Active Tactile Exploration using Shape-Dependent Reinforcement Learning

Shuo Jiang[1] and Lawson L.S. Wong[2]

***Abstract*— Tactile signals provide rich information about objects via touch and are essential for a robot to perform dexterous manipulation. Exploring actively via tactile perception collects important information about the workspace. However, designing an effective tactile exploration policy is challenging in unstructured environments. Typically, the geometric information is incomplete, and need to be completed by actively and repeatedly interacting with the environment. In this paper, we address the tactile exploration problem by proposing a shape-information-dependent exploration strategy, which consists of two components: (1) a Shape-Belief Encoder that encodes the explored area by learning effective 3-D reconstruction and predicts the complete object shape; (2) a shape-dependent exploration policy which incorporates the encoding in (1) to plan an exploration trajectory. The policy actively acquires new information about object surface by executing exploration actions. The Shape-Belief Encoder leverages the newly collected contact points to update the surface model and guides future exploration. We validate the proposed algorithm on simulated and real robots.**

I. INTRODUCTION

In this paper, we addressed the tactile exploration problem from a learning perspective: given a high-level goal (e.g., maximizing exploration efficiency) and some training scenarios, the robot discovers the exploration behavior by repeatedly interacting with the environments and learns how to transfer such ability to new scenarios.

To effectively plan an exploration trajectory covering the surface of an unknown object in an online fashion, the robot should make decisions based on the shape of the area already explored and the untouched area to be explored next. The shape inference ability enables the robot to predict the shape of the unknown area given the information of the explored shape (Figure 1). The exploration efficiency can be increased accordingly when the robot infers that in some area, there will be no object surface (such as beyond the object boundary), so the robot only needs to focus on planning touches in the potentially informative area. Based on such requirements, we aim to address three problems in this work: (1) how to process information from the explored area? (2) how to infer the shape of the unknown area? (3) how to plan trajectory incorporating information from (1) and (2)?

The task to be solved in this paper is that for any robot arm and dexterous hand with tactile sensors, it learns to perceive the shape information of an unknown object by actively planning and executing a trajectory of touches. To generate effective exploration behavior, the three questions in the last paragraph should be answered. For problem (1), we designed a Shape-Belief Encoder to transform the point cloud representing the explored area to a fixed-length latent code. To ensure the latent code captures object shape information, Shape-Belief Encoder reconstructs the original point cloud from such latent code by minimizing the difference between the reconstructed and the original point clouds. For problem (2), the latent code is then processed by a Hopfield network to predict complete shape from the partially explored area. The Hopfield network generates a new latent code encoding the predicted complete object shape. For problem (3), the two latent codes from (1) and (2) are provided to the exploration policy as the scene information to assist decision making. The exploration policy is trained by reinforcement learning to maximize the information gain by contacts.

Fig. 1: (Left) Point cloud of incomplete object already explored. (Middle) Predicted complete object shape from partial observation. (Right) robot hand with tactile sensors explores the unknown surface area by incorporating information from explored and predicted shapes.

Our main contributions can be summarized as (1) we propose a Shape-Belief Encoder, which can encode a point cloud to a fixed-length vector, infer the object surface distribution, predict complete object shape from an incomplete part, and distinguish different objects; (2) we applied a Hopfield network for the 3-D shape completion task; (3) we propose a shape-dependent tactile exploration policy that generates effective exploration trajectories and generalizes to new objects.

II. BACKGROUND

A. Tactile Exploration

Planning an exploration trajectory on a touching surface to gather tactile readings efficiently is a long-standing problem, and it becomes harder when working in unstructured environments. Exploration following a predefined trajectory [1], [2] can only work in limited scenarios and is less adaptive to environmental uncertainty; small discrepancies between object position or surface shape and the model may cause

[1]Northeastern University, 360 Huntington Ave, Boston, MA, USA jiang.shuo@northeastern.edu
[2]Northeastern University, 360 Huntington Ave, Boston, MA, USA lsw@ccs.neu.edu

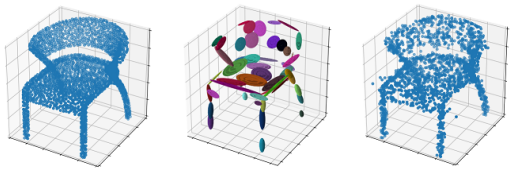

Fig. 2: (Left) Original object point cloud.
(Middle) Gaussian mixture model (GMM) representation.
(Right) Point cloud re-sampled from GMM.

the trajectory to fail. Other attempts based on local shape uncertainty model (e.g., based on Gaussian process regression) [3], [4], [5], [6], [7] could have the following problems: (1) gradient-based method do not fit gradient-discontinuous shapes well (like the edge of box); (2) calculating uncertainty can be costly in certain cases; (3) trajectory planned without a high-level strategy can be trapped in local optima; (4) hard to apply on multi-contact robots (like robot hand). The mentioned works are all planning methods which require the model to be precise and the solution is not transferable.

B. 3D Representation and Surface Distribution

An accurate and computation-efficient shape representation is essential for our shape-aware policy learning. Many surface or bulk representations of objects have been proposed, including point cloud [8], [9], [10], voxels [11], [12], meshes [13], and implicit surfaces [14]. Within these, mathematical models trying to capture the representations have also been proposed. Gaussian mixture model (GMM) representations, as shown in Figure 2, have been shown effective on capturing object surfaces represented by point clouds or meshes [15], [9].

C. Markov Decision Process

A Markov Decision Process (MDP) is a 4-tuple $M = \langle S, A, P, R \rangle$ where S is the set of states, A is the set of actions, $P(s_{t+1}|s_t, a_t)$ is the transition probability that action a in state s at time t will lead to state s at time $t+1$, $R(a_t, s_t)$ is the distribution of reward when taking action a_t in state s_t. A policy π is defined as the probability distribution of choosing action a_t given state s_t. The learning goal is to find a policy π^* that maximizes the accumulated reward in given horizon T, $\pi^* = \underset{\pi}{\operatorname{argmax}} \underset{a_t, s_t \sim \pi}{\mathbb{E}} \left[\sum_{t=0}^{T-1} \gamma^t R(a_t, s_t)\right]$, where γ is discount factor. Our tactile exploration task will be formulated as an MDP and solved by reinforcement learning algorithms.

D. Hopfield Neural Network

Hopfield network [16] is a special type of recurrent neural network which can store and retrieve memory. It is widely applied in image completion and image denoising tasks. Patterns $\mathbf{Z}$ can be stored by directly constructing the weights of the neural network instead of propagating gradient iteratively. A stored pattern (a row in $\mathbf{Z}$) can be retrieved by setting a query vector z as the initial state and iteratively updating the state by Equation 1.

$$z' = \text{softmax}\left(\beta \cdot z \cdot \mathbf{Z}^T\right) \cdot \mathbf{Z} \tag{1}$$

β is a hyperparameter. The final output z' will resemble one particular or several items in $\mathbf{Z}$. The term softmax $\left(\beta \cdot z \cdot \mathbf{Z}^T\right)$ is usually considered as a distribution measuring the likelihood between query vector z and each item in $\mathbf{Z}$. We will use this neural network to predict complete 3D shape from partial tactile information.

III. Related Works

A. Tactile Exploration

Tactile exploration by dexterous robots has been studied in the past decades, and many of them reached satisfactory performance in grasping, reconstruction, or manipulation tasks. [7] proposed a visuo-tactile exploration strategy. A pre-trained neural network first provides an initial prediction of the voxel-based object shape given the RGBD camera input. Then the high-resolution tactile signals are used to refine the 3D model. The grid-based uncertainty drives robot to explore the most uncertain area. The method shares a similar idea with ours that it is important to execute exploration based on shape posterior, which is the predicted complete shape from partial observation. Similarly, [14] learns a 3D posterior from depth image by representing the surface as a Signed Distance Function. [17] proposed a visuo-tactile framework doing peg insertion tasks with deep reinforcement learning. The mentioned works give shape posterior by depth image input while ours are based on raw tactile signal. In visually occluded environments (such as dirty water), such systems easily fail.

[18] proposed a query path exploration which can be more efficient than probing. The object surface is represented as a Gaussian process implicit surface [19], and the sliding direction is towards the highest local variance. The proposed methods could potentially have the following problems: (1) local uncertainty based trajectory planning can lead to a sub-optimal path (such as a dead-end); (2) Gaussian process implicit surface update can be computationally intensive as updating inverse kernel matrix requires $O\left(n^3\right)$ complexity; (3) the surface must be continuous and gradient-continuous, which can be alleviated by introducing local information [20]. Such work has been extended in [3]. However, it is limited on bi-manual robot without considering the physical constraints with multi contacts. Sometime multiple goals can be achieved by two arms but not reachable by multiple fingers. Similar exploration strategies based on Gaussian process surface have also been proposed in [5], [21], [1], [4]. [4] leverages local gradient of Gaussian process implicit surface to balance exploration and exploitation.

[6] proposed an exploration strategy for robot hand with full-hand tactile mapping sensor by null-space control. However, this method does not tackle with high-level strategy and the desired contact points are pre-defined, which makes the work less general.

[2] employed a high-resolution tactile sensor to reconstruct the 3D model combining robot kinematics. However, their exploration only follows several predefined grasping directions.

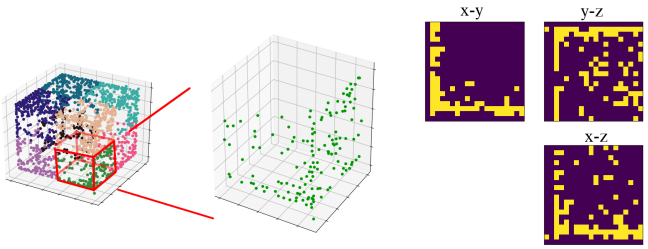

Fig. 3: Visualization of point-cloud local projection. (Left) Point cloud segmented into $2^3 = 8$ cells. (Right) Orthogonal projections from one such cell.

B. Point Cloud Encoder

To process point cloud data by a neural network, it is required the point cloud data with arbitrary length can be encoded into a fixed-length input vector. As point cloud data is permutation invariant, previous encoders [22], [23], [24] process such data using permutation equivariant functions. However, we have only limited choices of permutation invariant operations in practice such as max, min, sum, or mean. Such operation aggregates the local features from each point to form a global feature that may lose local details.

IV. METHOD

In this section, we show how our shape-dependent exploration system works. The first part introduces the structure of Shape-Belief Encoder which can translate a point cloud to a fixed-length latent vector while learning a GMM to represent the shape of such point cloud. A Hopfield network is then used for shape completion. The second part explains how the MDP of tactile exploration task is defined and how the Shape-Dependent Policy is learned. At the end, we show the overall structure of our system combining Shape-Belief Encoder and Shape-Dependent Policy implemented on robot.

A. Shape-Belief Encoder

In order to use neural networks to process a point cloud containing an arbitrary number of points, we need to encode the point cloud into a fixed-length vector. The vector should contain important geometric information about the point cloud. Based on such requirements, we propose Shape-Belief Encoder, designed to (1) encode a point cloud; (2) infer surface distribution; (3) predict the complete shape. Any point cloud with arbitrary number of points can be the input. Features of the point cloud are first extracted by local projection. Then, an internal encoder translates the features to a latent vector. The latent vector has two paths: a decoder that translates the latent vector to a GMM, representing the surface distribution; and a Hopfield network that generates a new latent vector, representing the predicted complete object shape.

Local Projection: The explored area is represented as a point cloud of contact points. The point cloud is first segmented evenly into local cells (Figure 3 left). Each cell contains a subset of the original point cloud. In each cell, a series of 2-D projections from different angles are generated. The projection is a 2-D array, where the value of each entry is 0 if no point is projected inside, or 1 if any point is projected inside. Here, we only use orthogonal projection because it requires no linear transformation, but projections from other directions can also be considered. A demonstration of such projection is shown in Figure 3. The projections from all cells are used as the feature map of the point cloud for further processing.

Encoder: The encoder structure is shown in Figure 4 left. For each local feature map, there will be an independent local receptor which is composed by multiple convolutional layers to encode it. The outputs of receptors will be concatenated and translated to latent code z by multi-layer perceptrons (MLP). We use the reparameterization trick [25] to learn the distribution of latent codes, where the encoder predicts μ_z and σ_z as the parameters of independent Gaussian distributions. A latent code z is then sampled from such distribution.

Decoder: The decoder structure is shown in Figure 4 right. A mixture density network (MDN) [26] translates the latent code predicted by the encoder to GMM parameters with K clusters. In each cluster, weight π_k, mean μ_k and covariance matrix Σ_k are outputs of the neural network. To ensure the covariance matrix predicted by the decoder is positive definite, the neural network first predicts eigenvalues to form a diagonal matrix $\mathrm{diag}(\lambda_1, \lambda_2, \lambda_3)$, then predict the roll-pitch-yaw angles γ, β, α in $[-\pi, \pi]$. A rotation matrix $\mathbf{R}$ can be calculated from α, β, γ by Equation 2 and the final covariance matrix of one cluster is calculated by Equation 3. $\mathbf{R}_x$, $\mathbf{R}_y$ and $\mathbf{R}_z$ are rotation matrices around roll, pitch, and yaw axes.

$$\mathbf{R} = \mathbf{R}_z(\alpha)\,\mathbf{R}_y(\beta)\,\mathbf{R}_x(\gamma) \tag{2}$$

$$\Sigma = \mathbf{R} \cdot \mathrm{diag}(\lambda_1, \lambda_2, \lambda_3) \cdot \mathbf{R}^T \tag{3}$$

Training: We adopt variational inference [25] to train the model. The Evidence Lower Bound (ELBO) to be maximized is shown in Equation 4. $\mathbf{X}$ is the point cloud data, θ and ϕ are parameters of decoder and encoder respectively. By maximizing this bound, the encoder matches the GMM parameters to the shape of the original point cloud.

$$\mathrm{ELBO}(\theta, \phi, \mathbf{X}) = \mathbb{E}_{q_\phi(z|\mathbf{X})}\left[\log p_\theta(\mathbf{X}|z)\right] - D_{KL}\left(q_\phi(z|\mathbf{X})\,||\,p(z)\right) \tag{4}$$

B. Shape Completion using a Hopfield Network

To plan the exploration trajectory, it is important to predict the shape of unexplored area given the shape of explored area, which can be seen as a 3-D completion task. As Hopfield networks have been successfully applied on image completion tasks [16], we apply a similar idea to our 3-D completion task by adding a Hopfield network at the end of the encoder, as shown in Figure 4 left (dashed component). The point clouds of several reference objects are selected as reference patterns and processed by the encoder to generate a batch of reference latent codes $\mathbf{Z}$. These latent codes are stored as the internal memory of the Hopfield network. For a test object with latent code z, a new latent code z' is generated by propagating z through the Hopfield network (by iterating Equation 1); the final z' encodes the shape of the

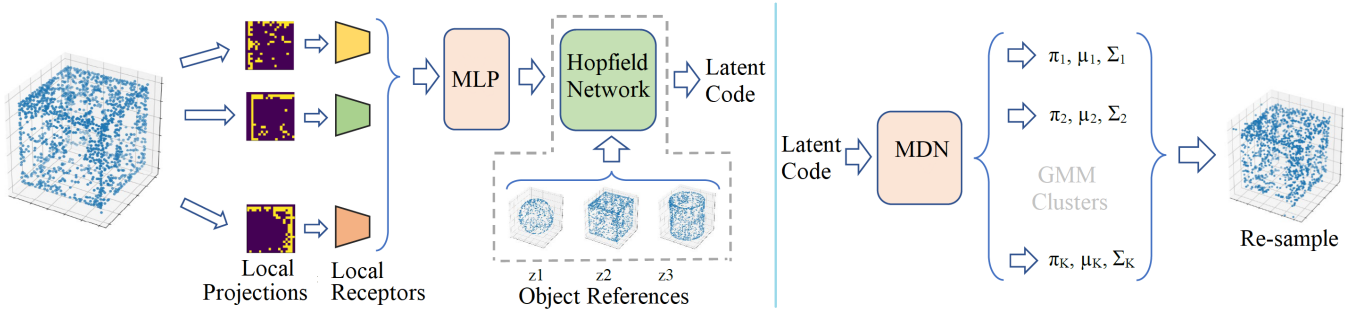

Fig. 4: Shape-Belief Encoder. (Left) Encoder: Local projections are processed by local receptors and concatenated, then translated to latent code by multi-layer perceptrons (MLP). Hopfield network (dashed component) is only used for object completion, not for object reconstruction (i.e., not during training of the encoder-decoder itself).
(Right) Decoder: The latent code is incorporated by a mixture density network (MDN) to predict cluster parameters of a GMM, representing the surface distribution. A new point cloud can then be re-sampled from the GMM.

predicted complete object. We show in our experiments that using the Hopfield network to perform 3-D shape completion significantly improves the recognition ability, even when the provided point cloud is only a fraction of the original object.

C. Shape-Dependent Policy

The robot executes tactile exploration following a Shape-Dependent Policy $a = \pi(s)$, which incorporates state s and generates action a. The state s includes the robot configuration and shape information from the Shape-Belief Encoder. The MDP of tactile exploration task is defined as:

- **State**: The state includes (1) all robot joint angles and velocities; (2) all Cartesian coordinates of robot links; (3) shape latent code z before entering Hopfield Network; (4) z' after propagating through the Hopfield Network by iterating Equation 1. We assume z encodes the shape of the explored area while z' encodes the belief of what the complete object should be.
- **Action**: The action is defined as the incremental angles $\Delta\theta$ for all joints. The goal joint angle in next time step is $\theta_{t+1} = \theta_t + \Delta\theta_t$, which will be controlled by low-level controllers. We did not use Cartesian-space actions because it is a multi-contact trajectory planning system. Planning goals for multiple fingers in Cartesian space may not be correctly executed due to mechanical constraints.
- **Reward**: To encourage the robot arm to actively explore the workspace, we propose an entropy-based reward at each time step, which aims to guide the policy to minimize future spatial uncertainty of the workspace. First, we voxelize the working space as shown in Figure 7 left. For each cell in the voxels, there is an uncertainty value p stored, ranging from 0 (free space) to 1 (object). All cells' initial uncertainty values are set as 0.5 (unsure). At each time step, the tactile sensors will update the uncertainty value in the corresponding cell by increasing two counters. The object counter N_o counts the number of times when the tactile sensors make contact in the cell. The free space counter N_f counts the number of times sensors indicates free space (when the sensor is in that cell but with no reading). The uncertainty value p_c in cell c is updated by Equation 5. We expect such soft update strategy instead of hard assignment to be more robust to noise.

$$p_c = \frac{N_o}{N_o + N_f} \tag{5}$$

The uncertainty values in each cell c can be used to calculate the entropy by Equation 6.

$$H(c) = -p_c \cdot \log(p_c) - (1 - p_c) \cdot \log(1 - p_c) \tag{6}$$

We use Information Gain as the reward in MDP, calculated as the reduction of entropies between two consecutive time steps (Equation 7).

$$I_t = \sum_c H_{t-1}(c) - H_t(c) \tag{7}$$

The policy π^* aims to maximize future Information Gain by Equation 8, which can be trained by reinforcement learning algorithms; we use Soft Actor-Critic [27] in our case (see Section V-D) as it has been demonstrated to be an efficient learning algorithm in robotics [28].

$$\pi^* = \underset{\pi}{\operatorname{argmax}} \mathbb{E}\left[\sum_{t=0}^{T-1} \gamma^t \cdot I_t\right] \tag{8}$$

The voxelization of workspace indicates the progression of space exploration as the entropy (uncertainty) of the space is decreasing while exploring. Also, only the cells that overlap with the robot finger tip are updated at each time step, which is computationally efficient.

D. Overall System Structure

Combining Shape-Belief Encoder and Shape-Dependent Policy, the overall system structure is shown in Figure 5. Starting from the top left corner, the explored area of the object surface, represented as a point cloud, is encoded by the Shape-Belief Encoder. A reconstruction head during training ensures that the latent code captures full shape information. The latent code z is then passed into the Hopfield network (blue box) to generate a second latent code z' encoding the predicted complete object shape. Both latent codes are concatenated with the robot configuration, forming the state

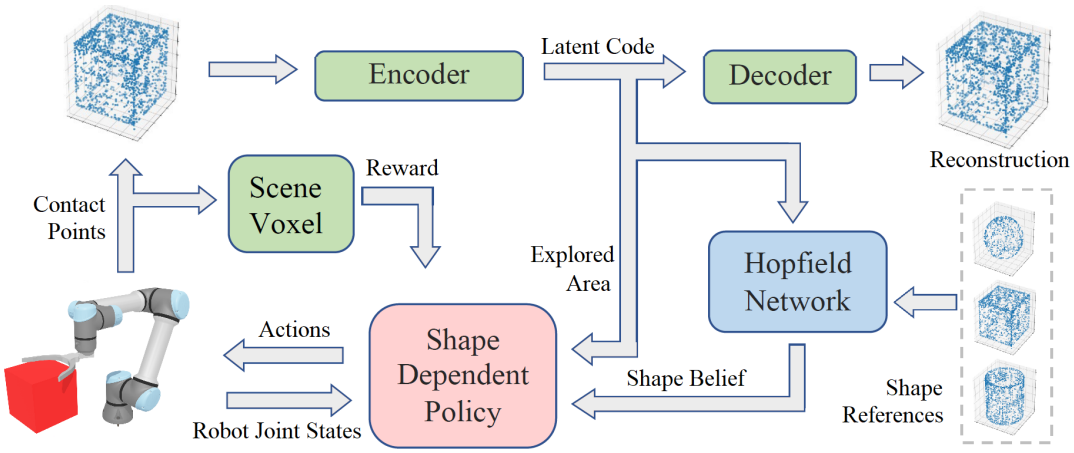

Fig. 5: Overall system. The explored area of the object surface, represented as a point cloud (top left corner), is processed by the encoder; the encoder is learned by reconstructing the point cloud via a GMM surface representation. The latent code z encoding the explored area is further processed by the Hopfield network (blue box) by generating the predicted shape-belief code z'. The two codes, encoding the incomplete and predicted complete shapes respectively, are used as state information for the Shape-Dependent Policy (red box) to control the robot. The newly generated contact points (left) are used to calculate information gain (reward) and update the explored area (top left corner).

vector, which is used by the Shape-Dependent Policy (red box) to generate actions for each robot joint. After moving the joints, new contact points are used to update the explored point cloud (top left corner) and to calculate information gain (reward) for training the policy.

V. Experiments

In this section, we show our proposed system accomplishes tactile exploration tasks. We first demonstrate the validity of each of our system's components: (1) 3-D reconstruction performance of Shape-Belief Encoder; (2) 3-D completion performance on *incomplete* point clouds. Then we combine the Encoder with Shape-Dependent Policy to (3) generate exploration trajectories on different objects and generalize to unseen objects. Finally, we demonstrate the performance of our system on a real robot.

A. 3-D Reconstruction

Shape-Belief Encoder compresses objects' geometric information into latent codes by conducting accurate 3-D reconstruction from such latent codes. For the 3-D reconstruction task, we train our Shape-Belief Encoder on the ShapeNet dataset [29]. The input of the Encoder is a point cloud, from which the Encoder learns the surface distribution. We compare the reconstruction results with TopNet [30], PCN [31], and FoldingNet [32]. We evaluate on 8 randomly sampled shapes from ShapeNet using the Chamfer Distance metric, shown in Table I. Our model achieves the smallest Chamfer Distance in most cases, which indicates that our model captures the shape and density of the point cloud with high precision. We also compared F-score proposed in [33] in Table II. Qualitative reconstruction results are shown in Figure 6. We also emphasize that our model trains very quickly (several seconds vs. hours for baselines), because of two reasons: (1) We turn a 3D recognition problem to a 2D recognition problem by local projections; (2) the GMM loss only needs to calculate cluster likelihood functions, instead of comparing points in a pairwise fashion.

×0.001	plane	cabin	car	chair	lamp	sofa	table	craft
PCN	5.4	**9.9**	12.2	11.0	10.5	**9.5**	13.4	9.3
TopNet	**5.0**	12.3	11.9	13.3	15.3	10.6	13.1	10.9
FoldingNet	7.0	16.1	18.2	24.8	23.1	16.3	22.3	16.4
Ours	**5.0**	12.8	**11.5**	**8.9**	**8.4**	10.3	**11.1**	**8.5**

TABLE I: 3-D Reconstruction Results on ShapeNet: Chamfer Distance comparison (lower is better)

	plane	cabin	car	chair	lamp	sofa	table	craft
PCN	87.8	99.5	93.7	80.9	**94.4**	93.5	89.9	87.8
TopNet	89.4	**99.9**	85.3	80.7	92.1	89.3	71.7	74.1
FoldingNet	72.0	97.8	66.8	63.2	77.0	69.7	40.0	40.4
Ours	**92.4**	99.5	**94.9**	**95.2**	94.3	**97.7**	**97.6**	**97.2**

TABLE II: 3-D Reconstruction Results on ShapeNet: F-score comparison (higher is better)

B. Object Shape Completion

To infer complete object shape from a fraction of the original point cloud is important in tactile exploration. We show that adding the Hopfield network layer in the testing phase is essential to recognize the object correctly. Figure 7 (a-c) shows the reconstruction results by full object point cloud, 1/2 point cloud, and 1/8 point cloud without Hopfield network layer. We see with fewer points, the reconstructions become more blurred and indistinguishable, which means the encoder has difficulty recovering complete object shape from partial information. The result after adding Hopfield network is shown in Figure 7 (d). We see that the Hopfield Network improves recognition ability significantly.

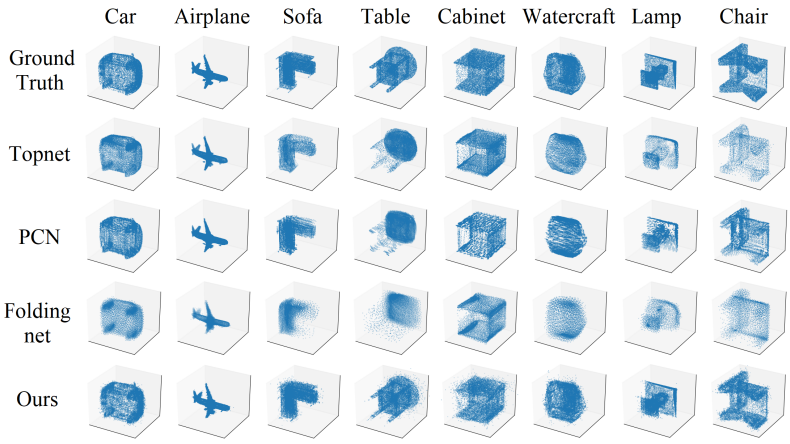

Fig. 6: Qualitative reconstruction results on some objects in the ShapeNet dataset [29].
From top to bottom: (1) Ground Truth; (2) TopNet [30]; (3) PCN [31]; (4) FoldingNet [32]; (5) Ours.

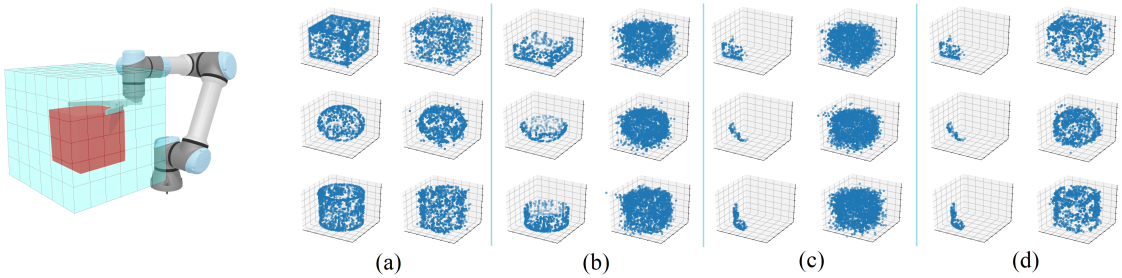

Fig. 7: (Left) Voxelization of working space. (Right a-c) Reconstruction from the full, 1/2, and 1/8 object point cloud *without* shape completion. The left of each pair is the input of the model and the right of each pair is the reconstruction result. (d) Reconstruction from 1/8 object point cloud *with* shape completion using the Hopfield Network.

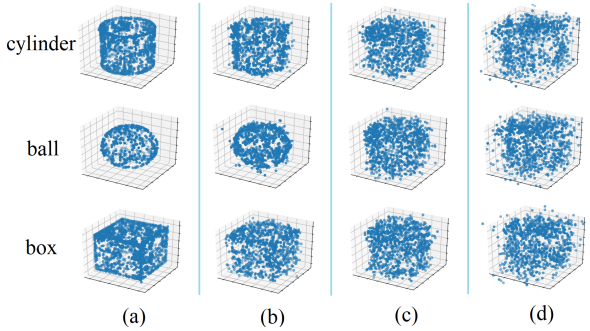

Fig. 8: Reconstruction performance. (a) Original objects. (b) Our method. (c) DeepSets. (d) PointNet++.

C. Encoder Comparison

We compare our proposed encoder (local projection) with DeepSets [22] encoder and PointNet++ encoder [24]. We found that when the encoders and objects are trained in pair (one encoder is only used to train one object), the reconstruction performances of DeepSets and PointNet++ are comparable to our method (Figure 8 b). However, when one encoder is used to train multiple objects (by randomly sampling from the training set), DeepSets and PointNet++ cause the model to learn the average shape instead of correctly distinguishing them (Figure 8 c,d).

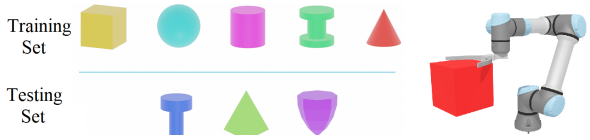

Fig. 9: (Left) Objects used for Shape-Dependent Policy. (Right) Robot in simulation.

D. Tactile Exploration Policy

We tested our Shape-Dependent Policy by executing tactile exploration tasks on different objects, shown in Figure 9 left. To show the generalization ability of our work, we use five objects with basic shapes including box, ball, cylinder, bell, and cone as the training set. Three testing objects including screw, tetrahedron, and diamond are used as novel objects to show generalization. A UR5e robot with pneumatic gripper attached executes exploration tasks in a Gazebo simulation (Figure 9 right). The tactile sensors are mounted on fingertips. We use the Soft Actor-Critic [27] algorithm to train the exploration policy. The task is episodic and each episode lasts for 120 seconds with 1-second duration for each step. At the beginning of each episode, one of the five objects is re-spawned at the center of the workspace. The robot has no prior information about what object to be explored in each episode but only perceives by tactile exploration.

	box	ball	cyl.	bell	cone	DR	rand	SDP
box	342.6	256.3	202.3	211.6	117.4	153.5	7.5	312.3
ball	185.3	374.3	138.3	219.3	115.4	169.7	7.5	372.2
cylinder	252.3	262.2	383.7	231.2	140.2	251.2	9.4	365.2
bell	361.2	295.4	135.6	604.0	185.2	194.1	2.3	432.3
cone	213.5	182.1	92.0	205.6	288.4	142.6	13.1	243.1
screw	252.3	210.1	183.6	323.2	61.2	202.1	4.5	314.2
tetra-	211.1	172.8	113.2	35.7	113.5	162.8	7.3	318.6
diamond	269.2	265.7	143.2	235.3	99.6	242.8	17.9	333.2

TABLE III: Explored area on various training and testing objects, using the shape-independent policies, domain-randomization policy (DR), random policy, and our Shape-Dependent Policy (SDP). Values are accumulated information gain from each policy (10 trials; higher is better).

For comparison, we train five shape-*independent* policies, whose state only consists of robot configuration but without shape information (latent codes). Each shape-independent policy is only trained on one particular object (e.g. "box" policy is only trained on the box object, but can be tested on a box or ball). We expect such shape-independent policies to work well on their paired training object, but to have bad exploration performance when the object is mismatched. The accumulated exploration area in one testing episode for all policies are shown in Table III; the values are averaged across 10 independent trials. The area is expressed by the total number of non-overlapping contact points. The columns indicate the policies, while the rows indicating the test objects. For example, "box" row and "bell" column means the policy was trained on exploring "bell" but tested on "box" (without shape information). The "DR" column is the policy trained on all five objects, based on domain randomization in reinforcement learning [34], still without shape information. The "random" column is the random policy in joint space. The "SDP" column is our Shape-Dependent Policy, which is also trained on all five objects, but the shape information is explicitly provided by the inferred shape encoding.

From Table III, we can conclude that our policy (SDP) generalizes well on all five training objects and three testing objects. The shape-independent policies (first 5 columns) only perform well in their own scenarios, which can be verified by the diagonal pattern in the upper part of the table. Even if shape information of all five objects are implicitly provided to the DR policy during training, it still generalizes poorly without directly incorporating shape encoding as part of its state. The exploration result is shown in Figure 10.

E. Object Recognition

To show the Hopfield network works well as an object shape recognizer, we recorded the output beliefs for the five test objects in the SDP experiment section. One result is shown in Figure 11. The belief is given by Equation 9

$$\text{belief} = \text{softmax}\left(\beta \cdot z \cdot \mathbf{Z}^T\right) \tag{9}$$

where z is the latent code of object being touched, and $\mathbf{Z}$ is the latent code matrix of the five training objects. The belief

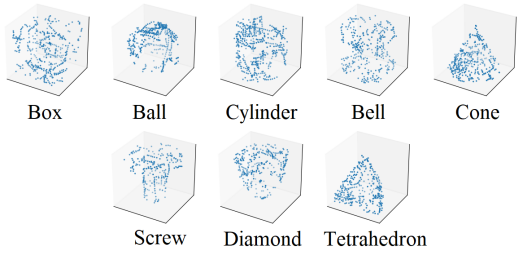

Fig. 10: Explored area by SDP

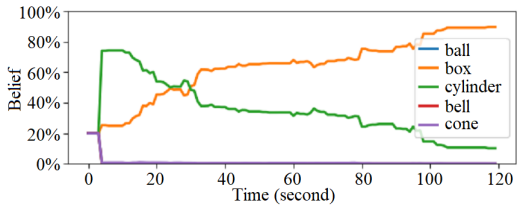

Fig. 11: Object belief in SDP, ground truth object is box.

in Equation 9 gives a distribution measuring the likelihood between latent code of the current object and latent codes of all training objects. The object to be explored is a box but unknown to the robot, so the robot has to recognize the object's shape by active tactile exploration. The graph shows the beliefs for the five objects. At the beginning, there is no contact point, beliefs for all objects are equal (20%). At about 5 seconds, the hand reaches the object, some contact points are collected and used to update beliefs. With just a small set of points explored by this time step, one can see the probabilities of cylinder and box increase. The robot concludes the explored area could not belong to a ball, bell, or cone. With more area explored, the robot gradually assigns higher probability to the true object, the box. At the very end, the robot is about 90% sure that the object touched is a box.

F. Application on a Physical Robot

We tested the Shape-Dependent Policy on a real robot, a UR5e robot arm with a pneumatic gripper [35]. One tactile sensor (Adafruit Round Force-Sensitive Resistor (FSR) [ADA166]) is mounted on each finger of the gripper. A contact is detected when the contact force on the sensor is above a certain threshold. Three objects with soft materials are used for testing: a yoga ball, a block box, and a cookie box (Figure 12). The explored area through time and the collected contact points are shown in Figure 13. The blue and red dots correspond to points collected from the two fingers respectively. The result is less dense compared to

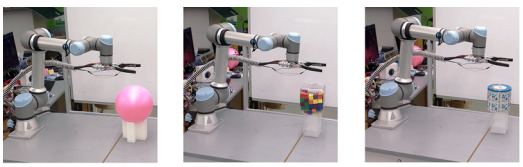

Fig. 12: Robot exploring (a) ball; (b) box; (c) cylinder

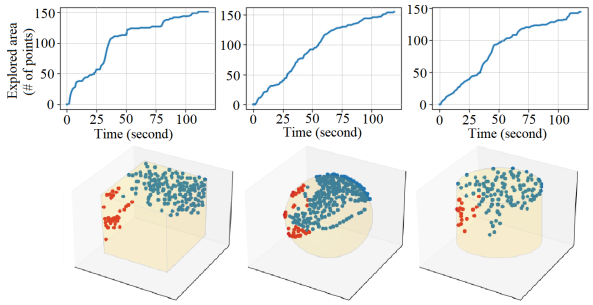

Fig. 13: (Top) Explored area with box, ball and cylinder. (Bottom) Explored point cloud.

simulation due to a sim-to-real gap; some points touched in simulation are not reached precisely on the real robot.

VI. CONCLUSION

In this work, we proposed a shape-dependent reinforcement learning framework for tactile-only exploration of objects in the robot workspace. We used a Shape-Belief Encoder to encode the point cloud of the explored area of an object, and used a Hopfield network to predict the complete object shape. The encoded and predicted shape information is then given to a Shape-Dependent Policy for context-based tactile exploration. Results in simulation and on a physical robot showed that the proposed framework can (1) effectively learn a GMM for object-surface representation; (2) predict complete object shape from a fraction of the explored area; (3) guide robots to comprehensively touch and explore object surfaces using tactile sensors.

REFERENCES

[1] Sommer, Nicolas, Miao Li, Aude Billard. "Bimanual compliant tactile exploration for grasping unknown objects." ICRA, 2014.

[2] Bauza, Maria, Oleguer Canal, Alberto Rodriguez. "Tactile mapping and localization from high-resolution tactile imprints." ICRA, 2019.

[3] Driess, Danny, Daniel Hennes, Marc Toussaint. "Active multi-contact continuous tactile exploration with Gaussian process differential entropy." ICRA, 2019.

[4] Dragiev, Stanimir, Marc Toussaint, Michael Gienger. "Uncertainty aware grasping and tactile exploration." ICRA, 2013.

[5] Jamali, Nawid, Carlo Ciliberto, Lorenzo Rosasco, Lorenzo Natale. "Active perception: Building objects' models using tactile exploration." Humanoids, 2016.

[6] Sommer, Nicolas and Aude Billard. "Multi-contact haptic exploration and grasping with tactile sensors." Robotics and Autonomous Systems 85: 48–61, 2016.

[7] Wang, Shaoxiong, Jiajun Wu, Xingyuan Sun, Wenzhen Yuan, William T. Freeman, Joshua B. Tenenbaum, Edward H. Adelson. "3D shape perception from monocular vision, touch, shape priors." IROS, 2018.

[8] Fan, Haoqiang, Hao Su, Leonidas J. Guibas. "A point set generation network for 3D object reconstruction from a single image." CVPR, 2017.

[9] Hertz, Amir, Rana Hanocka, Raja Giryes, Daniel Cohen-Or. "PointGMM: A neural GMM network for point clouds." CVPR, 2020.

[10] Watkins-Valls, David, Jacob Varley, Peter Allen. "Multi-modal geometric learning for grasping and manipulation." ICRA, 2019.

[11] Liu, Min, Zherong Pan, Kai Xu, Kanishka Ganguly, Dinesh Manocha. "Generating grasp poses for a high-DOF gripper using neural networks." IROS, 2019.

[12] Lu, Qingkai, Mark van der Merwe, Balakumar Sundaralingam, Tucker Hermans. "Multifingered grasp planning via inference in deep neural networks: Outperforming sampling by learning differentiable models." IEEE Robotics & Automation Magazine 27(2): 55–65, 2020.

[13] Gao, Jun, Wenzheng Chen, Tommy Xiang, Alec Jacobson, Morgan McGuire, Sanja Fidler. "Learning deformable tetrahedral meshes for 3D reconstruction." NeurIPS, 2020.

[14] van der Merwe, Mark, Qingkai Lu, Balakumar Sundaralingam, Martin Matak, Tucker Hermans. "Learning continuous 3D reconstructions for geometrically aware grasping." ICRA, 2020.

[15] Eckart, Benjamin, Kihwan Kim, Alejandro Troccoli, Alonzo Kelly, Jan Kautz. "Accelerated generative models for 3D point cloud data." CVPR, 2016.

[16] Ramsauer, Hubert, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Thomas Adler, Lukas Gruber, Markus Holzleitner, Milena Pavlović, Geir Kjetil Sandve, Victor Greiff, David Kreil, Michael Kopp, Günter Klambauer, Johannes Brandstetter, Sepp Hochreiter. "Hopfield networks is all you need." arXiv:2008.02217, 2020.

[17] Lee, Michelle A., Yuke Zhu, Krishnan Srinivasan, Parth Shah, Silvio Savarese, Li Fei-Fei, Animesh Garg, Jeannette Bohg. "Making sense of vision and touch: Self-supervised learning of multimodal representations for contact-rich tasks." ICRA, 2019.

[18] Driess, Danny, Peter Englert, Marc Toussaint. "Active learning with query paths for tactile object shape exploration." IROS, 2017.

[19] Williams, Oliver and Andrew Fitzgibbon. "Gaussian Process implicit surfaces." Workshop on Gaussian Processes in Practice, 2006.

[20] Ottenhaus, Simon, Martin Miller, David Schiebener, Nikolaus Vahrenkamp, Tamim Asfour. "Local implicit surface estimation for haptic exploration." Humanoids, 2016.

[21] Yi, Zhengkun, Roberto Calandra, Filipe Veiga, Herke van Hoof, Tucker Hermans, Yilei Zhang, Jan Peters. "Active tactile object exploration with Gaussian processes." IROS, 2016.

[22] Zaheer, Manzil, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R. Salakhutdinov, Alexander J. Smola. "Deep sets." Neural Information Processing Systems (NeurIPS), 2017.

[23] Qi, Charles R., Hao Su, Mo Kaichun, Leonidas J. Guibas. "PointNet: Deep learning on point sets for 3D classification and segmentation." CVPR, 2017.

[24] Qi, Charles Ruizhongtai, Li Yi, Hao Su, Leonidas J. Guibas. "PointNet++: Deep hierarchical feature learning on point sets in a metric space." Neural Information Processing Systems (NeurIPS), 2017.

[25] Kingma, Diederik P., Max Welling. "Auto-encoding variational Bayes." International Conference on Learning Representations (ICLR), 2014.

[26] Bishop, Christopher M. "Mixture density networks." Technical Report NCRG/94/004, Aston University, 1994.

[27] Haarnoja, Tuomas, Aurick Zhou, Pieter Abbeel, Sergey Levine. "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor." International Conference on Machine Learning (ICML), 2018.

[28] Haarnoja, Tuomas, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, Sergey Levine. "Soft actor-critic algorithms and applications." arXiv:1812.05905, 2018.

[29] Chang, Angel X., Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, Fisher Yu. "ShapeNet: An information-rich 3D model repository." arXiv:1512.03012, 2015.

[30] Tchapmi, Lyne P., Vineet Kosaraju, Hamid Rezatofighi, Ian Reid, Silvio Savarese. "TopNet: Structural point cloud decoder." CVPR, 2019.

[31] Yuan, Wentao, Tejas Khot, David Held, Christoph Mertz, Martial Hebert. "PCN: Point completion network." International Conference on 3D Vision (3DV), 2018.

[32] Yang, Yaoqing, Chen Feng, Yiru Shen, Dong Tian. "FoldingNet: Point cloud auto-encoder via deep grid deformation." CVPR, 2018.

[33] Tatarchenko, Maxim, Stephan R. Richter, René Ranftl, Zhuwen Li, Vladlen Koltun, Thomas Brox. "What do single-view 3D reconstruction networks learn?" CVPR, 2019.

[34] OpenAI, Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, Jonas Schneider, Nikolas Tezak, Jerry Tworek, Peter Welinder, Lilian Weng, Qiming Yuan, Wojciech Zaremba, Lei Zhang. "Solving Rubik's cube with a robot hand." arXiv:1910.07113, 2019.

[35] Schwarm, Eric, Kevin M. Gravesmill, John P. Whitney. "A floating-piston hydrostatic linear actuator and remote-direct-drive 2-DOF gripper." ICRA, 2019.