Tactile Transfer Learning and Object Recognition With a Multifingered Hand Using Morphology Specific Convolutional Neural Networks

Satoshi Funabashi, *Member, IEEE*, Gang Yan, *Student Member, IEEE*, Fei Hongyi, *Student Member, IEEE*, Alexander Schmitz, *Member, IEEE*, Lorenzo Jamone, *Member, IEEE*, Tetsuya Ogata, *Member, IEEE*, and Shigeki Sugano, *Fellow, IEEE*

***Abstract*— Multifingered robot hands can be extremely effective in physically exploring and recognizing objects, especially if they are extensively covered with distributed tactile sensors. Convolutional neural networks (CNNs) have been proven successful in processing high dimensional data, such as camera images, and are, therefore, very well suited to analyze distributed tactile information as well. However, a major challenge is to organize tactile inputs coming from different locations on the hand in a coherent structure that could leverage the computational properties of the CNN. Therefore, we introduce a morphology-specific CNN (MS-CNN), in which hierarchical convolutional layers are formed following the physical configuration of the tactile sensors on the robot. We equipped a four-fingered Allegro robot hand with several uSkin tactile sensors; overall, the hand is covered with 240 sensitive elements, each one measuring three-axis contact force. The MS-CNN layers process the tactile data hierarchically: at the level of small local clusters first, then each finger, and then the entire hand. We show experimentally that, after training, the robot hand can successfully recognize objects by a single touch, with a recognition rate of over 95%. Interestingly, the learned MS-CNN representation transfers well to novel tasks: by adding a limited amount of data about new objects, the network can recognize nine types of physical properties.**

***Index Terms*— Convolutional neural network (CNN), multifingered hand, object recognition, tactile sensing.**

I. Introduction

MULTIFINGERED hands are useful for the exploration and recognition of objects or environments by using multiple fingers dexterously (see Fig. 1). To achieve such multifingered tasks stably and effectively, tactile sensing was applied to a lot of tasks, such as grasp stability, detecting tactile events, and tactile exploration [1]. Those skills can also be crucial for multifingered manipulation where quick tactile feedback or recognition is required.

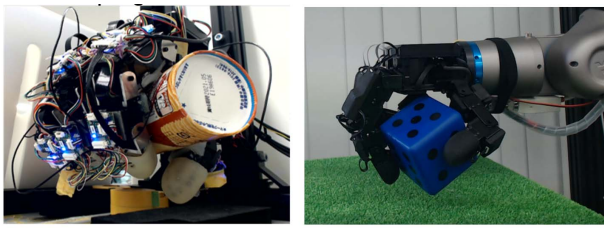


Fig. 1. Examples of applications for multifingered hands. Picking objects with diverse object properties is difficult even though the motion seems the same because the properties change the motion. During those kinds of multifingered tasks, acquiring object features through tactile information is important to achieve dexterous and stable manipulation.

Tactile sensing is considered to be complementary to other sensing modalities [2], especially when there is visual occlusion. In camera-based situations [3], the hand needs to be simple in shape to avoid occlusions, but there is a limitation in realizing difficult tasks, such as with multifingered hands. Therefore, for multifingered tasks, tactile sensing becomes a more important modality. In multifingered hand tasks, a diverse and relatively large area, including not only the fingertips but also the phalanges, comes into contact with the object, and the forces act in various directions as the fingers touch the grasped object from different directions [4]. A multifingered hand, such as a human mimetic hand, is capable of performing multipurpose tactile tasks [5], [6], but it is difficult to process such a rich amount of tactile information. Much research has been done on how a robotic hand equipped with tactile sensors can accomplish a task [7]. By considering the grasping state and fingertip position analytically, it has become possible to optimize the grasp and recognize the slip of the grasped object [8], [9]. However, complex grasping states are difficult to model analytically, and there are cases where only two fingers are used [10] or the touch is limited to the fingertips [11].

Manuscript received 15 September 2021; revised 16 August 2022; accepted 4 October 2022. This work was supported by the Japan Science and Technology Agency ACT-I Information and Future under Grant JPMJPR18UP and its Acceleration Phase under Grant JPMJPR18UP. *(Corresponding author: Satoshi Funabashi.)*

Satoshi Funabashi is with the Institute for AI and Robotics, Future Robotics Organization, Waseda University, Tokyo 169-8555, Japan (e-mail: s.funabashi.suganolab@gmail.com).

Gang Yan, Fei Hongyi, Alexander Schmitz, and Shigeki Sugano are with the Department of Modern Mechanical Engineering, Waseda University, Tokyo 169-8555, Japan.

Lorenzo Jamone is with the School of Electronic Engineering and Computer Science, Queen Mary University of London, E1 4NS London, U.K.

Tetsuya Ogata is with the Department of Intermedia Art and Science, Waseda University, Tokyo 169-8555, Japan.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2022.3215723.

Digital Object Identifier 10.1109/TNNLS.2022.3215723

Various machine learning methods are used for the recognition task; for example, a random forest was used for object classification for a simple two-fingered robot hand [12] and also for slip prediction for an arbitrary number of robot fingers [13]. Also, a support vector machine (SVM) has been used for zero-shot learning of object recognition tasks even though it has tactile information limited to fingertips [11].

Some other researchers have experimented with the fast estimation of object shape recognition when there is only one tactile sensor at a fingertip [14]. Active exploration methods, which use SVMs to learn by combining tactile and visual information, have been shown to be less accurate than learning from either alone [15]. This is one example where machine learning methods were not able to process sensor information of enormous size. Distributed tactile sensors have also been studied [16]. Even with SVM and self-organized maps (SOMs), the recognition rate remained low [17]. Also, research has already been done focusing on the features of multifingered hands [18]. By concatenating the information of each finger, a high recognition rate of objects can be obtained. While many studies have already been reported on the combination of robot hands and tactile sensors, there are still challenges in dealing with a large number of tactile sensors and learning high-dimensional tactile information.

Deep learning has been used for tactile sensors as one of the methods to process a large amount of tactile information. Deep learning has achieved better recognition rates compared to other machine learning methods, such as SVM [19]. Another example is the use of deep reinforcement learning to change the orientation of a cylindrical object. Although the hand is multifingered, the small number of pressure sensors and the high degree of freedom of the finger joints could make the situation difficult; moreover, it has not been used to manipulate various objects [20]. Many researchers have been working on convolutional neural networks (CNNs) for image and speech recognition because of their robust performance in extracting features from multidimensional information. This advantage of CNNs is well suited for distributed tactile sensors because the sensors are physically distributed on a 2-D surface. Therefore, CNNs have become widely used in distributed tactile sensors for robotic hands [21]. A state-of-the-art method focusing on CNNs and distributed tactile sensors has been developed [22]. CNNs have also been applied to tactile sensors for multifingered hands [23]. In our previous works, CNNs were also applied for in-hand manipulation and object recognition of an Allegro hand with three-axis tactile sensors [24], [25], and the results were better than the results of other modeling and machine learning methods.

While CNNs have achieved prominent results, it is still necessary to consider the problem of how to input tactile information from such sensors into the CNN when it comes to multifingered hands, as some hands have tactile sensors at the fingertips [26], while others have distributed sensors of different sizes and shapes [27]. This is particularly difficult because the size and shape of tactile patches on hands vary as much as the size of fingers, and in general, CNNs require rectangular input, which makes the implementation of CNNs difficult.

It also makes it difficult to extract multifingered level meaning from tactile information, such as object grasping and in-hand manipulation, due to its incapability of processing tactile information on multifingered hands at the same time. Previously, we used the uSkin three-axis tactile sensors [28], [29] attached to the Allegro hand, and in this article, we use the same dataset as in our papers [25], [30]. Both in [25] and [30], we obtained sufficient object recognition rates (95%). However, in [25], the input was a sequence of tactile states from different regions of the hand, while, in [30], it was a single, tactile state of the entire hand. Therefore, the architecture in [30] permits to have a much faster recognition. This is crucial especially if the information has to be used for real-time object manipulation (e.g., retrieving a stored 3-D model of the object after the object has been correctly recognized).

However, Funabashi et al. [30] just showed recognition rates and did not analyze how the CNNs process tactile information, and thus, it was not clear why this specific structure of the CNN was beneficial for multifingered hands. A more careful and detailed analysis of the internal representations created by the morphology-specific CNN (MS-CNN) would explain why the proposed structure is superior to other possible arrangements of the data, and whether it is beneficial for other tasks rather than object recognition. Finally, whether the CNNs are usable with other efficient training methods, such as transfer learning, to achieve better results has not been confirmed yet. It has to be evaluated whether CNNs can embrace a useful viewpoint, which is based on robotic morphology.

Even though the proposed method could be applied to multiple tasks, the data collection would require a high hardware load especially because tactile sensors directly touch objects and, in general, get worn out easily (e.g., [22]). This is especially important for multifingered hands because a lot of tactile sensors on the hands can break. In this case, transfer learning that is widely used for image recognition tasks [31] can be useful to reduce the size of the required training dataset with tactile sensors. There are some tactile transfer leaning methods that achieved high recognition rates [32], [33], yet they focus on only fingertips. N-shot learning was also used by specifying the size of training data [34]. Multimodal transfer learning, including vision and tactile information, was conducted [35], which can be difficult to implement in the case of multifingered hands due to occlusions. Sim-to-real transfer learning is one of the effective ways to collect training data [36], [37]. However, touching an object from diverse orientations happens due to multiple fingers and, thus, requires three-axis tactile information. That information is difficult to implement in simulation systems. Even though many tactile transfer learning methods have been used, transfer learning with a multifingered hand with tactile information from not only the fingertips but also the phalanges has not been investigated yet. Therefore, transfer learning was chosen for evaluating the MS-CNNs.

Therefore, the contributions of this study are given as follows. First, we review and compare the possible architectures of convolutional layers of the MS-CNN and how their combinations affect the object recognition rate.

Second, we visualize the internal representations of the different network structures using Grad-CAM++ [38] and highlight why and how such representations are the best candidates to support a wide range of robotic tasks related to multifingered hands. Finally, and most importantly, we demonstrate how these learned representations permit efficient transfer learning from the recognition of object instances to the recognition of physical properties (i.e., heaviness, slipperiness, and softness) of novel objects during in-hand manipulation with complex contact states on several fingers and show whether the CNNs are useful with transfer (WT) learning.

II. System Architecture

A. Hardware Design

This study uses the Allegro hand, a commercially available robotic hand from Wonik Robotics. Our uSkin tactile sensor, which was designed to detect forces in three axes, is used to cover the fingertips [28] and phalanges [29] of the Allegro hand, as shown in Fig. 2(a). As a multifingered robot hand with 16 degrees of freedom, the Allegro hand generates forces in many directions during manipulation. Therefore, the uSkin sensor has been implemented to detect such complicated grasping states. A total of 15 uSkin patches are installed: four on the index, middle, and little fingers, and three on the thumb. Thus, the customized Allegro hand has a total of 15 (sensor patches) $\times$ 16 (sensors) $\times$ 3 (tactile axes) + 16 (joint angles) = 736 measurements.

B. How to Input Tactile Information?

The locations of the sensors on the phalanges and fingertips are shown in Fig. 2(a) and (b). There are 16 sensors in each sensor patch. The phalanges and fingertips, however, differ in size and shape [see Fig. 2(b)]. Regarding the position of the sensors, the input map of the phalanges has a shape of 4×4, and the input map of the fingertips has a shape of 6×4. In the fingertip input map, the number "0" [the red number in Fig. 2(b)] is received at the position where each fingertip is not equipped with a sensor, resulting in a rectangular input map. This allows to convolute the input map with a filter of a size equal to or larger than 2×2. Some studies on image recognition use three channels of "RGB" input. This is because, in an image, each pixel has "RGB" information. Likewise, in this article, the input to CNN is set to three channels. This is because each of the sensors (or "taxel") provides xyz information [see Fig. 2(b)], and the orientation of xyz information on the patch is shown in [see Fig. 2(a)]. For the fingertips, the xyz-direction varies between taxels as the fingertips are curved. Specifically, the z-axis corresponds to the tactile information in the direction perpendicular to the sensor surface, and the other axes correspond to the tactile information in the tangential direction. This method of processing was used in [24], [30]. One way to process the tactile sensor patches is shown in Fig. 2(c). Here, as the first convolution layer, a convolution layer is prepared for each tactile patch. In the second layer, all the convolution layers of the first layer are combined in accordance with the position of the tactile sensor on the hand. In order to make the shape of the convolution layer rectangular by filtering, the first convolution

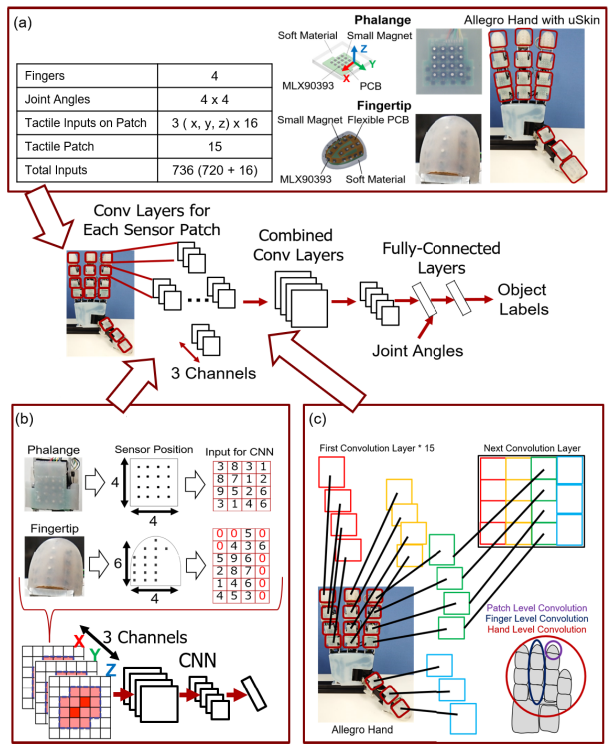


Fig. 2. Hardware setup and architecture of the proposed CNN. (a) Allegro hand with a set of uSkin three-axis tactile sensors mounted on the phalanges and fingertips (hardware setting). There are four uSkin sensor patches on the index, middle, and little fingers, and three patches on the thumb. (b) From left to right, the mounted sensor patches, the location of the sensors in black dots, and the input map for the CNN (how to input tactile information). Each red "0" stands for a position where no sensor is mounted on the corresponding actual sensor patch (other values are arbitrary). The map is used for input to the CNN in three channels (x, y, z). (c) One example is how to combine the convolution layers from each tactile sensor patch (how to combine tactile features). The current robotic hand platform has different sizes of sensor patches for fingertips and phalanges. In addition, there are different numbers of patches for the thumb and other fingers. In the example shown in this figure, each tactile patch has its own convolution layer (patch-level convolution) and is combined at a later stage (hand-level convolution).

layer convolutes the input from each sensor patch and converts it into an output of the required size. The height and width sizes of the filters in the convolution layer are adjusted as follows:

$$\mathrm{OH} = \frac{H + 2P - \mathrm{FH}}{S} + 1$$
$$\mathrm{OW} = \frac{W + 2P - \mathrm{FW}}{S} + 1 \qquad (1)$$

where H and W are the height and width of the current convolution layer's input, OH and OW are the height and width of the current convolution layer's output, and FH and FW are the height and width of the filters that convolute the input into the output of the next convolution layer. P is the padding, which typically adds "0" around the input map, so as to keep the outputs of the convolution layers the same size. Since the size of the input varies depending on the combination of convolution layers, padding is not used in this article. In addition, as we focus on how the convolution layers are combined, there is no pooling layer in the CNN used in our experiments that change the size of the input.

III. Experiment Design

A. Data Collection

Instead of collecting tactile information only when an object was statically grasped, we recorded tactile information through a series of manipulations in the hand as an active tactile sensing method. Since the objects were provided to the Allegro hand in random positions for training, recognition should be according to the object, not the way it was grasped. Note that, because the tactile measurements are not calibrated, the sensitivity varies, and furthermore, crosstalk between measurement axes may occur (see [28], [29] for details). However, we assume that, as long as the features from the measurements are extracted by neural networks, there is no need to calibrate the measurements.

1) Object Recognition: For the training data of object recognition, we use the same data as in our previous paper [25], [30], where we confirmed the effect of uSkin sensors on a multifingered hand by focusing on time series information and spatial information to improve the accuracy of object recognition by CNNs. In this article, we focus on spatial information and its analysis. The data were collected in a way that mimicks an infant's manipulation and exploration of objects to obtain information about their physical properties. Fig. 3(a) shows the 20 common objects used in the experiment, which also includes ten objects from the Yale-CMU-Berkeley Model Set. In particular, the objects in the top row of Fig. 3(a) and 20: spray bottle are elongated in shape, and when grasping the object, the palm of the hand always faces roughly the elongated side, e.g., the bottle was never grasped from the cap or bottom, and the orientation varied, with the object being grasped close to the center, but not necessarily exactly at the center. It is also important to note that, in this experiment, even after repeated grasping, the objects do not always have the same orientation, as the final grasping posture depends on the weight distribution. The other ten objects were roughly spherical and grasped in random orientations. Some examples of active tactile sensing for elongated objects can be seen in Fig. 3(c). Note that the Allegro hand controlled the position with a constant controller gain in all trials, and the reaction force varied depending on the size and shape of the object.

Thirty manipulation trials were performed for each object, for a total of 600 trials. Twenty-five trials for each object were used for training the CNNs, and five trials were used for each CNN's test set [see Fig. 3(a)]. Data were collected at a sampling rate of 30 Hz. Red is the raw data recorded, and green is the data extracted for training and testing shown in Fig. 3(e). The last step of the 250 steps was set less than one second before the end of the recording (the recording always stops at the same time after the movement stops). Note that the hand remains grasping the object even after the movement has stopped. Twenty-five timesteps from 250 timesteps in each of the 25 trials used for training were randomly sampled and used as the training dataset. The training dataset contains a total of 12 500 samples. The test dataset contained 2500 samples randomly sampled from five trials for each object.

2) Object Property Recognition: For the training data of object property recognition, the target in-hand manipulation chosen in this study was a movement of a precision grasp to a power grasp by starting with picking motion of fingertips from a ground [see Fig. 3(d)]. Since this motion embraces a variety of complicated contact states (e.g., slip, rolling contact, touch, and not touch by finger gating) during executing the manipulation, it was chosen to evaluate the proposed CNN's adaptability and whether it is applicable to other tactile recognition tasks or not. Since it is difficult to derive a formula to generate the motion, the training data were collected via the CyberGlove [40], which is a dataglove that enables teleoperate human-mimetic robot hands by sending joint motions of their fingers to robot fingers. One experimenter collected all data because the motion can be drastically different among trials when different people do the experiment, i.e., human hands are different from each other and require different calibration settings for the dataglove.

As shown in Fig. 3(b), there are 45 objects, including three objects from the Yale-CMU-Berkeley Model Set. The objects were separated into three heaviness classes [Heaviness High (more than 136 g), Heaviness Medium (77–114 g), and Heaviness Low (under 68 g)], three softness classes [Softness High (deformable), Softness Medium (only surface deformable), and Softness Low (stiff objects)], and three slipperiness classes [Slipperiness High (plastic or coated paper), Slipperiness Medium (paper or bumpy), and Slipperiness Low (textile or rubber)]. Each class has at least one object and sometimes several objects in terms of outer properties so that the networks can acquire a generalization skill for the inner properties (e.g., different-sized balls and differently shaped plastic fruits).

Two trials of the in-hand manipulation motion for each of the 45 objects were collected resulting in 90 trials in total. Data were collected with a sampling rate of 100 Hz. The objects were placed near the Allegro hand in random positions. Fig. 3(f) shows tactile trajectories obtained from one trial, where red indicates the raw data recorded and green indicates the data extracted for training and testing. The data were recorded for 17 s, and the manipulation was executed during the period. Since the motion was executed by a human, the motion itself and when the motion finishes are always differently shown in Fig. 4. Therefore, the data for training in each trial were extracted especially when the hand touches the grasped object with a threshold of tactile measurements. Several timesteps where the movement stops were also sampled as a static state. 291 timesteps were randomly sampled from each trial. In total, 26 189 samples were randomly chosen and used. The samples were split randomly into the training dataset and the test dataset consisting of 23 570 and 2619 samples, respectively. A different random split was conducted for each training trial of CNNs.

3) Object Property Recognition With Transfer Learning: For the training data of object property recognition, the datasets described in Sections III-A1 and III-A2 were used. First, the datasets for object recognition were used for a pretraining phase of the transfer learning, and thus, the task for the pretraining was the same object recognition task described in Section III-A1. Therefore, the size of samples for the training dataset and the test dataset is 23 570 and 2619, respectively. Second, the datasets for object recognition property recognition were used for a fine-tuning phase of the

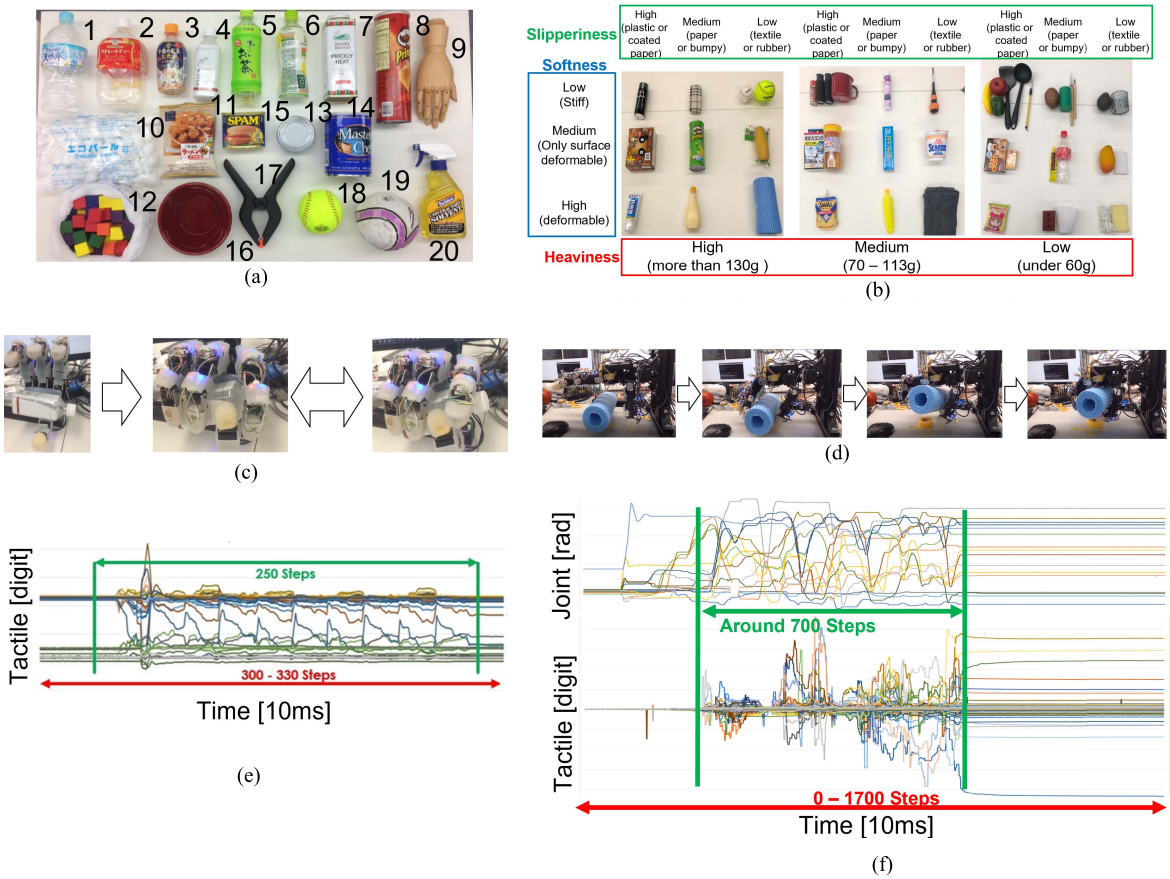


Fig. 3. Collection of objects and training data. (a) Selection of 20 daily objects (target objects for object recognition). Ten objects were selected from the "Yale-CMU-Berkeley Model Set" [39]. To increase the difficulty of recognition, we chose ten slightly more difficult identifiable objects. The bottles are labeled L (large), M (medium), and S (small). 1: bottle (L, spheric); 2: bottle (L, cornered); 3: bottle (S, spheric); 4: bottle (S, cornered); 5: bottle (M, spheric); 6: bottle (M, waisted); 7: powder can; 8: pringles; 9: hand model; 10: pack of Styrofoam dices; 11: pack of snacks; 12: pack of solid dices; 13: tuna can; 14: large can; 15: spam can; 16: bowl; 17: clipper; 18: baseball; 19: soccer ball; and 20: spray bottle. (b) 45 daily objects that were selected in terms of inner object properties (heaviness, softness, and slipperiness) (target objects for object property recognition). (c) shows how the training data were collected (target motion for object recognition). The motion is to mimic a human baby's squishing. Fingers move left and right. The skin sensors are rubbed to the grasped object. (d) Target motion of the object property recognition is to start a grasping posture by pinching it with fingertips. (e) Example of tactile time-series data throughout squishing trial (tactile trajectories during object recognition motion). The repetitive motion is reflected in the tactile data stream. Red indicates the complete data stream; green indicates the extracted raw data for learning object recognition. (f) Example of tactile time-series data throughout object picking trial (tactile trajectories during object properly recognition motion). Red indicates the complete data stream; green indicates the extracted raw data for learning object property recognition. The data are extracted when the hand touches objects, and thus, the extracted timesteps depend on each trial.

transfer learning. The task for the fine-tuning was the same object property recognition task described in Section III-A2 but with different training settings to the one for object property recognition without transfer learning.

B. Training Setting

All the CNNs were built with the Tensorflow library for Python and a GTX Geforce 1080, 1080Ti and RTX 2080 were used as the GPU. Object recognition and object property recognition have different training settings because the object recognition task requires the CNNs to generate one-hot vectors for so-called multiclass classification, while the object property recognition task requires the CNNs to generate "multi"-hot vectors for so-called multilabel classification.

Otherwise, with the exception of the size of the convolution layer, all the CNNs used in this article have the same network parameters and training settings, and they are described in Fig. 5(b). For the CNN input, we used samples, including 16 (joint angles) + 15 (sensor patches) × 16 (sensor) × 3 (tactile axis); thus, 736 measurements at each time step were obtained. To train the CNN, we added "0" to the input from the fingertip sensor, as shown in Fig. 3(b). Consequently, the number of dimensions of the input is 736 + 8 (number of "0"s for one fingertip) × 3 (tactile axis) × 4 (number of fingertips) = 832 dimensions.

Transfer learning was conducted with object property recognition. In this study, acquired features (or trained weights and biases) from the object recognition task were reused for (transferred to) the object property recognition task to get higher recognition rates. As far as the authors know, this is the first time to achieve transfer learning for tactile tasks with a multifingered hand.

1) Object Recognition: Each CNN was trained with 12 500 samples, and the test set consisted of 2500 different samples for up to 10 000 epochs until the training loss converged and the test loss for the test set did not go up. Weights for all CNNs were initialized with a random number from a

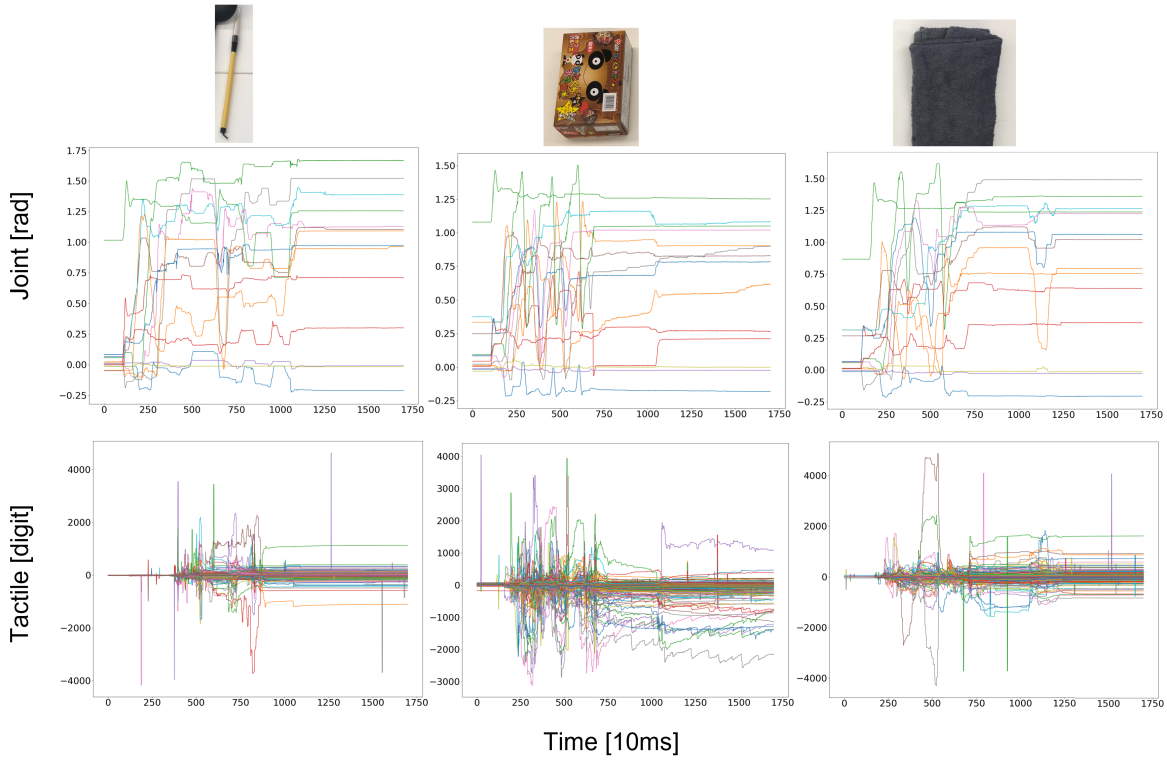


Fig. 4. Examples of the used objects for object property recognition and corresponding time-series (joint and tactile measurements) data. As shown in the graphs, even though each trial of manipulation starts and ends at a different time and the trajectories of joint and tactile information differently change, the proposed CNNs could get high recognition rates of the object properties.

truncated normal distribution with a standard deviation of 0.1, with a random seed defined as 1. All layers except the output layer used ReLU as the activation function. Softmax was used as the activation function for the output layer

$$f(x_i) = \frac{e_i^x}{\sum_k^C e^{x_k}} \tag{2}$$

where x_i is a feature from the ith neuron in the output layer and $f(x_i)$ is the activated output from the output layer. Since the sum of the values from the output layer is 1.0 through Softmax, it can be regarded as a probability. It is suitable for categorical cross (CC) entropy as the loss function. This CC Loss is defined as

$$\text{CCLoss} = -\log \frac{e^{x_p}}{\sum_k^C e^{x_k}} \tag{3}$$

where t_k and x_k are the target label and the output of the CNN for class k of C classes. The optimizer used was Adam, and the minibatch size was set to 100. The step size of the optimizer α was 0.0001, the first exponential decay rate $\beta 1$ was 0.9, the second exponential decay rate $\beta 2$ was 0.999, and the small value of numerical stability ε was $1e-08$. Moreover, the learning rate was 0.00001.

2) Object Property Recognition: For the training of the object property recognition task, each CNN was trained with 23 490 samples, and the test set consisted of 2610 different samples for up to 10 000 epochs until the training loss converged and the test loss for the test set did not go up. ReLU was used as an activation function for all the layers except the output layer. The output layer has a sigmoid activation function unlike object recognition

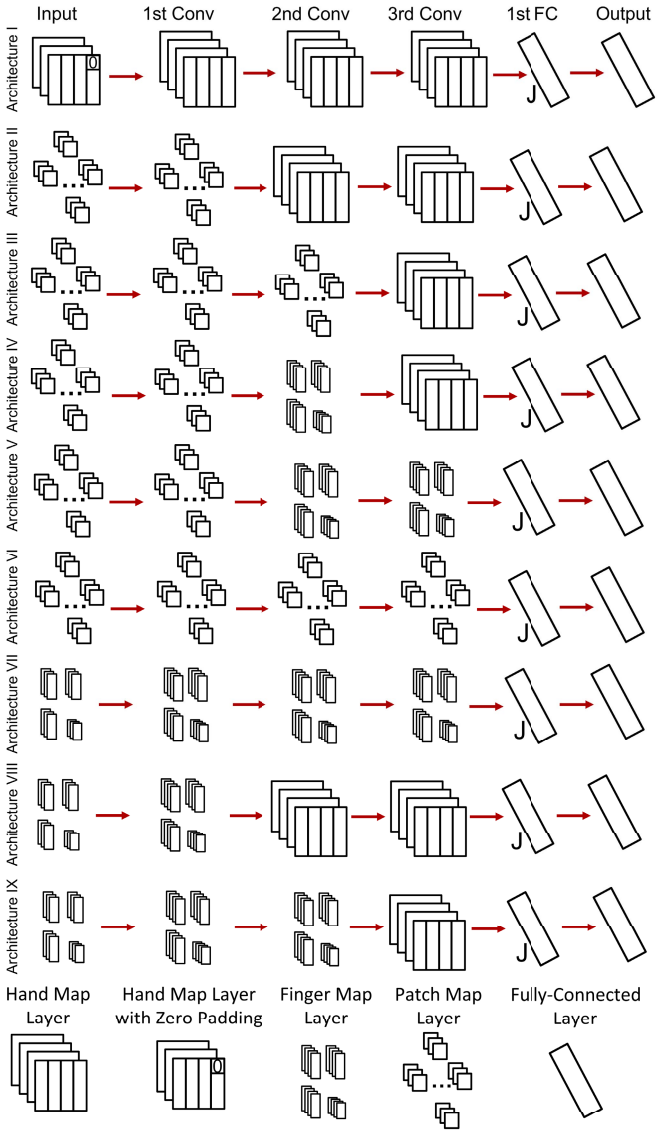
$$f(x_i) = \frac{1}{1 + e^{-x_i}} \tag{4}$$

where x_i is a feature from ith neuron in the output layer and $f(x_i)$ is activated output from the output layer. The output values from the output layer take a range of 0 to 1 through the sigmoid. The loss function was binary cross (BC) entropy. This BC loss is defined as

$$\text{BCLoss} = -\sum_k^C t_k \log(x_k) \tag{5}$$

where t_k and x_k are the target label and the output of the CNN for class k in C classes. The optimizer, the learning rate, and the minibatch size were the same as that of object recognition.

3) Object Property Recognition With Transfer Learning: For pretraining in transfer learning, the models that were made in the object recognition task described in Section III-B1 were prepared as pretrained models for object property recognition. Those pretrained models were fine-tuned for the object property recognition task. Each MS-CNN has differently shaped convolution layers, as shown in Fig. 5, the difference is investigated by considering the recognition results for transfer learning in Sections IV-D–IV-F, and the size of the output layer for the two tasks is different (20 for object recognition and nine for object property recognition). From these points, the weights and biases in the fully connected (FC) layers



(a)

		1st Conv	2nd Conv	3rd Conv	1st FC	Output
Architecture I	In	3	14	28	1136	1200
	Out	14	28	56	1200	20
	Filter Size	(3,3)	(3,3)	(2,2)	-	-
	Stride	(2,2)	(1,1)	(1,1)	-	-
Architecture II	In	3	14	28	1136	1200
	Out	14	28	56	1200	20
	Filter Size	(3,3)/(2,3)	(3,3)	(4,3)	-	-
	Stride	(1,1)	(1,1)	(1,1)	-	-
Architecture III	In	3	14	28	912	1200
	Out	14	28	56	1200	20
	Filter Size	(3,3)	(2,2)/(3,2)	(2,1)	-	-
	Stride	(1,1)	(1,1)	(1,1)	-	-
Architecture IV	In	3	14	28	1024	1200
	Out	14	28	56	1200	20
	Filter Size	(3,3)	(2,2)/(4,2)	(2,2)	-	-
	Stride	(1,1)	(1,1)	(1,1)	-	-
Architecture V	In	3	14	28	1192	1200
	Out	14	28	56	1200	20
	Filter Size	(2,2)	(5,2)	(5,2)	-	-
	Stride	(1,1)	(1,1)	(1,1)	-	-
Architecture VI	In	3	14	28	1304	1200
	Out	14	28	56	1200	20
	Filter Size	(2,2)	(2,2)	(2,2)	-	-
	Stride	(1,1)	(1,1)	(1,1)	-	-
Architecture VII	In	3	14	28	1024	1200
	Out	14	28	56	1200	20
	Filter Size	(4,2)	(3,2)	(2,1)	-	-
	Stride	(2,2)	(1,1)	(1,1)	-	-
Architecture VIII	In	3	14	28	1024	1200
	Out	14	28	56	1200	20
	Filter Size	(4,2)	(3,2)	(2,1)	-	-
	Stride	(2,2)	(1,1)	(1,1)	-	-
Architecture IX	In	3	14	28	1024	1200
	Out	14	28	56	1200	20
	Filter Size	(4,2)	(3,2)	(2,1)	-	-
	Stride	(2,2)	(1,1)	(1,1)	-	-

(b)

Fig. 5. Nine architectures for combining convolution layers. (a) First FC gets features from the third convolution (Conv) layer and joint angles (J) (combining patterns of MS-CNN architectures). Architectures II and III, and VII and IX are prepared to see the difference in when to combine convolution layers. (b) Each parameter for constructing convolution layers is described (parameter settings for MS-CNN architectures).

(FC1 and output layers in Fig. 5) were not transferred, only the ones in the convolution layers.

For fine-tuning in transfer learning, the weights and biases in the convolution layers were not updated during fine-tuning. The pretrained models were trained with 13 094 samples and 13 095 samples as the test set for up to 1000 epochs as training losses of the models got converged, and the test loss for the test set did not go up (the original dataset with 26 189 samples was split half and a half). The samples were randomly chosen for the training and test datasets for each training trial. All the hyperparameters for training, such as learning rate or training epoch, were the same as the object property recognition task in Section III-B2. However, in the case of the transfer learning, the size of the training dataset was almost 50% smaller than the training of object property recognition task (and the test set was almost 50% larger) as transfer learning is supposed to transfer useful knowledge to train networks for another task and result in being able to reduce the size of the training dataset. This was also because, when the training and test dataset sizes for object property recognition tasks were used, it was difficult to see a difference among recognition results from the CNNs used for transfer learning. Those datasets were created by randomly choosing samples every time the object property recognition trial was conducted.

C. Combining Architectures

Different architectures were considered to combine the convolutional (Conv) layers. The location of the hand sensors was taken into account when defining these architectures. Particularly, the choice of applying filters across the boundaries of

finger segments and different stages of fingers was considered. This idea considering merging convolution layers for extracting features at the different stages with each resolution can be useful for achieving high object recognition performance [41]. As shown in Fig. 5(a), there are nine architectures, and the parameters of each architecture are shown in Fig. 5(b). For example, in architecture I, the first Conv layer has an input of size $18 \times 16 \times 3$. When this is passed to the first Conv layer, the output from the first Conv layer is $10 \times 8 \times 14$. In a previous paper of ours [30], we evaluated the comparison between three-axis and one-axis tactile information and showed that using three-axis tactile information is more effective than that of one-axis. Therefore, Fig. 5(b) shows the parameter settings for the three axes only, which are determined heuristically. The joint angle measurements were added to the first FC layer, represented by J in Fig. 5(a), for all architectures. Weight sharing was tried to reduce the number of filters between fingers. However, when we evaluated the results with and without weight sharing, we found that weight sharing did not improve the recognition rate compared to the case where different weights were used in [30]. Therefore, only different weights of CNNs were used in this study. To adjust the size of the convolution layer, we used filters of different sizes in architectures II, III, and IV.

The measurements of joint angles and tactile sensors were, respectively, normalized by their full measurement range. In architecture I, a "Hand Map Layer with Zero Padding" was constructed as an input layer with a size of 18 rows $\times$ 16 columns. Three tactile patches [4 $\times$ 4 $\times$ (2 patches) and 6 $\times$ 4 $\times$ (1 patch)] for the thumb and four patches [4 $\times$ 4 $\times$ (3 patches) and 6 $\times$ 4 $\times$ (1 patch)] for the other fingers were implemented, respectively. Therefore, to construct a rectangular input layer, we had to add four rows $\times$ four columns of "0" on top of the tactile input map from the thumb fingertip, as shown in Fig. 5(a). As a result, the dimensionality of the input is $736 + 8$ (number of "0"s on one fingertip) $\times$ 3 (tactile axis) $\times$ 4 (number of fingertips) $+$ 16 (4 (rows) $\times$ 4 (columns) above the thumb's fingertip) $\times$ 3 (tactile axis) $=$ 880 dimensions. The filter was applied across the boundaries of the finger segments already set up in the first layer. The purpose of this architecture is to see whether it is sufficient to combine all the information into one large input layer or not.

In Architecture II, each one of the 15 sensor patches is processed separately in the first convolution layer (as a "Patch Map Layer"). In this case, the first convolution layer is the "Patch Map Layer," i.e., no filters are applied across the boundaries of the sensor patches, and hence, for every different sensor patch, a different filter is trained. In the next layer, the output of the first convolution layer is combined taking into account the location of the tactile patches of the hand in Fig. 2(c) ("Hand Map Layer"). For the input map from the phalanges of the thumb, a filter of size (2, 3) is used in the first Conv layer. The details are shown in Fig. 5(b).

In Architecture III, a "Patch Map Layer" is also used for the second Conv layer. A "Hand Map Layer" is implemented in the third Conv layer. For the thumb fingertip, a filter of size (2, 2) is used in the second Conv layer. The only difference between architectures II and III is when the "Hand Map Layer" is constructed. This idea comes from [16], which studied the timing of information fusion for grasp stability.

Architecture Pattern	Accuracy / Variance (%)
Architecture I	94.17 / 0.69
Architecture II	93.46 / 1.26
Architecture III	94.03 / 1.71
Architecture IV	95.59 / 0.86
Architecture V	94.01 / 0.78
Architecture VI	94.13 / 2.47
Architecture VII	94.45 / 0.40
Architecture VIII	94.85 / 0.54
Architecture IX	94.13 / 0.75

Fig. 6. Average and variance of recognition rate for ten times from each architecture. Architecture IV shows the best recognition rate.

In Architecture IV, we have a "Patch Map Layer" as the first Conv layer. Thereby, the maps of the four fingers (index finger, middle finger, little finger, and thumb) are constructed as the "Finger Map Layer." The "Hand Map Layer" is constructed in the third Conv layer. This architecture changes the map architecture from a "Patch Map Layer" to a "Hand Map Layer," one Conv layer at a time. A filter of size (4, 2) of the second Conv layer is used for the Conv layer from the phalanges and fingertip of the thumb.

In Architecture V, a "Patch Map Layer" is used for the input layer and the first Conv layer. A "Finger Map Layer" consists of the second and third Conv layers. Architecture V differs from Architecture IV in that it does not have a "Hand Map Layer" but still gradually combines the Conv layers.

Architecture VI consists of only a "Patch Map Layer" for the input and all Conv layers. This architecture is to check whether we need to combine convolution layers according to the position of the patch on the hand.

Architecture VII uses only a "Finger Map Layer" for the input and all Conv layers, to check whether it is sufficient to consider the position of the patches on the fingers.

Architecture VIII uses the "Finger Map Layer" in the input and the first Conv layers so that we confirm if a morphological fusion of convolution layers (i.e., finger- and hand-shaped convolution layers) extracts more useful features for tactile recognition tasks or not.

Architecture IX uses the "Finger Map Layer" in the input, first, and second Conv layers. The difference between architectures VIII and IX lies in when to build the "Hand Map Layer" with the same reason as for architectures II and III.

IV. Evaluation

A. Combining Architectures and Object Recognition Rates

In Fig. 6, the accuracy and variance presented are the mean values of ten recognition trials. In each trial, 1500 samples were randomly selected from the 2500 samples in the test set.

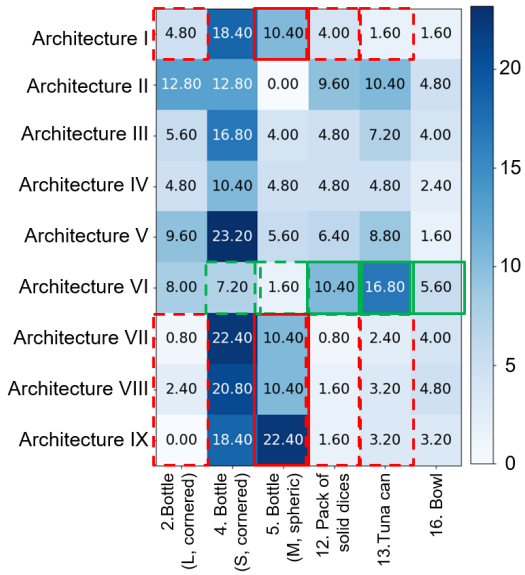


Fig. 7. Error rates of object recognition for six objects from each architecture. The red solid square shows where architectures that do not have the "Patch Map Layer" have relatively higher error rates. The red dot square shows where architectures that do not have the "Patch Map Layer" have lower error rates. The green solid square shows where architecture VI that only has "Patch Map Layer" has relatively higher error rates. The green dot circle shows where architecture VI that has lower error rates.

Architecture II showed a slightly lower recognition rate than the others. The variance for architecture VI is the highest as the architecture has the largest number of weight parameters in the convolution filters. The best recognition rate was achieved by architecture IV, which includes the Patch, Finger, and Hand Map Layers.

B. Object Recognition Rate for Each Object

In this section, the object recognition rate for each object is investigated to elaborate on the effect of morphological convolution architectures. Specifically, six objects (2: bottle (L, cornered), 4: bottle (S, cornered), 5: bottle (M, spheric), 12: pack of solid dices, 13: tuna can, and 16: bowl) are investigated, as shown in Fig. 7. For 2: bottle (L, cornered), 12: pack of solid dices, and 13: tuna can, four CNN architectures [architectures I, VII, VIII, and IX (with red dot squares)] achieved a lower error rate than other architectures. On the other hand, they have relatively higher error rates (with red solid squares) than the other architectures for 5: bottle (M, spheric). Those architectures do not have Patch Map Layers, and tactile sensor patches are combined from their input layer as the "Finger Map Layer" or the "Hand Map Layer."

Moreover, architecture VI has a lower error rate for 4: bottle (S, cornered) and 5: bottle (M, spheric) (with green dot squares), and the others have the highest wrong recognition rate with 12: pack of solid dices, 13: tuna can, and 16: bowl (with green solid squares). From this result, depending on having the "Patch Map Layer" or not, the recognition rate for each object can be changed.

C. Visualization of Sensor Map With Weights From Convolution Layers

Since the error rates of object recognition can change according to the structure of the convolution layers, how the weights in the last convolution layer (third Conv layer) in each architecture react to tactile measurements was investigated. Grad-CAM++ and guided Grad-CAM++ [38], which provides a saliency map of calculated weights from the last convolution layer corresponding to tactile measurements, were used, as shown in Fig. 8. The saliency map is defined as follows [38]:

$$L_{ij}^{c} = \mathrm{ReLU}\left(\sum_{k} \boldsymbol{w_k^c} \cdot \boldsymbol{A_{ij}^k}\right) \tag{6}$$

where w_k^c is a weight for a feature map. The map is defined as A_{ij}^k at the ith and jth spatial location for class c. For the map provided by Grad-CAM++, blue to red represent the lowest to highest values, respectively, and the pixels where high values are placed show where the network regards the information as important. For the map provided by guided Grad-CAM++, the pixels where convolution layers focus on are emphasized with colors, while the other pixels are depicted with gray. This section focuses on the objects that are described in Section IV-B. For architectures I, II, and IV, the saliency maps provided by Grad-CAM++ show that the tactile information is wholly focused on by their last convolution layer. On the other hand, architecture VI shows that the layer focuses on a small part of the tactile information, relatively. Also, architecture VII focuses on tactile information in a line-shaped fashion.

It seems that these differences happen because one or several convolution layers in the last layer (third Conv layer) are weighed heavily among the convolution layers. Since there is only one convolution layer in the last layer of architectures I, III, and IV, they seem to regard entire tactile information as important. Some objects picked in Figs. 7 and 8 (2: bottle (L, cornered), 12: pack of solid dices, 13: tuna can, and 16: bowl) have a relatively complicated shape compared to the rest of the objects [4: bottle (S, cornered) and 5: bottle (M, spheric)]. This shape can change contact patterns on each part of the multifingered hand. Therefore, when the hand grasps 12: pack of solid dices, for example, contact patterns on each finger segment can be different which are provided by an edge, side, and plane of the dices. On the other hand, 5: bottle (M, spheric) has small enough size that the hand grasps it wholly and has a cylindrical shape, which can produce a similar contact pattern on the hand while grasping. From this result, we deduce that it was easy for architecture VI to recognize relatively simple shaped objects [4: bottle (S, cornered) and 5: bottle (M, spheric)] because the network focuses on a small part of the contact areas, which can be enough to perform object recognition due to the similar contact pattern on any part of the Allegro hand. However, when it comes to complicated shaped objects, such as 2: bottle (L, cornered), 12: pack of solid dices, 13: tuna can, and 16: bowl, they were difficult for architecture VI to recognize because it focuses on small contact areas, but contact patterns on the contact areas are diverse. This misleads the network to recognize objects wrongly. For the other architectures,

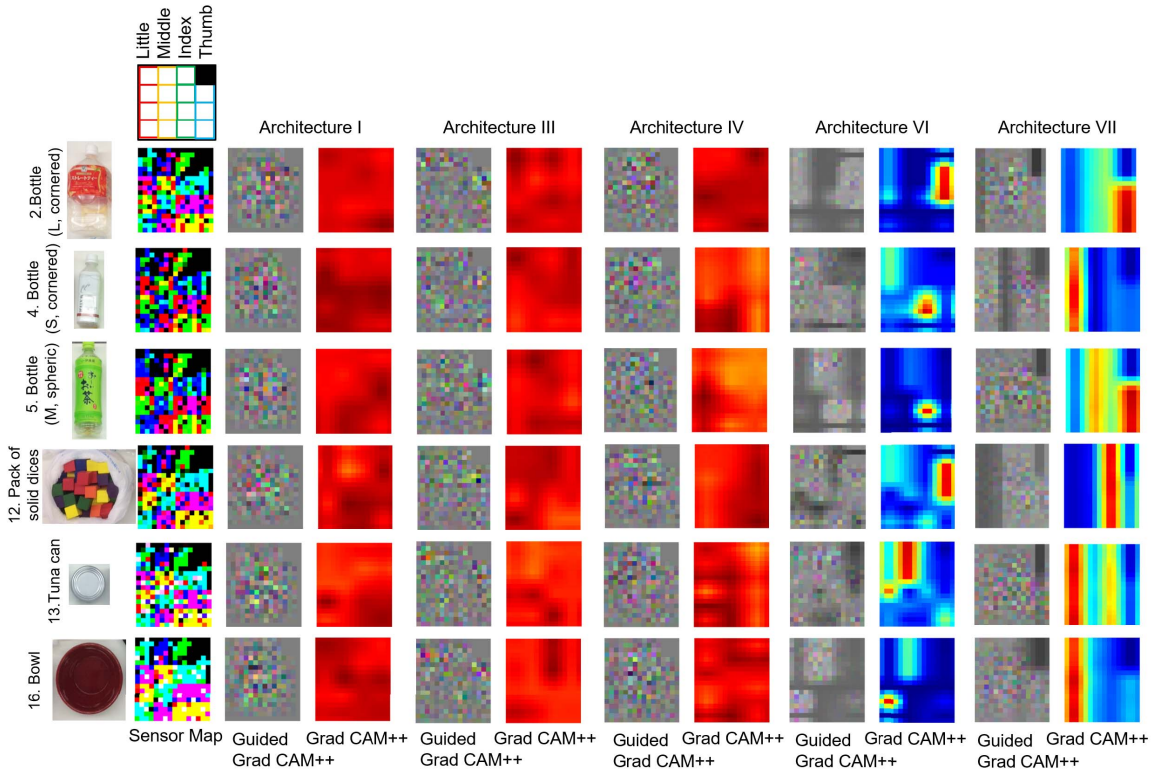


Fig. 8. Saliency maps from the last convolution layer in each MS-CNN generated by Guided Grad-CAM++ and Grad-CAM++ from [38]. The maps generated by both visualization methods for architectures I, III, and IV that have the "Hand Map Layer" clarify that the networks focus on the entire tactile information. Architecture VI that only has the "Patch Map Layer" shows it focuses on small areas compared to the other networks. Interestingly, architecture VII seems to see tactile information in a line fashion. Since those maps are generated by the last convolution layer and the layer shape is different for each architecture, the difference in the saliency maps among networks seems to happen as each network chooses which layer in the last convolution layer to focus on.

architectures I, II, and IV focus on whole tactile information, and architecture VII focuses on larger areas compared to architecture VI. Thus, they have better recognition rates when the hand grasps the relatively complicated shaped objects. Also, as shown in Fig. 7, architectures I, VII, VIII, and IX have lower error rates for the complicated objects specifically because they have combined inputs from the input layer. Furthermore, it can be considered that architecture IV could have the best object recognition rate with a variety of objects in terms of shape (or contact patterns) in total because the network has a "well-balanced" network architecture (i.e., the "Patch Map Layer," the "Finger Map Layer," and the "Hand Map Layer"). From this result, we hypothesize that a CNN that has combined convolution layers (i.e., the "Hand Map Layer") sees the entire tactile information on the multifingered hand and is well suited for recognizing complicated contact states.

D. Object Property Recognition and Transfer Learning

Object property recognition was chosen as the target task. The accuracy was calculated as a mean of all nine outputs of the CNNs. The recognition trials were conducted five times for each CNN. Fig. 9(a) shows the mean accuracies and their variances of the five trials. Architectures I, III, IV, VI, and VII were chosen. As a result, architecture IV got again the best recognition rate.

Architecture Pattern	Accuracy / Variance (%)	
	No transfer	With transfer
Architecture I	86.78 / 7.65	96.96 / 4.44
Architecture III	88.07 / 0.45	96.29 / 2.73
Architecture IV	89.20 / 0.88	98.27 / 3.65
Architecture VI	82.87 / 2.35	96.07 / 5.12
Architecture VII	81.80 / 2.90	93.60 / 7.18

(a)

Models	Accuracy / Variance (%)
Architecture IV	95.59 / 0.86
Resnet-18	4.045 / 0.14
Resnet-34	4.622 / 0.30
ShuffleNetV2	5.022 / 0.24
MnasNet	4.555 / 0.12
SVM	63.90 / 1.29

(b)

Fig. 9. (a) Recognition rate of the object property (comparison of the proposed CNN architectures). On the left-hand side of the table, averages and variances of recognition rate for five times of no transfer (NT) models are shown. The accuracy difference among CNNs for object property recognition is larger than that of object recognition shown in Fig. 6. Furthermore, architectures I, III, and IV have a large difference in comparison with architectures VI and VII. On the right-hand side of the table, averages and variances of recognition rate for five times WT models are shown. The recognition rate got around 10% better than without transfer learning. Note that architecture IV still has the highest recognition rate, and architectures I, III, and IV that have a "Hand Map Layer" have better recognition rates than the others. (b) Recognition rate of the object property (comparison with popular CNNs). On the right-hand side of the table, averages and variances of recognition rate for five times of each model are shown.

Also, architectures I and II got better recognition rates than the others, which do not have a "Hand Map Layer." However, a "Patch Map Layer" can improve the recognition rate from the result that architecture I has less recognition rate and huge variance compared to architectures III and IV. The "Finger

Map Layer" also contributes to a good result as architecture IV got better results than architecture III.

Regarding transfer learning, most architectures got around 10% better recognition rates, and architecture IV got the best accuracy. Furthermore, the trend that networks get better results with the "Hand Map Layer" and/or the "Patch Map Layer" was kept.

E. Classifier Comparison for Object Property Recognition

From Section IV-E, architecture IV got the best accuracy of object property recognition. To validate the recognition performance of the proposed CNNs, other CNN-based neural networks and machine learning models were used for comparison. Since each network has a different architecture, training epochs where each network converged was different. Resnet with 18 layers (trained for 800 epochs) and 34 layers (trained for 900 epochs), MobileNetV2 (trained for 900 epochs), ShuffleNetV2 (trained for 800 epochs), and MnasNet (trained for 750 epochs) as CNN-based models (provided by PyTorch) and SVM (provided by scikit-learn) as a machine learning model were prepared. The "Hand Map Layer with Zero Padding" was constructed as an input layer with a size of 18 rows $\times$ 16 columns (same as architecture I) to be input to the deep learning models. Note that the deep learning models were chosen, which could process the tactile input with a size of 18 $\times$ 16, which is relatively small-sized input compared to visual inputs from a camera with a larger size, such as 224 $\times$ 224. Moreover, the convolution layers soon before FC layers of the deep learning models were used as a feature extractor and a first FC, and output layers were prepared to be applied to a new domain referring [22]. During the training, the layers of the deep learning models were not updated, but only the first FC and output layers were, so that the effect of convolution layers of the deep learning models were validated in the same way as the transfer learning in Section IV-D, all the models were trained with the same training setting as one used in Section III-B-2 except the training epoch. The accuracy was calculated as a mean of all nine outputs of the networks. The recognition trials were conducted five times for each network. Fig. 9(b) shows the mean accuracies and their variances of the five trials. In the proposed CNN, architecture IV could achieve the best recognition rate. Especially, the deep learning models that are larger than architecture IV and do not have combined convolution layers following tactile sensor positions on the hand produced very low recognition rates.

F. Analysis of Weights in CNNs

Even though transfer learning is useful for problems with insufficient training data in general [42], it is not clear why the CNNs could achieve better recognition rates for object property recognition, as shown in Fig. 9(a). The difference between WT and NT models is how to prepare the weights in a neural network for a new task, i.e., the weights from a pretrained model trained in the other task or the weights generated from a random initialization method. Therefore, we compared the weights in pretrained models from the object recognition task as a WT model and the weights initialized by a normal distribution with a mean of 0.0 and a standard deviation of 0.1 as an NT model. Note that the transferred weights were from only convolution layers (not FC layers) as this study focused on the convolution mechanism for tactile information.

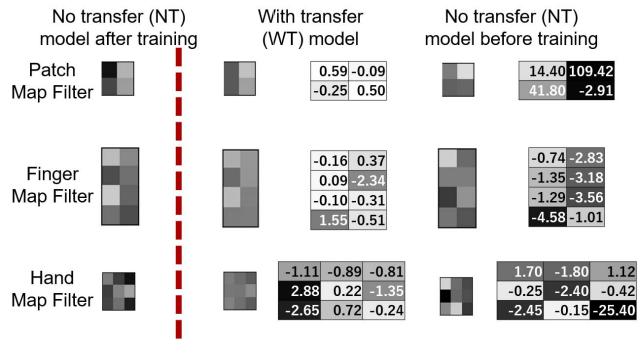


Fig. 10. Weight values (filters) from the last convolution layers. The top row shows one of the filters consisting of the weights from the last convolution layer in architecture VI. The middle row shows one of the filters consisting of the weights from the last convolution layer in architecture VII. The bottom row shows one of the filters consisting of the weights from the last convolution layer in architecture IV. The errors between weights values from the NT model after training and the transferred weights' values from the WT model are shown in the center. Also, the errors between the weights from the NT model after training and the NT model before training (the weights were randomly initialized by a normal distribution with a mean of 0.0 and a standard deviation of 0.1) are shown on the right-hand side. Most of the errors between the NT model after training and the transferred weights of the WT model are less than 1.00, but the errors between the NT model after training and the NT model before training are various and huge.

In Fig. 10, from the top row, one of the filters (weights) in the last convolution layer of architectures VI (2 $\times$ 2 filter), VII (2 $\times$ 4 filter), and IV (3 $\times$ 3 filter) as ones of examples are shown in a gray scale. On the left-hand side, the weights in the NT model after training is shown as target weights and the weights are supposed to be the optimized (trained and converged) ones for object property recognition. In the middle of Fig. 10, the weights in the WT model, which are the transferred weights of convolution layers from a model trained for object recognition, are shown. On the right-hand side, the weights in the NT model before training are shown as a baseline for this comparison study. The values shown in Fig. 10 are the errors of the weights between the NT model after training and the WT model in the center, and the NT model after training and the NT model before training on the right-hand side. The weights in the WT model are similar to the weights in the NT model after training with the dataset of object property recognition as most of the errors between weights of the NT model after training and weights of the WT model are under 1.00. This shows that the weights in the WT model were already optimized enough that the WT model required less size of training dataset and training epochs for object property recognition. Then, the WT model focused on updating only weights in the FC layers was enough to achieve high recognition rates (technically, the weights in the convolution layers were fixed and were not updated). On the other hand, as most of the errors between weights of the NT model after training and weights of the NT model before training are over 1.00 and some of the errors are even more than 100, the NT model needs to update weights in convolution

	Heaviness Low	Heaviness Medium	Heaviness High	Softness Low	Softness Medium	Softness High	Slipperiness Low	Slipperiness Medium	Slipperiness High
Architecture I	0.993	0.962	0.500	0.990	0.991	0.969	0.985	0.987	0.987
Architecture II	0.993	0.979	0.500	0.988	0.988	0.961	0.984	0.987	0.983
Architecture III	0.996	0.993	0.987	0.990	0.970	0.500	0.995	0.991	0.986
Architecture IV	0.995	0.979	0.994	0.995	0.993	0.995	0.992	0.995	0.996
Architecture VI	0.997	0.992	0.972	0.994	0.500	0.500	0.995	0.987	0.987
Architecture VII	0.995	0.973	0.987	0.994	0.977	0.500	0.989	0.990	0.992

Fig. 11. AUC of architectures I–IV, VI, and VII. AUC is an indicator of how much the models are capable of distinguishing between classes and its value, which is an area under the ROC curve. The AUC of the label "Heaviness High" is very low from architecture I. The AUC of the label "Softness High" is very low from architecture VI. This also implies that the fusion combination of CNN architectures affects the accuracy of each physical property of objects.

layers and the weights in FC layers. This comparison result can be a reason why the recognition rates of the WT models were better than those of the NT models.

G. Object Property and CNN Architectures

By using the WT models, recognition performance for each object property was investigated. Since the binary-cross entropy was used as the cost function, recognition results would change by cutoff values that decide an output of CNNs as 0 or 1. In this case, the receiver operating characteristic (ROC) and the area under the curve (AUC) were calculated. The ROC curve is depicted in a map with the true positive rate (TPR) axis and the false positive rate (FPR) axis, which are defined as

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$
$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}} \tag{7}$$

where true positive (TP) is that the actual class and prediction are positive (correct answer), true negative (TN) is that the actual class and the prediction are negative (correct answer), false positive (FP) is that the actual class is negative but the prediction is positive (incorrect answer), and false negative (FN) is that the actual class is positive but the prediction is negative (incorrect answer).

The AUC is an area under the ROC curve and is defined as

$$\mathrm{AUC} = \int_0^1 \mathrm{TPR}(\mathrm{FPR})d\mathrm{FPR}$$
$$= \int_0^1 \mathrm{TPR}\big(\mathrm{FPR}^{-1}(x)\big)dx \tag{8}$$

where x is a continuous random variable. Fig. 11 shows AUCs for object properties from each CNN. Architecture IV achieved a high value of AUC for each physical property. Interestingly, there are some properties that each CNN model is good at recognizing. The AUC of "Heaviness High" in architectures I–III shows a very low value of 0.500, which is theoretically the same value as what a predictor randomly outputs. There is a reason why 0.500 appears for some property labels. First, CNNs output 0 for the labels at all grasping states. Although a cutoff value, which is a threshold to decide whether the CNNs recognize an object property or not, varies from 0 to under 1, the output from CNNs is always classified as 0. Only when the cutoff value is set as 0, the output from CNNs are always classified as 1. Therefore, when the cutoff value is set to 0, TPR and FPR derived from (7) are 1 (i.e., all outputs from the CNNs are positive, and thus, TN and FN are 0); otherwise, they are 0 (i.e., all outputs from the CNNs are negative, and thus, TP and FP are 0). From this point, an ROC curve can be built only by coordinates TPR and FPR = (0, 0) and (1, 1). Finally, an AUC derived from (8) would be 0.500. On the other hand, the AUC of "Softness High" in architectures VI and VII shows a very low value. The difference is whether each model has the "Hand Map Layer." Furthermore, architectures II and III have similar network architecture, while each model has a low AUC value for different properties. Architecture II has a lower number of the "Patch Map Layers" and a low AUC value for "Heaviness High." Architecture III has a lower number of the "Hand Map Layers" and a low AUC value for "Softness High." Therefore, we deduce that the fusion of convolution layers affects not only the recognition of objects but also the physical property of objects.

H. Object Property and Tactile Information

Finally, tactile information was analyzed for a better understanding of CNN architectures for object property recognition. Specifically, the architectures that have the "Patch Map Layer" showed a low AUC value for "Softness High," while the other architectures that have "Hand Map Layer" showed a low AUC value for "Heaviness High." Grad-CAM++ showed how the architectures that have a "Patch Map Layer" are good at recognizing simpler contact patterns, while the other architectures that have the "Hand Map Layer" are good at recognizing complicated contact patterns.

From this point, how each object property changes contact patterns was analyzed. In Fig. 12, each tactile trajectory represents an average on each tactile sensor patch for simpler visualization of the tactile information. This information is taken from the third to fourth grasping postures in Fig. 3(d) as the last 70 timesteps of the motion. The red line is on a digital value of 2400 where noises and responses to the grasped object are clearly separated. The tactile trajectories over the digital value are regarded as where the hand touches the object firmly. Fig. 12(a) shows that five sensor patches have tactile value. We deduce that the soft object deformed and followed a grasping posture of the hand, and thus, the soft object affected relatively many tactile sensor patches. Also, tactile trajectories are dynamic due to the softness of the object, which means that contact patterns are complicated. Fig. 12(b) shows that a smaller number of tactile sensor patches (three patches) have tactile values over 2400 to the contrary. Also, tactile trajectories are relatively flat due to the stiffness of the

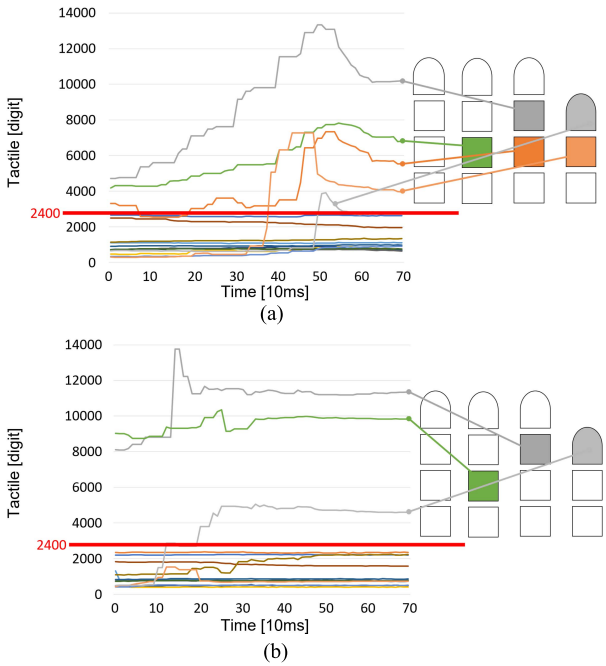


Fig. 12. Tactile information during grasping: (a) kitchen paper with "Softness High," "Slipperiness Low," and "Heaviness High" labels and (b) spray bottle with "Softness Low," "Slipperiness High," and "Heaviness High" labels. Those objects are chosen to compare tactile measurements varied by their softness. Despite a difference in the slipperiness labels of the objects, they are not taken into consideration for this comparison as AUCs of CNN architectures are barely different. They have the same heaviness label so that tactile measurements largely vary. The last 70 timesteps of tactile information during in-hand manipulation are shown. (a) Tactile information with kitchen paper (soft object). (b) Tactile information with a spray bottle (stiff object).

object, which means that contact patterns are not complicated. We deduce that the heavy and stiff objects are held at a small number of phalanges because it does not deform. These physical properties of objects change the recognition rates of an object property. This result revealed that each architecture has the robustness to process contact patterns that depend on the physical properties of objects.

V. Conclusion

This study investigated how the MS-CNN architecture affects tactile-based multifingered hand tasks with distributed three-axis tactile sensors. Object recognition and object property recognition were targeted to evaluate the CNN. The best object recognition rates over 95% were achieved in the experiments by initially separating and subsequently combining convolution layers following the robot's configuration, especially when making the patch, finger, and hand maps (architecture IV). Moreover, we clarified that the CNNs can make better results by synergy with another generally useful training method, i.e., transfer learning, which achieved prominent object property recognition rates up to 98% with the CNNs. Since this recognition was achieved at a single touch of the multifingered hand, the CNN could also be applied to in-hand manipulation and grasp stability tasks in which quick processing is required.

Most importantly, this study revealed an interesting finding when convolution layers are not fused (architecture VI); it has better recognition rates with simpler shaped objects for object recognition and with heaviness labels for object property recognition. On the other hand, when the convolution layers are fused and built in the "Hand Map Layer" (e.g., architecture IV), it has better recognition rates with more complicated shaped objects for object recognition and with softness labels for object property recognition. Thus, the CNN architecture can be customized depending on tactile measurements or tasks (i.e., simple or complicated touch), reflecting outer (e.g., size and shape), and inner (e.g., softness, slipperiness, and heaviness) properties of objects. This finding is investigated by a visualized localization skill of the Grad-CAM++ and thoroughgoing analyses of recognition results for each object and object property using not only a variety of proposed CNN architectures and also CNN-based neural network and machine learning models. Moreover, transferred weights and tactile measurements are analyzed to investigate the factors behind successful recognition. This kind of approach to geometrical features of grasped objects from tactile sensors can be a key for further understanding of tactile sensing as a tactile system in a human's skin also recognizes such object geometries [43].

Overall, considering robotic configuration represented by distributed tactile sensors with different scales (patch, finger, and hand mappings) can be a useful approach to achieving robotic tactile tasks, including simple and complicated contact states, and the approach can be used with other useful methods (i.e., transfer learning that is specifically important for tasks with many tactile sensors as they can be worn out) for achieving better results.

Nowadays, some of the differently shaped robots have distributed tactile sensors on their surface, for example, humanoid or disaster robots with distributed tactile sensors [44], [45]. The proposed concept, i.e., building network architectures following robot configurations, could be a suggestion for how to process the tactile information with CNNs for such robots. Also, the proposed concept can be applied to infant robots to identify their body via distributed tactile sensors for further understanding of cognitive science. Not only tactile information but also thermal information following robot configuration could be another application for our proposed methods and used for thermal imaging tasks [46], [47], [48]. As the proposed networks achieved high accuracies in recognition tasks, real-time recognition during multifingered manipulation is the next step for dexterous manipulation [49] using the graph convolutional network [50] inspired by the results of the MS-CNNs with morphology-related convolution. Moreover, transfer learning can be applied to more different domains, such as in-hand manipulation and measuring grasping stability.

Acknowledgment

The authors would like to thank Dr. S. Somlor and Dr. T. Tomo Pradhono for their technical support. They would also like to thank the Editor-in-Chief, the Associate Editor, and all the anonymous reviewers for their time and helpful comments.

References

[1] A. Yamaguchi and C. G. Atkeson, "Recent progress in tactile sensing and sensors for robotic manipulation: Can we turn tactile sensing into vision?" *Adv. Robot.*, vol. 33, no. 14, pp. 661–673, 2019, doi: 10.1080/01691864.2019.1632222.

[2] M. A. Lee et al., "Making sense of vision and touch: Self-supervised learning of multimodal representations for contact-rich tasks," in *Proc. Int. Conf. Robot. Automat. (ICRA)*, May 2019, pp. 8943–8950.

[3] Y. Gao, L. A. Hendricks, K. J. Kuchenbecker, and T. Darrell, "Deep learning for tactile understanding from visual and haptic data," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2016, pp. 536–543.

[4] I. Akkaya et al., "Solving Rubik's cube with a robot hand," 2019, *arXiv:1910.07113*.

[5] A. Billard and D. Kragic, "Trends and challenges in robot manipulation," *Science*, vol. 364, no. 6446, Jun. 2019. [Online]. Available: https://science.sciencemag.org/content/364/6446/eaat8414

[6] Y. Chebotar, K. Hausman, Z. Su, G. S. Sukhatme, and S. Schaal, "Self-supervised regrasping using spatio-temporal tactile features and reinforcement learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 1960–1966.

[7] H. Yousef, M. Boukallel, and K. Althoefer, "Tactile sensing for dexterous in-hand manipulation in robotics-A review," *Sens. Actuators A, Phys.*, vol. 167, no. 2, pp. 171–187, 2011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0924424711001105

[8] M. Andrés and R. Suárez, "Manipulation of unknown objects to improve the grasp quality using tactile information," in *Proc. SENSORS*, vol. 18, 2018, pp. 1628–1635.

[9] T. Narita, S. Nagakari, W. Conus, T. Tsuboi, and K. Nagasaka, "Theoretical derivation and realization of adaptive grasping based on rotational incipient slip detection," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 531–537.

[10] B. Sundaralingam and T. Hermans, "In-hand object-dynamics inference using tactile fingertips," *IEEE Trans. Robot.*, vol. 37, no. 4, pp. 1115–1126, Aug. 2021, doi: 10.1109/TRO.2020.3043675.

[11] Z. Abderrahmane, G. Ganesh, A. Crosnier, and A. Cherubini, "Haptic zero-shot learning: Recognition of objects never touched before," *Robot. Auto. Syst.*, vol. 105, pp. 11–25, Jul. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889017307492

[12] A. J. Spiers, M. V. Liarokapis, B. Calli, and A. M. Dollar, "Single-grasp object classification and feature extraction with simple robot hands and tactile sensors," *IEEE Trans. Haptics*, vol. 9, no. 2, pp. 207–220, Apr./Jun. 2016.

[13] F. Veiga, B. Edin, and J. Peters, "Grip stabilization through independent finger tactile feedback control," *Sensors*, vol. 20, no. 6, p. 1748, Mar. 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/6/1748

[14] T. Matsubara and K. Shibata, "Active tactile exploration with uncertainty and travel cost for fast shape estimation of unknown objects," *Robot. Auto. Syst.*, vol. 91, pp. 314–326, May 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S092188901630522X

[15] P. Falco, S. Lu, A. Cirillo, C. Natale, S. Pirozzi, and D. Lee, "Cross-modal visuo-tactile object recognition using robotic active exploration," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 5273–5280.

[16] J. Kwiatkowski, D. Cockburn, and V. Duchaine, "Grasp stability assessment through the fusion of proprioception and tactile signals using convolutional neural networks," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2017, pp. 286–292.

[17] A. Vasquez, Z. Kappassov, and V. Perdereau, "In-hand object shape identification using invariant proprioceptive signatures," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 965–970.

[18] H. Liu, D. Guo, and F. Sun, "Object recognition using tactile measurements: Kernel sparse coding methods," *IEEE Trans. Instrum. Meas.*, vol. 65, no. 3, pp. 656–665, Mar. 2016.

[19] S. S. Baishya and B. Bauml, "Robust material classification with a tactile skin using deep learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 8–15.

[20] V. Kumar, A. Gupta, E. Todorov, and S. Levine, "Learning dexterous manipulation policies from experience and imitation," 2016, *arXiv:1611.05095*.

[21] M. Meier, F. Patzelt, R. Haschke, and H. J. Ritter, "Tactile convolutional networks for online slip and rotation detection," in *Artificial Neural Networks and Machine Learning–$_{ICANN}$*, vol. 9887. 2016, pp. 12–19.

[22] W. Yuan, Y. Mo, S. Wang, and E. Adelson, "Active clothing material perception using tactile sensing and deep learning," 2017, *arXiv:1711.00574*.

[23] M. Lambeta et al., "DIGIT: A novel design for a low-cost compact high-resolution tactile sensor with application to in-hand manipulation," *IEEE Robot. Automat. Lett.*, vol. 5, no. 3, pp. 3838–3845, Jul. 2020.

[24] S. Funabashi et al., "Stable in-grasp manipulation with a low-cost robot hand by using 3-axis tactile sensors with a CNN," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Jan. 2020, pp. 9166–9173.

[25] S. Funabashi et al., "Object recognition through active sensing using a multi-fingered robot hand with 3D tactile sensors," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 2589–2595.

[26] B. Romero, F. Veiga, and E. Adelson, "Soft, round, high resolution tactile fingertip sensors for dexterous robotic manipulation," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 4796–4802.

[27] Q. Li, O. Kroemer, Z. Su, F. F. Veiga, M. Kaboli, and H. J. Ritter, "A review of tactile information: Perception and action through touch," *IEEE Trans. Robot.*, vol. 36, no. 6, pp. 1619–1634, Dec. 2020.

[28] T. P. Tomo et al., "A modular, distributed, soft, 3-axis sensor system for robot hands," in *Proc. IEEE-RAS 16th Int. Conf. Humanoid Robots (Humanoids)*, Nov. 2016, pp. 454–460.

[29] T. P. Tomo et al., "Covering a robot fingertip with uSkin: A soft electronic skin with distributed 3-axis force sensitive elements for robot hands," *IEEE Robot. Automat. Lett.*, vol. 3, no. 1, pp. 124–131, Jan. 2018.

[30] 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 *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 57–63.

[31] Z. Peng, Z. Li, J. Zhang, Y. Li, G.-J. Qi, and J. Tang, "Few-shot image recognition with knowledge transfer," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 441–449.

[32] C. Sferrazza and R. D'Andrea, "Transfer learning for vision-based tactile sensing," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 7961–7967.

[33] J. M. Gandarias, A. J. Garcia-Cerezo, and J. M. Gomez-de-Gabriel, "CNNbased- methods for object recognition with high-resolution tactile sensors," *IEEE Sensors J.*, vol. 19, no. 16, pp. 6872–6882, 2019.

[34] B. Bauml and A. Tulbure, "Deep n-Shot transfer learning for tactile material classification with a flexible pressure-sensitive skin," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 4262–4268.

[35] P. Falco, S. Lu, C. Natale, S. Pirozzi, and D. Lee, "A transfer learning approach to cross-modal object recognition: From visual observation to robotic haptic exploration," *IEEE Trans. Robot.*, vol. 35, no. 4, pp. 987–998, Aug. 2019.

[36] H. Lee, H. Park, G. Serhat, H. Sun, and K. J. Kuchenbecker, "Calibrating a soft ERT-based tactile sensor with a multiphysics model and sim-to-real transfer learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 1632–1638.

[37] Z. Ding, N. F. Lepora, and E. Johns, "Sim-to-real transfer for optical tactile sensing," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 1639–1645.

[38] A. Chattopadhay, A. Sarkar, P. Howlader, and V. N. Balasubramanian, "Grad-CAM++: Generalized gradient-based visual explanations for deep convolutional networks," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2018, pp. 839–847.

[39] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollár, "The YCB object and model set: Towards common benchmarks for manipulation research," in *Proc. Int. Conf. Adv. Robot. (ICAR)*, Jul. 2015, pp. 510–517.

[40] [XXXX].

[41] H. Zhou, Z. Li, C. Ning, and J. Tang, "CAD: Scale invariant framework for real-time object detection," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops (ICCVW)*, Oct. 2017, pp. 760–768.

[42] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A survey on deep transfer learning," *Artificial Neural Networks and Machine Learning–(ICANN)* (Lecture Notes in Computer Science), vol. 11141. 2018, pp. 270–279.

[43] J. A. Pruszynski and R. S. Johansson, "Edge-orientation processing in first-order tactile neurons," *Nature Neurosci.*, vol. 17, no. 10, pp. 1404–1409, 2014, doi: 10.1038/nn.3804.

[44] G. Cheng, E. Dean-Leon, F. Bergner, J. R. G. Olvera, Q. Leboutet, and P. Mittendorfer, "A comprehensive realization of robot skin: Sensors, sensing, control, and applications," *Proc. IEEE*, vol. 107, no. 10, pp. 2034–2051, Oct. 2019.

[45] D. Inoue, M. Konyo, and S. Tadokoro, "Distributed tactile sensors for tracked robots," in *Proc. 5th IEEE Conf. Sensors*, Oct. 2006, pp. 1309–1312.

[46] A. Glowacz, "Thermographic fault diagnosis of ventilation in BLDC motors," *Sensors*, vol. 21, no. 21, p. 7245, Oct. 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/21/7245

[47] A. Glowacz, "Ventilation diagnosis of angle grinder using thermal imaging," *Sensors*, vol. 21, no. 8, p. 2853, Apr. 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/8/2853

[48] A. Glowacz, "Fault diagnosis of electric impact drills using thermal imaging," *Measurement*, vol. 171, Feb. 2021, Art. no. 108815. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0263224120313099

[49] S. Funabashi et al., "Multi-fingered in-hand manipulation with various object properties using graph convolutional networks and distributed tactile sensors," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 2102–2109, Apr. 2022.

[50] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," Tech. Rep., 2017.

Satoshi Funabashi (Member, IEEE) received the B.E., M.E., and Ph.D. degrees from Waseda University, Tokyo, Japan, in 2015, 2017, and 2021, respectively.

Since 2021, he has been a Junior Researcher with the Institute for AI and Robotics, Future Robotics Organization, Waseda University. From 2018 to 2019, he was a Visiting Student with the Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, USA. His current research interests include multifingered hands, tactile perception, and dexterous manipulation.

Dr. Funabashi received the Research Fellowship for Young Scientists DC1 from the Japan Society for the Promotion of Science (JSPS) and the Strategic Basic Research Programs ACT-I from the Japan Science and Technology Agency (JST) in 2017 and 2018, respectively.

Gang Yan (Student Member, IEEE) received the B.E. degree from Northeastern University, Shenyang, China, in 2016, and the M.E. degree from Waseda University, Tokyo, Japan, in 2020, where he is currently pursuing the Ph.D. degree with the Department of Modern Mechanical Engineering.

In 2022, he was a Visiting Student with the Robotouch Laboratory, The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA. He has been working on research about grasping stability estimation and slip detection relying on either tactile or multimodal tactile-visual sensing using a data-driven approach. His research results have been published at the International Conference on Robotics and Automation and the IEEE ROBOTICS AND AUTOMATION LETTERS. His current research interests include tactile perception, tactile sensor simulation, robotic manipulation, and human–robot interaction.

Fei Hongyi (Student Member, IEEE) received the B.E. degree from China Jiliang University, Zhejiang, China, and the M.E. degree from Waseda University, Tokyo, Japan, in 2022.

Since 2020, he has been a student with the Department of Modern Mechanical Engineering, Waseda University. Since 2020, he has been working on research about object recognition and dexterous in-hand manipulation with multifingered hands and tactile sensors. He has developed a controlling system of anthropomorphic hands and machine learning methods. He worked on the project for generating in-hand manipulation published for IEEE ROBOTICS AND AUTOMATION LETTERS (RA-L) in 2022. His research interests include machine learning, tactile sensing, and dexterous manipulation.

Alexander Schmitz (Member, IEEE) received the master's degree (Hons.) from the University of Vienna, Vienna, Austria, in 2007, and the Ph.D. degree from The University of Sheffield, Sheffield, U.K., in 2011.

He performed his Ph.D. research as part of a joint location program with the Italian Institute of Technology, Genoa, Italy. He is currently an Associate Professor with the Department of Modern Mechanical Engineering, Waseda University, Tokyo, Japan. He has published 14 journal articles, one book chapter, and 34 international conference papers. Furthermore, he has applied for three national and five international patents. His research interests include tactile sensing, intrinsically safe actuation, human symbiotic robotics, and robotic object handling.

Dr. Schmitz received a grant of 117 million JPY from the JST START Program (Program for Creating STart-ups from Advanced Research and Technology) in 2016.

Lorenzo Jamone (Member, IEEE) received the M.S. degree (Hons.) in computer engineering from the University of Genoa, Genoa, Italy, in 2006, and the Ph.D. degree in humanoid technologies from the Italian Institute of Technology, Genoa, in 2010.

He was an Associate Researcher with the Takanishi Laboratory, Waseda University, Tokyo, Japan, from 2010 to 2012, and Vislab, Instituto Superior Técnico, Lisbon, Portugal, from 2012 to 2016. He is currently a Senior Lecturer in robotics with the School of Electronic Engineering and Computer Science, Queen Mary University of London, London, U.K. He is part of Advanced Robotics at Queen Mary (ARQ), London. He is the Founder and the Director of the CRISP Group: Cognitive Robotics and Intelligent Systems for the People. He has over 100 publications with an H-index of 26. His current research interests include cognitive robotics, robotic manipulation, and tactile sensing.

Dr. Jamone has been a Turing Fellow since 2018.

Tetsuya Ogata (Member, IEEE) received the B.S., M.S., and D.E. degrees in mechanical engineering from Waseda University, Tokyo, Japan, in 1993, 1995, and 2000, respectively.

He was a Research Associate with Waseda University from 1999 to 2001. From 2001 to 2003, he was a Research Scientist with the RIKEN Brain Science Institute, Saitama, Japan. From 2003 to 2012, he was an Associate Professor with the Graduate School of Informatics, Kyoto University, Kyoto, Japan. From 2009 to 2015, he was a JST (Japan Science and Technology Agency) PREST Researcher. Since 2012, he has been a Professor with the Faculty of Science and Engineering, Waseda University. Since 2017, he has been a Joint-Appointed Fellow with the Artificial Intelligence Research Center, National Institute of Advanced Industrial Science and Technology, Tokyo.

Shigeki Sugano (Fellow, IEEE) received the B.S., M.S., and D.E. degrees in mechanical engineering from Waseda University, Tokyo, Japan, in 1981, 1983, and 1989, respectively.

Since 1986, he has been a Faculty Member of the Department of Mechanical Engineering, Waseda University, where he is currently a Professor. Since 2014, he has been the Dean of the School/Graduate School of Creative Science and Engineering, Waseda University. Since 2020, he has been the Senior Dean of the Faculty of Science and Engineering, Waseda University.

Dr. Sugano is a fellow of four academic societies: IEEE, the Japan Society of Mechanical Engineers, the Society of Instrument and Control Engineers, and the Robotics Society of Japan. He has served as the General Chair of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics in 2003 and the IEEE/RSJ International Conference on Intelligent Robots and Systems in 2013. From 2001 to 2010, he has served as the President of the Japan Association for Automation Advancement. In 2017, he has served as the President of SICE.