Common Dimensional Autoencoder for Learning Redundant Muscle-Posture Mappings of Complex Musculoskeletal Robots

doi: https://doi.org/10.1109/IROS40897.2019.8968605

Common Dimensional Autoencoder for Learning Redundant Muscle-Posture Mappings of Complex Musculoskeletal Robots

Hiroaki Masuda[1], Arne Hitzmann[1], Koh Hosoda[1] and Shuhei Ikemoto[1]

***Abstract*— It has been widely considered that one of the distinctive features of musculoskeletal structures is to change both a joint angle and stiffness by exploiting the agonist-antagonist driving of the joint. In animals and humans, however, musculoskeletal systems are typically very complex, and the simple agonist-antagonist driving is hardly found. Therefore, in accordance with the increasing complexity of musculoskeletal robots, the feature that makes the robot take a posture with different stiffness values becomes difficult to achieve due to the difficulty of modeling the kinematics. Although data-driven approach such as Neural Network is thought of as suitable for modeling complex relationships, the training data is difficult to obtain because measuring a joint stiffness usually is very difficult in contrast to measuring an actuator's state and a posture. To this problem, in this paper, we propose the common dimensional autoencoder where the encoded feature has identical dimensions to the original input vector. In the proposed network, in parallel with the original unsupervised training using the data of actuators' states, supervised training at a part of the encoded features is carried out by using the data of postures. As a result, features expressing redundancy of inverse kinematics appear at the rest part of encoded features without using the data such as stiffness of joints. The validity of the proposed method was successfully confirmed through an experiment using a 10 DOF complex musculoskeletal robot arm driven by pneumatic artificial muscles.**

I. INTRODUCTION

Living organisms can adaptively behave in complex environments by manipulating their complex body skillfully. It has been thought of as that their distinctive body structures, namely musculoskeletal structures, greatly contributes to achieve adaptive behaviors. Therefore, many musculoskeletal robots have been developed in the field of bioinspired robotics to clarify and exploit functionalities provided by musculoskeletal systems through experiments[1], [2], [3], [4], [5]

One of the well-known functionalities of musculoskeletal systems is to control both an angle and stiffness of a joint. In contrast with that a joint is typically driven by a single rotatory actuator in conventional robots, if a joint is driven via tendons by two agonist-antagonist pair of springy actuators, the redundancy in the drive system can be used to change the stiffness[6], [7]. So far, this property has been intensively studied in the field of tendon-driven systems. It is well-known that musculoskeletal systems having actuators not only driving single joints but also multiple joints (i.e., actuators corresponding to multiarticular muscles) are still sufficiently modeled and controlled if agonist-antagonist pairs can be analytically identified[8], [9], [10]. However, because animals and humans have too complex musculoskeletal systems to identify agonist-antagonist pairs of muscles in advance, musculoskeletal robots have been increasing their structural complexity, and there are already several robots that have complex musculoskeletal robot enough to give up the conventional analytic approach[11], [12], [13].

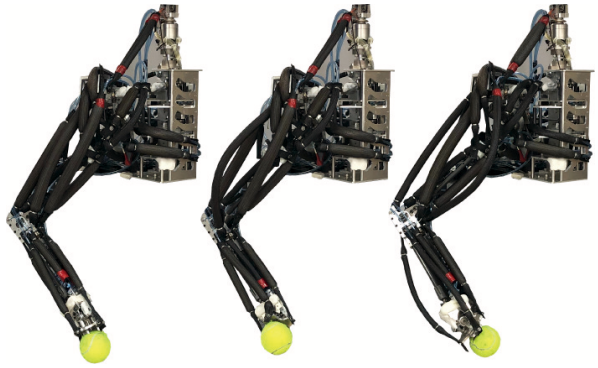

Fig. 1. Example of redundancy in a complex musculoskeletal robot. The same hand position can be achieved in different postures with different actuators' states. This robot has a three dimensional complex musculoskeletal structure driven by pneumatic artificial muscles. Different sets of internal pressures of pneumatic artificial muscles exist while the same hand position maintains.

Data-driven methods such as frameworks using Neural Network (NN) will be a promising candidate to overcome the difficulty in modeling a complex musculoskeletal structure. For instance, Jantsch et al. approximated a muscle Jacobian of a spherical joint by using a multilayered perceptron network[14]. Also, Jamone et al. modeled the muscle Jacobian by adopting the receptive field neural network for their tendon-driven neck structure inspired by the structure of humans[15]. However, in these studies, the feature of the musculoskeletal system that varies stiffness has been neither approximated nor exploited. Because a NN typically provides a one-by-one mapping, redundancy that appears in mapping from a posture to an actuators' state has to be explicitly handled by preparing training data that fully explain the difference in the same posture realized by different actuators' states appears. It is evident that preparing this type of data is much harder than preparing data of actuators' states and postures. To overcome this problem, Okubo et al. adopted a functional approximation of forwarding kinematics and solved the redundant inverse kinematics problem numerically based on the differentiation of the forward mapping[16].

*This work was supported by JSPS KAKENHI Grant Number 18H01410.

[1]The authors are with Department of System Innovation, Graduate School of Engineering Science, Osaka University, 1-3, Machikaneyama, Toyonaka, Osaka, Japan ikemoto@sys.es.osaka-u.ac.jp

Although they proposed a method of a bidirectional mapping for redundant kinematics problems, exploiting the redundancy (e.g., varying actuators' state while keeping the same posture) is fully responsible for the numerical method and can be time-consuming comparing with approaches using NN. If training a NN to express both forward and inverse mappings between actuators' states and postures becomes available only based on easily obtainable data while redundancy provided by not only different stiffness but also redundant joints for the task accomplishment is stored at the same time, it will be the viable alternative.

In this study, we propose a new NN-based method for highly redundant kinematics problems found in complex musculoskeletal robots. Because measuring the stiffness of all joints is typically difficult, the proposed method is required to train the NN without relying on this type of data. Also, since even measuring angles of all joint in musculoskeletal robots is often difficult (e.g., measuring three angles of a spherical joint), it is also required to deem it as the redundancy as the same as the above, if it is not explicitly considered in the task accomplishment. Fig.1 shows the example of this problem that appears practically in our musculoskeletal robot arm. To this end, in the proposed method, both the unsupervised learning and the supervised learning are carried out simultaneously for creating both forward and inverse mappings of kinematics. Briefly speaking, it directly trains a part of feature units in a general autoencoder by supervised learning manner during the original unsupervised learning, so that the rest of feature units finds the information, which is not contained in the supervised data but required to accomplish the unsupervised learning, i.e., the redundant information such as joint stiffness. Thus far, we verified the core idea through a simulation of a 2DOF planer manipulator driven by the typical 3-pairs-6-muscles configuration[17]. However, it has not been evaluated that the proposed method can be applied to complex musculoskeletal robots that actually require data-driven approaches. Therefore, in this paper, we conducted an experiment using a 10 DOF complex musculoskeletal robot arm driven by pneumatic artificial muscles[]. The experiment resulted in a successful confirmation of the method's validity.

II. PROPOSED METHOD

In this section, we explain the proposed method in a general setup where a mapping from high dimensional data D_H to low dimensional data D_L is a many-to-one mapping. For instance, $u \in D_H \in R^N$ and $q \in D_L \in R^M$, where $M < N$ holds, can be assumed to be corresponding to actuators' states and postures, respectively. Fig.2 shows the network used in the proposed method. The proposed method consists of unsupervised learning and supervised learning of NN. The unsupervised learning aims to transform high dimensional data to feature vectors and transforming back. Therefore, this is precisely the same as the general autoencoder. On the other hand, the supervised learning aims to force a part of the feature vector (i.e., M-dimensional subvector out of N-dimensional vector) to mean corresponding low dimensional

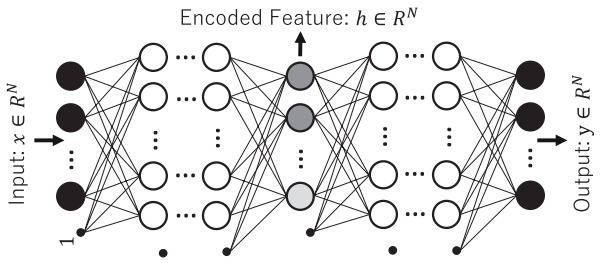

Fig. 2. The network used in the proposed method is a simple autoencoder trained by feeding high dimensional data D_H to both the input and the output units depicted as black circles. Because the bottleneck hidden layer, where low dimensional feature vectors are typically expressed, has the identical dimensionality with the input and the output layers, we name this network - the common dimensional autoencoder. At the same time, supervised learning is carried out by feeding low dimensional data D_L to a part of the feature units on the bottleneck layer (depicted as dark gray circles).

data. It indicates that the forward mapping from the high dimensional data to the low dimensional data is explicitly trained in the subnetwork of the autoencoder. Note that the unsupervised training and supervised training are both carried out simultaneously. In this case, the rest of the feature vector has to retain the information which is not contained in the low dimensional data but required to reproduce the high dimensional data. In other words, it has to show vectors that distinguish many input vectors of the many-to-one forward mapping. The important feature of the proposed method is that the remaining part is not trained explicitly but implicitly guided to express such information meaning the redundancy in the inverse mapping. Once the training was sufficiently accomplished, a variety of high dimensional vectors that realizes the same low dimensional vector can be obtained by fixing the trained part of the feature units (i.e., dark gray circles in Fig.2) to the low dimensional vector and by varying the rest (i.e., light gray circles in Fig.2). See Fig.3 for more specific comprehension.

As shown in Fig.2, the bottleneck layer of the autoencoder has the identical dimensionality with the input and the output and is named the common dimensional autoencoder. The reason is that because the dimensionality required for expressing the redundancy does not exceed $N-M$, N units common to the input and the output of the autoencoder are required for the feature vector at a maximum. Note that other hidden layers have more than N units.

The unsupervised learning is performed by referring the following cost function:

$$L_{\mathrm{AE}} = \frac{1}{\|D_H\|} \sum_{u_i \in D_N} (y(u_i) - u_i)^T Q (y(u_i) - u_i) \qquad (1)$$

where $y(u_i)$ indicates the output of the common dimensional autoencoder obtained by feeding u_i to the input layer and by transforming back via the bottleneck hidden layer. In addition, Q indicates a weighting matrix. This is the general form of the cost function of the autoencoder.

The supervised learning is performed only in the subnetwork of the common dimensional autoencoder. Let us denote vectors that appear on the bottleneck hidden layer

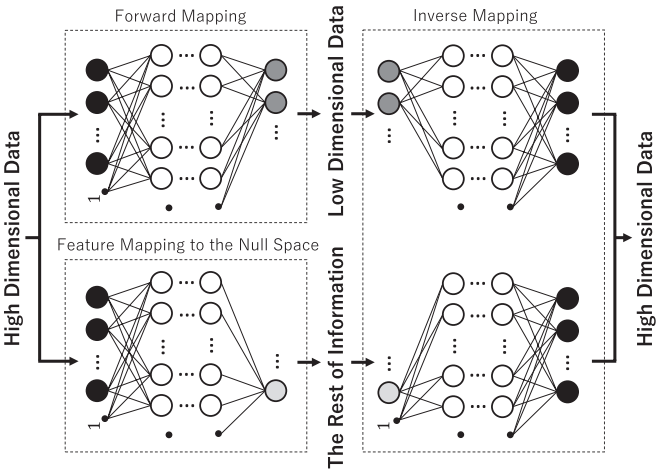

Fig. 3. Functions of subnetworks achieved in the common dimensional autoencoder. The network is sandwiched between the input layer, and the part of the bottleneck hidden layer where units received D_L (upper left) is explicitly trained to be the forward mapping. On the other hand, the rest half of the encoder (lower left) is not trained explicitly but implicitly guided to have the remaining information lost in the forward mapping as the result of training the entire autoencoder. Once the training was successful, it was obvious that the decoder (right) can be used for the inverse mapping. In particular, a variety of high dimensional vectors that conduces the same low dimensional data can be obtained by feeding the low dimensional vector to units depicted dark gray (upper right) and by feeding a variety of adequate vectors to units depicted light gray (lower left).

as h. Assuming a column vector $h_I^T \in R^M$ as the subvector of h where $(h_I^T, h_K^T)^T = h$ holds, the cost function for the supervised learning is written as follows:

$$L_{\mathrm{FM}} = \frac{1}{\|D_L\|} \sum_{\{u_i,q_i\}\in\{D_H,D_L\}} (h_I(u_i)-q_i)^T R (h_I(u_i)-q_i). \quad (2)$$

Where R and $h_I(u_i)$ indicates a weighting matrix and a vector that appears at h_I when u_i was fed onto the input layer, respectively. This is the general cost function of the simple feedforward neural network and governs training of the forward mapping.

The entire cost function for the common dimensional autoencoder is simply defined as:

$$L = L_{\mathrm{AE}} + L_{\mathrm{FM}}. \quad (3)$$

In the training, we do not assume any restriction of techniques about activation functions, optimization technique, and so on. Additionally, even for network structures of the encoder and the decoder, the proposed method allows using any configuration except for shortcut connections beyond the bottleneck layer.

As shown in the above, the rest of the units on the bottleneck hidden layer h_K is not trained explicitly. However, qualitatively speaking, h_K is required to have the information that has been lost by the many-to-one forward mapping from D_H to D_L because the entire network has to reproduce D_H on the output layer eventually. From the analogy of the linear algebra, h_I and h_K are thought of as playing as the image and the kernel of the function $f : u \to q$, respectively. The kernel (null space) is commonly employed to explain

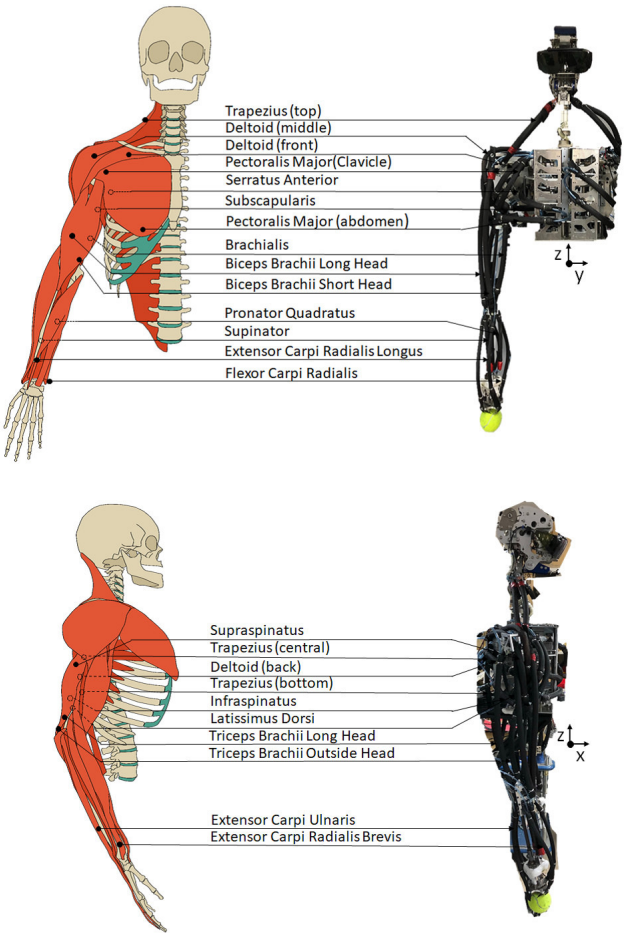

Fig. 4. Muscle arrangements of the musculoskeletal robot arm and a human. This robot arm has a 10 DOF skeletal structure where the scapula, the shoulder joint, the elbow joint, the forearm, and the wrist have 3 DOFs, 3 DOFs, 1 DOF, 1 DOF, and 2 DOFs, respectively. All joints are redundantly driven by 24 PAMs arranged similarly to the corresponding human muscles. Each PAM has a pressure sensor (SMC PSE-540), and the internal pressure is controlled by an air-flow control valve (Festo MPYE-5). A yellow ball is attached instead of the hand, and the Cartesian position is tracked using a depth camera (Intel RealSense ZR 300). The origin is located at the base of the upper torso. A visualization of the robot's coordinate system's origin is also referenced in the image above.

how the stiffness of the joint and other variables appear as redundancy of kinematics models in tasks, and the common dimensional autoencoder introduces the same viewpoint to complex musculoskeletal robots.

III. EXPERIMENT

In this section, we verify the proposed method by using the 10 DOF musculoskeletal robot arm[18]. Fig.4 shows the musculoskeletal structure which the robot arm has. In this robot arm, 24 pneumatic artificial muscles (PAMs) corresponding to muscle arrangement of humans are employed. Further, the Cartesian position of the hand, where the yellow ball is attached, is measured by a depth camera. In the experiment, we focus on the bidirectional mapping between the internal pressures of all PAMs and their resulting hand positions. In other words, in this experiment, the actuators'

state and the posture in the previous section are specifically corresponding to the internal pressures of PAMs and the Cartesian position of the hand. Therefore, $N = 24$ and $M = 3$ hold in this problem. It is clear that the forward mapping that transforms internal pressures to the corresponding hand position is a many-to-one mapping and redundancy appears in the inverse mapping.

In the following subsections, at first, the procedure to gather training data D_H and D_L is explained in detail. The structure and the training of the common dimensional autoencoder employed in this setup are also further explained. The second subsection contains the result of control based on the trained common dimensional autoencoder as well as the discussion.

A. Training of the Common Dimensional Autoencoder

To train the common dimensional autoencoder, D_H and D_L are collected by moving the musculoskeletal robot arm. A data point in D_H indicates a vector consisting of internal pressures of PAMs. To sample the data points, the pressure of each PAM was randomly selected set between 0 to 0.5 [MPa] with the interval 0.01. The corresponding data point in D_L that indicates the hand position was measured when the musculoskeletal robot arm reached the equilibrium at the given pressures. By repeatedly executing this procedure, 1 million points of D_H and D_L were gathered from the musculoskeletal robot arm.

Based on D_H and D_L, the training of the common dimensional autoencoder was carried out. About the network structure, because 24 actuators are used in this experiment, 24 units are used on the input layer, the bottleneck hidden layer, and the output layer. There are 3 hidden layers between the input layer and the bottleneck hidden layer as well as between the bottleneck hidden layer and the output layer, For these hidden layers, 180, 120, and 60 units are employed between the input and bottleneck layer, and in reverse order between the bottleneck and output layer. Each layer - except the output layer - has an additional bias unit that always outputs 1. Because the hand position is expressed in the form of a three dimensional vector, 3 units out of 24 units on the bottleneck hidden layer are explicitly trained to output the corresponding data points in D_L. For the training, the Adam optimization algorithm was used, and the entire cost value was sufficiently reduced.

B. Posture control based on inverse mapping

To verify whether or not the common dimensional autoencoder learned the bidirectional mapping between PAM pressures and hand positions, we conducted the hand position control experiment using the musculoskeletal robot arm. For the experiment, desired hand positions were generated as follows:

$$h_I^* = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0.15\cos\theta + 0.05 \\ 0.15\sin\theta - 0.30 \\ -0.27 \end{bmatrix} \tag{4}$$

where θ varies from 0 to 2π with the interval $\pi/10$. Thus, there are 20 desired hand positions on a circular path on a horizontal plane. In the decoder part of the common dimensional autoencoder (The right half of the network. See in Fig.3), the h_I, which three units trained by referring the D_L, receives one by one from these 20 desired hand positions.

If the proposed method works as expected and the training was done sufficiently, although the h_K meaning the rest of units on the bottleneck hidden layer affects the values on the output layer, the same hand position will be achieved by using these values as desired internal pressures of PAMs. Therefore, we can randomly generate values of 21 units contained in h_K in intervals where each unit could have during the training. However, how these values change joints' stiffness and the rest of the linkage positions and postures is unknown at this moment. Once both h_I and h_K are obtained, the vector $(h_I^T, h_K^T)^T$ is fed to the decoder part, and the output vector is used as desired pressures for the pressure feedback controller of the musculoskeletal robot arm.

Fig.5 shows the sequential snapshots when the hand position control was performed. In addition, Fig.6 shows the performance of the control. In these results, the h_K was fixed after the random generation. From these figures, although the accuracy was obviously low, it can be seen that the hand position control was qualitatively achieved.

In the next, we prepared five different h_K and executed the hand position control for each of five different h_K. Fig.7 shows the results of the hand position control. Ideally, these five trajectories should be the same, but errors can be seen in this figure. However, because these errors are reasonably smaller than the variance made by changing in h_I, it is thought of as that the proposed method worked as expected.

Fig.8 shows the variety of pressures of selected PAMs seen during trials shown in Fig.7. From this figure, it can be seen that where a large difference of pressures in each PAMs. Therefore, it indicates that changes in h_K actually resulted in the changes in pressures while keeping hand positions.

During the experiment, the common dimensional autoencoder was able to approximate both the forward and the inverse mapping of a redundant kinematics problem. Notably, on the bottleneck hidden layer, the vital feature that h_K automatically becomes to express the information about the redundancy in the forward mapping despite it is not trained explicitly was successfully validated. Therefore, we conclude that the validity and features of the proposed method are confirmed.

IV. DISCUSSION AND CONCLUSION

In this study, we proposed a new method that learn bidirectional mapping between redundant actuators' states and low dimensional postures. The proposed method employed an autoencoder which the number of units on the bottleneck layer is the same to that of the input layer and the output layer and imposes supervised training to a part of the bottleneck units. Therefore, We named this method as the common dimensional autoencoder. To train the network, high dimensional redundant vectors are used for the unsupervised training of autoencoder and low dimensional corresponding vectors are used for the supervised training

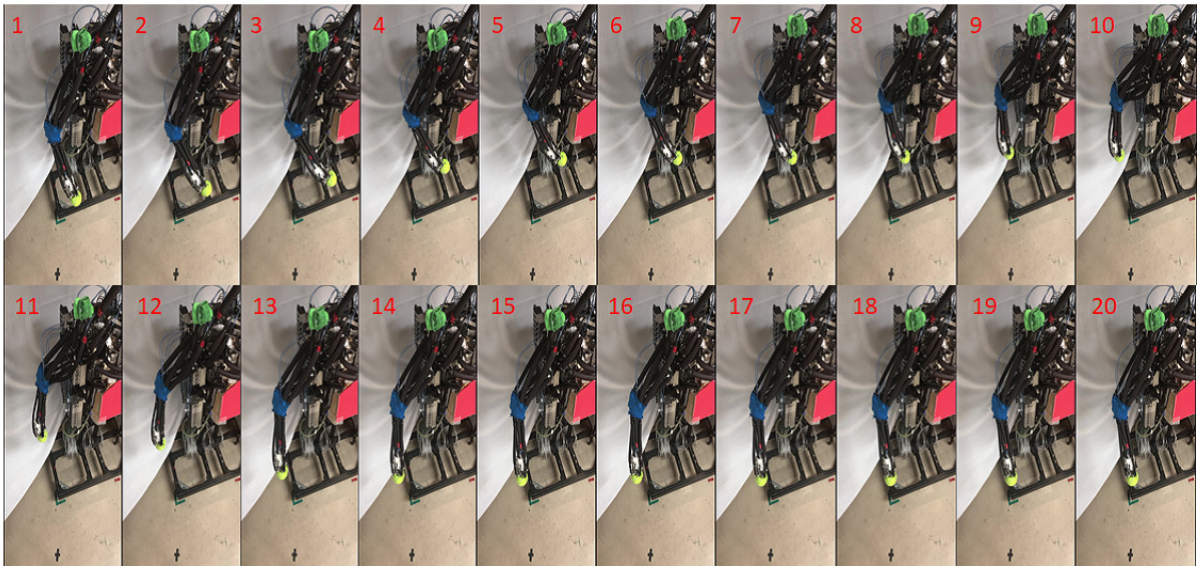

Fig. 5. Sequential snapshots of the hand position control of the musculoskeletal robot arm. PAM pressures were obtained by using the inverse mapping constituted in the right half of the trained common dimensional autoencoder. The hand drew a circular trajectory by feeding circular desired positions to the h_I of the decoder part.

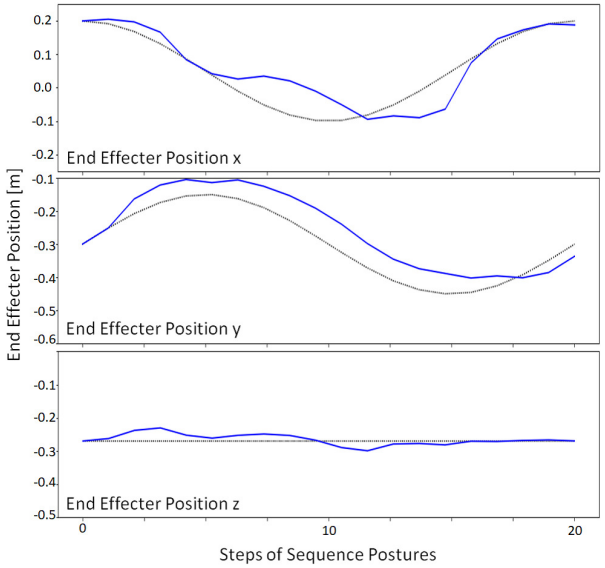

Fig. 6. The performance of the hand position control of the musculoskeletal robot arm shown in Fig.5. The top, the middle, and the bottom plots show the time-series of the hand position on x, y, and z axes, respectively. Black dotted lines indicate the trajectory obtained by smoothing desired hand positions given to h_I. Blue solid lines indicate the trajectory obtained by connecting realized hand positions by giving calculated pressure values to the feedback controller.

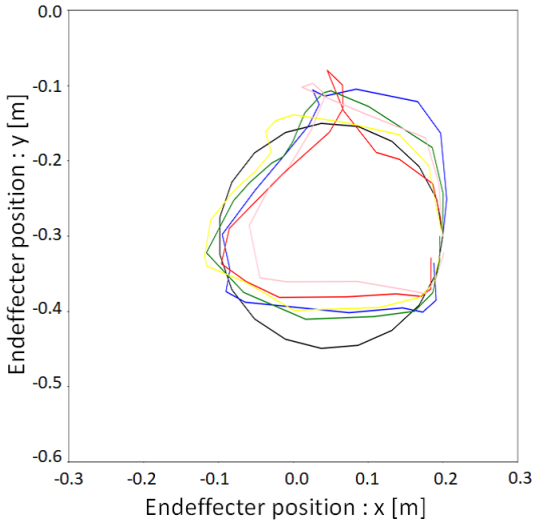

Fig. 7. Trajectories of hand positions obtained by executing the control with different h_K^T. The black line indicates the trajectory drawn by connecting by desired hand positions given to h_I. The other five different colored lines indicate trajectories obtained by connecting realized hand positions in trials with different h_K^T.

of a part of the bottleneck units. As the result, it is assumed that the remaining units on the bottleneck layer will keep the information that is going to be lost in the forward mapping. It indicates that both the forward and the inverse mapping are approximated in subnetworks of the common dimensional autoencoder. In order to validate the method, we conducted an experiment using 10 DOFs musculoskeletal robot arm driven by 24 pneumatic artificial muscles. Through the experiment, it was successfully confirmed that the common dimensional autoencoder learned both the forward and the inverse mapping between 24-dimensional pressure vectors and 3-dimensional hand positions.

Gathering training data was the largest difficulty for conducting the experiment. As explained in the last section, 1 million data points were collected by actually moving the musculoskeletal robot. Because approximately one second was taken for each points in order to wait the convergence to the equilibrium, more than 10 days are consumed for

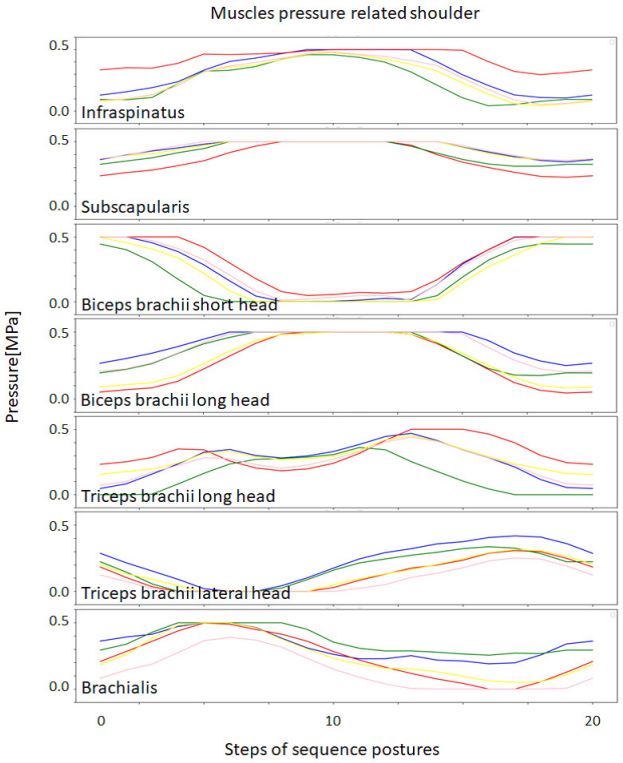

Fig. 8. Histories of changes in desired pressures of selected PAMs in the musculoskeletal robot arm. These muscles are all related to driving the scapula link. Five different colors are corresponding to them used in Fig.7.

just gathering data even if the robot arm can keep moving. In the long trials, PAMs were often repaired to cope with severe damage and the consistency of the hardware decayed. Furthermore, unignorable noise was also existing on sensor signals. To cope with these problem by improving the hardware will be necessary to improve the performance shown in this paper. On the other hand, the fact that the proposed method could still exhibit qualitatively valid results indicates its robustness.

Variable stiffness provided by musculoskeletal systems is one of the typical features that pose redundancy in the inverse mapping. In the proposed method, h_K bears the role to express the information corresponding to stiffness of joints. However, in this study, although it was confirmed that stiffness is possibly changed by changing the h_K, how h_K should be modulated to achieve desired stiffness was not investigated. Analyzing h_K and extracting the information of stiffness and other valuable factors will be one of important future works.

REFERENCES

[1] K. Hosoda, et al. Anthropomorphic muscularskeletal robotic upper limb for understanding embodied intelligence. Advanced Robotics 26.7 (2012): 729-744.

[2] K. Radkhah et al. Concept and design of the biobiped1 robot for human-like walking and running. International Journal of Humanoid Robotics 8.03 (2011): 439-458.

[3] Sharbafi, Maziar Ahmad, et al. A new biarticular actuator design facilitates control of leg function in BioBiped3. Bioinspiration biomimetics 11.4 (2016): 046003.

[4] B. Tondu, et al. A seven-degrees-of-freedom robot-arm driven by pneumatic artificial muscles for humanoid robots. The International Journal of Robotics Research 24.4 (2005): 257-274.

[5] D. Lau, et al. Musculoskeletal static workspace analysis of the human shoulder as a cable-driven robot. IEEE/ASME Transactions on Mechatronics 20.2 (2015): 978-984.

[6] Jacobsen, Stephen C., et al. Control strategies for tendon-driven manipulators. IEEE Control Systems Magazine 10.2 (1990): 23-28.

[7] K. Koganezawa, et al. Stiffness and angle control of antagonistially driven joint. The First IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics, 2006. BioRob 2006.. IEEE, 2006.

[8] Y. Nakata, et al. Hopping of a monopedal robot with a biarticular muscle driven by electromagnetic linear actuators. 2012 IEEE International Conference on Robotics and Automation. IEEE, 2012.

[9] S. Oh et al. Reaction force control of robot manipulator based on biarticular muscle viscoelasticity control. 2010 IEEE/ASME International Conference on Advanced Intelligent Mechatronics. IEEE, 2010.

[10] K. Radkhah, et al. Detailed dynamics modeling of BioBiped's monoarticular and biarticular tendon-driven actuation system. 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, 2012.

[11] Marques, Hugo Gravato, et al. ECCE1: The first of a series of anthropomimetic musculoskeletal upper torsos. 2010 10th IEEE-RAS International Conference on Humanoid Robots. IEEE, 2010.

[12] Y. Nakanishi, et al. Development of musculoskeletal humanoid kenzoh with mechanical compliance changeable tendons by nonlinear spring unit. 2011 IEEE International Conference on Robotics and Biomimetics. IEEE, 2011.

[13] T. Kozuki, et al. Design of upper limb by adhesion of muscles and bonesDetail human mimetic musculoskeletal humanoid kenshiro. 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, 2013.

[14] M. Jntsch, et al. A scalable joint-space controller for musculoskeletal robots with spherical joints. 2011 IEEE International Conference on Robotics and Biomimetics. IEEE, 2011.

[15] L. Jamone, et al. Machine-learning based control of a human-like tendon-driven neck. 2010 IEEE International Conference on Robotics and Automation. IEEE, 2010.

[16] S. Ookubo, et al. Learning nonlinear muscle-joint state mapping toward geometric model-free tendon driven musculoskeletal robots. 2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids). IEEE, 2015.

[17] H. Masuda, S. Ikemoto, and K Hosoda. Common Dimensional Autoencoder for Identifying Agonist-Antagonist Muscle Pairs in Musculoskeletal Robots. International Conference on Intelligent Autonomous Systems. Springer, Cham, 2018.

[18] A. Hitzmann, et al. Anthropomorphic musculoskeletal 10 degrees-of-freedom robot arm driven by pneumatic artificial muscles. Advanced Robotics 32.15 (2018): 865-878.