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

Deep Tactile Experience: Estimating Tactile Sensor Output from Depth Sensor Data

Karankumar Patel, Soshi Iba, Nawid Jamali

***Abstract*— Tactile sensing is inherently contact based. To use tactile data, robots need to make contact with the surface of an object. This is inefficient in applications where an agent needs to make a decision between multiple alternatives that depend the physical properties of the contact location. We propose a method to get tactile data in a non-invasive manner. The proposed method estimates the output of a tactile sensor from the depth data of the surface of the object based on past experiences. An experience dataset is built by allowing the robot to interact with various objects, collecting tactile data and the corresponding object surface depth data. We use the experience dataset to train a neural network to estimate the tactile output from depth data alone. We use GelSight tactile sensors, an image-based sensor, to generate images that capture detailed surface features at the contact location. We train a network with a dataset containing 578 tactile-image to depth-map correspondences. Given a depth-map of the surface of an object, the network outputs an estimate of the response of the tactile sensor, should it make a contact with the object. We evaluate the method with structural similarity index matrix (SSIM), a similarity metric between two images commonly used in image processing community. We present experimental results that show the proposed method outperforms a baseline that uses random images with statistical significance getting an SSIM score of 0.84 ± 0.0056 and 0.80 ± 0.0036, respectively.**

I. Introduction

Perception of object properties such as texture, roughness and slipperiness require an agent to make contact with the environment. Humans use their tactile sensors to perceive finger-object contact properties [1]. With sufficient experience, however, humans are also able to perceive such properties from visual perception alone [2], [3], albeit, a rough qualitative estimate of a certain physical property [4], for example, the object surface appears to be smooth and slippery. Estimates of contact properties of an object such as roughness and slipperiness can help an agent, in advance, to decide how to interact with the object without making contact with the object.

The proposed method can be applied to any problem where knowledge of physical properties of the point of contact is needed when making a decision, for example, a legged robot can use the method to plan where to plant its feet to avoid a slippery surface. Another application is in grasp planning. To successfully grasp an object, often, robots need to make multiple contacts with the object to determine a suitable contact location for a stable grasp. Grasping objects that involve multiple contact attempts to generate the tactile data is time consuming and inefficient for the robot. The proposed method can be used during planning to make better

The authors are with Honda Research Institute USA, Inc. {karankumar_patel, siba, njamali}@honda-ri.com

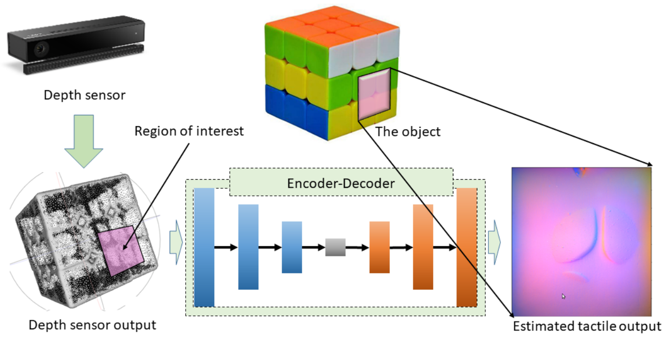

Fig. 1. Illustrates a schematic view of the proposed system. The system consists of a depth sensor to sense the structure of the surface of the object. To estimate the tactile sensor output, a region-of-interest, equal to the size of the tactile sensor is selected as the input of an encoder-decoder based neural network. The neural network estimates the output the tactile sensors would have produced if a contact with the object was established.

informed grasps. It can increase the efficiency of dexterous robots and help them gracefully manipulate objects in their environment. Other applications include haptic rendering where it can be used by blind persons to perceive their surroundings [5].

In this paper we propose a novel idea that uses depth data of the surface of an object to estimate the tactile output, which, can be used by an agent to estimate physical properties of the object. As illustrated in Fig. 1, the system consists of a depth sensor to sense the structure of the surface of the object of interest. In order to estimate the tactile output, a region-of-interest[1], equal to the size of the tactile sensor is selected. The data from the depth sensor in the region-of-interest is used as the input of an encoder-decoder neural network. The neural network estimates the output the tactile sensor would have produced if a contact with the object was established. In other words, the neural network imagines how the tactile sensor will be excited without making any contact with the object.

II. Background

Tactile perception consists of two components: the artificial tactile sensors and the artificial intelligence to interpret the tactile sensor data. Many physical phenomena have been used to create artificial tactile sensors. These include capacitive [7], peizoresistive [8], optical [9]–[11],

[1]The region-of-interest can be given by a planner, for example, a grasp synthesizer [6] can propose a number of possible grasp locations. It can use tactile sensor estimates to select a better grasp location based on the object's surface features.

and magnetic [12]. Tactile sensors that have been successfully commercialized include BioTac [13] and GelSight [11] sensors. BioTacs are inspired by human mechanoreceptors that are capable of sensing vibrations and contact forces. GelSight sensors, on the other hand, are image based. It uses an elastomer and a camera to encode the shape of a contact surface in the form of an image.

Early work in tactile perception demonstrated that, in principle, tactile sensor signals can be used to differentiate materials [14], [15]. Machine learning has been used to fuse vision and tactile sensory information to improve the accuracy of 3-D object recognition [16], differentiate between objects found commonly in a shopping bag [17], detect slip [18], [19], and classify materials [20], [21].

Robotic manipulation is a common task that can benefit from tactile sensing. In order to manipulate an object, the robot needs to plan a grasp configuration that satisfies a set of criteria relevant for the grasping task [6], this is referred to as grasp synthesis. Analytical and data-driven approaches have been used to solve the problem of grasp synthesis. Data driven approaches sample grasp candidates and use a metric to rank them [6]. Mahler et al. [22] propose a network that learns policies to successfully grasp objects in a heap, for a given set of grippers, by training in simulation on synthetic depth data. The authors show that using domain randomization it is possible to transfer the learned policy to a robot. However, the method does not take into account the effect of the contact interaction between the manipulator and the object surface, which may be critical for a successful grasp.

Recently, Calandra et al. [23] showed that including tactile sensors as additional information can improve over vision only based grasping methods. However, their method requires a contact with the object to generate the tactile sensor data so that the model can predict the stability of the chosen contact location, often, leading to a re-grasp action. The method proposed in this paper, by estimating the tactile output, can eliminate the gripper-object contact required to acquire tactile sensor data. Thereby, making it possible to account for the tactile sensor output without making a contact with the object surface and select a better informed grasp location.

Hogan et al. [24] propose a tactile-based grasp quality metric. They also propose a re-grasping method that uses the tactile sensor output from the initial contact and use rigid body transformation to simulate tactile sensor output and select a re-grasp location that maximizes the grasp quality metric. The rigid body transformation is not based on the sensed structure of the object, it is a translation of the initial contact tactile data, which may not correspond to the object shape.

A closely related work is presented by Takahashi and Tan [25]. The authors propose an encoder-decoder network to learn latent features that map RGB images to tactile data. The mapping is learned by training the network with RGB images of an object as the input and tactile sensor signals of an interaction with the object as the output. After training, the authors propose, use of the latent features to correlate object images to object properties. The authors present experimental results in which tactile data is generated by stroking objects with varying textures and rigidity, and recording an RGB image of the object. The authors show that in the latent space objects are arranged according to friction and rigidity.

Similar to Takahashi and Tan [25], Li et al. [26] propose an encoder-decoder network approach to estimate tactile sensor output. The authors use GelSight tactile sensors, which we also use for our experiments. The input to the network is an RGB image of the entire scene, a reference tactile sensor output and two frames before and after the contact event. The authors present results of a study that used Amazon Mechanical Turk. Human subjects were presented with the ground truth tactile videos and the predicted tactile results along with the vision input. The subjects were asked which tactile video corresponds better to the input vision signal. The authors report that participants identified 46% of the predicted videos to correspond to the input signal for objects in the training set. In case of novel objects, they report 38% of predicted videos were identified to correspond to the input signal.

The method presented in this paper is different from Takahashi and Tan [25] in sensing and the way we interpret the sensor data. We focus on estimating tactile sensor output, which is different from Takahashi and Tan [25] who explicitly refrain from inferring the tactile sensor output. Takahashi and Tan [25], and Li et al. [26] use RGB images as input. We use a different sensing modality as the input for our network. We use depth sensor data instead of RGB images. Depth sensor data captures surface structures that directly affect the tactile sensor output.

III. Methodology

We present a method that estimates the output of a tactile sensor from depth sensor data of the surface of an object. The intuition behind the method is that a depth sensor captures essential object-surface features that can be used to estimate the output of a tactile sensor. We postulate that it is possible to learn a neural network model that leverages past experience to estimate the tactile sensor output. As illustrated in Fig. 1, the input to the algorithm is depth sensor data and a region-of-interest. The network outputs an estimate of the tactile sensor output, which, for the tactile sensors used in this paper is an image. Therefore, to train such network we need: a depth-map for the region-of-interest, and the corresponding tactile sensor output. In this section we first give a brief description of the sensors (Section III-A), then explain how we generate the depth-map for a given tactile-object contact (Section III-B). This is followed by a description of the encoder-decoder network used to model the tactile experience (Section III-C).

A. Sensors

We use two Microsoft Kinect-V2 sensors to get depth data on the surface of the object. The tactile data is generated by GelSight [11] tactile sensors. GelSight sensors are optical

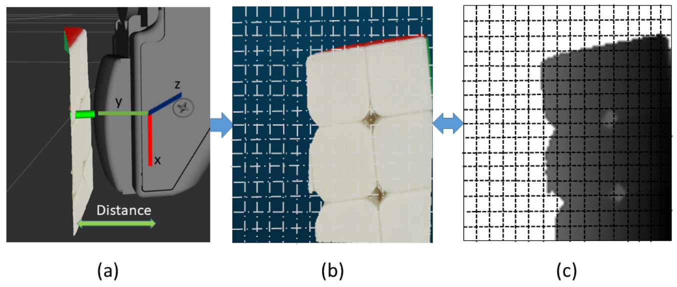

Fig. 2. Illustration of depth-map construction. (a) Object point cloud in the finger's frame of reference. (b) Extracted point cloud of the object surface that is in contact with the tactile sensors. The white lines indicate the bins for depth-map generation. (c) The resulting depth-map after binning. Note: the bins in this example are only for illustration purposes. They do not represent the actual bins used.

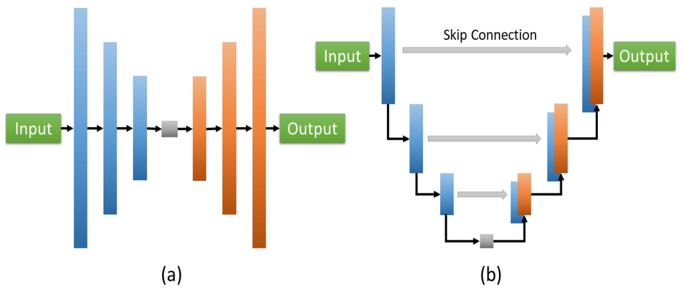

Fig. 3. Illustrations of encoder-decoder architecture. (a): An encoder-decoder network. (b): A U-Net encoder-decoder network with skip connections. Skip connections help to shuttle low level information of an input from encoder to decoder.

sensors, unlike other tactile sensors that sense forces and vibrations, a GelSight sensor consists of an elastomer covered with a reflective coating membrane. When the elastomer makes contact with an object surface, the membrane distorts to take on the shape of the surface of the object. A camera is used to record the image of the membrane, which encodes the shape and features of the contact surface. We use these two sensors to generate data for training the neural network.

B. *Tactile-Object Contact Surface Depth-Map Generation*

We use two Kinect-V2 sensors to get RGB-D data of the environment. Two sensors help to cover a lager area and reduce occlusion. This helps the robot to autonomously sample the object surface. However, two depth sensors are not sufficient to capture the entire object. To ensure that each tactile image has a corresponding complete depth data, we used complete object point clouds provided with the Yale-CMU-Berkeley (YCB) dataset [27]. In this section we describe how we build a surface depth-map for each tactile-object contact. Please refer to the accompanied video for a visual illustration of the process.

1) Object Segmentation: The first step is to segment the object of interest. Any algorithm can be used to segment the object. We used Euclidean clustering algorithm [28] to segment the object of interest. We crop the point cloud to remove other objects, e.g., objects outside the work space of the robot. We also remove planar surfaces. The filtered point could is used as the input of the Euclidean clustering algorithm, which gives us a single object cluster.

2) Registration with Pre-Captured Point Cloud: Once we have the object point cloud, then we register a pre-captured point cloud of the object to get a complete point cloud for the object. As described earlier, we use the complete object point clouds that come with the YCB dataset. We used sample consensus alignment algorithm [29] to get an initial coarse alignment. Then, we used the iterative closest point algorithm [30] to register the YCB point cloud with the point cloud of the segmented object.

3) Object-Tactile Contact Surface Depth-Map: Once the registration of the complete point cloud of the object described in the previous step is complete, the robot approaches the object to make contact with the object at a region-of-interest to collect tactile sensor output. We extract a point cloud patch for the region-of-interest using robot's kinematics. We use this point cloud patch to construct the depth-map.

Figure 2 illustrates how a depth-map is generated from the object point cloud. First, the point cloud data is transformed to the tactile sensor's frame of reference with the origin at the center of the tactile sensor. As shown in the Fig. 2 (a), the $x-z$ plane is parallel to the surface of the tactile sensor, the y-axis corresponds to the distance of the object from the surface of the tactile sensor. That is, a point on the surface of the tactile sensor will have a y value of zero, a value of greater than zero means the point is not making contact with the surface of the tactile sensor. Similarly, a point with value less than zero indicates that it has made an indentation in the tactile sensor's elastomer.

The first step is to filter a volume (x, y, z) of point cloud that corresponds to the contact area between the object and the tactile sensor. To construct the depth-map, we divide the $x - z$ plane into $m \times k$ bins, indicated by the white dashed lines in Fig. 2 (b). The value of each bin is set to the minimum y value of points in that bin's $(x\ ,\ z)$ range. Figure 2(C) shows a depth-map after binning the cloud patch into 100×100 bins.

C. *Network Architecture*

We adapt our network architecture from [31]. We use conditional generative adversarial networks (cGAN) [32] to model the tactile sensor output estimator. Generative adversarial networks [33] consist of a generator network, G, and a discriminator network, D. The discriminator tries to learn a loss that can distinguish whether a given input is real or fake. On the other hand, the generator tries to learn how to minimize this loss for fake input. Conditional GANs are a form of GANs where the discriminator network is conditioned with an input. So, it can learn conditional generative models. This makes cGANs suitable for our problem, where we condition on an input depth-map and generate the corresponding tactile image. The objective of GAN can be described as:

$$\mathcal{L}_{cGAN}(G,D) = \mathbb{E}_{x,y}[logD(x,y)] + \mathbb{E}_{x,z}[1 - logD(x, logG(x,z))], \quad (1)$$

where x, y and z are network input, label and random noise respectively. G tries to minimize this objective against an adversarial D that tries to maximize it, i.e.

$$G^* = arg \min_G \max_D L_{cGAN}(G,D). \quad (2)$$

We use L1 loss as described in [31] rather ran L2 loss because it encourages less blurring.

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[||y - G(x,z)||_1]. \quad (3)$$

Hence, our final network objective is:

$$G^* = arg \min_G \max_D \mathcal{L}_{cGAN}(G,D) + \lambda \mathcal{L}_{L1}(G). \quad (4)$$

where λ is the learning rate.

The generator is based on the U-Net [34], an encoder-decoder with skip connections as shown in Figure 3(b). Using the encoder-decoder based network (Figure 3(a)) for a generator is well-known. In this type of network, we progressively downsample our input until a bottleneck layer. Later, we try to upsample the data from a bottleneck layer. So, there is a chance of losing information while going from the high dimensional space (input) to a low dimensional space (bottleneck layer). U-Net includes skip connections in between each layer i and layer $n - i$, where n is the total number of layers. Skip connection concatenates information from layer i to layer $n - i$. This helps to shuttle low level information of an input from encoder to decoder. We use same discriminator as a PatchGAN [35]. It penalizes structure in terms of patches. This helps the network to learn a structural loss. It outputs whether the given image is real or fake by averaging responses for all the patches. All modules of the generator and discriminator use convolution-BatchNorm-ReLu [36].

IV. Experimental Setup

To train and test a model for the proposed tactile sensor estimator we used the experimental setup in Figure 4. It consists of a 7-DoF Sawyer robot by Rethink Robotics. The robot is equipped with a Weiss two-finger parallel jaw gripper. A GelSight [37] sensor is mounted on each finger of the robot. Two Kinect-V2 sensors are used to produce depth data.

We used three objects (Fig. 5) from the YCB Object and Model Set [27] to build the tactile experience dataset. A total of 289 grasps were performed on the objects. With two fingers, resulted in a dataset of 578 (tactile, depth-map) correspondences. The dataset was split into 70%, 15% and 15% for training, validation and testing, respectively.

In the following section we describe the data collection process, then in Section IV-B we explain how we train the neural network.

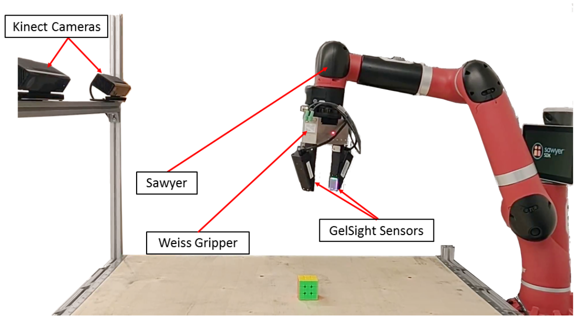

Fig. 4. Experimental Setup. It consists of 7-DoF Sawyer robot. The robot is equipped with a Weiss gripper: two-finger parallel jaw gripper. A GelSight sensor is mounted on each finger of the robot. Two Kinect-V2 sensors are used to produce depth data.

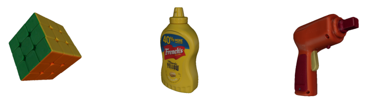

Fig. 5. Objects used for the data collection.

A. Data collection Process

The data collection process is illustrated in Fig. 6. During data collection, the object is rigidly attached to the table. This ensures that the object does not shift when the robot makes contact with it[2]. Then the robot samples the surface of the object at random locations.

Algorithm 1 describes the data collection process. The input to the process is the point cloud from the depth sensors and the object model, that is the complete point cloud from the YCB dataset. The output is a dataset, D, of (tactile-image, depth-map) couples. That is, for each tactile image from a contact with the surface of the object, the process generates the corresponding depth-map.

For each object, first, the object point cloud is segmented (described in Section III-B.1). The next step is to register the object model to the current state of the object (described in Section III-B.2). Once the robot has the registered model, it selects a random location (x, y, z, θ) to sample the object surface. The (x, y, z) is calculated by adding a random value to the centroid of the object point cloud. The orientation, θ_x, θ_y, are fixed such that the palm of the gripper is parallel to the table surface. Only θ_z, of the gripper is changed. The contact force is set to a constant value.

When the robot makes contact with the object, a volume, (x_c, y_c, z_c), of the point cloud at the contact location is extracted for each finger. The dimensions of the contact-volume are determined by the size of the surface of the tactile sensor, (x_t, y_t), and the depth of the surface elastomer (z_t). We use the robot's kinematics to determine the contact location. In practice, we noticed that inaccuracies in the

[2]This is only required during data collection. At run-time the object can be placed freely.

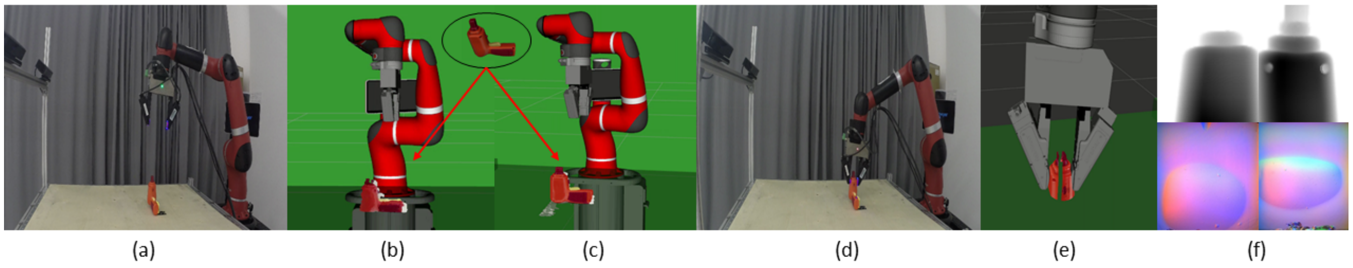

Fig. 6. Data collection overview: (a) System setup (b) Object segmentation (c) Registering complete point cloud model of the object (d) Random grasp (e) Extracted point cloud patches (f) Generated depth-maps (top) and tactile sensors output (bottom).

Algorithm 1: Tactile Experience Data Collection

```
Input: depth data, object model
Output: dataset(tactile-image, depth-map)
 1: for object in selected objects do
 2:     object ← segmentObject(depth)
 3:     registered-model ← registerObject(object, model)
 4:     for grasp in total grasps do
 5:         $(x, y, z, \theta)$ ← randomGrasp(object)
 6:         contact-volume ← extractPatch(registered-model)
 7:         depth-map ← generateDepth(contact-volume)
 8:         add dataset(tactile-image, depth-map)
 9:     end for
10: end for
11: return dataset
```

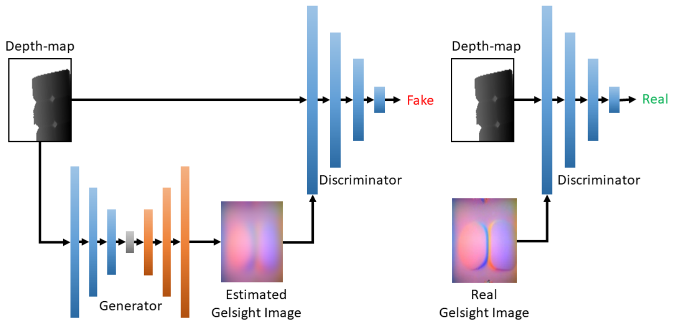

Fig. 7. Training overview. As shown in figure we provide depth-map as an input to our generator and get estimated tactile image as an output. Discriminator also takes in depth-map and tactile image as an input and outputs whether given tactile image is real or not.

calibration of the robot with the external cameras resulted in a systematic shift in the point cloud with respect to the calculated contact position. Since the error is systematic, taking a larger patch than the size of the tactile sensor will give a learning algorithm sufficient information to account for the systematic error.

The point cloud in the contact-volume for each finger is converted into a depth-map using the method described in Section III-B.3. The tactile experience database is then updated with the recorded GelSight images and the corresponding depth-map. The process is repeated for each object.

B. *Training the Network*

As illustrated in Fig. 7, we use the depth-map and the GelSight images to train the conditional generative adversarial network (cGAN) described in Section III-C.

The input to the network is a 100×100 image both for the depth-map and the GelSight image. The model is trained using a 70×70 PatchGAN architecture as the discriminator. We used mini-batch Stochastic Gradient Descent with Adam solver [38]. We used a learning rate of 0.0002 and momentum parameters $\beta_1 = 0.5, \beta_2 = 0.999$.

V. Results

A. *Qualitative Results*

Figure 8 shows the result of estimating a tactile output from depth-maps. In the first column we present a color image of the object to aid in visualizing the surface. The second column is the input to the network, followed by the ground truth tactile image. The last column is the estimate of the network. Visually we can see that the estimated tactile output captures the key features of the object, such as ridges, and surface patterns.

To further analyse the results, we used OpenCV's template matching, which searches an input image for areas that are similar to a given template image. A match is found by sliding the template image across the input image and a similarity metric is calculated. We used the normalized correlation coefficient as the similarity metric. The templates were defined by a human (the authors of this paper) to select a bounding box that captures interesting features (deformed area in GelSight images) of the ground-truth tactile image. In Fig. 8 the blue bounding boxes show the template and the red bounding boxes show the result of template matching. In some cases, for example, in the ninth example it is not easy to see a correspondence between the ground truth and the estimate. However, in the context of template matching, it is easy to see that the estimate captures important features of the tactile image.

B. *Quantitative Results*

Structural similarity matrix (SSIM) [39] is a metric to quantify perceived image quality. It is commonly used in image processing to evaluate the quality of processed images.

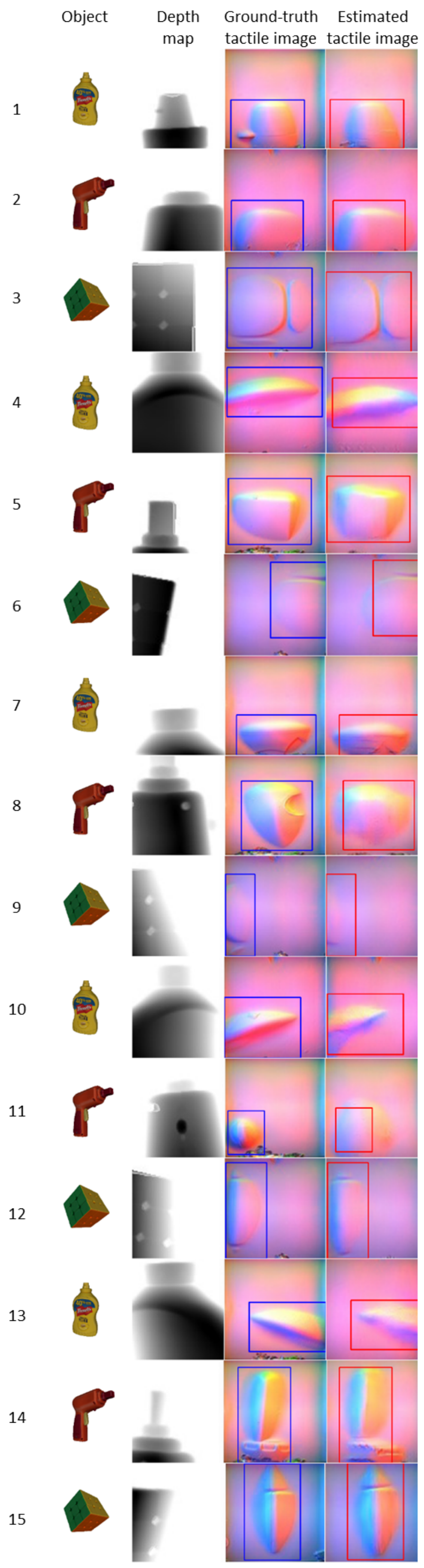

Fig. 8. Qualitative results on test data. First column: object used for data collection. Second column: input to the network – a depth-map. Third column: ground truth tactile image with a selected template (blue bounding box). Fourth column: estimated tactile sensor output with the template matching result (red bounding box).

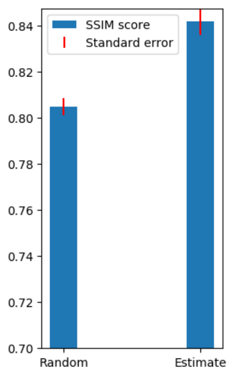

Fig. 9. Quantitative results. Left bar shows average SSIM score for the baseline. Right bar shows average SSIM score for our method – between ground truth images and the estimated tactile images.

We used SSIM to quantify the quality of the estimated tactile images. It takes two inputs: the ground truth image, and the estimated image. A sliding window approach is used to calculate the similarity metric. For each window in both images, SSIM considers luminance, contrast and structural value. SSIM score ranges between 0 to 1. A score of 1 means two images are perfectly structurally similar while score of 0 means that two images have no structural similarity. A study by Flynn et al. [40] shows that humans can not perceive the difference between a distorted image and a real image at the SSIM score approaches 0.95. We calculated SSIM scores between ground truth images and the estimated tactile images. We compare the SSIM scores with a baseline SSIM score.

The baseline score is calculated by taking an average SSIM score between an estimated tactile image and 15 randomly selected ground truth images from the database. That is, for each estimated tactile image, 15 random images were selected and an average SSIM score was calculated. Figure 9 shows the result of this analysis. Using our trained model, the average SSIM score between the estimated image and the ground truth is 0.84 ± 0.0056, compared to the baseline which achieves an average score of 0.80 ± 0.0036. The results suggest that our model outperforms the baseline with statistical significance.

C. Effect of Depth Sensor Resolution

We also studied the effect of depth sensor resolution on the tactile estimate. We reduced the depth sensor resolution of our point clouds by downsampling the object's point cloud. We used the voxelgrid filter from the point cloud library, which downsamples a point cloud by using a centroid of all points present in a voxel as an approximation. Figure 10 shows an example of estimated tactile sensor outputs using depth-maps from different point cloud densities. Table I shows the SSIM scores for the tactile estimates on point clouds with different densities. The results suggest that a reduction in the point cloud density has negligible adverse effect on the tactile estimate.

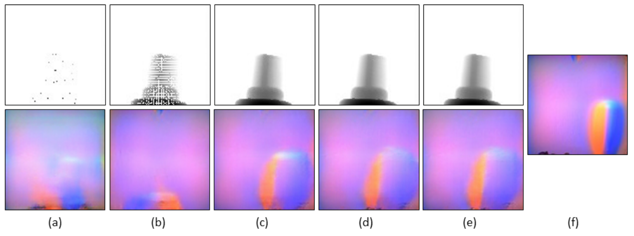

Fig. 10. Figures a–e show in the top row the depth-maps generated from different point cloud densities. Point cloud densities are 1, 10, 20, 40 and 80 points/cm^3 from left to right, respectively. In the bottom row are the corresponding estimates of the GelSight sensor output produced by our method. Figure f shows the ground truth GelSight output.

TABLE I

SSIM SCORES AS A FUNCTION OF POINT CLOUD DENSITY

Cloud density (points/cm^3)	SSIM $\pm$ std error
1	0.8238 $\pm$ 0.0052
10	0.8278 $\pm$ 0.0057
20	0.8478 $\pm$ 0.0056
40	0.8480 $\pm$ 0.0056
80	0.8476 $\pm$ 0.0057

VI. Conclusion and Future work

We presented a method that estimates the output of a tactile sensor from depth sensor data. The novelty of this work lies in the way we use depth sensor data to estimate tactile sensor output. A neural network based on conditional GANs is trained with a dataset containing couples of contact-surface depth-map and the corresponding tactile sensor output. At run-time, given a depth-map of a region-of-interest, the network produces an estimate of the tactile sensor output should it make contact with the environment at the region-of-interest. Advantage of using depth data is twofold: first, depth sensor data captures surface structures that directly affect the tactile sensor output. Second, depth data is robust to lighting changes compared to RGB images.

We presented qualitative and quantitative results that suggest the proposed method can estimate tactile sensor output from the depth data. We also presented results of a study that suggests reduction of point cloud density has negligible adverse affect on the quality of the tactile sensor estimates.

An interesting future direction is to study whether inclusion of RGB data and auditory information to complement the depth data will lead to an improved estimate of the tactile sensor output.

In future, we would like to increase the number of objects. We would also like to study the effectiveness of the tactile estimates for tasks such as classifying objects. Another area that can benefit from the system is grasp synthesis. We would like to explore application of the proposed method to rank grasps such that a robotic system can grasp an object in a single attempt, eliminating the need to sample the surface for a better grasp location.

References

[1] W. M. B. Tiest, "Tactual perception of material properties," *Vision research*, vol. 50, no. 24, pp. 2775–2782, 2010.

[2] M. Tanaka and T. Horiuchi, "Investigating perceptual qualities of static surface appearance using real materials and displayed images," *Vision research*, 2015.

[3] H. Yanagisawa and K. Takatsuji, "Effects of visual expectation on perceived tactile perception: An evaluation method of surface texture with expectation effect," *International Journal of Design*, 2015.

[4] R. W. Fleming, "Visual perception of materials and their properties," *Vision research*, vol. 94, pp. 62–75, 2014.

[5] P. Wacker, C. Wacharamanotham, D. Spelmezan, J. Thar, D. A. Sánchez, R. Bohne, and J. Borchers, "Vibrovision: an on-body tactile image guide for the blind," in *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, 2016.

[6] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-driven grasp synthesis—a survey," *IEEE Transactions on Robotics*, 2013.

[7] N. Jamali, M. Maggiali, F. Giovannini, G. Metta, and L. Natale, "A new design of a fingertip for the icub hand," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015.

[8] O. Kerpa, K. Weiss, and H. Worn, "Development of a flexible tactile sensor system for a humanoid robot," in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453)*, vol. 1. IEEE, 2003, pp. 1–6.

[9] G. Hellard and R. A. Russell, "A robust, sensitive and economical tactile sensor for a robotic manipulator," in *Australian Conference on Robotics and Automation*. Citeseer, 2002, pp. 100–104.

[10] M. Ohka, Y. Mitsuya, Y. Matsunaga, and S. Takeuchi, "Sensing characteristics of an optical three-axis tactile sensor under combined loading," *Robotica*, vol. 22, no. 2, pp. 213–221, 2004.

[11] M. K. Johnson and E. H. Adelson, "Retrographic sensing for the measurement of surface texture and shape," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 1070–1077.

[12] E. Torres-Jara, I. Vasilescu, and R. Coral, "A soft touch: Compliant tactile sensors for sensitive manipulation," 2006.

[13] N. Wettels, J. A. Fishel, and G. E. Loeb, "Multimodal tactile sensor," in *The Human Hand as an Inspiration for Robot Hand Development*. Springer, 2014, pp. 405–429.

[14] Y. Tada, K. Hosoda, and M. Asada, "Sensing ability of anthropomorphic fingertip with multi-modal sensors," in *IEEE International Conference on Intelligent Robots and Systems*. Citeseer, 2004, pp. 1005–1012.

[15] Y. Tanaka, M. Tanaka, and S. Chonan, "Development of a sensor system for collecting tactile information," *Microsystem Technologies*, vol. 13, no. 8-10, pp. 1005–1013, 2007.

[16] J. K. Kim, J. W. Wee, and C. H. Lee, "Sensor fusion system for improving the recognition of 3d object," in *IEEE Conference on Cybernetics and Intelligent Systems, 2004.*, vol. 2. IEEE, 2004, pp. 1207–1212.

[17] D. Taddeucci, C. Laschi, R. Lazzarini, R. Magni, P. Dario, and A. Starita, "An approach to integrated tactile perception," in *Proceedings of International Conference on Robotics and Automation*. IEEE, 1997.

[18] I. Fujimoto, Y. Yamada, T. Morizono, Y. Umetani, and T. Maeno, "Development of artificial finger skin to detect incipient slip for realization of static friction sensation," in *Proceedings of IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems, MFI2003.* IEEE, 2003, pp. 15–20.

[19] N. Jamali and C. Sammut, "Slip prediction using hidden markov models: Multidimensional sensor data to symbolic temporal pattern learning," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012.

[20] ——, "Majority voting: Material classification by tactile sensing using surface texture," *IEEE Transactions on Robotics*, vol. 27, no. 3, pp. 508–521, 2011.

[21] J. A. Fishel and G. E. Loeb, "Bayesian exploration for intelligent identification of textures," *Frontiers in neurorobotics*, vol. 6, p. 4, 2012.

[22] J. Mahler, M. Matl, V. Satish, M. Danielczuk, B. DeRose, S. McKinley, and K. Goldberg, "Learning ambidextrous robot grasping policies," *Science Robotics*, vol. 4, no. 26, p. eaau4984, 2019.

[23] 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," *IEEE Robotics and Automation Letters*, 2018.

[24] F. R. Hogan, M. Bauza, O. Canal, E. Donlon, and A. Rodriguez, "Tactile regrasp: Grasp adjustments via simulated tactile transformations," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 2963–2970.

[25] K. Takahashi and J. Tan, "Deep visuo-tactile learning: Estimation of tactile properties from images," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8951–8957.

[26] Y. Li, J.-Y. Zhu, R. Tedrake, and A. Torralba, "Connecting touch and vision via cross-modal prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[27] B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, "Benchmarking in manipulation research: Using the yale-cmu-berkeley object and model set," *IEEE Robotics & Automation Magazine*, 2015.

[28] R. B. Rusu, "Semantic 3d object maps for everyday manipulation in human living environments," *KI-Künstliche Intelligenz*, 2010.

[29] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *2009 IEEE international conference on robotics and automation*. IEEE, 2009, pp. 3212–3217.

[30] P. J. Besl and N. D. McKay, "Method for registration of 3-d shapes," in *Sensor fusion IV: control paradigms and data structures*, vol. 1611. International Society for Optics and Photonics, 1992, pp. 586–606.

[31] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[32] M. Mirza and S. Osindero, "Conditional generative adversarial nets. arxiv 2014," *arXiv preprint arXiv:1411.1784*, 2014.

[33] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014.

[34] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[35] C. Li and M. Wand, "Precomputed real-time texture synthesis with markovian generative adversarial networks," in *European conference on computer vision*. Springer, 2016, pp. 702–716.

[36] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[37] W. Yuan, S. Dong, and E. H. Adelson, "Gelsight: High-resolution robot tactile sensors for estimating geometry and force," *Sensors*, 2017.

[38] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[39] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, 2004.

[40] J. R. Flynn, S. Ward, J. Abich, and D. Poole, "Image quality assessment using the ssim and the just noticeable difference paradigm," in *International Conference on Engineering Psychology and Cognitive Ergonomics*. Springer, 2013.