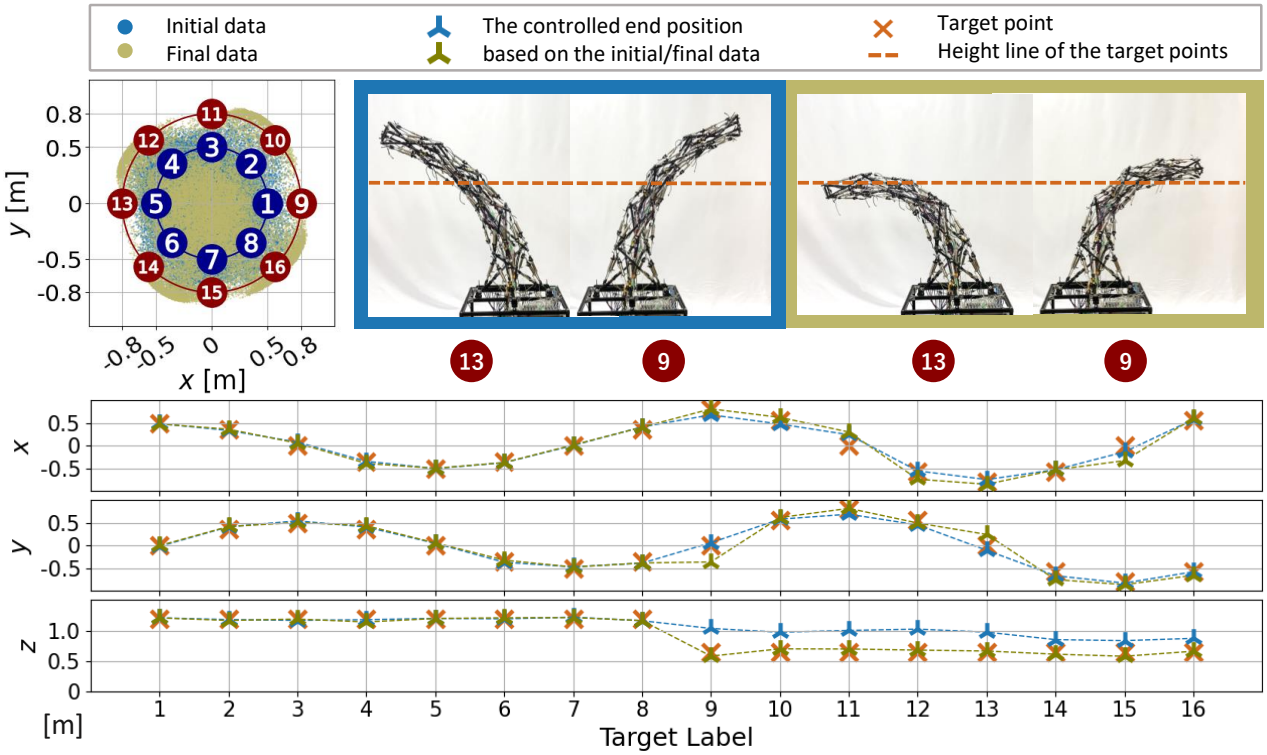

Fig. 5. The result of feedforward end position control based on the initial and final models. Decoder subnetworks of the two models, which receive the desired end position $\boldsymbol{y}^*$ with a designated null space vector $\boldsymbol{z}^*$ and output the corresponding target pressure $\boldsymbol{u}$, are used to generate control input vectors fed to the manipulator.

B. Evaluation of the Feedforward End-Position Control

Distributions of the initial and final acquisition datasets are distinctly different. We then investigated the differences between the two kinematic models trained on these two datasets. Fig.5 shows the results of using the decoder part of the model, an inverse kinematics model that receives the desired end position $\boldsymbol{y}^*$ with a designated null space vector $\boldsymbol{z}^*$ and outputs the corresponding target pressure $\boldsymbol{u}$. The actual end position $\boldsymbol{y}$ is measured by feeding it to the tensegrity manipulator. The null space vector $\boldsymbol{z}^*$ was sampled from a multivariate normal distribution with a mean and a variance calculated by the encoder part and the datasets. The mean end position is calculated based on 100 results recorded by feeding 100 $\boldsymbol{z}^*$ samples. The top left of Fig.5 represents the 16 desired end positions to be evaluated. The initial dataset $\mathcal{D}^0_{\text{train}}$ has fewer data at the periphery than the final acquisition dataset $\mathcal{D}^T_{\text{train}}$, suggesting that the initial model is less accurate at the periphery than the final model. The top right of Fig.5 shows the postures obtained by the initial and final models when reaching the two desired end positions around the periphery as targets. They show that the initial model could not perform deep bending while the final model could and achieved better accuracy. The bottom of Fig.5 shows the average end position at all 16 points. As initially expected, there is no significant difference in the positional accuracy in the center region in both the initial and final models. However, around the periphery of the target No.9-16 points , especially in the z direction, the positional accuracy of the initial model got worse, whereas the final model showed good positional accuracy. Note that the average errors around the periphery in the z direction based on the initial/final model were 295 and 41 [mm], respectively. This shows that the proposed active learning framework can update the data points appropriately to collect the necessary information for modeling the kinematics.

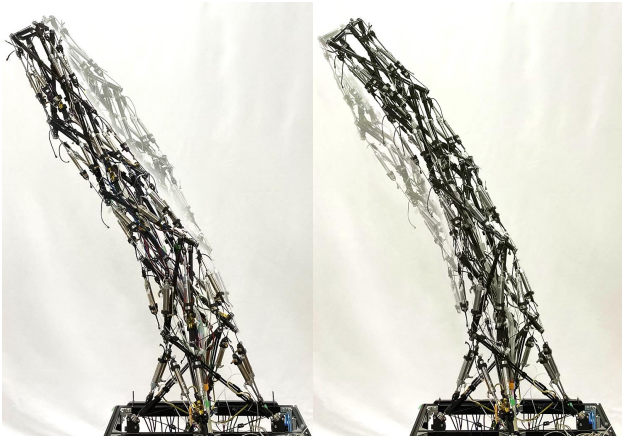

Fig. 6. Variation of postures obtained by changing z_{13} from -3.0 to 3.0. The posture is changed in response to changes in z_{13} while the end position is maintained.

C. Evaluation of Null Space Effects on Postures

In the [11], we evaluated a model corresponding to the initial model and reported that a change in the null space

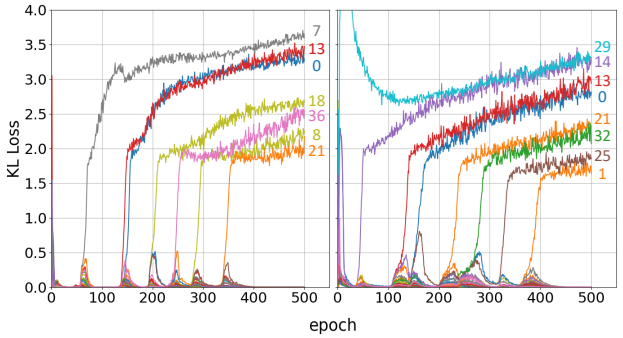

Fig. 7. Kullback-Leibler (KL) divergence of latent variable z of conventional random and new adaptive sampled data

vector can generate control space vectors that result in different stiffness with the same end position. However, control space vectors resulting in different postures with the same end position could not be generated. Therefore, we confirm whether the proposed active learning framework can exhibit redundant postures.

Fig.6 shows the postures produced by the final model with different null space vectors $\boldsymbol{z}$. The elements in the null space vector have to have a larger Kullback-Leibler divergence with $N(0, 1)$, the second term of Eq.7, to contribute to the mapping. This means that the element that will produce the significant change in the control space vector can be selected in advance. In particular, we chose z_{13} and varied it in [-3.0, 3.0] continuously while the other elements were fixed at 0. As shown in Fig.6, various bending postures were realized to achieve the same end position. This confirms that the proposed active learning framework can collect data points for diverse solutions in inverse kinematics and that the proposed stochastic forward/inverse kinematics model can obtain a better representation of control space, task space, and null space.

VI. DISCUSSION

Since the null space has at most 37 dimensions, 37 units were prepared without dimensionality reduction. As mentioned above, it can be determined which of these units has a useful representation by the amount of the second term in the Eq.7: $|D_{\mathrm{KL}}(q_\phi(\boldsymbol{z}|\boldsymbol{u}^*)||p(\boldsymbol{z}))|$. Fig.7 shows the change in the $|D_{\mathrm{KL}}(q_\phi(\boldsymbol{z}|\boldsymbol{u}^*)||p(\boldsymbol{z}))|$ when our network is trained from the beginning using the initial dataset $\mathcal{D}^0_{\mathrm{train}}$ and final dataset $\mathcal{D}^T_{\mathrm{train}}$. From this figure, it can be seen that as learning progresses, as one unit of $\boldsymbol{z}$ begins to have a larger value of $|D_{\mathrm{KL}}(q_\phi(\boldsymbol{z}|\boldsymbol{u}^*)||p(\boldsymbol{z}))|$ than the other, the difference increases and becomes fixed. Increasing the $|D_{\mathrm{KL}}(q_\phi(\boldsymbol{z}|\boldsymbol{u}^*)||p(\boldsymbol{z}))|$ is inherently a burden in Eq.5 because it decreases Eq.6. Therefore, this change is unacceptable without an increase in the first term of Eq.7 or an increase in Eq.8. In Fig.7, seven units in the initial model and eight units in the final acquired model showed such a change in the $|D_{\mathrm{KL}}(q_\phi(\boldsymbol{z}|\boldsymbol{u}^*)||p(\boldsymbol{z}))|$. This means that the final acquired model could find a higher dimensional null space vector to model the kinematics accurately. In the future, we aim to actively acquire a null space representation that can easily generate posture and stiffness changes by adjusting the balance of $|D_{\mathrm{KL}}(q_\phi(\boldsymbol{z}|\boldsymbol{u}^*)||p(\boldsymbol{z}))|$ carried by each unit in $\boldsymbol{z}$.

VII. CONCLUSION

This paper proposed an active learning framework for forward and inverse modeling of complex kinematics that maximizes the information gain in control, task, and null spaces. It consists of two key components: 1) a VAE-type network that internally holds expressions of control space, task space, and null space, and 2) an algorithm for selecting new data using the cross-entropy method. The validity of the proposed system was verified using the tensegrity manipulator driven by 40 pneumatic cylinders. As a result, it was confirmed that active learning contributed to achieving the covered range of motion and a well-organized representation of the null space.

ACKNOWLEDGMENT

This work was supported by JSPS KAKENHI Grant Numbers 22H03671 and 22K19815.

REFERENCES

[1] R. Pfeifer, M. Lungarella, and F. Iida, "Self-organization, embodiment, and biologically inspired robotics," *Science*, vol. 318, no. 5853, p. 1088, 2007. [Online]. Available: http://science.sciencemag.org/content/318/5853/1088.abstract

[2] R. E. Skelton and M. C. Oliveira, *Tensegrity Systems*. Springer Nature, 2009.

[3] K. Kim, A. K. Agogino, and A. M. Agogino, "Rolling locomotion of cable-driven soft spherical tensegrity robots," *Soft Robotics*, vol. 7, no. 3, pp. 346–361, 2020. [Online]. Available: https://doi.org/10.1089/soro.2019.0056

[4] Y. Zheng, Y. Li, Y. Lu, M. Wang, X. Xu, C. Zhou, and Y. Luo, "Robustness evaluation for rolling gaits of a six-strut tensegrity robot," *International Journal of Advanced Robotic Systems*, vol. 18, no. 1, p. 1729881421993638, 2021. [Online]. Available: https://doi.org/10.1177/1729881421993638

[5] R. Kobayashi, H. Nabae, G. Endo, and K. Suzumori, "Soft tensegrity robot driven by thin artificial muscles for the exploration of unknown spatial configurations," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5349–5356, 2022.

[6] S. Lessard, D. Castro, W. Asper, S. D. Chopra, L. B. Baltaxe-Admony, M. Teodorescu, V. SunSpiral, and A. Agogino, "A bio-inspired tensegrity manipulator with multi-dof, structurally compliant joints," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Conference Proceedings, pp. 5515–5520.

[7] E. Jung, V. Ly, N. Cessna, M. L. Ngo, D. Castro, V. SunSpiral, and M. Teodorescu, "Bio-inspired tensegrity flexural joints," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, Conference Proceedings, pp. 5561–5566.

[8] S. Ikemoto, K. Tsukamoto, and Y. Yoshimitsu, "Development of a modular tensegrity robot arm capable of continuous bending," *Frontiers in Robotics and AI*, vol. 8, p. 347, 2021. [Online]. Available: https://www.frontiersin.org/article/10.3389/frobt.2021.774253

[9] C. Woods and V. Vikas, "Design and modeling framework for dexter: Dexterous continuum tensegrity manipulator," *Journal of Mechanisms and Robotics*, vol. 15, no. 3, 2023. [Online]. Available: https://doi.org/10.1115/1.4056959

[10] Y. Yoshimitsu, K. Tsukamoto, and S. Ikemoto, "Development of pneumatically driven tensegrity manipulator without mechanical springs," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Conference Proceedings, pp. 3145–3150.

[11] Y. Yoshimitsu, T. Osa, and S. Ikemoto, "Forward/inverse kinematics modeling for tensegrity manipulator based on goal-conditioned variational autoencoder," 10 2023, pp. 6668–6673.

[12] B. Settles, *Active Learning Literature Survey*. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2010.

[13] P. Ren, Y. Xiao, X. Chang, P.-Y. Huang, Z. Li, B. B. Gupta, X. Chen, and X. Wang, "A survey of deep active learning," *ACM Comput. Surv.*, vol. 54, no. 9, p. Article 180, 2021. [Online]. Available: https://doi.org/10.1145/3472291

[14] X. Zhan, Q. Wang, K.-H. Huang, H. Xiong, D. Dou, and A. B. Chan, "A comparative survey of deep active learning," *ArXiv*, vol. abs/2203.13450, 2022.

[15] D. Cohn, L. Atlas, and R. Ladner, "Improving generalization with active learning," *Mach. Learn.*, vol. 15, no. 2, p. 201–221, 1994. [Online]. Available: https://doi.org/10.1023/A:1022673506211

[16] A. T. Taylor, T. A. Berrueta, and T. D. Murphey, "Active learning in robotics: A review of control principles," *Mechatronics*, vol. 77, p. 102576, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957415821000659

[17] H. Liu, Y.-S. Ong, and J. Cai, "A survey of adaptive sampling for global metamodeling in support of simulation-based complex engineering design," *Structural and Multidisciplinary Optimization*, vol. 57, no. 1, pp. 393–416, 2018. [Online]. Available: https://doi.org/10.1007/s00158-017-1739-8

[18] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, "Taking the human out of the loop: A review of bayesian optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2016.

[19] J. C. Latombe, "Probabilistic roadmaps: A motion planning approach based on active learning," in *2006 5th IEEE International Conference on Cognitive Informatics*, vol. 1, Conference Proceedings, pp. 1–2.

[20] D. Coleman, I. A. Şucan, M. Moll, K. Okada, and N. Correll, "Experience-based planning with sparse roadmap spanners," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, Conference Proceedings, pp. 900–905.

[21] B. Ichter, J. Harrison, and M. Pavone, "Learning sampling distributions for robot motion planning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, Conference Proceedings, pp. 7087–7094.

[22] S. Chernova and M. Veloso, "Interactive policy learning through confidence-based autonomy," *J. Artif. Int. Res.*, vol. 34, no. 1, p. 1–25, 2009.

[23] D. Silver, J. A. Bagnell, and A. Stentz, "Active learning from demonstration for robust autonomous navigation," in *2012 IEEE International Conference on Robotics and Automation*, Conference Proceedings, pp. 200–207.

[24] M. Y. Ju and J. R. Lee, "Vision-based mobile robot navigation using active learning concept," in *2013 International Conference on Advanced Robotics and Intelligent Systems*, Conference Proceedings, pp. 122–129.

[25] G. Maeda, M. Ewerton, T. Osa, B. Busch, and J. Peters, "Active incremental learning of robot movement primitives," in *Proceedings of the 1st Annual Conference on Robot Learning (CoRL)*, ser. Proceedings of Machine Learning Research, vol. 78. PMLR, Nov. 2017, pp. 37–46. [Online]. Available: http://proceedings.mlr.press/v78/maeda17a.html

[26] J. Li, A. Wahrburg, and N. Enayati, "Active learning of time-optimal motion for an industrial manipulator with variable payload," *IFAC-PapersOnLine*, vol. 56, no. 2, pp. 306–313, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896323019936

[27] A. Saviolo, J. Frey, A. Rathod, M. Diehl, and G. Loianno, "Active learning of discrete-time dynamics for uncertainty-aware model predictive control," *IEEE Transactions on Robotics*, vol. 40, pp. 1273–1291, 2024.

[28] E. Daş and J. W. Burdick, "An active learning based robot kinematic calibration framework using gaussian processes," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, Conference Proceedings, pp. 11 495–11 501.

[29] M. Rolf, J. J. Steil, and M. Gienger, "Online goal babbling for rapid bootstrapping of inverse models in high dimensions," in *2011 IEEE International Conference on Development and Learning (ICDL)*, vol. 2, Conference Proceedings, pp. 1–8.

[30] A. Baranes and P.-Y. Oudeyer, "Active learning of inverse models with intrinsically motivated goal exploration in robots," *Robotics and Autonomous Systems*, vol. 61, no. 1, pp. 49–73, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0921889012000644

[31] Z. Liu, A. Hitzmann, S. Ikemoto, S. Stark, J. Peters, and K. Hosoda, "Local online motor babbling: Learning motor abundance of a musculoskeletal robot arm," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Conference Proceedings, pp. 6594–6601.

[32] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[33] R. Y. Rubinstein and D. P. Kroese, *The Cross Entropy Method: A Unified Approach To Combinatorial Optimization, Monte-carlo Simulation (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2004.

[34] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[35] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1412.6980