Can I Pour into It? Robot Imagining Open Containability Affordance of Previously Unseen Objects via Physical Simulations

Hongtao Wu and Gregory S. Chirikjian

Abstract—Open containers, i.e., containers without covers, are an important and ubiquitous class of objects in human life. In this letter, we propose a novel method for robots to “imagine” the open containability affordance of a previously unseen object via physical simulations. The robot autonomously scans the object with an RGB-D camera. The scanned 3D model is used for open containability imagination which quantifies the open containability affordance by physically simulating dropping particles onto the object and counting how many particles are retained in it. This quantification is used for open-container vs. non-open-container binary classification (hereafter referred to as open container classification). If the object is classified as an open container, the robot further imagines pouring into the object, again using physical simulations, to obtain the pouring position and orientation for real robot autonomous pouring. We evaluate our method on open container classification and autonomous pouring of granular material on a dataset containing 130 previously unseen objects with 57 object categories. Although our proposed method uses only 11 objects for simulation calibration (training), its open container classification aligns well with human judgements. In addition, our method endows the robot with the capability to autonomously pour into the 55 containers in the dataset with a very high success rate. We also compare to a deep learning method. Results show that our method achieves the same performance as the deep learning method on open container classification and outperforms it on autonomous pouring. Moreover, our method is fully explainable.

I. INTRODUCTION

The understanding of object affordances [1], i.e., the set of possible interactions with an object, plays a crucial role in human perception of real-world objects despite the tremendous variance in their appearances [2]. As robots begin to enter our daily life, it is necessary for them to assess previously unseen real-world objects and understand how they can be interacted with via affordance-based object perception. Despite the necessity and advantages of affordance reasoning in object perception, the majority of work on this topic is appearance-based. Appearance-based methods can be brittle when encountering objects with large intra-class variations. Moreover, they can be challenged when functional reasoning about previously unseen objects and inter-class function generalization are needed. For example, consider a robot trying to prepare a bowl of M&M’s® candies (hereafter referred to as candies) in a kitchen. What if the bowl in the kitchen has a different appearance from the bowls that the robot has seen before? What if the bowl is broken with a hole at the bottom while still visually appears to be a bowl? If there are no bowls but only cups in the kitchen, can the robot be programmed to improvise? Given these unexpected situations, we argue that appearance-based object perception alone is not sufficient for robotics application in the real world; what is missing is the capability to physically “understand” object affordances. If the bowl is previously unseen, the robot should recognize its containability affordance; if the bowl is broken, it should reason that the bowl cannot afford to contain candies; if there are no bowls but only cups, it should generalize the containability affordance from bowls to cups and use a cup instead to contain the candies (Fig. 1).

To address the problem of affordance reasoning in object perception, we introduced the interaction-based definition of an object and a method to “imagine” object affordances as a chair via physical simulations in our previous work [3]. In this paper, we extend our idea of physical simulation-based object affordance reasoning to the class of open containers and propose a novel method for robots to imagine the open...
containability affordance. We are inspired by the fact that humans mentally simulate high-level physical interactions to reason about complex systems [4]. Our long-term goal is to endow robots with similar mental capabilities to imagine potential interactions with an object and reason about its object affordances accordingly (Fig. 1(a)).

We first discuss the physical attributes to imagine for the class of open containers. Containers can be generally categorized into open containers (e.g., cups) and closed containers (e.g., bottles with a cap). The containability closure [5] of a closed container makes the physical interaction with it different from that with an open container. An agent needs to open a closed container to transfer contents into it while there is no such need for an open container. The open containability we describe in this paper does not take into account the containability closure. The pose of an open container also affects the interaction with it. An upside down open container needs to be reoriented upright to afford open containability. Finally, containability is related to the physical properties of the container and the contents. For example, the origami cup in Fig. 2 cannot contain water because it will be soaked but it is able to contain candies. In this paper, we focus on reasoning the open containability for granular material of an upright object. Based on the containability definition in [5], we propose the interaction-based definition of open containers:

“an object which can be placed such that liquid or granular material can be poured into it from above and retained therein under small perturbations in pose.”

The interaction-based definition defines open containers from the perspective of object interaction which robots can “understand” and “imagine” accordingly. Fig. 2 shows our method’s pipeline. In our method, the robot performs open containability imagination (Sec. III-A) to quantify the open containability affordance of a previously unseen object. This quantification is used as a cue for open-container vs. non-open-container binary classification (hereafter referred to as open container classification). If the object is classified as an open container, the robot further performs pouring imagination (Sec. III-B) to obtain the pouring position and orientation for the robot to pour autonomously in the real world (Sec. IV-D). We implement our imagination method on a UR5 manipulator. We evaluate our method on open container classification and autonomous pouring of granular material (M&M’s® candies) with a dataset containing 130 previously unseen real-world objects. The dataset has 57 object categories. Results show that our method is able to achieve human-like judgements on open container classification, even though it uses only 11 objects for simulation calibration (training). In addition, our method is able to autonomously pour into the 55 containers in the dataset with a very high success rate. We also compare to a deep learning method [6] which we find most relevant in the literature. Our method is able to achieve the same performance as the deep learning method on open container classification and outperform it on autonomous pouring.

II. RELATED WORK

Containability Reasoning. Containability reasoning has been studied in the field of cognitive science [7]–[9] and computer vision [5], [10]–[12]. Liang et al. [7] compare human judgements of containing relations between different objects with physical simulation results. Yu et al. [10] voxelize the world containing an object of interest and determine the state of each voxel to reason about whether the object is a container and its best filling direction. Our method diverges from these approaches by reasoning open containability affordance with a real robot and leveraging the affordance reasoning on robot autonomous pouring.

Learning-based Affordance Detection. There is a growing interest in learning object affordances and object functionalities in the computer vision and robotics community [6], [13]–[21]. [6], [14]–[17] leverage convolutional neural networks (CNNs) to segment affordance regions within an image. Manuelli et al. [18] use deep learning to identify key-point affordances for objects to reinforce robot manipulation. Instead of learning appearance-based cues, our method digs into the underlying physics by using physical simulations to encode object affordances.

Physics-based Reasoning. The concept of object affordances has also been studied by introducing physics to reason about the functionality of an object [3], [10], [11], [21]–[24].
Kunze and Beetz [23] reason the outcome of robot actions with simulation-based projections to guide robots in planning actions. The concept of digital twins [25] uses simulations to replicate physical systems for operation optimization and failure prediction in manufacturing. Instead of reasoning physics or using physics to predict action outcomes, we exploit physics to imagine potential interactions with an object for object classification and robot-object interaction.

**Autonomous Pouring.** Autonomous pouring is an important task in robot manipulation [26]–[30]. Yamaguchi and Atkeson [27] explore temporally decomposed dynamics in pouring simulation experiments with differential dynamic programming. Scheneck and Fox [28] learn to estimate the liquid volume in a container and incorporate this sensory feedback into robotic pouring. Unlike these methods which know the object class (container) and/or the object 3D model *a priori*, our method categorizes a previously unseen object and predict the pouring position and orientation without knowing the 3D model or the category of the object *a priori*.

**III. METHOD**

Our method takes two steps to explore the open containability affordance of a previously unseen object. In the first step, the robot performs open containability imagination, which simulates dropping particles of granular material from above the object and perturbing the object, to determine if the object is an open container. If the object is identified as an open container, the robot proceeds to pouring imagination, which simulates pouring particles from a cup, to obtain the pouring position and orientation for autonomous pouring in the real world. Given the scanned 3D model of the object, our method attaches the body frame to its center of mass (CoM) and set the axes parallel to those of the world frame. The object and particles are considered as rigid bodies. Rigid body transformation can be specified by $g = (R, t) \in SE(3)$ where $R \in SO(3)$ is the rotation matrix and $t = [x, y, z]^T \in \mathbb{R}^3$ parameterizes the translation.

**A. Open Containability Imagination**

Fig. 3 shows the setting of open containability imagination. Particles are placed above the axis-aligned bounding box (AABB) of the object and patterned in a grid to cover the top face of the AABB (Fig. 3(b)). We denote the numbers of particles along the $x$-, $y$-, and $z$-axes as $N_x$, $N_y$, $N_z \in \mathbb{Z}^+$. $N_x$, $N_y$ are set as follows:

$$N_x(s) = \left\lfloor \frac{sl_x}{l_x^o} \right\rfloor, N_y(s) = \left\lfloor \frac{sl_y}{l_y^o} \right\rfloor$$

where $l_x^o$ and $l_y^o$ are the extents of the object (particle) AABB along the $x$- and $y$-axes, respectively; $s$ is a scale factor to avoid cases where there are too many particles to drop which will greatly increase the runtime; $\left\lfloor \cdot \right\rfloor$ is the floor function. We first set $s = 1$, $N_z = 1$ and calculate $N_x(1)$ and $N_y(1)$. If $N_x(1)N_y(1) > N_{\text{max}}$, i.e., there are too many particles, we decrease $s$ by setting $s = \sqrt{N_{\text{drop}} / N_x(1)N_y(1)}$ and recalculate $N_x(s)$ and $N_y(s)$. If $N_x(1)N_y(1) < N_{\text{min}}$, i.e., there are too few particles which will make later calculation of open containability (Eqn. 1) unreliable, we increase $N_z$ by setting $N_z = \left\lceil \frac{N_{\text{drop}}}{N_x(1)N_y(1)} \right\rceil$.

After the particles are set up, they are dropped in the physical simulation with gravity. According to the interaction-based definition, if an object can robustly retain particles, small perturbations in pose shall not spill the particles. Therefore, perturbations are introduced by simulating motions equivalent to rotating and translating the object when the drop is finished. The object is rotated by a small degree ($\pm \pi/60$) along the $x$- and $y$-axes to simulate rotation; horizontal force fields ($g_h = 0.5N/kg$) along the positive and negative directions of $x$- and $y$-axes are applied sequentially to simulate translation (Fig. 3(a)). Note that applying a horizontal force field is equivalent to accelerating the object horizontally along the negative direction of the force field. Translation is simulated with accelerations instead of constant velocities because based on common experience, spillage mostly happens during accelerations instead of moving with constant velocities. The number of particles retained within the objects, $N_{\text{in}}$, is obtained by counting the number of particles within the object AABB at the end of the simulation. The open containability of the object $\Omega$ is quantified as:

$$\Omega = \frac{N_{\text{in}}}{N_{\text{drop}}}$$

where $N_{\text{drop}} = N_xN_yN_z$ is the number of particles dropped. If $\Omega > \Omega_{\text{thr}}$, the object is considered as an open container. $\Omega_{\text{thr}}$ is a threshold. The $xy$-plane projection of the initial dropping position of the particles retained within the object after the drop constitutes what we called the *footprint* of the object (indicated in red in Fig. 3(b)).
For each \( \theta \) r cup from different positions and orientations (Fig. 4). To angle. (c) Snapshots of the pouring imagination at different timesteps (TS).

Pouring Imagination. (a) Pouring setting. (b) Pouring imagination of a red bag. We set \( \gamma_0 = 62^\circ \) which is a hyperparameter specifying the total pouring angle. For all pourings, we set \( \theta \) is a parameter that specifies the rotation which rotates about the z-axis of the cup mouth, \( l_{\text{cup}} \) is the slant height passing through the rim of the cup at \( \theta \) pivot (Fig. 4(a)). For all pourings, the cup is initially placed such that \( l_{\text{cup}} \in E \) and \( r_{\text{pivot}} \) is the lowest end of the cup mouth. This constrains the initial pose of the cup to \( SE(2) \). We parameterize a pouring with a tuple \( (\theta_{\text{pour}}, \mathbf{p}_{\text{pour}}) \in SE(2) \) where \( \theta_{\text{pour}} \in S^1 \) and \( \mathbf{p}_{\text{pour}} \in \mathbb{R}^2 \). At the beginning of each pouring, \( l_{\text{cup}} \) is aligned with a ray (red dashed line Fig. 4(a)) originated from \( \mathbf{p}_E \). The angle between the x-axis of \( E \) and the ray equals \( \theta_{\text{pour}} \). \( \mathbf{p}_{\text{pour}} \), which lies along the ray, specifies the x and y components of \( r_{\text{pivot}} \) in \( E \). The pouring is a one degree of freedom motion: the cup is rotated about an axis \( l_{\text{pour}} \in E \) (blue dashed line in Fig. 4(a)) which is tangent to the rim of the cup at \( r_{\text{pivot}} \).

For an object, the imagination simulates pouring from 8 different \( \theta_{\text{pour}} \):

\[
\theta_{\text{pour}}(i) = \frac{i \pi}{4}, \; i = 0, 1, ..., 7 \tag{2}
\]

For each \( \theta_{\text{pour}} \), it simulates pouring at 3 different \( \mathbf{p}_{\text{pour}} \):

\[
\mathbf{p}_{\text{pour}}(i,j) = R(\theta_{\text{pour}}(i)) \begin{bmatrix} l_{\text{idt}}(j) \\ 0 \end{bmatrix} \tag{3}
\]

where \( l_{\text{idt}}(j) = L_0 + j \Delta L, j = 0, 1, 2 \) defines the j-th indentation distance; \( L_0 \) and \( \Delta L \) are two hyperparameters:

\[
R = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}
\]

The same method described in Sec. III-A is used to count the number of particles retained within the object \( N_{\text{in}} \) after each pouring. The particle-in-object ratio \( P \) of pouring with \( (\theta_{\text{pour}}, \mathbf{p}_{\text{pour}}) \) is specified as:

\[
P(\theta_{\text{pour}}, \mathbf{p}_{\text{pour}}) = \frac{N_{\text{in}}}{N_{\text{pour}}} \tag{4}
\]

where \( N_{\text{pour}} \) is the number of particles poured. In total, the imagination simulates 24 pourings for \( 24 (\theta_{\text{pour}}, \mathbf{p}_{\text{pour}}) \) tuples and get 24 \( P \). We add indentation \( l_{\text{idt}} \) instead of directly pouring at \( \mathbf{p}_E \) (Eqn. 3) for two reasons. First, \( \mathbf{p}_E \) is obtained by freely dropping the particles, i.e., the particles have no horizontal velocities. Whereas in pouring, the particles have a horizontal velocity when exiting the cup mouth. We indent the cup “backward” horizontally to compensate for this velocity. Also, we want to explore more area for pouring. Indenting the cup towards the object periphery provides a wider exploration area and a higher chance to find a \( (\theta_{\text{pour}}, \mathbf{p}_{\text{pour}}) \) with a large \( P \) (Fig. 5).

The \( (\theta_{\text{pour}}, \mathbf{p}_{\text{pour}}) = (\theta_{\text{pour}}(i^*), \mathbf{p}_{\text{pour}}(i^*, j^*)) \) for robot autonomous pouring is chosen by setting \( i^* \) and \( j^* \) as follows:

\[
i^* = \arg \max_i \sum_j P(\theta_{\text{pour}}(i), \mathbf{p}_{\text{pour}}(i,j))
\]

\[
j^* = \arg \max_j P(\theta_{\text{pour}}(i^*), \mathbf{p}_{\text{pour}}(i^*, j))
\]

We select \( \theta_{\text{pour}} \) as above because pouring with this \( \theta_{\text{pour}}(i^*) \) has higher chances to get a large \( P \), i.e., less spillage, in the real world. The cup initial pose of pouring \( (R_{\text{init}}, t_{\text{init}}) \in SE(3) \) in the world frame can be retrieved as follows:

\[
R_{\text{init}} = R_z(\theta_{\text{pour}})R_0
\]

\[
t_{\text{init}} = \left[ \begin{array}{c} \mathbf{p}_{\text{pour}} + \mathbf{p}_E \\ z_E \end{array} \right] - R_{\text{init}}r_{\text{pivot}}
\]

where \( R_z(\theta_{\text{pour}}) \) specifies the rotation which rotates about the z-axis of the world frame by \( \theta_{\text{pour}} \); \( R_0 \) rotates the cup such that \( l_{\text{cup}} \) is parallel to the x-axis of \( E \) and \( r_{\text{pivot}} \) is the lowest end of the cup mouth in the world frame; \( r_{\text{pivot}} \in \mathbb{R}^3 \) is the position of the pivot in the object body frame; \( \mathbf{p}_E \) and \( z_E \) are defined at the beginning of this section.
IV. Experiments

Fig. 1(b) shows the experiment setup. We implement our method on a UR5 robot mounted with an afag EU-20 UR gripper. A PrimeSense Carmine 1.09 RGB-D camera is also mounted on the end effector. The robot base frame is used as the world frame throughout the experiment.

A. Robot 3D Scanning

In the experiment, the object of interest is placed randomly on a transparent platform in its upright pose. The transparent platform is also randomly placed on a table within limits that are described below. Since the depth sensor of the camera has a shortest range of 0.35m, although random, the position of the object needs to satisfy: 1) the object does not collide with the robot during the scanning; 2) the object falls within the range of the depth sensor in all capturing views (e.g., the front-to-back view shown in Fig. 6(a)); 3) the object is in the robot workspace so that the robot can pour into the object if it is classified as an open container. Using the wrist mounted RGB-D camera, we move the robot’s end effector to 24 pre-defined configurations to capture depth images of the scene. From the robot’s forward kinematics, we are able to obtain the pose of the camera at each view. This allows us to use TSDF Fusion [31], [32] to densely reconstruct the scene. Since the object is placed on a transparent platform which will not be captured by the depth sensor of the camera, we are able to segment the object from the scene by simply cropping the 3D reconstruction with a box. Finally, we get the 3D mesh of the object. It is worth noting that object segmentation could also be achieved without the transparent platform by 3D object segmentation [33] or physically picking up the object for scanning [34].

As we are not able to capture views from underneath the object, the reconstructed 3D model is partial. It only covers surfaces which can be seen from the top and side views. The inner containing surface of open containers can be seen from the top and side views given the object is placed upright, and thus is included in the reconstructed model. Therefore, the partially reconstructed model is sufficient for the purpose of open containability reasoning. Reconstructing the complete model of an object can be achieved by methods involving manipulating the object such as [34], [35].

B. Data

Our dataset contains 141 real-world objects which are commonly encountered in daily life (Fig. 1(b,c,d)). We use 11 objects for simulation calibration (training set) and the remaining 130, which are previously unseen by the robot, for testing (test set). The 130 objects in the test set covers 57 object categories. Since the objects have no labels of open container, we recruit 20 human subjects to annotate the data for open container classification. Every object is annotated 5 times by 5 different subjects. Given an object, we asked each of the 5 subjects: 1) Given the object in this pose, is it able to contain M&M’s® candies? 2) Given the object in this pose, are you able to pour M&M’s® candies into it? If the annotations to these two questions are both true, we consider the object is annotated as an open container by the subject; otherwise, it is annotated as a non-open container. If 3 or more subjects annotate an object as an open container, the ground truth label of the object is an open container; otherwise, the label is a non-open container. The test set contains 55 open containers and 75 non-open containers.

C. Physical Simulation

We use Pybullet [36] as the physical engine for open containability imagination and pouring imagination. The object, cup, and particles are imported with the URDF files which specify the mass, CoM, inertia matrix, collision model, friction coefficient, and joint properties. The particles are modelled as ellipsoids with similar dimensions of an M&M’s® candy. We use Volumetric Hierarchical Approximate Convex Decomposition (V-HACD) [37] to decompose the model of the object and the cup into convex pieces as the object collision model in the simulation. The default Coulomb friction model is used for friction modeling. The collision between the particles and the object is modelled as inelastic (coefficient of restitution $e = 0.1$).

We calibrate the simulation by running the algorithm on the training set. The simulation parameters of the open containability imagination are manually calibrated such that the open container classification result matches with the human annotation. The simulation parameters of the pouring imagination are manually calibrated such that the pouring results, i.e. $P$ in Eqn. 4, matches with the real-world pouring results to the greatest extent. From the simulation calibration, $\Omega > 0$ for all the open containers in the training set; $\Omega = 0$ for all the non-open containers. Thus, we set $\Omega_{br} = 0$ which physically means that an object is classified as an open container if there are particles retained within the object after the drop and non-open container if otherwise. We set $N_{pour} = 60$ for all pourings in the pouring imagination, a number of which the volume can be contained by all the open containers in the dataset. However, we want to point out that $N_{pour}$ can be set according to how much the user wants to pour and the containing volume of the object, which we did not cover in this paper. A further investigation of reasoning $N_{pour}$ is left for future work. The cup used in pouring imagination has the same dimensions as the one used in robot autonomous pouring (Sec. IV-D).

D. Robot Autonomous Pouring

In robot autonomous pouring, the robot pours the same number of candies as in pouring imagination. At the beginning of each experiment, we fill the cup with candies and place it at a pre-defined position. After the pouring imagination, the robot first picks up the cup and execute a pre-defined trajectory to move the cup to a pre-pour pose. We did not directly ask whether the object is an open container because the concept of open container is not clear to some human subjects.
The trajectory is designed such that the cup does not spill the candies during the execution. Then, the robot is controlled to move the cup to the imagined initial pose of pouring $(R_{init}, l_{init})$ obtained from the pouring imagination (Eqn. 6). After that, similar to the pouring imagination, the robot rotates the cup about $l_{pour}$ to pour the candies (Fig. 6(e)). We use the python-urx library [38] to control and generate motion plans for the robot.

V. RESULTS

In this section, we show the results of open container classification and robot autonomous pouring on the test set. We evaluate our method on a computer running Intel Core i7-8700 @ 3.2GHz CPU. Our single-threaded unoptimized implementation takes about 50 seconds for 3D scanning, 15 seconds for open containability imagination, 80 seconds for pouring imagination, and 40 seconds for autonomous pouring. We denote our method as Imagination. The most comparable method is [10]. However, its source code is not available for comparison. Thus, we compare with a deep learning method [6], denoted as AffordanceNet, which we find most relevant in the literature. AffordanceNet takes as input an RGB image. It has an object detection branch which localizes the object with a bounding box and classifies the object and an affordance detection branch which assigns each pixel within the bounding box with an affordance label. AffordanceNet is trained on the IIT-AFF dataset [14] which contains 8,835 real-world images with 10 object classes and 9 affordance classes. Among the 10 object classes, 3 classes are open containers (cup, bowl, pan). Among the 9 affordance classes, one of them is “contain”. To ensure fairness, the comparison with AffordanceNet is evaluated on a subset of the full test set, denoted as the restricted test set. It contains only 51 objects (23 open containers (Fig. 6(b)) and 28 non-open containers) which falls in the object classes of the IIT-AFF dataset. For each of these objects, at least four of the five human annotators were in agreement.

A. Open Container Classification

We use the classification accuracy (accuracy) and area under the Receiver Operating Characteristic curve (AUC) to evaluate the open container classification performance of different methods. The results are shown in Table I. For our proposed method, the open containability $\Omega$ (Eqn. 1) is used as the confidence score to calculate the AUC. For AffordanceNet, we use its object detection branch for open container classification. Although our method only uses the RGB-D camera, we also save the RGB images. These RGB images are used for testing AffordanceNet. Our method segments the RGB-D images for each view before inputting into AffordanceNet. From the output of AffordanceNet’s detection branch, we first select the detection (bounding box + classification score) with the highest classification score. The maximum classification score of the 3 open container classes of this detection is used as the confidence score of the object being an open container. We evaluate the classification accuracy and AUC with AffordanceNet for the 24 views respectively, resulting in 24 accuracies and AUCs in total. The highest accuracy and AUC is selected as the accuracy and AUC of AffordanceNet.

| Method             | Dataset    | Accuracy | AUC   |
|--------------------|------------|----------|-------|
| AffordanceNet      | restricted | 100.00   | 100.00|
| Imagination        | restricted | 100.00   | 100.00|
| Imagination        | full       | 96.15    | 100.00|

Both our method and AffordanceNet achieve perfect performance on the restricted test set. On the full test set, our method fails to classify some objects (indicated with green I’s in Fig. 6(d)), labelled as non-open containers, the same way as the human annotation for two reasons. Firstly, the reconstructed 3D model of the object traps particles with small concavities due to inaccurate reconstruction and

---

3Videos are available at: [https://chirikjianlab.github.io/realcontainerimagination/](https://chirikjianlab.github.io/realcontainerimagination/)

4We use the code and pretrained weight provided by the author for testing.
approximate convex decomposition. Secondly, the object (e.g., a spoon) possesses small concavities which can retain particles. Interestingly, the spoon, classified as an open container by our method, is annotated as a non-open container by 3 out of 5 human subjects, i.e., whether it is an open container diverges among human subjects.

B. Robot Autonomous Pouring

We use the method described in Sec. [II-B] and [IV-D] to pour candies into the open containers in the test set. We compare with AffordanceNet by using the affordance detection branch. We derive \((\theta_{\text{pour}}, p_{\text{pour}})\) with the affordance detection and use the same method described in Sec. [IV-D] to pour. The view with the highest AUC is used to test AffordanceNet. It is a top-down view which is similar to the view used for pouring in [6]. As in the open container classification, we crop the object from the image to ensure fair comparison. Using the same method described in [6], we set \(p_{\text{pour}}\) by calculating the centroid of all the pixels predicted as the “contain” affordance in 2D and project it to 3D with the corresponding depth image and camera intrinsic parameters. However, directly pouring at the projected point will result in collision with the object if the point is on the object surface. Therefore, we set the \(z\) component of \(r_{\text{pivot}}\) by adding a fixed offset \(\Delta h\) to the \(z\) component of the projected point. Since AffordanceNet is not able to specify \(\theta_{\text{pour}}\), we use a fixed \(\theta_{\text{pour}}\) in the experiment. We also compare with a variation of AffordanceNet, denoted as AffordanceNet w/ 3D Scanning which incorporates the robot 3D scanning module in our method to provide the \(z\) component of \(r_{\text{pivot}}\). Similar to our method, the \(z\) component is set such that \(r_{\text{pivot}}\) is 1cm above the object AABB. In addition, we compare to a variation of our method, denoted as Imagination w/ fixed \(\theta_{\text{pour}}\), in which \(\theta_{\text{pour}}\) is fixed and equals to that used for AffordanceNet. We want to study the importance of reasoning about pouring orientations.

| Method                      | Dataset      | Success Rate |
|-----------------------------|--------------|--------------|
| AffordanceNet               | restricted   | 82.61        |
| AffordanceNet w/ 3D Scanning| restricted   | 91.30        |
| Imagination w/ fixed \(\theta_{\text{pour}}\)| restricted   | 100.00       |
| Imagination                 | restricted   | 100.00       |
| Imagination w/ fixed \(\theta_{\text{pour}}\)| full         | 94.55        |
| Imagination                 | full         | 98.18        |

For AffordanceNet and its variation, if the network detects “contain” affordance for an object and pour without spillage, we consider the pouring a success and failure if otherwise. For our method and its variation, if the open container imagination classifies the object as an open container and pour without spillage, we consider the pouring a success and failure if otherwise. The results are shown in Table II. Failure cases of all methods are shown in Fig. [5](b)(c). Since AffordanceNet is not able to specify the \(z\) component of \(r_{\text{pivot}}\), it struggles with very shallow containers (e.g., pans) due to candies bouncing off the object. With the robot 3D scanning to provide the object height, AffordanceNet w/ 3D scanning is able to cope with this problem but still fails on some objects (e.g., the wooden bowl with a similar color as the table in Fig. [5](b)) due to incorrect affordance detection. Both Imagination and its variation perform perfectly on the restricted test set. On the full test set, the variation fails on a gravy boat with an opening resembling a slim rectangle. Without the reasoning of \(\theta_{\text{pour}}\), it is not able to find the pouring orientation in which the cup mouth points along the length of the rectangle, providing larger tolerances for the randomness in the candy’s pouring trajectory. The only failure case for Imagination on the full test set is a vase with a small opening which was reconstructed as a dent with very limited depth due to limitations on the 3D scanning. The robot classifies it correctly but fails to pour without spillage.

VI. DISCUSSION & FUTURE WORK

Despite only trained with 11 objects, our method is able to achieve the same open container classification performance as the deep learning method, which has been trained with thousands of annotated images, on the restricted test set. Moreover, our method’s performance on the full test set aligns well with human judgements. The pouring experiment results show that our method outperforms the deep learning method on the restricted test set and is able to maintain similar performance on the full test set. Both the classification and pouring results of our method on the full test set, which covers 57 object categories, show that our method is able to achieve inter-class function (open containability) generalization which is challenging for appearance-based method for the reasons of data burden and the implicit nature of appearance representation. We believe this generalization is a fundamental strength of the affordance-based object perception which physically captures the most essential cue of open containers through reasoning interactions with objects.

Unlike the black box nature of the deep learning method, our method is explainable. It explains the open containability affordance of an object by the number of particles the object is able to retain in the physical simulation. Our method explores object affordances based on object instances instead of object categories. For objects which do not belong to a typical open container categories but are able to afford open containability, e.g., the candlestick in Fig. [5], our method is able to improvise and recognize its affordance. This provides robots with intelligence to tackle previously unseen object instances and/or categories when deployed in our daily life.

In future study, we plan to introduce more accurate and adaptive scanning to tackle the failure cases in the classification and pouring experiments. We also plan to combine reasoning the amount of particles to pour with the evaluation of the open container’s volume for more adaptive pouring.

5 [6] demonstrates robot pouring using the affordance detection branch. However, the authors did not do quantitative experiments and the source code for pouring is not available. The code for testing AffordanceNet on pouring is implemented based on the description in the paper and correspondence with the author.

6This view also has the highest classification accuracy.
Finally, we want to cut down the runtime via parallelization of pouring from different poses in the pouring imagination.

VII. CONCLUSIONS

In this work, we develop a novel method for robots to imagine the open containability of previously unseen objects via physical simulations. Our method is able to perform open container classification and endows robots with the capability to autonomously pour into objects via open containability imagination and pouring imagination, respectively. We evaluate our method on a dataset containing 130 previously unseen objects with 57 object categories. Results show that our method’s performance on open container classification aligns well with human judgements and it is able to pour into the 55 open containers in the dataset with a success rate of 98.18%. This ability to access and generalize function to previously unseen object instances and object category provides robots with high intelligence on robot-object interaction. We hope that our method will serve as an effective approach to reinforce robot-object interaction in future research.

REFERENCES

[1] J. J. Gibson, The ecological approach to visual perception. Boston, MA, USA: Houghton Mifflin, 1979.
[2] J. J. DeCarlo, D. Zoccoli, and N. C. Rust, “How does the brain solve visual object recognition?”, Neuron, vol. 73, no. 3, pp. 415–434, 2012.
[3] H. Wu, D. Misra, and G. S. Chirikjian, “Is that a chair? imagining affordances using simulations of an articulated human body,” arXiv preprint arXiv:1909.07572, 2019.
[4] F. W. Battaglia, J. B. Hamrick, and J. B. Tenenbaum, “Simulation as an engine of physical scene understanding,” Proceedings of the National Academy of Sciences, vol. 110, no. 45, pp. 18327–18332, 2013.
[5] K. M. Varadarajan and M. Vincez, “Afnet: The affordance network,” in Asian Conference on Computer Vision. Springer, 2012, pp. 512–523.
[6] T.-T. Do, A. Nguyen, and I. Reid, “Afferencenet: An end-to-end deep learning approach for object affordance detection,” in 2018 IEEE international conference on robotics and automation (ICRA). IEEE, 2018, pp. 1–5.
[7] W. Liang, Y. Zhao, Y. Zhu, and S.-C. Zhu, “Evaluating human cognition of containing relations with physical simulation.” inCogSci, 2015.
[8] E. Davis, G. Marcus, and A. Chen, “Reasoning from radically incomplete information: The case of containers,” Advances in Cognitive Systems, vol. 2, pp. 1–18, 2013.
[9] S. Ullman, N. Dorfman, and D. Harari, “A model for discovering ‘containment relations’,” Cognition, vol. 183, pp. 67–81, 2019.
[10] L.-F. Yu, N. Duncan, and S.-K. Yeng, “Fill and transfer: A simple physics-based approach for containability reasoning,” in Proceedings of the IEEE international conference on computer vision (ICCV), 2015, pp. 711–719.
[11] L. Hinkle and E. Olson, “Predicting object functionality using physical simulations,” in 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2013, pp. 2784–2790.
[12] W. Liang, Y. Zhu, and S.-C. Zhu, “Tracking occluded objects and recovering incomplete trajectories by reasoning about containment relations and human actions,” in Thirty-Second AAAI Conference on Artificial Intelligence, 2018.
[13] A. Myers, C. L. Teo, C. Fermüller, and Y. Aloimonos, “Affordance detection of tool parts from geometric features,” in 2015 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2015, pp. 1374–1381.
[14] A. Nguyen, D. Kanoulas, D. G. Caldwell, and N. G. Tsagarakis, “Object-based affordances detection with convolutional neural networks and dense conditional random fields,” in 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2017, pp. 5908–5915.
[15] F.-J. Chu, R. Xu, and P. A. Vela, “Learning affordance segmentation for real-world robotic manipulation via synthetic images,” IEEE Robotics and Automation Letters, vol. 4, no. 2, pp. 1140–1147, 2019.
[16] J. Sawatzky, A. Srikantha, and J. Gall, “Weakly supervised affordance detection,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 2795–2804.
[17] A. Roy and S. Todorovic, “A multi-scale cnn for affordance segmentation in rgb images,” in European conference on computer vision (ECCV). Springer, 2016, pp. 186–201.
[18] I. Manuelli, W. Gao, P. Florence, and R. Tedrake, “kpam: Keypoint affordances for category-level robotic manipulation,” in International Symposium on Robotics Research (ISRR), 2019.
[19] E. Ruiz and W. Mayol-Cuevas, “Geometric affordance perception: leveraging deep 3d saliency with the interaction tensor,” Frontiers in Neuroinformatics, 2020.
[20] Y. Zhu, A. Fathi, and L. Fei-Fei, “Reasoning about object affordances in a knowledge base representation,” in European conference on computer vision (ECCV). Springer, 2014, pp. 408–424.
[21] Y. Zhu, “Visual commonsense reasoning: Functionality, physics, causality, and utility,” Ph.D. dissertation, The University of Wisconsin-Madison, 1987.
[22] L. Kunze and M. Beetz, “Envisioning the qualitative effects of robot manipulation actions using simulation-based projections,” Artificial Intelligence, vol. 247, pp. 352–380, 2017.
[23] S.-B. Ho, “Representing and using functional definitions for visual recognition,” Ph.D. dissertation, The University of Wisconsin-Madison, 2018.
[24] P. Abelha and F. Guerin, “Learning how a tool affords by simulating 3d models from the web,” in 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2017, pp. 4923–4929.
[25] S. Boschert and R. Rosen, “Digital twin—the simulation aspect,” in Mechatronic futures. Springer, 2016, pp. 59–74.
[26] T. L. Guevara, N. K. Taylor, M. U. Gutmann, S. Ramamoorthy, and K. Subr, “Adaptable pouring: Teaching robots not to spill using fast but approximate fluid simulation,” in Proceedings of the Conference on Robot Learning (CoRL), 2017.
[27] A. Yamaguchi and C. G. Atkeson, “Differential dynamic programming with temporally decomposed dynamics,” in 2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids). IEEE, 2015, pp. 696–703.
[28] C. Schenck and D. Fox, “Visual closed-loop control for pouring liquids,” in 2017 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2017, pp. 2629–2636.
[29] Z. Pan and D. Manocha, “Motion planning for fluid manipulation using simplified dynamics,” in 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2016, pp. 4224–4231.
[30] M. Kennedy, K. Schmeckpeper, D. Thakur, C. Jiang, V. Kumar, and K. Daniilidis, “Autonomous precision pouring from unknown containers,” IEEE Robotics and Automation Letters, vol. 4, no. 3, pp. 2317–2324, 2019.
[31] B. Curless and M. Levoy, “A volumetric method for building complex models from range images,” in Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, 1996, pp. 303–312.
[32] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser, “3dmatch: Learning local geometric descriptors from rgb-d reconstructions,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.
[33] R. Finnan, T. Whelan, M. Kaess, and J. J. Leonard, “Toward lifelong object segmentation from change detection in dense rgb-d maps,” in 2013 European Conference on Mobile Robots. IEEE, 2013, pp. 178–185.
[34] M. Krainin, P. Henry, X. Ren, and D. Fox, “Manipulator and object tracking for in-hand 3d object modeling,” The International Journal of Robotics Research, vol. 30, no. 11, pp. 1311–1327, 2011.
[35] K. Welke, J. Issac, D. Schiebener, T. Asfour, and R. Dillmann, “Autonomous acquisition of visual multi-view object representations for object recognition on a humanoid robot,” in 2010 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2010, pp. 2012–2019.
[36] E. Coumans and Y. Bai, “Pybullet, a python module for physics simulation for games, robotics and machine learning.” http://pybullet.org, 2016–2018.
[37] K. Mamou, “Volumetric hierarchical approximate convex decomposition,” Game Engine Gems 3, pp. 141–158, 2016.
[38] Olivier Roulet-Dubonnet et al., “python-urx.” [Online]. Available: https://github.com/SintefManufacturing/python-urx