Robotics: Science and Systems 2021
Held Virtually, July 12–16, 2021

Ab Initio Particle-based Object Manipulation

Siwei Chen, Xiao Ma, Yunfan Lu and David Hsu
National University of Singapore

Reconstructed particle representation of some household objects

Grasping Pushing Placing

Fig. 1: PROMPT constructs a particle representation for object manipulation.

***Abstract*—*Particle-based Object Manipulation* (PROMPT) is a new method for robot manipulation of novel objects, without prior object models or pre-training on a large object data set. The key element of PROMPT is a particle-based object representation, in which each particle represents a point in an object, the local geometric, physical, and other features of the point, and also its relation with other particles. The particle representation connects visual perception with robot control. Like data-driven methods, PROMPT infers the object representation online in real time from the visual sensor. Like model-based methods, PROMPT leverages the particle representation to reason about the object's geometry and dynamics, and choose suitable manipulation actions accordingly. PROMPT thus combines the strengths of model-based and data-driven methods. We show empirically that PROMPT successfully handles a variety of everyday objects, some of which are transparent. It handles various manipulation tasks, including grasping, pushing, etc.. Our experiments also show that PROMPT outperforms a state-of-the-art data-driven grasping method on everyday household objects, even though it does not use any offline training data. The code and a demonstration video are available online**[1]

I. Introduction

Object manipulation is a central question of robotics research. It embodies a broad spectrum of challenges from robot design to sensing and control algorithms. One key challenge of robot manipulation systems is object representation, which connects robot sensing and control. Take, for example, grasping. Classic model-based methods capture the object's geometric and physical properties in a compact analytic model, which enables sophisticated reasoning about the grasp quality, through quasi-static or dynamic analysis. However, acquiring such an object model from sensor data is often difficult. Also, simplifying assumptions and inaccuracies in the models often result in brittle grasps that fail during the robot execution. In contrast, data-driven methods learn the grasps directly from prior experiences. They focus on the perception of object features critical for predicting good grasps and are often more robust in practice. The data-driven methods, however, find difficulty in generalizing to novel objects, unseen in the training data. This work aims for a general, flexible object representation that enables both effective construction from sensor data and sophisticated reasoning of geometry and dynamics for manipulation, thus achieving the benefits of both data-driven and model-based methods.

Specifically, we introduce *Particle-based Object Manipulation* (PROMPT). The key element of PROMPT is a particle-based object representation, in which each particle represents a point in the object, the local geometric, physical, and other features of the point, and also its relation with other particles. The particle representation is general and approximates objects with diverse geometric shape and visual appearance (Fig. 1). Given a set of RGB images of an object from multiple camera views, PROMPT first constructs a particle representation of the object *online* in real time. For each camera view, it projects the particles into the image plane and matches with the foreground-segmented object in the image, using an improved Chamfer distance measure. PROMPT then uses the reconstructed particle set as an approximate state representation of the object for reasoning. It performs particle-based dynamics simulation to predict the effects of manipulation actions on the object and chooses the desired actions through model predictive control. PROMPT is capable of a variety of manipulations tasks: grasping, pushing, placing, ... (Fig. 1).

Our experiments show that PROMPT successfully constructs the particle representation for a wide variety of everyday household objects, including some that are transparent and particularly challenging for reconstruction from visual sensor data. PROMPT outperforms Dex-Net 2.0 [30], a state-of-the-art data-driven method for object grasping. It compares favorably with previous work [27] on object pushing. Finally, it manipulates an object with unknown mass distribution, grasps it, and balances it on a small stand.

PROMPT is a simple idea, but works surprisingly well. It

[1]The code and a demonstration video are available online at https://adacomp.comp.nus.edu.sg/2021/06/26/particle-based-robot-manipulation.

benefits significantly from the particle representation, which connects visual perception with dynamics reasoning. The particle representation is simple and general. It allows us to leverage the dramatic progress in particle-based dynamics simulation. Compared with model-based methods, PROMPT learns the object representation automatically from visual sensor data. Compared with data-driven methods, it requires no offline training data and "generalizes" to novel objects trivially. PROMPT also generalizes across many different rigid-body manipulation tasks.

II. Related Works

A. Manipulation with 3D Mesh Model

Classical analytic methods for object manipulation often assume a known 3D mesh model of an object and evaluate some specific metrics for action planning [43]. For example Pokorny and Kragic [37] extends the popular L^1 grasp quality measure that considers the ability to resist wrenches in object grasping tasks. Numerous works follow this approach to retrieve the object's 3D mesh model, estimate the pose and plan for actions [9, 15, 40, 19, 4]. However, given an unseen novel object with no available mesh model, classical methods often fails. To fix this issue, 3D mesh reconstruction with laser scanners are introduced [31, 32, 3, 2]. They consider an object as a combination of a construction of primitive parts, e.g.boxes and cylinders. Nonetheless, generalizing 3D mesh-based models to arbitrary shaped and translucent objects remains challenging.

B. Manipulation with Latent Representation

Latent state representation closely connects to modern data-driven approaches. Sensory observations are encoded into latent states and directly mapped to a robot policy by a parameterized function, which is trained from pre-collected labeled data [30]. Pioneer works use human labelled data for training [1, 21, 16, 25]. Self-supervised trial and error on real robots provide an alternative to human data labeling [33, 34]. Large-scale real robot experiments [35, 26, 20] are conducted to collect thousands of grasping experiences across few months. In contrast to training a policy with real-world data, other works train policy in simulators [47, 45]. Specifically, Dex-Net 2.0 [30] trains a grasp quality CNN network over a 6.7 million synthetics depth images grasping dataset and achieves 80% success rate without any real-world data. Similarly, PushNet [27] learns a pushing policy entirely in simulation and succeeds in all real robot pushing tasks.

Despite the success of data-driven approaches, the data-driven approaches still suffer from the following limitations: 1) Data gathering: data gathering in the real world is expensive and time-consuming. 2) Distribution Shift: the testing data may differ significantly from the training data, especially when the policy is trained in a simulator and applied in the real world. 3) Dynamics reasoning: the trained policy often lacks explicit dynamics reasoning and may result in sub-optimal actions.

C. Particle Representation & Reconstruction

Structure From Motion (SFM) [22] tracks the feature correspondences cross multi-view images and reconstruct a scene point cloud. However, the reconstructed point cloud is sparse and incomplete as the point cloud relies on tracked features that can be very sparse and noisy. Other approaches [29, 41, 42] related to multi-view stereo achieve high accuracy and completeness in reconstruction but require time ranging from minutes to even hours and is hence not practically useful in real robot tasks. Shapes from silhouettes (SFS) performs reconstruction from multi-view silhouettes [23, 24]. However, typical SFS, such as the polyhedral approach, often requires perfect silhouettes and problems arise when silhouettes contain errors [11, 44]. Moreover, it does not scale well as the number of images increases. In our experiments, we need to process dozens to hundreds of images online, and the masking images often contain errors.

We propose a new online approach for constructing an particle representation and use it for model-based planning. Instead of reconstructing the 3D mesh model using existing approaches that can be slow, our method can quickly achieve good point cloud reconstructions on various objects. On top of that, particle representation connects the visual perception to the known particle-based dynamics model and enables explicit reasoning for different robot actions. Furthermore, particle representation is general and has better potentials. For example, it can model deformable objects and liquid such as a bowl of porridge and a cup of coffee by using different types of particles.

III. Particle-based Object Manipulation

A. Overview

Our approach aims to generate a universal particle state representation for ab initio rigid object manipulation, such as grasping, pushing, and placing tasks. The input to the system is defined as a set of tuples (M_i, E_i), where M_i is the i_{th} masking image and E_i is the i_{th} camera extrinsic matrix. The camera's intrinsic matrix is known. We also assume we can have a sufficient coverage of views on the object and each object is manipulated in isolation. We define particle state below:

Definition 1. ***Particle*** *We define a particle as* $v = (x, y, z, \dot{x}, \dot{y}, \dot{z}, m)$ *with position* x, y, z*, velocity* $\dot{x}, \dot{y}, \dot{z}$*, and mass* m*. A particle represents a point of the object, which distributes on either the surface or the interior of an object.*

Definition 2. ***Particle State*** *We define a particle state as* $S = (V, R, \mu)$*, where* V *is the set of object particles defined above and* R *is a set of direct edges. Each edge* $r = (v_1, v_2, a)$ *represents the relations* a *from particle* v_1 *to* v_2 *(e.g. collision, spring connection).* μ *refers to the system friction coefficient.*

The particle state definition applies to both rigid and deformable objects, e.g., liquid and cloth, by setting different edge types R. In this work, we focus on the rigid objects

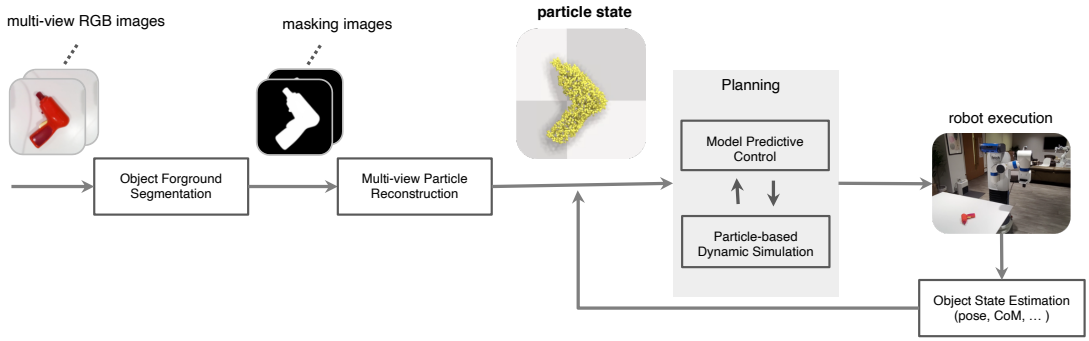


Fig. 2: Overview. Particle Model-based Grasping takes in multi-view RGB images and reconstruct a 3D particle state which is used for planning with model-predictive control with a state-of-the-art particle-based physics engine.

and leave deformable objects for future study. Specifically, we manually set edge types to specify a rigid object.

Given a particle state, a decision making system outputs an action, which can be adjusted for different tasks. For example, in a top-down object grasping task, we define action space as $A = (x, y, z, \theta)$, with the grasping center as (x, y, z) and the anti-clockwise angle θ from the positive x axis. Similarly, for an object pushing task, the action space could be defined as $A = (x_0, y_0, x_1, y_1)$, which refers to the start and end position of a horizontal push action with a fixed height.

With the particle state representation, we factorize a manipulation problem into two consecutive downstream tasks: 1) accurate estimation of a particle state; 2) effective motion planning with the predicted particle state.

We introduce a new approach called PROMPT: Ab Initio Particle-based Object Manipulation. The high-level structure can be found in Fig. 2. The robot first takes RGB images from different view angles, and produces segmentation masks M. Given the masking images, our approach predicts the particle state online, focusing on 3D shape reconstruction for the unseen objects. The predicted particle state is then fed into a particle-based simulator, which predicts the future particle states given the current state and an action. With the particle state and the dynamics model, we perform cross-entropy model predictive control (MPC) [39] to plan for the best action. The system then executes the best action on the real robot. After each real robot execution, we collect new images and use the observation to update the hidden properties of the object, e.g., mass distribution and position.

B. Multi-view Particle State Estimation

Data-driven methods learn to reconstruct point cloud by offline training with labeled data. Reconstructing an accurate point cloud for arbitrary objects is extremely challenging due to the limited available training data. However, we argue that recovering an accurate point cloud online for a specific object is much easier. In practice, we observe that with a small amount data, e.g., a few seconds of video, an accurate point cloud can be estimated.

Our approach differs from the traditional structure from motion (SFM). SFM relies on tracked image features that potentially lead to a sparse point cloud, and many essential geometry details are lost. In contrast, our approach generates a dense point cloud to capture all the critical details for particle-based dynamics modeling. We use N particles in the experiment. The original particle $v = (x, y, z, \dot{x}, \dot{y}, \dot{z}, m)$ consists of 7 attributes. However, at the reconstruction stage, we assume the object's mass is evenly distributed, and the object remains static. We hence simplify a single particle into $\hat{v} = (x, y, z)$. The particle generator function is shown below:

$$G : h \rightarrow \hat{V} \tag{1}$$

where G is the particle generator parameterized by a neural network we aim to train. h is a single number, and $\hat{V}$ are the target particles. A quick idea to train the generator G is to re-project the 3D particles $\hat{V}$ into 2D points on the image plane, and minimize the masking image reconstruction loss. The target particle states can then be achieved by iterative gradient descent with the loss function specified below.

$$\min \sum_{M_i} D(f_{proj}(\hat{V}), M_i) \tag{2}$$

where M_i refers to the i_{th} masking image, f_{proj} projects the particles from 3D into 2D points (x, y) on the image, and D measure the distance between the re-projected masking image and the ground truth masking image. Simple L1 or L2 loss between the re-projected masking images and ground truth masking images fails. It is because there is no gradient flows through the pixel value of the re-projected masking images, and only the coordinates of the mask are computed from $\hat{V}$.

In fact, measuring the distance $D(f_{proj}(\hat{V}), M_i)$ is equivalent to measure the distance between two non-ordered sets $\{p'\}$ and $\{p\}$, where $p'_n = (x'_n, y'_n) = f_{proj}(\hat{v}_n)$ is the n_{th} re-projected point from $\hat{v}_n$ on the image, and $p_n = (x_n, y_n)$ is the n_{th} point sampled from ground truth masking image M_i. Since $\{p\}$ may have a larger size than re-projected set $\{p'\}$, for computational efficiency, we uniformly sampled the

Binary Masking M_i Sampled Points $\{p\}$ Reprojected Points $\{p'\}$

Fig. 3: An example of $\{p\}$ and $\{p'\}$. A cup and a cube lay on the table. Left is one of the ground truth object masking images. The center image shows a set of points $\{p\}$ sampled from left. Right shows a set of points $\{p'\}$ re-projected from particles $\hat{V}$.

same number of the points from M_i to construct $\{p\}$. Figure 3 shows an example of $\{p\}$ and $\{p'\}$.

To effectively compute the loss, Chamfer distance is often used to measure the average distance between two sets. Original Chamfer distance $D_{CD}(\{p'\}, \{p\})$ has the form:

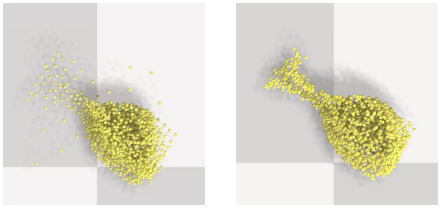
$$\sum_{p' \in \{p'\}} \min_{p \in \{p\}} ||p' - p||_2^2 + \sum_{p \in \{p\}} \min_{p' \in \{p'\}} ||p' - p||_2^2$$

However, we observe that the original Chamfer distance can be stuck in a local minimum. From the definition, each point only finds the closest point from other point sets. This pairing is not unique and can be problematic. For example, in Fig. 4 (a), standard Chamfer distance falls in a local minimum. If $\{p'\}$ only occupies part of $\{p\}$, the first term $\sum_{p' \in \{p'\}} \min_{p \in \{p\}} ||p' - p||_2^2$ has a low loss as every $\{p'\}$ finds a point nearby. However, each $p \in \{p\}$ selects the repeating nearest point in $\{p'\}$ to optimize, and all other points are not involved. After few iterations, $\{p'\}$ squeezes in a local area with very few points spread out.

To address this issue, we proposed a new variant of the Chamfer distance called K nearest neighbors Chamfer loss $D_{kN\_CD}(\{p'\}, \{p\})$, defined in (3). KN selects the K nearest neighbors that belong to the other set. When K is not small, e.g.100, the proposed loss penalizes those points with sparse neighbors and pushes the generated points to have better coverage. Fig. 4 (b) shows an example with $K = 100$ helps to mitigate the local minimum problem.

$$\frac{1}{K} \sum_{p' \in \{p'\}} \sum_{p \in KN} ||p' - p||_2^2 + \frac{1}{K} \sum_{p \in \{p\}} \sum_{p' \in KN} ||p' - p||_2^2 \quad (3)$$

After reconstruction, flying particles are filtered away by a threshold of average neighbour distance due to noisy masking images. Our approach to reconstructing the point cloud has several advantages: (1) It requires no offline dataset. Hence, we save the efforts from collecting data, which is non-trivial, expensive, and time-consuming. (2) Without using any offline dataset, we prevent over-fitting to existing training data. (3) Robust to Translucent Objects. In contrast to the depth sensor that fails on transparent objects, our system works on RGB images to detect the object masking and reconstruct the 3D shape.



(a) $K = 1$ (b) $K = 100$

Fig. 4: A real-world reconstruction example, a wine glasses, with noisy masking images as input. a) When $K = 1$, it is the standard Chamfer distance, and the reconstruction gets stuck in a local minimum. The algorithm always selects repeating nearest points to optimize. b) $K = 100$, the K nearest Chamfer distance mitigates the problem and provides better results.

C. Model-based Planning

Traditional approaches sample a set of action proposals and rank them by handcrafted heuristic metrics. Instead of handcrafting heuristic metrics, we connect the generated particles into the particle-based simulator, building upon a particle-based physics engine. Other existing simulators such as MuJoCo [46], and PyBullet [8] use the 3D mesh representation, which is difficult to obtain in the test time from the real world.

To explain the planning algorithms in detail, we use the grasping task as an example. Our approach leverages the particle-based physics engine to perform the virtual grasping and evaluate the final performance based on the two following criteria: (1) Whether the grasping is successful. We define the success criteria as to whether the object is above a height threshold. (2) How robust the grasp is. This metric is defined by the sum of displacement of all the particles along the x-axis and y-axis. The combined reward function is shown below:

$$\text{Return} = \mathbb{1} * -10^6 - \sum_{(x,y,z) \in \hat{V}} ||x - x_0||_1 + ||y - y_0||_1 \quad (4)$$

where $\mathbb{1}$ is the indicator function which suggests whether the grasp is successful or not, $\hat{V}$ is the set of particles predicted, (x, y, z) refers to the 3D position of a particle, and (x_0, y_0, z_0) is the initial position of the same particle.

Given the predicted particle state, the dynamics model, and the reward function, we perform model-based planning that explicitly reasons the dynamics. For simplicity, we use a cross-entropy model-predictive control (MPC) [39] with a uniform prior based on the predicted particles. Instead of randomly sample from the action space, we initialize actions near the predicted particles with random approaching angles and without collision. In each iteration, the system samples sequences of actions from a Gaussian distribution, compute the accumulative reward, update the mean of the Gaussian distribution with the K action sequences with the highest-achieved reward. The fist action of the best action sequence is used as the output.

D. Object State Estimation

For one-shot object grasping, a open-loop planner is able to give a robust policy with careful tuning. However, it is

Algorithm 1 Grasping Planning with Cross Entropy MPC

Require: H Planning horizon
I Optimization Iteration
J Candidates per Iteration
K Number of Top candidates to Track
$s = \{o\}$ Predicted Particle State
Ensure: The best estimated action

Initialize action $q(a_{t:t+H})$ with a prior distribution
for optimization iteration $i = 1..I$ **do**
  # Evaluate J actions' performance
  for candidate action $j = 1..J$ **do**
    $a^j_{t:t+H} \sim q(a_{t:t+H})$
    $s^j_{t:t+H+1} \sim \prod_{\tau=t+1}^{t+H+1} p(s_\tau | s_{\tau-1}, a^j_{\tau-1})$
    $r^j = \sum_{\tau=t+1}^{t+H+1} R(s^j_\tau, a^j_t au)$
  end for
  # Then select the best K actions to update the action distribution
  $\kappa = \leftarrow argsort(\{r^j\})_{1:K}$
  $u_{t:t+H} = \frac{1}{K} \sum_{k \in \kappa} a^k_{t:t+H}$
  $\sigma_{t:t+H} = \frac{1}{K-1} \sum_{k \in \kappa} |a^k_{t:t+H} - u_{t:t+H}|$
  $q(a_{t:t+H}) \leftarrow Normal(u_{t:t+H}, \sigma_{t:t+H})$
end for
Return the action mean u_t

insufficient for manipulation tasks with consecutive actions and unknown physical properties, e.g., object pushing. Specifically, in a pushing task, the robot has to estimate the position and orientation of the object after each push. Moreover, for an object with uneven mass distribution, e.g., a hammer, correctly estimating the center of mass (CoM) is essential for a successful pushing. PROMPT implements close-loop state estimation by dynamically updating the properties of a particle state through the interactions with the object. As illustrative examples, we demonstrate PROMPT can be applied to object pose estimation and CoM estimation during a pushing task.

1) Closed-loop Object Pose Estimation: We fix the particle state obtained from the reconstruction stage and only varies the pose parameters (x, y, θ), which represents (x, y) position on the table and orientation θ. The estimation is done by finding the best pose parameters to minimize the masking image reconstruction loss on new observed image. Similar to CE MPC, we use cross entropy estimation to search for the best pose. With the particle representation, many other optimization techniques can also be used such as particle filter, Bayesian optimization and gradient descent.

2) CoM Estimation: We define the CoM as a Gaussian distribution centered at (x_c, y_c) with variance Σ. Similar to cross-entropy MPC Algorithm 1, cross-entropy (CE) Estimation is used to find the best parameters (x_c, y_c) and the variance Σ that minimizes the gap between the predicted particle state from the particle simulator and the observed state from the real world. In each iteration, the system samples a set of (x_c, y_c) and passes it into the particle-based simulation with all other things fixed, such as the particle states and actions performed. We compare the simulated results with the observed object configuration to select the top candidates. After that, the system recomputes the mean and variance of the CoM Gaussian distribution based on the top candidates and starts a new iteration.

TABLE I: 3D Reconstruction Results

	Other Baselines		PROMPT (Ours)		
	Point [10]	3D-R2N2 [7]	CNN-100	FC-100	FC-1
Plane	0.601	0.561	0.718	0.692	0.664
Lamp	0.462	0.421	0.525	0.467	0.438
Fire arm	0.604	0.600	0.721	0.701	0.589
Chair	0.544	0.550	0.521	0.492	0.486
Watercraft	0.611	0.61	0.686	0.680	0.645
Speaker	0.737	0.717	0.509	0.540	0.587
Couch	0.708	0.706	0.502	0.546	0.606
table	0.606	0.580	0.517	0.490	0.493
Average IoU	0.609	0.593	0.587	0.576	0.563

Fig. 5: 3D Reconstruction Examples. We evaluate our reconstruction performance on the ShapeNet [6] dataset. There are three variants of our approach. PROMPT CNN-100 uses a CNN encoder structured generator with K=100 for K nearest Chamfer distance. FC-100 uses a generator that only consists of a fully-connected layer. FC-1 is an FC layer generator with standard Chamfer distance (K=1). We do not claim to outperform the two baselines but to show we can reconstruct the novel objects reasonably well for next-stage object manipulation. We cite Point Set [10] and 3D-R2N2 [7] as a performance indicator.

IV. EXPERIMENTS

We first evaluate our particle reconstruction approach on the dataset ShapeNet [6] to find out whether our approach can reconstruct the objects well. Next, we compare with DexNet 2.0 [30], a state-of-the-art data-driven method for object grasping without any real grasping data. On top of that, we compare with previous work PushNet [27] for object pushing. Finally, the system manipulates an object with unknown mass distribution, pushes it, grasps it, and stacks it on a small stand.

In summary, our experiments aim to answer the following questions: 1) Can we generate good quality particle reconstructions? 2) How does our approach PROMPT perform on a real robot grasping task? 3) Can our particle-based framework generalize to other robot tasks such as object pushing and placing tasks?

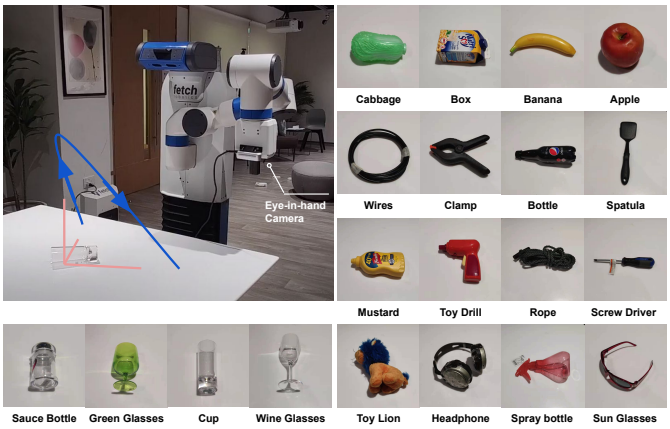


Fig. 6: Top left: a Fetch Robot with an eye-in-hand RGB camera for object grasping. The blue line indicates the trajectory of the eye-in-hand camera during the reconstruction phase. the RGB camera is always pointing at the working area and the view set follows the three-view convention to achieve a good coverage. The top right shows simple items; the center right shows everyday items; the bottom right shows difficult items and the bottom left shows translucent items. Each item is grasped separately.

TABLE II: Robot Grasping Results

		DexNet 2.0 [30]		PROMPT (Ours)	
		Count	Suc Rate	Count	Suc Rate
Simple Items	Banana	5/5	0.8	5/5	0.95
	Box	2/5		4/5	
	Cabbage	4/5		5/5	
	Apple	5/5		5/5	
Daily Items	Mustard	5/5	0.825	5/5	0.95
	Spatula	5/5		5/5	
	Toy Drill	4/5		5/5	
	Clamp	2/5		5/5	
	Screw Driver	3/5		4/5	
	Bottle	4/5		5/5	
	Wires (Circle)	5/5		4/5	
	Ropes	5/5		5/5	
Translucent Items	Cup	3/5	0.7	4/5	0.9
	Sauce Bottle	2/5		5/5	
	Wine Glasses	4/5		5/5	
	Green Glasses	5/5		4/5	
Complex Items	Headphone	5/5	0.7	5/5	0.95
	Sun Glasses	3/5		5/5	
	Toy Lion	4/5		4/5	
	Spray Bottle	2/5		5/5	
Average Success Rate			0.77		0.94

Fig. 7: Real world particle reconstruction and grasping results. Our approach PROMPT grasps various novel objects with pure RGB images as input without any offline training. With particle-based object representation, our method connects the image observation with a particle dynamics model and enables model-based planning on novel objects.

A. 3D Reconstruction

This section aims to find out whether our 3D reconstruction approach works on complicated, arbitrary-shaped objects. The reconstruction is conducted with 200 views and 4000 particles with a single 2080Ti GPU. We propose three variant implementations of the point cloud generators: FC-100, CNN-100, and FC-1. FC-100 is a single fully connected layer network that takes in a random number and outputs the $N * 3$ sized point cloud with K =100 for the proposed K nearest Chamfer distance. FC-1 is the same FC generator with standard Chamfer distance (K=1). CNN-100 is an auto-encoder structured neural network generator that consists of 12 convolution layers followed by an LSTM layer for a sequence of image inputs and 2 fully connected layers for point cloud output. The variant CNN-100 is used here to determine whether taking in a sequence of image inputs helps with the reconstruction performance.

We evaluate our 3D reconstruction quantitative results on the common benchmark dataset ShapeNet [6] with the results shown in table I. Total 8 categories, 50 objects per category, are evaluated. Shown in Table I, our reconstruction performance is not perfect but surprisingly decent. For some categories such as plane, lamp, firearm, and watercraft, our approach achieves better results with the intersection of union (IoU) metrics. It shows that our approach works on complicated objects. However, an interesting observation is that our approach's performance drops significantly on simple object categories such as speaker and couch, often a large box. It is because IoU measures the entire object, including the interior space, which is not meaningful as we do not know whether an object is empty or not given image observations. Our approach generates a point cloud from masking images and has no explicit objective for the interior space. However, other baselines such as Point Set [10] trains their generator with ground truth point cloud and hence knows the prior information. During the testing phase, categorical information is also provided to Point Set, and 3DR2N2 [7]. We do not claim to outperform the two baselines but to show we can reconstruct the novel objects reasonably well for next-stage object manipulation. We cite them as a performance indicator.

In addition to the deep learning-based approach, we favorably considered other options in the multi-view stereo approach. Typical multi-view stereo methods usually rely on object feature detection or the Lambertian assumption of photo-consistency [12, 13]. However, in the real world, transparent objects do not follow the photo-consistency assumption and many other objects have featureless surfaces, leading to a large number of failure cases in traditional multi-view stereo

TABLE III: Robot Pushing Results

	PushNet [27]		PROMPT (Ours)	
	Count	Avg Steps	Count	Avg Steps
Mustard	10/10	5.90	9/10	4.44
Toy Drill	9/10	6.22	10/10	3.30
Sun Glasses	-	-	10/10	4
Banana	10/10	9.80	10/10	4.30
Black Box	10/10	6.00	10/10	4.40
Clamp	9/10	7.11	9/10	5.11
Soup Ladle	9/10	7.56	9/10	6.11
Average	57/60	7.1	67/70	4.49

methods. Due to several problems mentioned above, we stick to our own reconstruction method.

Comparing FC-100 and FC-1, FC-100 improves the performance on most categories. CNN-100 achieves a slightly better performance than FC-100 with image inputs as the images provide some prior information that helps with the reconstruction. With the experiment results, we conclude that our 3D reconstruction approach works on novel objects given different views of masking images.

B. Object Grasping

This experiment aims to answer whether our approach PROMPT works on the real robot grasping tasks. We have an eye-in-hand RGB camera that follows a precomputed trajectory to take a short video. To obtain the object's masking image for arbitrary objects, we leverage the pre-trained foreground segmentation algorithm U2Net [38].

1) Experiment Setups: We deploy our method PROMPT on Fetch robot, see figure 6 top left. The robot has parallel grippers and can perform top-down grasping. We select some objects from the YCB object set [5], and other four translucent objects as well as daily objects. There are 20 objects in total. Figure 6 left bottom and right show the test set items, and each item is grasped in isolation.

2) Evaluation Protocol: We use the grasping success rate to evaluate the general performance of our approach. There are 20 unseen objects, and we test 5 trials each object with random pose and position. During all the trials, the object remains constant. Each grasp attempt (episode) is a one-time grasping action. If no object has been grasped at the end of the grasping action, the attempt is regarded as a failure.

3) Implementation Details: The robot takes a video of objects and generate masking images using U2Net [38] for particle state reconstruction with number of particles $N = 2000$ and number of views $I \approx 70$. The predicted particle state is then fed into the SOTA particle based simulator Nvidia-Flex [14] for action planning. We develop the simulator based on PyFlex [28]. Other particle-based simulators such as Taichi [17] and Differentiable Taichi [18] are also available. Three 2080Ti GPUs are used for action planning described in section 3C and one 2080Ti for reconstruction.

Fig. 8: Objects for pushing tasks. From left to right are: sun glasses, banana, clamp, soup ladle, mustard, black box, toy drill.

4) View Points Selection: We choose a set of views to maximize the coverage of the target object. In practice, we choose a fixed trajectory to take a short video and perform reconstruction on the collected images. We discover that the top views reveal the most information about the object geometry and the side views, which are very close to the table, reveal the height information. It indeed matches the three-view drawings convention: front, left and right views. We visualized the view points in Fig. 6.

The choice of views can be extended to active information gathering that contributes to 3D reconstruction, which we leave for future study. For example, we can formulate the problem of determining the set of views as a problem of "the Next Best View" (NBV) problem and leverage on previously developed NBV solutions [36].

5) Results: We visualize some of the real-world object particle reconstructions in figure 7. Those items have very different shapes and challenging appearances. For example, the spray bottle and the headphone have irregular shapes, and the transparent wine glasses and cups have a difficult appearance. Suggested by the results, our multi-view particle reconstruction approach can reconstruct the 3D structure reasonably well in the real robot task with a single RGB camera, even with noisy masking images. In real robot experiments, we pipeline the reconstruction process so that each new image collected is immediately available for reconstruction without waiting for the robot to stop. Reconstructing 2000 particles takes 6 seconds to converge, while 1500 particles takes 4 seconds, and the grasp planning takes 8 seconds to generate the required actions. The planning time can be further reduced if we train a policy and use the trained policy to guide the search. We leave this problem for future study.

The results are shown in table II. The real robot experiment results suggest a success rate of 0.94 over 100 grasping trials for our approach. We compare with DexNet 2.0 [30] that grasps object in isolation, as DexNet 4.0 is for grasping and suction in clutter and DexNet 3.0 is for Suction. The DexNet 2.0 achieves 0.77 success rate in the our experiment. The results suggest our particle model-based planning approach PROMPT outperforms DexNet 2.0 in all object categories, especially for translucent items and complex items.

Our object segmentation process relies on the foreground segmentation network U2Net [38], and some failures come from noisy object-masking images produced by U2Net, such as failures on translucent green glasses and cups. U2Net is trained on a large set of images. However, it is not explicitly optimized for translucent objects and shadows. Our approach assumes that we can detect object masking images, so in

(a) (b) (c) (d) (e) (f) (g)

Fig. 9: Pushing, grasping, and placing a hammer with an uneven mass distribution. We show the intermediate steps: (a) object reconstruction (b) push (c) position re-estimation (d) push (e) position re-estimation (f) grasp (g) object stacking. The robot manipulates the object interactively to estimate the mass distribution in order to stack the hammer.

principle, this problem is outside the scope of this work. In fact, we can address the issue in the future by fine-tuning U2Net with translucent objects and shadows. We conclude that our approach PROMPT works on the real robot grasping task with complicated object shapes based on the real-world experiment results.

C. Object Pushing

We further evaluate our approach on the object pushing tasks to find out whether our method generalizes to other object manipulation tasks. Similar to the Figure 6, in the pushing experiment we have the same robot setup. Figure 8 shows the test objects in the experiment. After each push, the robot takes an additional image to re-estimate the object's current position and configuration with the algorithm described in section 3D.

1) Evaluation Protocol: We use the push success rate and number of steps to evaluate the average performance. We test on 7 novel objects, 10 trials each, with random goal pose and position. Goal positions in x direction are sampled uniformly from (0.1m, 0.2m) $\cup$ (-0.1m, -0.2m) and y direction are sampled uniformly from (-0.15m, -0.25m). Rotational goals are sampled uniformly from (90, 30) $\cup$ (-30, -90). The push task is considered as a success when the object is 5cm closer to the goal with orientation error less than 15 degrees. The maximum allowed number of steps is 25.

2) Results: Table III shows the real robot pushing experiment results. We compare with the modern object pushing baseline PushNet [27] that trains its policy in a simulator and applies to a real robot. Our approach and PushNet achieve a high success rate, but our approach produces fewer steps to achieve the goal. We leverage on a known particle-based dynamics model and explicit model-based planning, which produces near-optimal actions for each push. Despite the high success rate of PushNet, we observed PushNet often generates sub-optimal actions that incur high costs on difficult objects. Taking the banana as an example, pushing bananas seems like an easy task to complete, but it turns out contrary to expectations. The toy banana is light, which means it rotates easily if the push direction does not accurately go through the center of mass. Sub-optimal push actions hence likely result in large unexpected rotations on the banana. Our approach relies on planning which is good at producing accurate actions and achieves better results on difficult objects e.g.banana.

A typical failure case occurs in both methods when the object is pushed out of the robot working area, resulting in unreachable positions. The following actions hence can not be executed, and the task fails. On top of that, the item sunglasses completely fails on PushNet as PushNet uses depth image-based object segmentation that fails on the partially translucent object. Since object segmentation is not the main contribution of PushNet, we exclude the sunglasses result for PushNet.

In summary, our approach naturally generalizes object-pushing tasks by changing the actions space and reward function for the MPC planner.

D. Object Pushing, Grasping, and Placing

We demonstrate the ability to perform pushing, grasping and placing task all together with our proposed approach. The robot setup is shown in figure 9. The system manipulates a hammer with unknown mass distribution, grasps it, and stacks it on a small stand. Without understanding the center of the mass, simply stacking the object's center on the small stand likely leads to a failure. The robot first reconstruct a particle state and then pushes the object for several times to estimate the novel object's center of mass (CoM). The video link can be found in the abstract.

V. Conclusion

We present PROMPT, a particle-based object manipulation framework, which enables robots to achieve dynamic manipulation on a variety of tasks, including grasping, pushing, and placing with novel objects. The particular particle state representation bridges the learning for complex observation and planning for decision making, connecting the complex visual observation to a known dynamics model. This idea enables the entire system to deal with complex objects with arbitrary shapes and at the same time has the understanding of the dynamics system for better action planning. We evaluate our approach to real robots with grasping, pushing, and placing tasks, showing the system's effectiveness and generalization to various robot tasks.

Acknowledgements

This research is supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG2-PhD-2021-08-015T) and A*STAR under the National Robotics Program (Grant No. 192 25 00054).

References

[1] Ravi Balasubramanian, Ling Xu, Peter D Brook, Joshua R Smith, and Yoky Matsuoka. Physical human interactive guidance: Identifying grasping principles from human-planned grasps. *IEEE Transactions on Robotics*, 28(4):899–910, 2012.

[2] Jeannette Bohg, Matthew Johnson-Roberson, Beatriz León, Javier Felip, Xavi Gratal, Niklas Bergström, Danica Kragic, and Antonio Morales. Mind the gap-robotic grasping under incomplete observation. In *2011 IEEE International Conference on Robotics and Automation*, pages 686–693. IEEE, 2011.

[3] Gary M Bone, Andrew Lambert, and Mark Edwards. Automated modeling and robotic grasping of unknown three-dimensional objects. In *2008 IEEE International Conference on Robotics and Automation*, pages 292–298. IEEE, 2008.

[4] Peter Brook, Matei Ciocarlie, and Kaijen Hsiao. Collaborative grasp planning with multiple object representations. In *2011 IEEE international conference on robotics and automation*, pages 2851–2858. IEEE, 2011.

[5] 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*, 22(3):36–52, 2015.

[6] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[7] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European conference on computer vision*, pages 628–644. Springer, 2016.

[8] Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. http://pybullet.org, 2016–2019.

[9] Dan Ding, Yun-Hui Liu, and Shuguo Wang. Computing 3-d optimal form-closure grasps. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 4, pages 3573–3578. IEEE, 2000.

[10] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017.

[11] Jean-Sébastien Franco and Edmond Boyer. Efficient polyhedral modeling from silhouettes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31 (3):414–427, 2008.

[12] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE transactions on pattern analysis and machine intelligence*, 32(8):1362–1376, 2009.

[13] Yasutaka Furukawa, Brian Curless, Steven M Seitz, and Richard Szeliski. Towards internet-scale multi-view stereo. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 1434–1441. IEEE, 2010.

[14] NVIDIA Gameworks. Nvidia flex, 2018.

[15] Corey Goldfeder, Peter K Allen, Claire Lackner, and Raphael Pelossof. Grasp planning via decomposition trees. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 4679–4684. IEEE, 2007.

[16] Alexander Herzog, Peter Pastor, Mrinal Kalakrishnan, Ludovic Righetti, Jeannette Bohg, Tamim Asfour, and Stefan Schaal. Learning of grasp selection based on shape-templates. *Autonomous Robots*, 36(1-2):51–65, 2014.

[17] Yuanming Hu, Tzu-Mao Li, Luke Anderson, Jonathan Ragan-Kelley, and Frédo Durand. Taichi: a language for high-performance computation on spatially sparse data structures. *ACM Transactions on Graphics (TOG)*, 38 (6):201, 2019.

[18] Yuanming Hu, Luke Anderson, Tzu-Mao Li, Qi Sun, Nathan Carr, Jonathan Ragan-Kelley, and Frédo Durand. Difftaichi: Differentiable programming for physical simulation. *ICLR*, 2020.

[19] Kai Huebner, Kai Welke, Markus Przybylski, Nikolaus Vahrenkamp, Tamim Asfour, Danica Kragic, and Rudiger Dillmann. Grasping known objects with humanoid robots: A box-based approach. In *2009 International Conference on Advanced Robotics*, pages 1–6. IEEE, 2009.

[20] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *arXiv preprint arXiv:1806.10293*, 2018.

[21] Daniel Kappler, Jeannette Bohg, and Stefan Schaal. Leveraging big data for grasp planning. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4304–4311. IEEE, 2015.

[22] Jan J Koenderink and Andrea J Van Doorn. Affine structure from motion. *JOSA A*, 8(2):377–385, 1991.

[23] Aldo Laurentini. The visual hull concept for silhouette-based image understanding. *IEEE Transactions on pattern analysis and machine intelligence*, 16(2):150–162, 1994.

[24] Aldo Laurentini. How far 3d shapes can be understood from 2d silhouettes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(2):188–195, 1995.

[25] Ian Lenz, Honglak Lee, and Ashutosh Saxena. Deep learning for detecting robotic grasps. *The International Journal of Robotics Research*, 34(4-5):705–724, 2015.

[26] Sergey Levine, Peter Pastor, Alex Krizhevsky, Julian Ibarz, and Deirdre Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research*, 37(4-5):421–436, 2018.

[27] Jue Kun Li, David Hsu, and Wee Sun Lee. Push-net : Deep planar pushing for objects with unknown physical properties. In *Robotics: Science and System*, 2018.

[28] Yunzhu Li. Pyflex. https://github.com/YunzhuLi/PyFleX, 2019.

[29] Zhaoxin Li, Kuanquan Wang, Wangmeng Zuo, Deyu Meng, and Lei Zhang. Detail-preserving and content-aware variational multi-view stereo reconstruction. *IEEE Transactions on Image Processing*, 25(2):864–877, 2015.

[30] Jeffrey Mahler, Jacky Liang, Sherdil Niyaz, Michael Laskey, Richard Doan, Xinyu Liu, Juan Aparicio Ojea, and Ken Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. *arXiv preprint arXiv:1703.09312*, 2017.

[31] Zoltan-Csaba Marton, Dejan Pangercic, Nico Blodow, Jonathan Kleinehellefort, and Michael Beetz. General 3d modelling of novel objects from a single view. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3700–3705. IEEE, 2010.

[32] Zoltan-Csaba Marton, Lucian Goron, Radu Bogdan Rusu, and Michel Beetz. Reconstruction and verification of 3d object models for grasping. In *Robotics Research*, pages 315–328. Springer, 2011.

[33] Luis Montesano and Manuel Lopes. Active learning of visual descriptors for grasping using non-parametric smoothed beta distributions. *Robotics and Autonomous Systems*, 60(3):452–462, 2012.

[34] John Oberlin and Stefanie Tellex. Autonomously acquiring instance-based object models from experience. In *Robotics Research*, pages 73–90. Springer, 2018.

[35] Lerrel Pinto and Abhinav Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 3406–3413. IEEE, 2016.

[36] Richard Pito. A solution to the next best view problem for automated surface acquisition. *IEEE Transactions on pattern analysis and machine intelligence*, 21(10):1016–1030, 1999.

[37] Florian T Pokorny and Danica Kragic. Classical grasp quality evaluation: New algorithms and theory. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3493–3500. IEEE, 2013.

[38] Xuebin Qin, Zichen Zhang, Chenyang Huang, Masood Dehghan, Osmar Zaiane, and Martin Jagersand. U2-net: Going deeper with nested u-structure for salient object detection. volume 106, page 107404, 2020.

[39] Arthur George Richards. *Robust constrained model predictive control*. PhD thesis, Massachusetts Institute of Technology, 2005.

[40] Maximo A Roa, Max J Argus, Daniel Leidner, Christoph Borst, and Gerd Hirzinger. Power grasp planning for anthropomorphic robot hands. In *2012 IEEE International Conference on Robotics and Automation*, pages 563–569. IEEE, 2012.

[41] Nikolay Savinov, Christian Hane, Lubor Ladicky, and Marc Pollefeys. Semantic 3d reconstruction with continuous regularization and ray potentials using a visibility consistency constraint. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5460–5469, 2016.

[42] Ben Semerjian. A new variational framework for multi-view surface reconstruction. In *European Conference on Computer Vision*, pages 719–734. Springer, 2014.

[43] Bruno Siciliano and Oussama Khatib. *Springer handbook of robotics*. Springer, 2008.

[44] Richard Szeliski. Rapid octree construction from image sequences. *CVGIP: Image understanding*, 58(1):23–32, 1993.

[45] Josh Tobin, Lukas Biewald, Rocky Duan, Marcin Andrychowicz, Ankur Handa, Vikash Kumar, Bob McGrew, Alex Ray, Jonas Schneider, Peter Welinder, et al. Domain randomization and generative models for robotic grasping. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3482–3489. IEEE, 2018.

[46] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.

[47] Ulrich Viereck, Andreas ten Pas, Kate Saenko, and Robert Platt. Learning a visuomotor controller for real world robotic grasping using simulated depth images. *arXiv preprint arXiv:1706.04652*, 2017.