2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Variable In-Hand Manipulations for Tactile-Driven Robot Hand via CNN-LSTM

Satoshi Funabashi [1], Shun Ogasa [1], Tomoki Isobe [1], Tetsuya Ogata [2], Alexander Schmitz [1], Tito Pradhono Tomo [1] and Shigeki Sugano [1]

***Abstract*— Performing various in-hand manipulation tasks, without learning each individual task, would enable robots to act more versatile, while reducing the effort for training. However, in general it is difficult to achieve stable in-hand manipulation, because the contact state between the fingertips becomes difficult to model, especially for a robot hand with anthropomorphically shaped fingertips. Rich tactile feedback can aid the robust task execution, but on the other hand it is challenging to process high-dimensional tactile information. In the current paper we use two fingers of the Allegro hand, and each fingertip is anthropomorphically shaped and equipped not only with 6-axis force-torque (F/T) sensors, but also with uSkin tactile sensors, which provide 24 tri-axial measurements per fingertip. A convolutional neural network is used to process the high dimensional uSkin information, and a long short-term memory (LSTM) handles the time-series information. The network is trained to generate two different motions ("twist" and "push"). The desired motion is provided as a task-parameter to the network, with twist defined as -1 and push as +1. When values between -1 and +1 are used as the task parameter, the network is able to generate untrained motions in-between the two trained motions. Thereby, we can achieve multiple untrained manipulations, and can achieve robustness with high-dimensional tactile feedback.**

***Index Terms*— Multi-in-hand manipulation, Tactile sensing, Neural networks.**

I. INTRODUCTION

Our everyday lives require various manipulation tasks, and in-hand manipulation is no exception. Humans can perform different in-hand manipulations with the same tool or object. For example, when using a fork, humans use it for poking, dipping and stirring as they eat a variety of foods with different characteristics. Moreover, curved (anthropomorphic) fingertips play an important role in achieving such a variety of tasks, and enable a richer set of contact states compared to grippers with flat fingertips for example. However, while anthropomorphic fingertips can perform dexterous manipulation, achieving multiple in-hand manipulation tasks with such fingertips can be difficult, because they cause complex unstable touch states when certain conditions are encountered (e.g., slipperiness and rolling-contact). Because each task includes different contact states, even if the initial grasping postures for various tasks are virtually the same, typically each task is learned individually when using machine learning approaches.

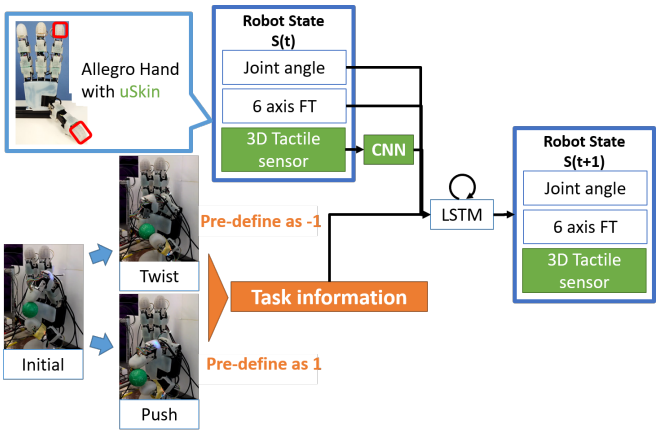


Fig. 1. The proposed architecture for variable in-hand manipulation tasks. Abundant tactile information is processed by a CNN (green color) and each task is defined by a task parameter in advance (orange color).

Our previous research has focused on generalization associated with in-hand manipulation tasks. First, the TWENDY-ONE hand achieved in-hand manipulation with objects of various sizes and shapes (generalization to objects) using tactile information, and we could show that with deep learning the required training data is reduced. Thereby we could demonstrate the importance of tactile information for in-hand manipulation and the potential of deep neural networks to adequately process such information [1]. Nevertheless, the network had to be trained for each in-hand manipulation task. Multi-fingered manipulation tasks with the Allegro anthropomorphic hand are also approached by one-shot learning to generalize to a different number of fingers used for the manipulation (adaptation to the number of fingers) [2]. The method in [2] again can also generalize to different sized and shaped objects. However, at least one training sample for each new manipulation is required, and if a new task requires motion that is totally different to the previously learned one, completely new training for each additional task is necessary. Therefore, the current paper approaches different motions by providing task information (as a task parameter) as shown in Fig. 1. This idea is similar to [1] in terms of providing a

This research was supported by the JSPS Grant-in-Aid No. 19H02116, No. 19H01130, the JST ACT-I Information and Future No. 50185, the Tateishi Science and Technology Foundation Research Grant (S), and the Research Institute for Science and Engineering, Waseda University.

[1]Satoshi Funabashi, Shun Ogasa, Tomoki Isobe, Alexander Schmitz, Tito Pradhono Tomo and Shigeki Sugano are with the Faculty of Science and Engineering, Dept. of Modern Mechanical Engineering, Waseda University, Tokyo 169-8555, Japan. (e-mail: s_funabashi@sugano.mech.waseda.ac.jp, s_ogasa@sugano.mech.waseda.ac.jp, t_isobe@sugano.mech.waseda.ac.jp, schmitz@aoni.waseda.jp, tito@toki.waseda.jp, sugano@waseda.jp)

[2]Tetsuya Ogata is with the Faculty of Science and Engineering, Dept. of Intermedia Art and Science, Waseda University, Tokyo 169-8555, Japan. (e-mail: ogata@waseda.jp)

task parameter to the network. In particular, in [1] the task parameter described the manipulated object, while in this paper the task parameter defines the motion. Furthermore, this method is used to generate untrained motions by using intermediate values between the trained task parameters. To the best of our knowledge, this is the first approach to multi-in-hand manipulation that employs such a technique. Already in our previous work we compared learned controllers with neural networks to predefined position control methods [1][2] and demonstrated that the neural networks achieved more stable and successful manipulation. Therefore, the current paper focuses on neural networks only, to achieve a deeper understanding of neural networks for in-hand manipulation, rather than comparing neural networks to other methods.

Therefore, this paper has two contributing ideas:

- A combination of a CNN for tactile data and LSTM for time-series information is implemented to achieve stable in-hand manipulation with rich tactile information.
- Providing task information to the network as a task parameter enables to perform untrained motions in-between the trained motions, thereby achieving a generalization of motions.

II. RELATED WORKS

A. In-Hand Manipulation with Tactile Sensors

When it comes to in-hand manipulation tasks, the shape of the hand's fingertips can also be important according to results reported in a previously-conducted investigation [3]. During in-hand manipulation with curved (e.g. anthropomorphic) and soft fingertips, the contact states such as the contact areas between the fingertips and objects which produce stable grasping change significantly, especially for achieving more versatile manipulation. To manage this, tactile sensors are essential for robotic hands so that the contact states can be detected directly [1], while the use of visual sensors in this scenario is problematic since the visual field may be occluded by the fingertips making it impossible to deduce the contact state at that time. Hence, some vision-based control methods achieve dexterous object reposing only when multiple cameras are facing the hand [4]. Contact modeling with external forces for in-hand object manipulations has also been previously proposed [5][6]. Modeling with tactile feedback is also used for achieving in-hand manipulation [7][8], but they are still specific to each target task and not adaptive to multiple in-hand manipulations. A Markov decision process is used as one of the reinforcement learning methods [9]. A combination of modeling and machine learning methods is useful for in-hand manipulation [10][11], however, it is difficult to generate complicated manipulations in these researches, because they use under-actuated hands with less controllability and therefore achieve only simple motions.

In our previous research, distributed 3-axis uSkin tactile sensors were developed and integrated into the fingertips [12] and phalanges [13] of Allegro hands. The sensors can be implemented in the curved skin of the fingertips and provide triaxial (x y z-axes) tactile information. Triaxial tactile information can be useful, specifically tactile information in the tangential direction along the fingertips and objects can improve many in-hand manipulation tasks with some research considering slip [14][15] or friction [16][17]. Also, using significantly more tactile information can improve the task performance (e.g. for object recognition [18]). However, too much tactile information could be counterproductive, in particular many machine learning approaches have difficulty in processing too high-dimensional data [19]. Nevertheless, especially deep learning has recently been shown to be able to process also high-dimensional information.

Therefore, CNNs which have demonstrated promising results for vision and robots [4], have been utilized with tactile sensors for recognition tasks [20][21]. In the current paper, CNNs are implemented for processing 3-axis tactile information for multi-in-hand manipulation tasks, for the first time to the best of our knowledge. Specifically, the CNN must learn features from several tasks at the same time, which could be difficult for the network due to the different touch states among different tasks.

B. Generalization to Multi-In-Hand Manipulation Tasks

Tactile sensing with deep neural networks could improve in-hand manipulation skills of robotic hands and generalization to a variety of objects, as we have previously confirmed [1].

However, generalization to a variety of in-hand manipulation tasks has yet to be achieved, as one network was used for each task [1]. Some studies have reported on the adaptation to multi-tasks [22][23]. Multi-time scale RNNs are applied to tasks which are then divided into sub-tasks [24]. Point clouds have been used to adapt to multiple demonstrations by optimizing probabilistic models [25]. Parametric bias (PB) has also been used for generalizing to not only multi-tasks but also untrained tasks including robotic arm manipulation tasks [26][27]. Since they use visual information but do not focus on tactile sensing, the manipulation does not include complicated in-hand manipulation. The PB itself is an effective method that is employed in many fields (e.g., action exploration in a map [28] and face detection [29]). As one of the state of the art methods for generalization to multi-tasks, one-hot vector has been exploited for generating multiple manipulation motions that share the same weights in the network architecture based on visual information and convolution-based VAE-GAN (variational autoencoder-generative adversarial network) and LSTM [30]. Even though there are some methods used for adapting to multi-manipulation tasks, including PB and one-hot vector, the network architectures for in-hand manipulation with tactile sensing that can manage multi-in-hand manipulation tasks have yet to be investigated.

III. PROPOSED METHOD

A. Allegro Hand with Tactile Sensors

As one of the commercially available multi-DOF robotic hands, an Allegro hand is used in this paper. Distributed 3-axis uSkin tactile sensors were mounted on the phalanges

[12] and fingertips [13] in our lab. Previous work confirmed the usefulness of 3-axis tactile information from uSkin for object recognition [18][31]. Therefore, we assumed uSkin could play an important role for in-hand manipulation as well. In addition, there are 6-axis F/T sensors in each fingertip to achieve more stable manipulation as effectively used in [2]. As described in Fig. 2, 6-axis F/T sensors provide 2 (fingertips) * 6 (force measurements) = 12 measurements and the uSkin sensors provide 2 (fingertips) * 3 (axis) * 24 (sensor chips) = 144. There are a total of 8 joints in the hand; thus, the hand provides 164 measurements in total.

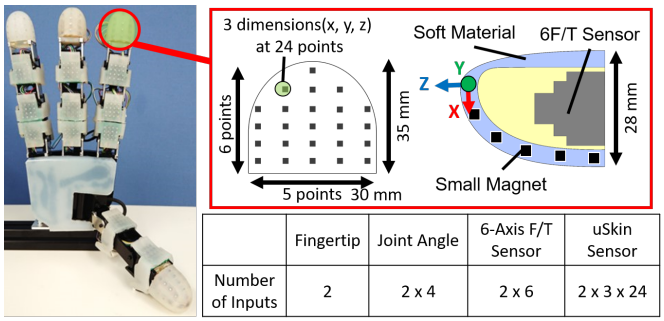
B. Network Architecture

Fig. 1 shows a schematic of our proposal. Shown in green, the 3-axis tactile information is processed by a CNN. Inputting tactile information from uSkin to the CNN is accomplished in the same manner as in [31], as described in Fig. 3. Twenty-four taxels on each fingertip have xyz information, as described in Fig. 3. The shape of the input maps from each fingertip is 6 * 5, with "0" (red colored) added to the positions where sensors are not installed, which makes it possible to construct rectangular input maps to convolute them by 2 * 2 or larger filters of CNNs. In the current paper, input maps have a size of 6 * 5 * 3 based on the uSkin coverage.

Task information corresponding to each task, which is highlighted in orange, is applied to the networks. In this research, two tasks are trained. Furthermore, parameters between the defined task information (-1 and 1) are used for generalizing to un-trained tasks.

A LSTM is also trained, as LSTMs in general are well suited to deal with time-series information. When the outputs from the CNN are input to the LSTM, the dimension is 20 because other inputs (joint angles and 6-axis F/T sensors measurements) to the LSTM have a similar number of dimensions and each input could be processed with equal importance by the LSTM.

In short, inputs to the LSTM are joint angles, 6-axis F/T sensors, 3-axis tactile information from uSkin, and task information.



	Fingertip	Joint Angle	6-Axis F/T Sensor	uSkin Sensor
Number of Inputs	2	2 x 4	2 x 6	2 x 3 x 24

Fig. 2. The customized Allegro hand with uSkin and 6-axis F/T sensors as used in this paper. Each fingertip has uSkin on the surface and a 6-axis F/T sensor inside of the fingertip. Twenty-four sensor points are placed in each uSkin and each point has x-, y-, and z-axis tactile information. Z-axis is for normal force. X- and y-axis are for shear forces along with the shape of the fingertip, i.e. uSkin is curved along with the anthropomorphic fingertip. The 6-axis F/T sensors are mounted the same way as in [1].

The outputs from the LSTM are the same as the inputs (but the output is the next time step from the input) except for the defined task information. We include the tactile measurements in the output, even though they are not used for control, because we expect that this makes the network more robust. The parameters for the network are described in Section IV-B, IV-C, and Table I.

As in previous work we already used and evaluated simpler network architectures, in particular multi-layer perceptrons [1], the current paper focuses on more advanced architectures, which are able to produce variable in-hand manipulations.

IV. EXPERIMENT DESIGN

A. Data Collection

In this study, posture interpolation control was used for collecting training data as it was used in our previous research [2][32]. Using this method, initial (starting) and final postures are defined and those postures (defined as joint angles) are interpolated resulting in generating in-hand manipulation motion. The details of this method can be found in [2][32]. Interpolation control can be used for achieving in-hand manipulation with robotic hands [33], but has limited adaptability to errors in the initial grasping positions on the fingertips and can generate too much force on objects which ends up breaking objects, as comparison experiments between neural networks and the interpolation control showed [1]. Interpolation control is a simple yet powerful technique to collect training data instead of using a data glove for teleoperation, especially for the Allegro hand. In particular, unlike the human hand, the Allegro hand lacks abduction/adduction in all finger, which makes precise inverse kinematics control of its fingertips and teleoperation impossible.

For each in-hand manipulation trial, the initial grasping position of the target object is determined randomly by the

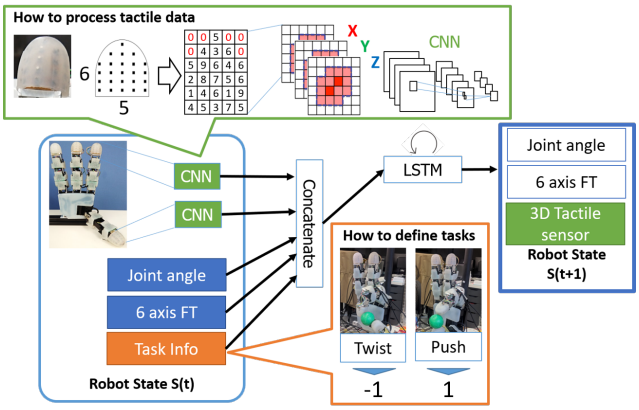


Fig. 3. Schematic of proposed method focusing on CNN processing of uSkin tactile information. Inputs from uSkin have a size of 6 * 5 by inputting "0" (red-colored) and 3 channels for x-, y- and z-axis. The compressed tactile features from both CNNs are concatenated with joint angles, 6-axis F/T sensors, and task information value before being used as input to the LSTM layer. The task information has a range from -1 (representing "Push") to 1 (representing "Twist"). The output from the LSTM layer is the next time step of the uSkin tactile information, joint angles, and 6-axis F/T sensors.

experimenter so that the neural networks can be trained effectively with training data that has motions starting from as many initial grasping positions as possible. An XYZ positioning stage was also used for deciding where to put an object on the fingertips (details are described in Section V-A).

The target in-hand manipulation motions (i.e., twisting and pushing) are shown in Fig. 4. For the twist motion, the grasping position of an object shifts from the top of the index finger to its side. For the push motion, the grasping position of the object changes from the top of the index finger and thumb to their bottom side. These motions were selected because they include various difficulties during in-hand manipulation. In particular, rolling-contact and slip can occur, caused by the shape of anthropomorphic fingertips. The contact areas between the object and fingertips can change gradually and lead to difficult but dexterous manipulation. The target motions are collected by interpolation control with selected final grasping postures. In this study, the target object was a spherical object made of styrene foam with a diameter of 40 mm. Only target motions were the focus of this study, unlike our previous research that focused on a variety of objects [1]. For collecting training data, each target task was executed 30 times successfully with a sampling rate of 100 Hz, resulting in a total of 60 trials and 400 time steps for each. Eighteen out of 30 trials from each task were used as the training dataset, and the remaining 12 trials from each task were used as the validation dataset to validate the trained neural network models. Therefore, the size of the training and validation dataset in total are 36 and 24 respectively. Each dataset was chosen randomly from 30 trials from each task. All the sensor values were normalized to values from -1 to 1 as input for the neural networks.

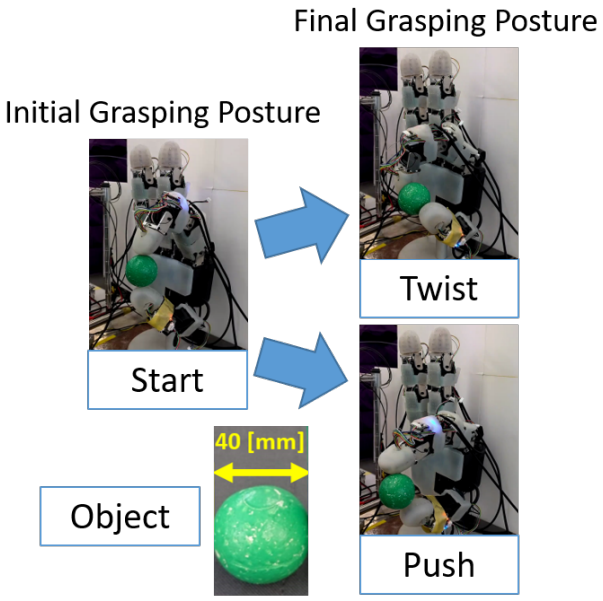


Fig. 4. There are two in-hand manipulation tasks (Twist and Push). The starting grasping posture for both tasks is the same. The manipulated object is a 40 mm sphere.

TABLE I
NEURAL NETWORK SETTING

		Architecture I	Architecture II	Architecture III	Architecture IV
Networks		LSTM	CNN-LSTM	CNN-LSTM	CNN-LSTMPB
All Inputs to Networks		165 uSkin 3-axis, 6-axis F/T, Joint Angle Defined Info	200 uSkin 3-axis, 6-axis F/T, Joint Angle	201 uSkin 3-axis, 6-axis F/T, Joint Angle, Defined Info	189 uSkin 3-axis, Joint Angle, Trained PB
Inputs to 1st Conv.		-	180		
1st Conv.	In/Out	-	3/14		
	Filter Size	-	3, 3		
	Stride	-	1, 1		
	Activation	-	Relu		
2nd Conv.	In/Out	-	14/28		
	Filter Size	-	3, 3		
	Stride	-	1, 1		
	Activation	-	Relu		
1st FC (uSkin)		-	200		
2nd FC (uSkin)		-	20		
Inputs to FC (all inputs)		165	40 (20 + 12 + 8)	41 (20 + 12 + 8 + 1)	29 (20 + 8 + 1)
1st FC (all inputs)		100	100		
2nd FC (all inputs)		200	200		
LSTM layer		200			
Outputs		164			8
Training Epochs		25000	27000	28000	
Batch Size		36	18	36	36

B. Neural Network Architectures

Four neural network architectures are described in Table I. The parameters for each network were heuristically selected by the experimenter with the smallest loss as a stopping criteria from each network during training and for successful motions on the real robot. The left side of Table I is a description of the layers. The dropout method was used in the 1st Conv., 2nd Conv., 1st FC (uSkin), and 1st FC (all) with a rate of 0.5. The output layer has a size of 164 (except for architecture IV), including the joint angles, the six-axis F/T sensors and uSkin sensors. We include the tactile measurements in the output, because we expect that this makes the learning more robust. However, only the joint angles were used for controlling the hand.

LSTM with defined task information (represented as "Defined Info" in Table I) was prepared as Architecture I. The LSTM itself was used for processing time-series information, and this was compared to CNN-LSTM with defined task information (Architecture III) to confirm the ability of convolution layers to process abundant tactile information. Since the other inputs (joint angles, 6-axis F/T sensors, and uSkin) are normalized from -1 to 1, the task parameter for the twist motion was defined as -1, and the task parameter for the push motion was defined as 1. The number of inputs for this architecture was 144 (24 (the number of sensors for a fingertip) * 3 (force axes) * 2 (number of fingertips)) from uSkin tactile sensors + 8 (joint angles) + 12 (6-axis

F/T sensors) + 1 (task info) = 165. The number of training epochs was 25,000 and the batch size was 36. Compared to [18], the number of training epochs is much larger, due to the difficulty of in-hand manipulation tasks.

Architecture II does not provide the task parameter as input to the neural network. Since this architecture has convolution layers that accept inputs from uSkin sensors, inputs from sensors on fingertips have the number 0 to provide input maps to the 1st Conv. Layer as described in Section III-A. Therefore, the number of inputs from the uSkin tactile sensors is 180 (24 (the number of sensors for a fingertip) + 6 (the number 0 for a fingertip) * 3 (force axes) * 2 (the number of fingertips), (6 * 5 * 3 * 2)). There are two fully-connected layers for uSkin sensor information (1st and 2nd FC (uSkin) layers) for compressing the features from the 2nd Conv. Layer into a similar size of the other inputs (joint angles and 6-axis F/T sensors) resulting in a uSkin feature dimension of 20. The number of total inputs for the 1st FC is 40 (20 uSkin features + 8 joint angles and 12 6-axis F/T sensors). After the 2nd FC layer (all inputs), the LSTM layer with a size of 200 provides outputs with the size of 164 which includes uSkin, joint and 6-axis F/T information. Each convolution layer had filters with a size of 3 * 3, stride of 1, and the activation function of Relu. The number of training epochs was 27,000. The batch size was 18 because this network was trained (and validated) separately for each of the two tasks due to the incapability of performing two tasks (pushing and twisting) by one network.

Architecture III is CNN-LSTM with defined task information (represented as "Defined Info" in Table I) which is almost the same as Architecture II, but defined task information is added to the inputs. Therefore, the input size to the 1st FC (all inputs) layer was 41 (40 + 1 task information). The output from the LSTM layer had a size of 164, which was the same as for the other architectures. The number of training epochs was 28,000. The batch size was 36.

Architecture IV CNN-LSTM with PB required the PB to be trained, as will be explained in more detail below. Inputs for this architecture are uSkin, joint angles and PB, and the output layer includes only joint angles because when 6-axis F/T sensor information was also input to the CNN-LSTM and the output layer includes not only joint angles but the others, the success rate of in-hand manipulation got worse. The number of training epochs was 28,000. The batch size was 36.

The Relu activation was used for all the layers including Conv., LSTM, and FC layers except for the output layer. the output layer did not have an activation function. The loss function used was the mean squared error. Adam was used as the optimizer for all the architectures except for Architecture IV (with a learning rate of 0.00001, a step size of 0.0001, a first exponential decay rate of 0.9, a second exponential decay rate 0.999, and a small value for numerical stability of 1e-08). The learning rate for Architecture IV was set to 0.0001 so that PB is optimally updated. Those four architectures were built with the Tensorflow library for Python, and the training of networks were accelerated using the GTX Geforce 1080 and RTX 2080 as the GPUs.

C. Parametric Bias for Task Generalization

As an alternative to the predefined task parameter, a PB can be trained. The PB was attached to a fully-connected layer where other inputs including joint angles, 6-axis F/T sensors and features of uSkin tactile information from CNNs were attached in the networks mentioned in Section III-B instead of defined task information (Architecture IV). The PB (used in Section V-B) was trained to update the PB value for each task. The value was initialized as 0, and it converged in a range of -1 to 1 after training. The PB was trained with corresponding in-hand manipulation tasks, with details described in [26].

V. EVALUATION

A. Adaptability to Initial Grasping Positions

First, the CNN is evaluated for its ability to adequately process tactile information from multi-in-hand manipulation tasks. The CNN was connected to the LSTM, as shown in Fig. 3 and was defined as CNN-LSTM in this paper. The CNN-LSTM with defined task information (Architecture III) was compared with LSTM with defined task information (Architecture I) regarding the success rate of in-hand manipulation. Success in this study was defined as follows: the object did not drop, while the final grasping posture was reached (pushing: the object reaches the bottom of the fingertips and both fingertips face each other; and twisting: the object reaches the side of the index fingertip and the middle of thumb). In all settings, the hand was controlled via closed-loop control, i.e. the current sensor readings are input to the neural network, and the joint angles at the output are used for control. The LSTM successfully performed the twist motion 3 times out of 10, and the push motion 5 times out of 10. On the contrary, CNN-LSTM could achieve the twist motion 7 out of 10 times and the push motion 8 out of 10 times.

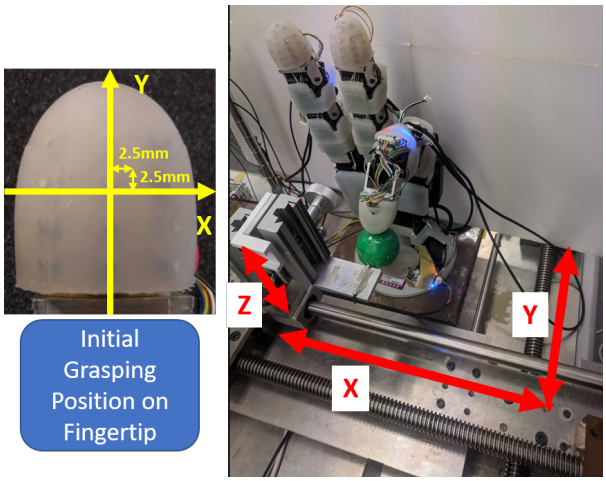


Fig. 5. The XYZ positioning stage for the experiment shown in Section V-A. The initial grasping position of the object is determined precisely by using the stage. The position (X, Y) = (0, 0) is the center of the fingertip.

Moreover, the adaptability of each network to the initial grasping position of the object was evaluated. An XYZ positioning stage was used, which makes it possible to put the object in x-, y-, and z-axis in steps of 0.1 mm. The results are presented in Fig. 6. The position of the object on the fingertips was modified in steps of 2.5 mm. The red color indicates the initial grasping positions on the fingertip from which a successful push or twist manipulation was performed three out of three times (100% success rate). For LSTM, 1 position for the twist motion and 5 positions for the push motion were successful. For CNN-LSTM, 23 positions for the twist motion and 39 positions for the push motion are where the manipulation was achieved.

With the two aforementioned experiments we confirmed that CNN-LSTM was better than only LSTM and that the CNN is useful for processing tactile information. Therefore, CNN-LSTM with defined task information (Architecture III) was used in the latter evaluation experiments.

B. In-Hand Manipulation with Task Information

As the first experiment for defined task information, comparison of CNN-LSTM with or without defined task information and with trained PB was conducted, by investigating the number of successful in-hand manipulations.

1) Training of Parametric Bias: The trained PB was also compared to using defined task information. While training the PB, the parameters in the PB were updated to acquire features which could represent each task. Fig. 8 shows the distribution of trained PBs for each task. PB values ranged from 0.2 to 0.8 and there was no overlap between tasks, which suggests that the PBs were sufficiently trained. For real-time control of the Allegro hand, the largest value for the twist motion and the smallest value for the push motion (indicated by arrows in Fig. 8) were used so that the network can tell apart the tasks clearly. PB values for each task were

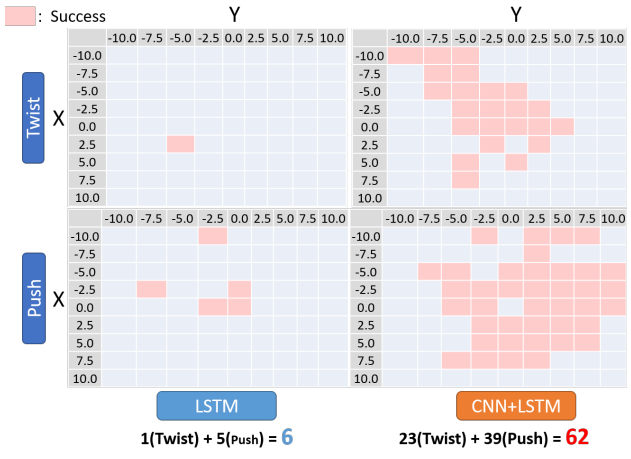


Fig. 6. The result of a comparison between Architecture I and Architecture III described in Section V-A. From the center of fingertips, the initial grasping positions are shifted in steps of 2.5 mm in x- and y- axes. Architecture III has more initial grasping positions where successful in-hand manipulation was achieved three times out of three. This result indicates that the CNN could extract informative features for executing stable in-hand manipulation.

TABLE II
SUCCESS RATES OF EACH IN-HAND MANIPULATION BY EACH ARCHITECTURE

	Architecture I	Architecture II	Architecture III	Architecture IV
Twist	3/10	2/10	7/10	7/10
Push	5/10	0/10	8/10	2/10

input to the network to generate in-hand manipulation for each task 10 times.

2) Result of In-Hand Manipulation: As shown in Table II, Architecture II CNN-LSTM could achieve only 2 successful in-hand manipulations in total (including both tasks), but Architecture III CNN-LSTM with defined task information could achieve 15 out of 20 times a successful in-hand manipulation in total (examples are shown in Fig. 7). One of the possible reasons could be that architecture III was trained with both twist and push motions, which resulted in the network getting more training data and becoming more robust with respect to not dropping the object, while still embracing adaptability to multi-tasks. Architecture II had one neural network for each task, so each network had only half the training data available, which seems to have been insufficient to learn stable in-hand manipulation. For Architecture IV, when the PB for the twist motion was used as input, manipulation for the twist motion was successfully generated 7 out of 10 times. On the other hand, when the PB for the push motion was input, successful push motion was generated only 2 out of 10 times, and the other 8 times included both dropping the object or performing twisting motion instead. This should be investigated further as part of future work, especially, how the trained PB values corresponded to each task, because it could be possible to confirm that the PB is also useful but that the PB values are just difficult to handle for in-hand manipulation.

C. Intermediate Grasping Motions with Task Information

By changing the value of the defined task information, the change in manipulation was investigated. Architecture III was used here. As shown in Fig. 9, the manipulation gradu-

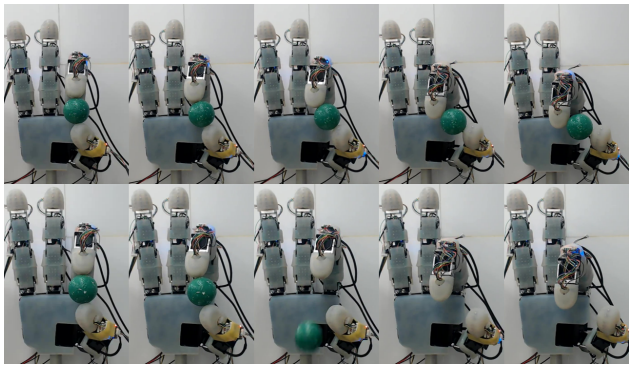


Fig. 7. The top row shows a successful in-hand manipulation with Architecture III (CNN-LSTM with defined task information). The bottom row is a sequence from Architecture II (CNN-LSTM without defined task information). While Architecture II is trained with only Twist training data, Architecture III was trained with Push and Twist, and Architecture III could achieve more stable in-hand manipulation.

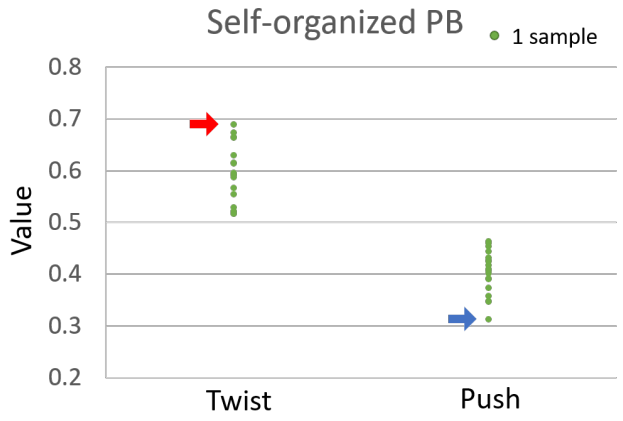


Fig. 8. The distribution of self-organized (trained) PBs for Twist and Push. All 18 points are where PB values converged after adequate training epochs. The PB values for Twist and Push are distributed in different ranges (Twist in a range from 0.5 to 0.7 and Push in a range from 0.3 to 0.5), which indicates that the training of PBs was successfully accomplished. For the real time control of the Allegro hand, the PB value pointed out by a red or blue arrow was used, respectively.

TABLE III
SUCCESS RATE OF EACH DEFINED TASK INFORMATION

Task Info	-1.0	-0.5	0.0	0.5	1. 0
Success Rate	7/10	7/10	8/10	8/10	8/10

ally changed from -1 as Twist to 1 as Push. When using a defined task information value of 0, an intermediate posture was successfully generated as the final grasping posture and the grasped object was not dropped by the hand. Successful in-hand manipulation was achieved at least 7 times out of 10 regardless of the defined task information value (-1 to 1 in 0.5 increments) shown in Table III. From this result, we confirmed that untrained tasks could be achieved by changing the task parameter.

VI. CONCLUSION

This paper presents the results of our neural network study with task information for multi-in-hand manipulation with 3-axis tactile sensors. CNNs and LSTMs were used for processing copious amounts of tactile and time-series information of in-hand manipulation. Task information was applied to tasks that required push and twist motions. The CNN was enough to process complex tactile information resulting in an improvement of robustness to initial grasping position. By using defined task information, the success rate of in-hand manipulation was higher than that without task information. Finally, defined task information was utilized to generate untrained tasks between two trained tasks. Importantly, we confirmed that it was possible to define the final grasping posture by changing the task parameter.

Future works could make in-hand manipulation more stable by changing hyper-parameters of CNN-LSTMs and increasing the size of the training dataset to avoid overfitting as our other work achieved stable manipulation when larger sized dataset was used [34]. Generalization to changing the orientation of the hand and increasing the number of tasks learned by a network are also possible in future work. This method could possibly scale up to be applied to multi-fingered and more complicated tasks.

REFERENCES

[1] S. Funabashi, A. Schmitz, T. Sato, S. Somlor, and S. Sugano, "Robust in-hand manipulation of variously sized and shaped objects," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept 2015, pp. 257–263.

[2] S. Funabashi, A. Schmitz, S. Ogasa, and S. Sugano, "Morphology specific stepwise learning of in-hand manipulation with a four-fingered hand," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 1, pp. 433–441, 2020.

[3] K. Or, A. Schmitz, S. Funabashi, M. Tomura, and S. Sugano, "Development of robotic fingertip morphology for enhanced manipulation stability," in *2016 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, July 2016, pp. 25–30.

[4] O. M. Andrychowicz, B. Baker, M. Chociej, R. Józefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, J. Schneider, S. Sidor, J. Tobin, P. Welinder, L. Weng, and W. Zaremba, "Learning dexterous in-hand manipulation," *The International Journal of Robotics Research*, vol. 39, no. 1, pp. 3–20, 2020. [Online]. Available: https://doi.org/10.1177/0278364919887447

[5] N. Chavan-Dafle and A. Rodriguez, "Prehensile pushing: In-hand manipulation with push-primitives," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2015, pp. 6215–6222.

[6] N. C. Dafle, A. Rodriguez, R. Paolini, B. Tang, S. S. Srinivasa, M. Erdmann, M. T. Mason, I. Lundberg, H. Staab, and T. Fuhlbrigge, "Extrinsic dexterity: In-hand manipulation with external forces," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 1578–1585.

[7] J. He, S. Pu, and J. Zhang, "Haptic and visual perception in in-hand manipulation system," in *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Dec 2015, pp. 303–308.

[8] J. Shi, J. Z. Woodruff, P. B. Umbanhowar, and K. M. Lynch, "Dynamic in-hand sliding manipulation," *IEEE Transactions on Robotics*, vol. 33, no. 4, pp. 778–795, Aug 2017.

[9] H. van Hoof, T. Hermans, G. Neumann, and J. Peters, "Learning robot in-hand manipulation with tactile features," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, Nov 2015, pp. 121–127.

[10] M. Liarokapis and A. M. Dollar, "Deriving dexterous, in-hand manipulation primitives for adaptive robot hands," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2017, pp. 1951–1958.

[11] M. V. Liarokapis and A. M. Dollar, "Learning task-specific models for dexterous, in-hand manipulation with simple, adaptive robot hands," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2016, pp. 2534–2541.

[12] T. P. Tomo, W. K. Wong, A. Schmitz, H. Kristanto, A. Sarazin, L. Jamone, S. Somlor, and S. Sugano, "A modular, distributed, soft, 3-axis sensor system for robot hands," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, Nov 2016, pp. 454–460.

[13] T. P. Tomo, A. Schmitz, W. K. Wong, H. Kristanto, S. Somlor, J. Hwang, L. Jamone, and S. Sugano, "Covering a robot fingertip with uskin: A soft electronic skin with distributed 3-axis force sensitive elements for robot hands," *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 124–131, Jan 2018.

[14] F. Veiga, H. van Hoof, J. Peters, and T. Hermans, "Stabilizing novel objects by learning to predict tactile slip," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2015, pp. 5065–5072.

[15] M. Stachowsky, T. Hummel, M. Moussa, and H. A. Abdullah, "A slip detection and correction strategy for precision robot grasping," *IEEE/ASME Transactions on Mechatronics*, vol. 21, no. 5, pp. 2214–2226, Oct 2016.

[16] M. Costanzo, G. D. Maria, and C. Natale, "Slipping control algorithms for object manipulation with sensorized parallel grippers," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 7455–7461.

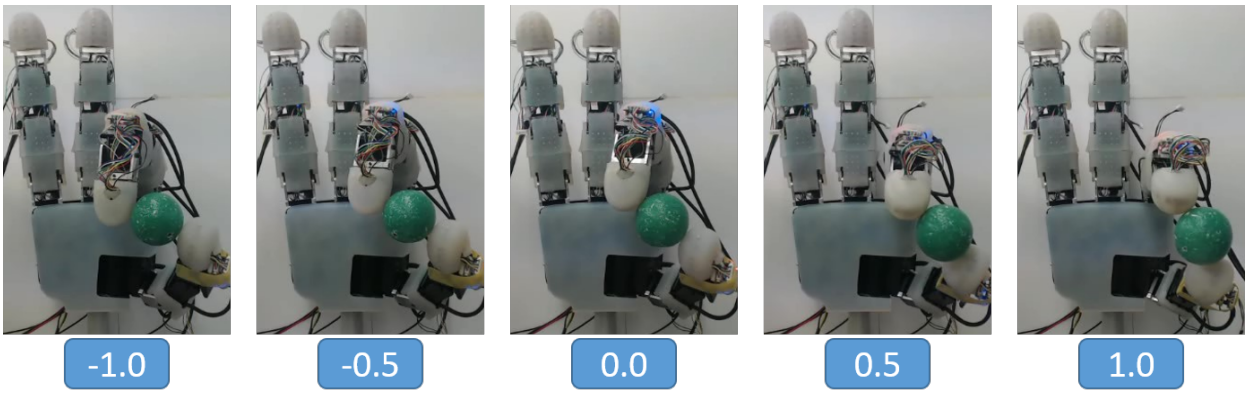


Fig. 9. The final grasping postures generated by changing defined task information. Push motion is clearly generated by defined task information value -1 and Twist is generated by task information value 1. From the defined task information value -0.5 to 0.5, the final grasping posture is shifting from Push to Twist, which implied that the desired final grasping postures could be determined in advance even they are not directly included in the training dataset.

[17] F. E. V. B., Y. Karayiannidis, C. Smith, and D. Kragic, "Adaptive control for pivoting with visual and tactile feedback," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 399–406.

[18] S. Funabashi, S. Morikuni, A. Geier, A. Schmitz, S. Ogasa, T. P. Torno, S. Somlor, and S. Sugano, "Object recognition through active sensing using a multi-fingered robot hand with 3d tactile sensors," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2018, pp. 2589–2595.

[19] S. S. Baishya and B. Bäuml, "Robust material classification with a tactile skin using deep learning," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 8–15.

[20] R. Calandra, A. Owens, D. Jayaraman, J. Lin, W. Yuan, J. Malik, E. H. Adelson, and S. Levine, "More Than a Feeling: Learning to Grasp and Regrasp using Vision and Touch," *ArXiv e-prints*, May 2018.

[21] W. Yuan, Y. Mo, S. Wang, and E. Adelson, "Active Clothing Material Perception using Tactile Sensing and Deep Learning," *ArXiv e-prints*, Nov. 2017.

[22] X. Long, M. Wonsick, V. Dimitrov, and T. Padır, "Anytime multi-task motion planning for humanoid robots," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2017, pp. 4452–4459.

[23] G. Raiola, X. Lamy, and F. Stulp, "Co-manipulation with multiple probabilistic virtual guides," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2015, pp. 7–13.

[24] K. Kase, K. Suzuki, P. Yang, H. Mori, and T. Ogata, "Put-in-box task generated from multiple discrete tasks by ahumanoid robot using deep learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 6447–6452.

[25] A. X. Lee, A. Gupta, H. Lu, S. Levine, and P. Abbeel, "Learning from multiple demonstrations using trajectory-aware non-rigid registration with applications to deformable object manipulation," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2015, pp. 5265–5272.

[26] J. Tani and M. Ito, "Self-organization of behavioral primitives as multiple attractor dynamics: A robot experiment," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 33, no. 4, pp. 481–488, July 2003.

[27] T. Ogata, S. Matsumoto, J. Tani, K. Komatani, and H. G. Okuno, "Human-robot cooperation using quasi-symbols generated by rnnpb model," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, April 2007, pp. 2156–2161.

[28] L. Rybicki, Y. Sugita, and J. Tani, "Reinforcement learning of multiple tasks using parametric bias," in *2009 International Joint Conference on Neural Networks*, June 2009, pp. 2732–2739.

[29] Y. Kim and X. Huynh, "Discrimination between genuine versus fake emotion using long-short term memory with parametric bias and facial landmarks," in *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, Oct 2017, pp. 3065–3072.

[30] R. Rahmatizadeh, P. Abolghasemi, L. Bölöni, and S. Levine, "Vision-based multi-task manipulation for inexpensive robots using end-to-end learning from demonstration," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 3758–3765.

[31] S. Funabashi, G. Yan, A. Geier, A. Schmitz, T. Ogata, and S. Sugano, "Morphology-specific convolutional neural networks for tactile object recognition with a multi-fingered hand," in *2019 International Conference on Robotics and Automation (ICRA)*, May 2019, pp. 57–63.

[32] K. Or, M. Tomura, A. Schmitz, S. Funabashi, and S. Sugano, "Interpolation control posture design for in-hand manipulation," in *2015 IEEE/SICE International Symposium on System Integration (SII)*, 2015, pp. 187–192.

[33] U. Scarcia, K. Hertkorn, C. Melchiorri, G. Palli, and T. Wimböck, "Local online planning of coordinated manipulation motion," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 6081–6087.

[34] S. Funabashi, T. Isobe, S. Ogasa, T. Ogata, A. Schmitz, T. P. Tomo, and S. Sugano, "Stable in-grasp manipulation with a low-cost robot hand by using 3-axis tactile sensors with a cnn," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, Accepted.