APPEARED IN
Proc. IEEE Int. Conf. on Robotics & Automation, 2019

Learning To Grasp Under Uncertainty Using POMDPs

Neha P Garg[1,2] David Hsu[1,2] Wee Sun Lee[1]

Abstract— **Robust object grasping under uncertainty is an essential capability of service robots. Many existing approaches rely on far-field sensors, such as cameras, to compute a grasp pose and perform open-loop grasp after placing gripper under the pose. This often fails as a result of sensing or environment uncertainty. This paper presents a principled, general and efficient approach to adaptive grasping, using both tactile and visual sensing as feedback. We first model adaptive grasping as a partially observable Markov decision process (POMDP), which handles uncertainty naturally. We solve the POMDP for sampled objects from a set, in order to generate data for learning. Finally, we train a grasp policy, represented as a deep recurrent neural network (RNN), in simulation through imitation learning. By combining model-based POMDP planning and imitation learning, the proposed approach achieves robustness under uncertainty, generalization over many objects, and fast execution. In particular, we show that modeling only a small sample of objects enables us to learn a robust strategy to grasp previously unseen objects of varying shapes and recover from failure over multiple steps. Experiments on the G3DB object dataset in simulation and a smaller object set with a real robot indicate promising results.**

I. INTRODUCTION

Robots in any modern e-commerce warehouse need to process thousands of orders every day. To finish their task, such robots should pick objects of various shapes and sizes quickly and reliably from a shelf amid action and perception uncertainty. Most state of the art solutions (see Sec. II) for autonomous grasping do open loop grasp execution which is fast but does not work well amid action and perception uncertainty. They mainly focus on computing a grasp pose from raw sensor data (generally RGBD point cloud).

A grasp pose is a 6 DOF pose of the robot hand and it's fingers' configurations such that when the robot hand is placed in that pose and the fingers are closed, it is able to grasp the object successfully. From the grasp pose, a pre-grasp pose is computed which is a slightly translated pose from the grasp pose so that a collision free motion plan exists. Then open loop grasp execution is done by placing the gripper in pre-grasp pose, moving forward to reach the grasp pose and closing fingers to pick the object.

Amid uncertainty, computed grasp pose is often inaccurate due to noisy sensors and algorithmic errors. Even for a correct grasp pose, grasp execution can fail due to robot control and calibration errors. For example, gripper may push the object away while moving forward and close without the object inside it. It might overshoot and not grasp the object in case of thin objects and might not be able to go into the gap for complex objects like a headphone (See open loop grasp executions in video[1]). To prevent and recover from such failures during grasp execution, we should compute closed loop plans conditioned on sensor feedback like touch, vision and proprioception. One of the naive closed loop plan is to repeatedly recompute grasp pose and do open loop grasp execution until success. In this solution, same error in grasp pose estimation and action outcomes may keep on recurring. A robust grasp execution plan should be able to decide the best action by taking into account the uncertainty in state estimation and robot control. It should be able to take information gathering actions when uncertainty is high and improve its next action by using the action observation history while reaching the goal.

Existing robust grasp execution approaches either use hand designed controllers or learning or model-based planning. It is difficult to provide a general solution which is suitable for different objects using hand designed controllers. However both learning based and planning based approaches have the potential to provide a general solution for fast and robust grasp execution under uncertainty. While learning based approaches can provide a fast policy, they require large amount of appropriate data and feature representation to deal with uncertainty. Planning based approaches can deal with uncertainty but are very slow and require a system model for a general solution. As we will see in section II, when learning or planning is used separately, it is difficult to fulfill these requirements. This leads to solutions that are applicable to very specific scenarios like grasping only a fixed known set of objects or simple shapes or following suboptimal plans like gripper always going back for re-grasping.

In this work, we combine model-based planning and learning approaches, to provide a general and principled approach for fast and robust grasp execution under uncertainty. We first model a simpler problem with known objects as a partially observable markov decision process (POMDP) (a principled way for planning under uncertainty). Then we use the plans containing information gathering actions generated by POMDP solver as training data to train a deep recurrent neural network policy for general, fast and robust autonomous grasping.

For model-based planning, one of the the main challenges is to define a system model which is very hard to specify for complex shaped objects. In our approach, we sample a few objects from distribution of objects and create a model for only sampled objects through simulation. Using practical

[1]Garg, Hsu and Lee are with School of Computing, National University Of Singapore `A0117042@u.nus.edu`, `(dyhsu,leews)@comp.nus.edu.sg`

[2]Garg and Hsu are with NUS Graduate School for Integrative Sciences and Engineering

[1]`https://youtu.be/X0l_XF_3nvM`

POMDP solvers like DESPOT [1], we can solve this simpler problem of grasping an unknown object selected from a set of known objects under pose and shape uncertainty. Theoretically, if there exists a small policy to successfully grasp the sampled objects under uncertainty, then we should be able to generalize to whole distribution of objects by planning only for a sample of objects (See [2]).

Another key issue with model-based planning under uncertainty is that solving a POMDP is computationally intensive which leads to slow decision making. To make decisions quickly, we need to avoid doing forward search at every step and make decisions based on previous experience. For this we do imitation learning. In this way we don't need the complete model for all the objects and can still get a fast and general policy for robust grasp execution under uncertainty.

For learning, one of the main challenge is gathering large amount of relevant data. To learn a policy which can succeed under uncertainty, along with large amount of data, another key consideration is that the data should contain enough examples of information gathering actions when uncertainty is high. This is difficult to provide through human demonstrations as humans do not perceive the environment like robots. We are able to generate this large amount of relevant data by using POMDP solver traces from grasping of the sampled objects under pose and shape uncertainty.

Using this approach, by modeling only 50 household object instances in simulation, we are able to learn autonomously many long term plans for robust grasp execution which generalize well to objects of varying shapes and sizes that are not modeled. We grasp 100 object instances that are not modeled in simulation and 5 real world objects that are representative of different types of shapes present in modeled objects but are not modeled.

II. RELATED WORK

State of the the art grasping approaches are data-driven ([3], [4], [5], [6], [7]) and aim to compute a grasp pose with high probability of success using vision feedback. Some ([8], [5]) even try to overcome action uncertainty and calibration errors by servoing the gripper to the computed grasp pose. However once the grasp pose with a high probability of success rate is computed and robot has been servoed to the computed pose, grasp execution is still open loop i.e. gripper is moved towards the object and closed to pick the object. Thus grasp failures due to slipping of object out of gripper or accidental hitting of object due to slight misplacements cannot be prevented. In fact they are often reported as main reasons of grasp failure in these works. Such failures have high probability especially when only depth or rgb images are used for grasp pose calculation and servoing, as there can be error in perception of distance from object. Thin objects and objects requiring gripper to move in a gap further increase probability of such failures. Side grasping provides additional challenges as gripper can move the object away by accidentally hitting it. Our work is meant to be used alongside these approaches to prevent and recover from such failures by incorporating additional touch and proprioception feedback during grasp execution.

Existing closed loop grasp execution approaches that take into account sensor feedback like touch, proprioception, optical sensors force feedback etc. during grasp execution can be divided into 3 categories: 1) Hand designed controllers; 2) Learning; 3) Model-based planning. Hand designed controllers ([9], [10], [11], [12], [13], [14]) use a predefined action plan which is dependent on sensor feedback after each action. Thus these approaches are limited in the variety of plans they can employ to prevent or recover from grasp failure. They either do small adjustments near the grasp pose or follow fixed plans to do re-grasping like moving back by a predefined distance or in a predefined direction. We want to obtain such plans autonomously in a principled manner through model-based POMDP planning.

Existing model-based planning approaches work for very simple cases like grasping a specific set of known objects [15] or grasping in a very simple simulated environment [16] due to difficulty in obtaining system model and time required for solving a POMDP. We can scale to large number of novel objects by using sampling and imitation learning.

Existing learning approaches ([17], [18], [19], [20], [21], [22]) mainly use labeled data which maps various near field sensor inputs to grasp success probability to learn simple grasp pose adjustments given sensor feedback. They are either applicable only when gripper is already touching object ([17], [18], [19]) or follow suboptimal plans like gripper always trying to re-grasp object ([20], [21], [22]). Such policies increase the risk of accidentally hitting the object during grasp execution. They have been either demonstrated only on a small set of objects or have low success rate ([22]). These issues can be avoided by learning long term plans which optimally do a trade off between information gathering and completing the task. Generating data for learning such plans is extremely hard. [23] demonstrate successful trajectories for grasping and then learn dynamic motion primitives for robustness. However they are also restricted to small set of objects due to requirement of human demonstration.

III. OUR APPROACH

A. Problem Statement

We want to compute a policy, which provides optimal actions based on our current belief of object pose, object shape and gripper pose for successful grasping. We are provided with a pre-grasp pose for gripper initially and touch sensor feedback, vision feedback, gripper pose and finger joint angles during each action. We first model a simpler problem with known set of objects as a POMDP, a principled way to do planning under uncertainty. In this work, we assume we do side grasping of objects with pose uncertainty in only x (forward/backward) and y (left/right) axis.

B. POMDP modeling

A POMDP is defined by a tuple $<S,A,Z,T,O,R>$ where S is the state space, A is the action space, Z is the observation space. State transition function $T(s,a,s') = p(s'|s,a)$ is probability of next state s' when action a is taken in state s. Observation function $O(s',a,z) = p(z|s',a)$ is probability of observing z in state s' reached by performing action a. $R(s,a)$ is the immediate reward obtained on taking action a in state s. Uncertainty is modeled by maintaining a belief b, which is probability distribution over S. Solution to POMDP is a policy $\pi : B \rightarrow A$ which maps belief $b \in B$ to an action $a \in A$ such that the expected total discounted reward $V_\pi(b)$ defined below is maximized.

$$V_\pi(b) = E\left(\sum_{t=0}^{\infty} \gamma^t R(s_t, \pi(b_t)) | b_0 = b\right)$$

where γ is the discount factor and b_0 is the initial belief.

1) State: To model grasp execution as a POMDP, we define state space S as $< Obj_{id}, Obj_{pose}, G_{pose}, F >$ where Obj_{id} is object id and determines object shape. Obj_{pose} and G_{pose} are object and gripper 2D position w.r.t world frame and F represents the finger joint angles. For out robot, $F =< J_1, J_2 >$, where J_i represents each finger's joint angle.

2) Action: To keep computation requirements minimal, we assume that gripper only moves forward/backward (x-axis) or sideways (y-axis) to grasp objects from side. Thus we define action space A as a set of 11 actions: Move until touch forward/backward/left/right by 1cm or 8cm, close gripper, open gripper, pick object. To grasp from any direction, the action space can be extended by including actions like move up/down or rotate the gripper. This will only increase the resources required for modeling objects but will not change our approach fundamentally. We use one small movement action (1cm) and one large movement action (8cm) in each direction as small movement actions help in gripper pose adjustment around object and large movement actions reduce the length of optimal action sequence which shortens the search horizon for POMDP solver. 8cm is defined based on our gripper workspace size of $20cm \times 16cm$ and can be increased for larger workspace.

3) Observation: We assume that G_{pose} and F are fully observable due to accurate proprioception. However Obj_{id} and Obj_{pose} are unknown and can only be estimated by sensor observations. If touch sensors are reliable and provide rich information, touch feedback alone should be sufficient to estimate object pose and shape. However our touch sensors are very noisy and are present only on gripper finger tips which often leads to pushing of object out of workspace. Therefore we use an additional binary vision observation which detects object movement based on the difference in point cloud of object at the start of an action and during an action. Thus Z is defined as $< G_{pose}, F, T, OM >$. $T =< T_1, T_2 >$, where T_i is the touch sensor value of finger i. OM is 1 if object moved.

4) State Transition And Observation Model: Obtaining a state transition and observation model is in general a hard problem. However since we need to obtain this model for only a small set of known objects, we can record the next state and observation values for different state action pairs using simulation. To explain this process, we first define grasp pose and object grasp pose.

a) Grasp Pose: Since we are considering uncertainty in only x any y axis, we calculate the grasp pose for side grasp using a simple heuristic for simulation. First we fix the height g_z of the gripper. Then we find the closest point (c_x, c_y, c_z) on the object in x axis, at the height of gripper i.e. $(|g_z - c_z| <= 0.005)$. This point is our grasp pose. Ideally we should place the gripper at $(c_x - 0.13, c_y, g_z)$. However because of gripper workspace restrictions, we always place gripper at same starting position (g_x, g_y, g_z) and place the object at a position (o_x, o_y, o_z) (we call this position object grasp pose) such that $(c_x, c_y, c_z) = (g_x + 0.13, g_y, g_z)$. Thus in our experiments, grasp pose defines position of object instead of position of gripper (See Fig. 1).

We place each object at its object grasp pose and gripper at 320 positions (each position is 1 cm apart in $20cm \times 16cm$ gripper workspace) around it and perform all the actions to generate $< s, a, s', z >$ tuples which are stored in a table. From each location, we also perform 2 action sequences consisting of one of the 8 move actions as first action and one of the 11 actions as second action. Thus we perform 11*(1+8) = 99 actions from each location. We need to perform 2 action sequences while collecting data because when initial position of gripper is colliding with object, simulation shows unrealistic behavior. Therefore we cannot record data on those positions. By performing 2 step actions, we are able to move gripper close to object through first action.

Using this data, we can generate s' approximately according to $p(s'|\tilde{s}, a)$ by matching $\tilde{s}$ with s of stored $< s, a, s', z >$ tuples. For fast matching, we discretize states in a 2d grid with 1cm resolution based on (x, y) coordinates of the relative position of gripper w.r.t object for each action. Thus each bin consists of the $< s, a, s', z >$ tuples whose s, a has been mapped to that bin. We find a matching tuple for $\tilde{s}, a$ by probabilistically selecting a tuple from the bin $\tilde{s}, a$ gets mapped to. The probability of a tuple $< s, a, s', z >$ matching to $\tilde{s}, a$ is $\propto \left(\frac{1}{2}\right)^{\alpha d(s, \tilde{s}|a)}$ where α is a parameter defined as 4 and $d(s, \tilde{s}|a)$ is the distance between states s and $\tilde{s}$ defined as follows for different actions:

Pick: Euclidean distance between gripper finger joint angles
Close/Open: Euclidean distance between the relative 2d positon of gripper w.r.t object
Move in x: Euclidean distance with relative gripper position in y axis and absolute gripper position in x-axis
Move in y: Euclidean distance with relative gripper position in x axis and absolute gripper position in y-axis

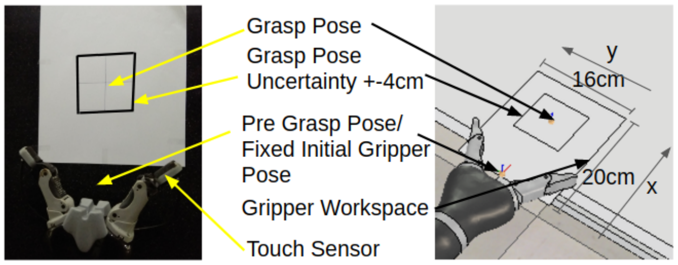

Grasp Pose
Grasp Pose Uncertainty +-4cm
Pre Grasp Pose/ Fixed Initial Gripper Pose
Gripper Workspace
Touch Sensor
y
16cm
20cm
x

Fig. 1: Real And Simulated Robot Setup. Kinect Sensor is also used but not shown.

We use absolute position for one dimension in move action so that we can do better matching near gripper workspace boundaries. For pick action, we match the finger joint angles as they are important for correctly picking an object.

Similarly, we can generate z according to $p(z|s',a)$ by matching $\tilde{s'}$ with the s' of tuples $< s,a,s',z >$. As gripper position is fully observable, we only need T and OM observations for which gripper workspace is not an issue. Therefore for all actions we take the euclidean distance between the relative 2d position of gripper w.r.t object to find the nearest neighbor in the bin.

If there is no data point in the bin, we assume that object is very far from the gripper as we cover all the positions close to object during data collection. Therefore we use a default function which moves gripper as much as intended and gives a default observation of no touch and no movement detection for move and open actions. For close action, default function closes the gripper fully and gives a default touch sensor observation determined by taking average of touch sensor observations when gripper is fully closed. For pick action, default function outputs an unsuccessful pick.

5) Reward: We give a reward of -1 for each step to encourage small trajectories. If touch is detected, an additional reward of +0.5 is given to encourage gripper to remain near object. A large positive reward of 100 is given for successfully picking object. A large negative reward of -10 is given if the object falls or is pushed out of workspace. To narrow the search, we give penalty of -100 on performing move actions after a close gripper action. To avoid actions which lead to no state change, we give penalty of -100 for actions like open action when the gripper is already open or move actions that result in no movement when the gripper is at the workspace boundary. This is important for searching for best action efficiently.

6) Belief And Belief Update: Since our state space is continuous, we represent our belief as a finite set of state instances (particles) weighted according to the probability of them being true state. For initial belief b_0, we sample each object with equal probability and for a given object, calculate the object pose using uniform distribution with $\pm 4cm$ uncertainty around the object grasp pose. Since gripper is assumed to be placed at the grasp pose, we fix gripper's initial position without loss of generality.

$p(z|s,a)$ is obtained by comparing the true observation z with the observation $\tilde{z}$ generated for the given particle using our stored data. It is defined as $\propto (\frac{1}{2})^{\alpha d(z,\tilde{z})}$ where α is a parameter defined as 5 and $d(z,\tilde{z})$ is the distance between observation z and $\tilde{z}$. We can assume that observations from different sensors are independent of each other given the state. Therefore we calculate the distance between gripper 2D position $d(G_{pose},\tilde{G}_{pose})$, joint angles $d(F,\tilde{F})$, touch sensor observation $d(T,\tilde{T})$ and object movement $d(OM,\tilde{OM})$ individually and define $d(z,\tilde{z})$ as their weighted sum.
$d1 = d(G_{pose},\tilde{G}_{pose})$ is the Euclidean distance between 2D gripper positions. To avoid over fitting to collected data, we set $d1 = 0$ if $d1 < 5mm$ and $d1 = 2$ if $d1 > 1.5cm$
$d2 = d(F,\tilde{F})$ is 0 if gripper is open (determined by action which led to the state) or closed without object (determined by action which led to the state and whether the finger joint values are above a manually defined threshold) according to both F and $\tilde{F}$. $d(F,\tilde{F})$ is 2 if gripper is closed with object according to F and closed without object according to $\tilde{F}$ or vice versa. If gripper is closed with object according to both F and $\tilde{F}$, then $d(F,\tilde{F}) = \frac{|J_1-\tilde{J}_1|+|J_2-\tilde{J}_2|}{2}$.
$d3 = d(T,\tilde{T}) = \frac{|T_1-\tilde{T}_1|+|T_2-\tilde{T}_2|}{2}$.
$d4 = d(OM,\tilde{OM}) = |OM - \tilde{OM}|$
$d(z,\tilde{z}) = \frac{w1*d1+w2*d2+w3*d3+w4*d4}{w1+w2+w3+w4}$
We set $w1 = 2$, $w2 = 1$, $w3 = 4$, $w4 = 2$. If action is close gripper, we set $w2$ to 2 and $w4$ to 1.

We can use such a manually designed distance to compare two observations for generating observation probability as belief update only needs approximate probability values which can give particles close to true state higher weight.

C. Imitation Learning

To learn a policy from the execution traces provided by DESPOT after solving the above POMDP for grasping known set of objects amid pose and shape uncertainty, we use 2-layered deep recurrent neural network (RNN). RNNs have an inductive bias that works well with sequence data and may potentially generalize to whole object distribution better than the policy tree used by DESPOT. Each RNN layer consists of 128 hidden units. The input to RNN consists of gripper pose, gripper finger joint values, touch sensor observation, vision movement observation and observed object class vector. Output is 11 dimensional for 11 actions.

IV. EXPERIMENTS

A. Robot Setup

We conducted experiments using Kinova Mico Arm. The gripper consists of 2 under-actuated fingers which adapt to the shape of object while grasping. Therefore we can grasp a variety of objects using side grasp. Each finger is controlled by one joint and has one numatac touch sensor on its tip which provides real number touch values (See Fig. 1). We make it binary by setting a threshold for POMDP planning. For OM, depth point cloud is obtained from Kinect sensor. We compare the depth point cloud above gripper height to filter out gripper movement and detect only object movement. For simulations, we use vrep simulator ([24]). We are able to map real numatac sensor touch values to vrep touch sensor values through a linear transformation. Gripper can move using cartesian control in a $20cm \times 16cm$ area.

As we are interested in seeing how we can prevent/recover from grasp failure when calculated grasp pose can be incorrect, we make the process of how the grasp pose was calculated and how the error was generated irrelevant. We place the object relative to gripper such that gripper is at the correct grasp pose (correct grasp pose is easy to determine for simulation and manually determined for real objects for sake of experiments) and then move the object in a small range ($\pm$4cm in our work) to simulate the error in grasp pose. Thus without loss of generality, we can fix the initial position of robot at (x=0,y=0.07), with (0,0) at the bottom right corner of gripper workspace. Then we calculate the true position of object relative to it as explained in section III-B.4.a.

B. Baseline

As we place gripper at true grasp pose w.r.t object, our baseline is to move gripper forward by a fixed distance, close the gripper and pick the object. Since the amount of distance to move forward can be different for various objects, we use 7 baseline policies which differ in the amount of forward movement (ranging from 10cm to 16cm). For each object instance, we choose the best performing baseline. Thus grasping success rate for baseline is the average of the success rate of best performing baseline associated with each different object instance.

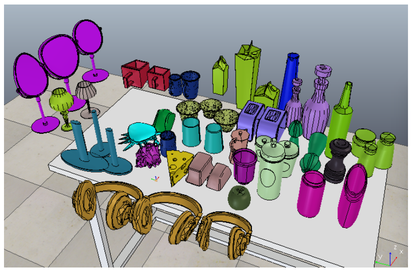
C. Experiments with Household objects in Simulation

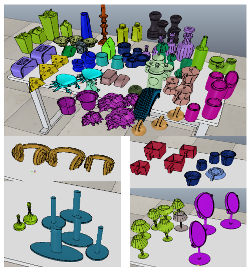
We use G3DB object dataset ([25]) to simulate household objects in vrep simulator. It has 92 types of objects, out of which 41 were side graspable and had stable simulation. Each object type can have many instances. We have 150 object instances from 41 object types for experiments.

Ideally we can consider all 150 household objects as one object class. However keeping all objects in one class makes policy complex as many information gathering actions need to be performed for differentiating between different objects like cylindrical objects or objects with 2 lobes. For estimating grasp stability before pick, we depend on only joint angle feedback. This is not sufficient to determine whether object is inside the gripper or not for very thin objects like cup handles or wine glasses. With a richer sensor like force feedback or touch with larger area or vision, it should be possible to put all objects in one class. In the current work, we divide the 150 object instances into 5 object classes: 1) Power grasp objects; 2) Pinch grasp objects; 3) Cup With handles; 4) Objects with thin sticks but support on top; 5) Objects with 2 lobes. Fig. 2b shows various objects in object classes. To automatically determine object class, we train a convolutional neural network which maps depth image to object class. Thus the object class observation is a vector with size as number of object classes and element i giving the probability of object belonging to class i. To add the prior about object class, the first action is always to get the object class observation. For belief update, we define $p(z|s,a)$ as the dot product of observed object class vector and true object class vector (which is 1 for true object class, 0 otherwise).

We randomly sample approximately 1/3 of objects (minimum 3) from each object class to form a set of known objects (See Fig. 2a). We grasp these modeled objects placed at different positions generated by uniform distribution with $\pm 4cm$ interval in x and y axis around object grasp pose for pose uncertainty using DESPOT. Then we train a 2 layer deep RNN to learn a fast and robust grasping policy from the generated action observation sequences.

For experiments, we place the object to be grasped at 81 different positions in x and y axis which form a grid of $\pm 4cm$


(a) 50 Training Objects


(b) Object Classes

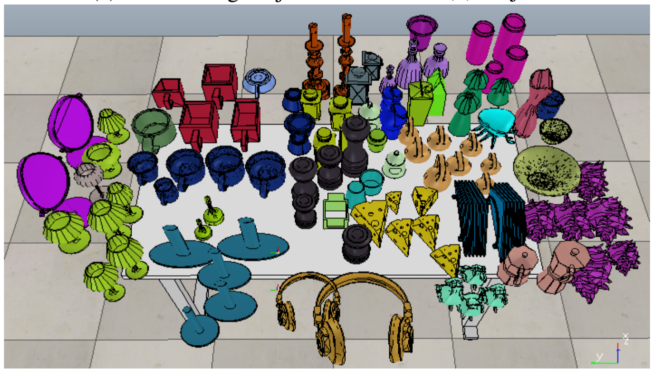

(c) 100 Test Objects

Fig. 2: b) Top: Power grasp objects; Middle Left: 2 lobe object; Middle Right: Cups With Handle; Bottom Left: Pinch grasp objects; Bottom Right: Objects with support on top

around object grasp pose. Then we try to grasp the object using baselines, despot and the learned policy. Table I shows the grasping success rate for the 50 modeled object instances (train) and 100 object instances that were not modeled (test) (Fig. 2c). We can see that we are able to significantly improve the grasping success rate of baseline for both training and test set of objects. Learned policy generalizes slightly better than the despot policy while takes only 0.9-1.5ms to compute next action as compared to 5 seconds taken by DESPOT.

D. Grasping With Real Robot

Next we conduct experiments on real robot. We choose 5 side graspable objects representative of different shapes in training objects as shown in Fig. 4. As in the simulator, we fix the true grasp pose for objects at 13cm from the fixed gripper position and then place the object at $\pm 4cm$ in x and y axis around object grasp pose (9 positions) (See Fig. 1). Then we try to grasp it using baselines, DESPOT and learned policy. DESPOT and learned policy are the same as the one obtained by modeling 50 simulation objects. We provide object class observation manually for real objects as our object class classifier was trained on simulator data. If we have enough real objects, we can train such a classifier on real objects also and automatically determine object class. Table II shows the grasping success rate for each of the 5 objects. We can see that we are able to significantly improve the grasping success rate of the baseline for real objects also.

E. Analysis

We see some interesting policies that are generated by DESPOT and learned by deep RNN. For power grasp objects,

TABLE I: Grasp success rate in simulator

Object	Baseline	Despot	Learned
Train (50 objects)	65.5%	82.1%	**83.4%**
Test (100 objects)	65.6%	71.7%	**73.8%**

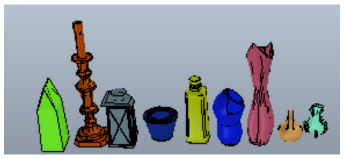


Fig. 3: Simulation Object Types Not Modelled

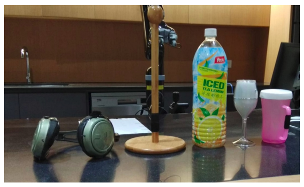


Fig. 4: Real Objects

policy is to move forward as much as possible before closing the gripper and re-grasp if object slips. If touch is detected, then gripper tries to get around object and avoids hitting it. (See learned policy grasp execution for bottle in video[1]). These simple policies generalize well for different objects. For example the object types shown in Fig. 3 have no instances in training objects. But they are still grasped successfully by the DESPOT and learned policy.

For pinch grasp objects, the policy is to localize object by moving left and right after moving forward and then move backward so that object comes in gripper fingers (See learned policy grasp execution for kitchen towel stand in video[1]). To move backward, learned policy takes two to three consecutive move back actions which shows that it has learned a long term plan. This policy should also generalize well for pinch grasps. Though in real arm experiments, we get many failures for kitchen towel stand object because joint angles of real arm are not able to differentiate between gripper closed with or without object for thin objects. So we have to rely on very noisy touch sensor feedback to determine this. With a better joint angle feedback we should be able to differentiate between gripper closed with or without object and this policy should work well.

For objects with 2 lobes like headphone also, policy is to localize the object by moving left and right and then try to move into the gap by slightly pushing the object. (See learned policy grasp execution for headphone in video[1]). Here also we see long term plans being learned when gripper tries to push object slightly thrice, when not able to move into the gap during execution 2 in video[1].

For objects with support on top, policy is to simply move forward by 16cm and close the gripper which gives almost 100% success rate. Ideally these objects should be merged with pinch grasp objects but because of their extremely thin sticks they have to be kept separate. For cup with handles, learned policy is very similar to power grasp objects. This policy works because when the cup is grasped from the main part instead of handle, it slips and handle comes in gripper fingers resulting in successful grasp. This is also seen in real robot experiments with cup with handle.

The main reasons of grasp failures in our work are: 1) Objects going out of reach of gripper when they are shifted forward as this puts them close to gripper workspace boundary. This is the main reason of failure for bottle and

TABLE II: Grasp success rate with real arm

Object	Baseline	Despot	Learned
Lemon tea bottle	66.7%	77.8%	**88.9%**
Kitchen towel stand	33.3%	**88.9%**	66.7%
Headphone	66.7%	**88.9%**	**88.9%**
Wine glass	100%	**100%**	**100%**
Cup With Handle	100%	88.9%	**100%**
Total	73.3%	**88.9%**	**88.9%**

headphone in real arm experiments. 2) Objects with small height and tapering shape or thin objects have grasping failure in simulator even when grasp is correct. 3) Joint angles are weak indicators of grasp stability. This leads to many failures specially for pinch grasp objects. 4) Objects like cups with handles rotate and the movement is not detected. This behavior is currently not modeled. While 1) can be avoided with larger robot workspace, 2) is just the artifact of simulator and 3) can be avoided by using force feedback. For 4) object rotation needs to be modeled which requires richer vision observation. This is difficult with state of the art POMDP solvers as it requires dealing with large observation spaces. We leave it for future work.

V. CONCLUSION AND FUTURE WORK

We showed that we can make autonomous grasping more accurate under uncertainty by modeling grasp execution as a POMDP for only a small sample of objects and then using the execution traces of POMDP to train a very fast policy for robust grasp execution under uncertainty which also generalizes to other objects of the object class that were not modeled. The learned policy takes 0.9-1.5 ms to compute the next action[2] which is comparable to open loop execution while planner policy takes 5 seconds to compute the next action although the success rate of learned policy is slightly better than the planner policy and significantly better than the open loop execution.

There are various ways in which we can further improve the grasping success rate. One way is to increase the variety of actions that can be performed. If we can move the gripper in z-axis then we can better deal with objects of small height. Similarly if we can rotate gripper then we can better deal with objects that rotate with gripper movement. Another option is to incorporate a richer vision feedback. This can make our policy simpler and can be helpful in modeling more complex object movements like rotation and tilting. This can also alleviate the need for specifying object classes and enable us to model uncertainty in object movement due to unknown object properties like center of mass and friction for successful grasping. Currently we do not vary these properties as they cannot be determined by touch, proprioception and basic vision feedback. Another interesting direction is to improve the learned policy online through user feedback if it does some undesired actions.

ACKNOWLEDGMENT

This work was supported by Singapore Ministry of Education AcRF grant MOE2016-T2-2-068.

[2]Note that the execution of learned policy in video appears slow because our system for sensor feedback gathering is not optimized.

References

[1] A. Somani, N. Ye, D. Hsu, and W. S. Lee, "Despot: Online pomdp planning with regularization," in *Advances in Neural Information Processing Systems 26*, C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, Eds. Curran Associates, Inc., 2013, pp. 1772–1780. [Online]. Available: http://papers.nips.cc/paper/5189-despot-online-pomdp-planning-with-regularization.pdf

[2] Y. Wang, K. Won, D. Hsu, and W. Lee, "Monte Carlo Bayesian reinforcement learning," in *Proc. Int. Conf. on Machine Learning*, 2012.

[3] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-driven grasp synthesis-a survey," *IEEE Transactions on robotics*, vol. 30, no. 2, pp. 289–309, 2014, qC 20140611.

[4] L. Pinto and A. Gupta, "Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours," in *2016 IEEE International Conference on Robotics and Automation, ICRA 2016, Stockholm, Sweden, May 16-21, 2016*, 2016, pp. 3406–3413. [Online]. Available: https://doi.org/10.1109/ICRA.2016.7487517

[5] K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige, S. Levine, and V. Vanhoucke, "Using simulation and domain adaptation to improve efficiency of deep robotic grasping," *CoRR*, vol. abs/1709.07857, 2017.

[6] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg, "Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics," 2017.

[7] J. Mahler, F. T. Pokorny, B. Hou, M. Roderick, M. Laskey, M. Aubry, K. Kohlhoff, T. Kröger, J. Kuffner, and K. Goldberg, "Dex-net 1.0: A cloud-based network of 3d objects for robust grasp planning using a multi-armed bandit model with correlated rewards," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 1957–1964.

[8] U. Viereck, A. ten Pas, K. Saenko, and R. P. Jr., "Learning a visuomotor controller for real world robotic grasping using simulated depth images," in *CoRL*, ser. Proceedings of Machine Learning Research, vol. 78. PMLR, 2017, pp. 291–300.

[9] K. Hsiao, S. Chitta, M. T. Ciocarlie, and E. G. Jones, "Contact-reactive grasping of objects with partial shape information." in *IROS*. IEEE, 2010, pp. 1228–1235. [Online]. Available: http://dblp.uni-trier.de/db/conf/iros/iros2010.html\#HsiaoCCJ10

[10] S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *CoRR*, vol. abs/1603.02199, 2016. [Online]. Available: http://arxiv.org/abs/1603.02199

[11] A. Leeper, K. Hsiao, E. Chu, and J. Salisbury, "Using near-field stereo vision for robotic grasping in cluttered environments," in *Experimental Robotics*, ser. Springer Tracts in Advanced Robotics, O. Khatib, V. Kumar, and G. Sukhatme, Eds. Springer Berlin Heidelberg, 2014, vol. 79. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-28572-1_18

[12] K. Hsiao, P. Nangeroni, M. Huber, A. Saxena, and A. Y. Ng, "Reactive grasping using optical proximity sensors." in *ICRA*. IEEE, 2009, pp. 2098–2105. [Online]. Available: http://dblp.uni-trier.de/db/conf/icra/icra2009.html\#HsiaoNHSN09

[13] S. Dragiev, M. Toussaint, and M. Gienger, "Uncertainty aware grasping and tactile exploration." in *ICRA*. IEEE, 2013, pp. 113–119. [Online]. Available: http://dblp.uni-trier.de/db/conf/icra/icra2013.html\#DragievTG13

[14] N. Vahrenkamp, S. Wieland, P. Azad, D. I. Gonzalez-Aguirre, T. Asfour, and R. Dillmann, "Visual servoing for humanoid grasping and manipulation tasks." in *Humanoids*. IEEE, 2008, pp. 406–412.

[15] K. Hsiao, L. P. Kaelbling, and T. Lozano-Pérez, "Robust grasping under object pose uncertainty," *Autonomous Robots*, vol. 31, no. 2–3, 2011. [Online]. Available: http://lis.csail.mit.edu/pubs/tlp/HsaioAutonomousRobots11.pdf

[16] K. Hsiao, L. P. Kaelbling, and T. Lozano-Pérez, "Grasping POMDPs," in *IEEE International Conference on Robotics and Automation*, 2007. [Online]. Available: http://lis.csail.mit.edu/pubs/tlp/pomdp-grasp-final.pdf

[17] J. Steffen, R. Haschke, and H. Ritter, "Experience-based and tactile-driven dynamic grasp control." in *IROS*. IEEE, 2007, pp. 2938–2943. [Online]. Available: http://dblp.uni-trier.de/db/conf/iros/iros2007.html\#SteffenHR07

[18] H. Dang and P. K. Allen, "Stable grasping under pose uncertainty using tactile feedback," *Auton. Robots*, vol. 36, no. 4, pp. 309–330, 2014. [Online]. Available: http://dx.doi.org/10.1007/s10514-013-9355-y

[19] H. Dang and P. K. Allen, "Grasp adjustment on novel objects using tactile experience from similar local geometry," pp. 4007–4012, 2013. [Online]. Available: http://dx.doi.org/10.1109/IROS.2013.6696929

[20] Y. Chebotar, K. Hausman, Z. Su, G. Sukhatme, and S. Schaal, "Self-supervised regrasping using spatio-temporal tactile features and reinforcement learning," in *International Conference on Intelligent Robots and Systems (IROS) 2016*, 2016.

[21] 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," *CoRR*, vol. abs/1805.11085, 2018.

[22] A. Murali, Y. Li, D. Gandhi, and A. Gupta, "Learning to grasp without seeing," *CoRR*, vol. abs/1805.04201, 2018.

[23] P. Pastor, L. Righetti, M. Kalakrishnan, and S. Schaal, "Online movement adaptation based on previous sensor experiences," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2011)*, 2011. [Online]. Available: http://www-clmc.usc.edu/publications/P/pastor-IROS2011

[24] M. F. E. Rohmer, S. P. N. Singh, "V-rep: a versatile and scalable robot simulation framework," in *Proc. of The International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[25] A. Kleinhans, B. S. Rosman, M. Michalik, B. Tripp, and R. Detry, "G3db: A database of successful and failed grasps with rgb-d images, point clouds, mesh models and gripper parameters," 2015.