VAT-MART: LEARNING VISUAL ACTION TRAJECTORY PROPOSALS FOR MANIPULATING 3D ARTICULATED OBJECTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Perceiving and manipulating 3D articulated objects (*e.g.*, cabinets, doors) in human environments is an important yet challenging task for future home-assistant robots. The space of 3D articulated objects is exceptionally rich in their myriad semantic categories, diverse shape geometry, and complicated part functionality. Previous works mostly abstract kinematic structure with estimated joint parameters and part poses as the visual representations for manipulating 3D articulated objects. In this paper, we propose *object-centric actionable visual priors* as a novel perception-interaction handshaking point that the perception system outputs more actionable guidance than kinematic structure estimation, by predicting *dense geometry-aware*, *interaction-aware*, and *task-aware* visual action affordance and trajectory proposals. We design an *interaction-for-perception* framework VAT-MART to learn such actionable visual representations by simultaneously training a curiosity-driven reinforcement learning policy exploring diverse interaction trajectories and a perception module summarizing and generalizing the explored knowledge for pointwise predictions among diverse shapes. Experiments prove the effectiveness of the proposed approach using the large-scale PartNet-Mobility dataset in SAPIEN environment and show promising generalization capabilities to novel test shapes, unseen object categories, and real-world data.

1 INTRODUCTION

We live in a 3D world composed of a plethora of 3D objects. To help humans perform everyday tasks, future home-assistant robots need to gain the capabilities of perceiving and manipulating a wide range of 3D objects in human environments. Articulated objects that contain functionally important and semantically interesting articulated parts (*e.g.*, cabinets with drawers and doors) especially require significantly more attention, as they are more often interacted with by humans and artificial intelligent agents. Having much higher degree-of-freedom (DoF) state spaces, articulated objects are, however, generally more difficult to understand and subsequently to interact with, compared to 3D rigid objects that have only 6-DoF for their global poses.

There has been a long line of research studying the perception and manipulation of 3D articulated objects in computer vision and robotics. On the perception side, researchers have developed various successful visual systems for estimating kinematic structures (Abbatematteo et al., 2019; Staszak et al., 2020), articulated part poses (Li et al., 2020; Jain et al., 2020; Liu et al., 2020), and joint parameters (Wang et al., 2019b; Yan et al., 2019). Then, with these estimated visual articulation models, robotic manipulation planners and controllers can be leveraged to produce action trajectories for robot executions (Klingbeil et al., 2010; Arduengo et al., 2019; Florence et al., 2019; Urakami et al., 2019; Mittal et al., 2021). While the commonly used two-stage solution underlying most of these systems reasonably breaks the whole system into two phases and thus allows bringing together well-developed techniques from vision and robotics communities, the current handshaking point – *the standardized visual articulation models* (*i.e.* kinematic structure, articulated part poses, and joint parameters), may not be the best choice, since essential geometric and semantic features for robotic manipulation tasks, such as interaction hotspots (*e.g.* edges, holes, bars) and part functionality (*e.g.* handles, doors), are inadvertently abstracted away in these canonical representations.

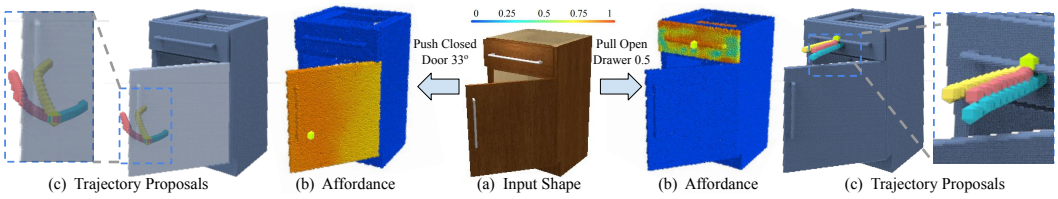

Figure 1: Given an input 3D articulated object (a), we propose a novel perception-interaction handshaking point for robotic manipulation tasks - *object-centric actionable visual priors*, including per-point *visual action affordance predictions* (b) indicating *where* to interact, and *diverse trajectory proposals* (c) for selected contact points (marked with green dots) suggesting *how* to interact.

We propose a new type of *actionable visual representations* (Do et al., 2018; Nagarajan et al., 2019; Mo et al., 2021) exploring a more *geometry-aware*, *interaction-aware*, and *task-aware* perception-interaction handshaking point for manipulating 3D articulated objects. Concretely, we train the perception system to predict action possibility and visual action trajectory proposals at every point over parts of 3D articulated objects (See Figure 1). In contrast to previous work that use standardized visual articulation models as visual representations, our framework VAT-MART predicts per-point dense action trajectories that are adaptive to the change of geometric context (*e.g.*, handles, door edges), interactions (*e.g.*, pushing, pulling), and tasks (*e.g.*, open a door for 30°, close up a drawer by 0.1-unit-length). Abstracting away from concrete external manipulation environments, such as robot arm configurations, robot base locations, and scene contexts, we aim for learning unified *object-centric visual priors* with *a dense* and *diverse* superset of visual proposals that can be potentially applied to different manipulation setups, avoiding learning separate manipulation representations under different circumstances.

The proposed *actionable visual priors*, as a "preparation for future tasks" (Ramakrishnan et al., 2021) or "visually-guided plans" (Wang et al., 2019a; Karamcheti et al., 2021), can provide informative guidance for downstream robotic planning and control. Sharing a similar spirit with Nagarajan et al. (2019); Mo et al. (2021), we formulate our visual action possibility predictions as per-point affordance maps, on which the downstream robotic planners may sample a position to interact according to the predicted likelihood of success. Then, for a chosen point for interaction, the discrete task planner may search for applicable interaction modes (*e.g.*, whether to attempt a grasp) within a much smaller space formed by the visual action trajectory distribution, instead of searching in the entire solution space. Next, considering the robot kinematic constraints and physic collisions, the continuous motion planner can further select an open-loop trajectory from the set of proposed visual action trajectory candidates as an initial value for optimization, and finally pass to the robot controller for execution. More recent reinforcement learning (RL) based planners and controllers can also benefit from our proposed solution spaces for more efficient exploration.

To obtain such desired *actionable visual priors*, we design an *interaction-for-perception* learning framework VAT-MART, as shown in Figure 2. By conducting trial-and-error manipulation with a set of diverse 3D articulated objects, we train an RL policy to learn successful interaction trajectories for accomplishing various manipulation tasks (*e.g.*, open a door for 30°, close up a drawer by 0.1-unit-length). In the meantime, the perception networks are simultaneously trained to summarize the RL discoveries and generalize the knowledge across points over the same shape and among various shapes. For discovering diverse trajectories, we leverage curiosity feedback (Pathak et al., 2017) for enabling the learning of perception networks to reversely affect the learning of RL policy.

We conduct experiments using SAPIEN (Xiang et al., 2020) over the large-scale PartNet-Mobility (Chang et al., 2015; Mo et al., 2019) dataset of 3D articulated objects. We use 562 shapes in 7 object categories to perform our experiments and show that our VAT-MART framework can successfully learn the desired *actionable visual priors*. We also observe reasonably good generalization capabilities over unseen shapes, novel object categories, and real-world data, thanks to large-scale training over diverse textureless geometry.

In summary, we make the following contributions in this work:

- We formulate a novel kind of *actionable visual priors* making one more step towards bridging the perception-interaction gap for manipulating 3D articulated objects;

- We propose an *interaction-for-perception* framework VAT-MART to learn such priors with novel designs on the joint learning between exploratory RL and perception networks;
- Experiments conducted over the PartNet-Mobility dataset in SAPIEN demonstrate that our system works at a large scale and learns representations that generalize over unseen test shapes, across object categories, and even real-world data.

2 RELATED WORK

Perceiving and Manipulating 3D Articulated Objects has been a long-lasting research topic in computer vision and robotics. A vast literature (Yan & Pollefeys, 2006; Katz et al., 2008; Sturm et al., 2009; 2011; Huang et al., 2012; Katz et al., 2013; Martin & Brock, 2014; Höfer et al., 2014; Katz et al., 2014; Schmidt et al., 2014; Hausman et al., 2015; Martín-Martín et al., 2016; Tzionas & Gall, 2016; Paolillo et al., 2017; Martın-Martın & Brock, 2017; Paolillo et al., 2018; Martín-Martín & Brock, 2019; Desingh et al., 2019; Nunes & Demiris, 2019) has demonstrated successful systems, powered by visual feature trackers, motion segmentation predictors, and probabilistic estimators, for obtaining accurate link poses, joint parameters, kinematic structures, and even system dynamics of 3D articulated objects. Previous works (Peterson et al., 2000; Jain & Kemp, 2009; Chitta et al., 2010; Burget et al., 2013) have also explored various robotic planning and control methods for manipulating 3D articulated objects. More recent works further leveraged learning techniques for better predicting articulated part configurations, parameters, and states (Wang et al., 2019b; Yan et al., 2019; Jain et al., 2020; Zeng et al., 2020; Li et al., 2020; Liu et al., 2020; Mu et al., 2021), estimating kinematic structures (Abbatematteo et al., 2019; Staszak et al., 2020), as well as manipulating 3D articulated objects with the learned visual knowledge (Klingbeil et al., 2010; Arduengo et al., 2019; Florence et al., 2019; Urakami et al., 2019; Mittal et al., 2021). While most of these works represented visual data with link poses, joint parameters, and kinematic structures, such standardized abstractions may be insufficient if fine-grained part geometry, such as drawer handles and faucet switches that exhibit rich geometric diversity among different shapes, matters for downstream robotic tasks.

Learning Actionable Visual Representations aims for learning visual representations that are strongly aware of downstream robotic manipulation tasks and directly indicative of action probabilities for robotic executions, in contrast to predicting standardized visual semantics, such as category labels (Russakovsky et al., 2015; Wu et al., 2015b), segmentation masks (Lin et al., 2014; Mo et al., 2019), and object poses (Hinterstoisser et al., 2011; Xiang et al., 2016), which are usually defined independently from any specific robotic manipulation task. Grasping (Montesano & Lopes, 2009; Lenz et al., 2015; Mahler et al., 2017; Fang et al., 2020; Mandikal & Grauman, 2021; Corona et al., 2020; Kokic et al., 2020; Yang et al., 2020; Jiang et al., 2021) or manipulation affordance (Kjellström et al., 2011; Do et al., 2018; Fang et al., 2018; Goff et al., 2019; Nagarajan et al., 2019; Nagarajan & Grauman, 2020; Nagarajan et al., 2020; Xu et al., 2021; Mo et al., 2021) is one major kind of actionable visual representations, while many other types have been also explored recently (*e.g.*, spatial maps (Wu et al., 2020; 2021), keypoints (Wang et al., 2020; Qin et al., 2020), contact points (You et al., 2021), etc). Following the recent work Where2Act (Mo et al., 2021), we employ dense affordance maps as the actionable visual representations to suggest action possibility at every point on 3D articulated objects. Extending beyond Where2Act which considers task-less short-term manipulation, we further augment the per-point action predictions with task-aware distributions of trajectory proposals, providing more actionable information for downstream executions.

Learning Perception from Interaction augments the tremendously successful learning paradigm using offline curated datasets (Russakovsky et al., 2015; Lin et al., 2014; Chang et al., 2015; Mo et al., 2019) by allowing learning agents to collect online active data samples, which are more task-aware and learning-efficient, during navigation (Anderson et al., 2018; Ramakrishnan et al., 2021), recognition (Wilkes & Tsotsos, 1992; Yang et al., 2019; Jayaraman & Grauman, 2018), segmentation (Pathak et al., 2018; Gadre et al., 2021), and manipulation (Pinto et al., 2016; Bohg et al., 2017). Many works have also demonstrated the usefulness of simulated interactions for learning perception (Wu et al., 2015a; Mahler et al., 2017; Xu et al., 2019; Ramakrishnan et al., 2021; Nagarajan & Grauman, 2020; Lohmann et al., 2020; Mo et al., 2021) and promising generalizability to the real world (James et al., 2019; Chebotar et al., 2019; Hundt et al., 2019; Liang et al., 2020; Kadian et al., 2020; Anderson et al., 2020; Rao et al., 2020). Our method follows the route of learning perception from interaction via using the action trajectories discovered by an RL interaction policy to supervise a jointly trained perception system, which reversely produces curiosity feedback (Pathak et al., 2017) to encourage the RL policy to explore diverse action proposals.

3 ACTIONABLE VISUAL PRIORS: *Action Affordance and Trajectory Proposals*

We propose novel *actionable visual representations* for manipulating 3D articulated objects (see Fig. 1). For each articulated object, we learn *object-centric actionable visual priors*, which are comprised of: 1) an *actionability map* over articulated parts indicating *where* to interact; 2) per-point distributions of *visual action trajectory proposals* suggesting *how* to interact; and 3) estimated success likelihood scores rating the *outcomes* of the interaction. All predictions are *interaction-conditioned* (*e.g.*, pushing, pulling) and *task-aware* (*e.g.*, open a door for 30°, close a drawer by 0.1-unit-length).

Concretely, given a 3D articulated object O with its articulated parts $\mathscr{P} = \{P_1, P_2, \cdots\}$, an interaction type T, and a manipulation task θ, we train a perception system that makes *dense* predictions at each point p over each articulated part $p \in \cup\mathscr{P}$: 1) an actionability score $a_{p|O,T,\theta} \in [0,1]$ indicating how likely there exists an action trajectory of interaction type T at point p that can successfully accomplish the task θ; 2) a distribution of *visual action trajectories* $\mathbb{P}_p(\cdot|O,T,\theta)$, from which we can sample *diverse* action trajectories $\tau_{p|O,T,\theta} \sim \mathbb{P}_p(\cdot|O,T,\theta)$ of interaction type T to accomplish the task θ at point p; and 3) a per-trajectory *success likelihood score* $r_{\tau|O,p,T,\theta} \in [0,1]$.

Inputs. We represent the input 3D articulated object O as a partial point cloud S_O. We consider two typical interaction types: *pushing* and *pulling*. A *pushing* trajectory maintains a closed gripper and has 6-DoF motion performing the pushing, whereas a *pulling* trajectory first performs a grasping operation at the point of interaction by closing an initially opened gripper and then has the same 6-DoF motion during the pulling. For articulated objects we use in this work, we only consider 1-DoF part articulation and thus restrict the task specification $\theta \in \mathbb{R}$. For example, a cabinet drawer has a 1-DoF prismatic translation-joint and a refrigerator door is modeled by a 1-DoF revolute hinge-joint. We use the absolute angular degrees in radian (*i.e.* $\theta \in [-\pi, \pi]$) for revolute joints and use the units of length (*i.e.* $\theta \in [-1, 1]$) relative to the global shape scale for prismatic joints.

Outputs. Both the actionability score $a_{p|O,T,\theta}$ and per-trajectory success likelihood score $r_{\tau|O,p,T,\theta}$ are scalars within $[0,1]$, where larger values indicate higher likelihood. One can use a threshold of 0.5 to obtain binary decisions if needed. Every action trajectory $\tau_{p|O,T,\theta}$ is a sequence of 6-DoF end-effector waypoints $(wp_0, wp_1, \cdots, wp_k)$, with variable trajectory length ($k \leq 5$). In our implementation, we adopt a residual representation $(wp_0, wp_1 - wp_0, \cdots, wp_k - wp_{k-1})$ for the action trajectory, as it empirically yields better performance. Each 6-DoF waypoint is comprised of a 3-DoF robot hand center x and 3-DoF orientation R. We use the 6D-rotation representation (Zhou et al., 2019) for the orientation of wp_0 and predict 3-DoF euler angles for subsequent orientation changes.

4 VAT-MART: AN *Interaction-for-perception* LEARNING FRAMEWORK

The VAT-MART system (Fig. 2) consists of two parts: an RL policy exploring diverse action trajectories and a perception system learning the proposed *actionable visual priors*. While the RL policy collects interaction trajectories for supervising the perception networks, the perception system provides curiosity feedback (Pathak et al., 2017) for encouraging the RL policy to further explore diverse solutions. In our implementation, we first pretrain the RL policy, then train the perception network with RL-collected data, and finally finetune the two parts jointly with curiosity-feedback enabled. We describe key system designs below and will release code for our implementation.

4.1 THE RL POLICY FOR INTERACTIVE TRAJECTORY EXPLORATION

For every interaction type T and part articulation type, we train a single conditional RL policy using TD3 (Fujimoto et al., 2018) to collect trajectories τ that can accomplish the interaction of varying task specifications θ across all shapes O and contact points p. In the RL training, since the RL policy is trained in the simulation for only collecting training data to supervise the perception networks, we can have access to the ground-truth state information of the simulation environment, such as the part poses, joint axis, and gripper poses. At the test time, we discard the RL network and only use the learned perception networks to predict the proposed *actionable visual priors*.

Fig. 2 (left) illustrates the RL training scheme and example explored diverse trajectory proposals for the task of pulling open the microwave door for 30 degrees. Below, we describe the RL specifications.

Task Initialization. For the shape to interact with, we first randomly sample a shape category with equal probability, alleviating the potential category imbalance issue, and then uniformly sample a

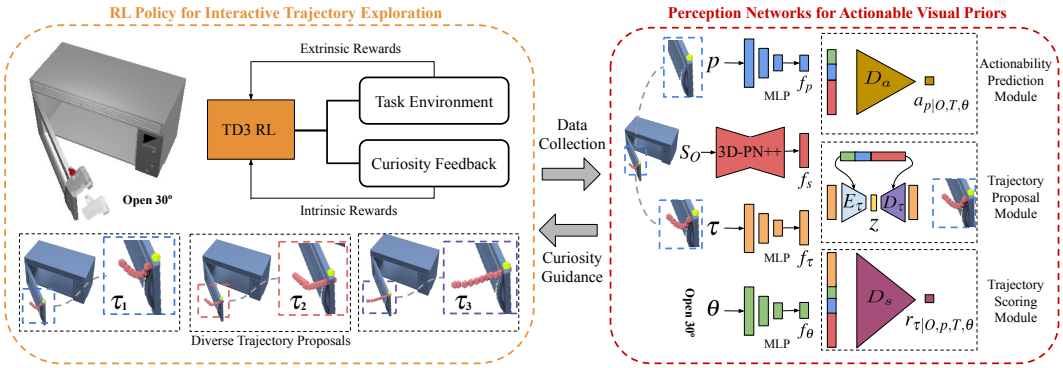

Figure 2: Our proposed VAT-MART framework is composed of an RL policy (left) exploring interaction trajectories and a perception system (right) learning the desired actionable visual priors. We build bidirectional supervisory channels between the two parts: 1) the RL policy collects data to supervise the perception system, and 2) the perception system produces curiosity feedbacks encouraging the RL networks to explore diverse solutions.

shape from the selected category for training. For the task specification θ, we randomly sample within $[10°, 70°]$ for revolute parts and $[0.1, 0.7]$ for prismatic parts. We also randomly sample a starting part pose θ_0 with the guarantee that the task θ can be accomplished. For the gripper, we initialize it with fingertip 0.02-unit-length away from the contact point p and pointing within a cone of 30 degrees along the negative direction of the surface normal at p.

State Space. The RL state includes the 1-DoF part pose change $\Delta\theta_i = \theta_i - \theta_0$ at the current timestep, the target task θ, the difference $\theta - \Delta\theta_i$, the gripper pose $wp_0 = (x_0, R_0) \in SE(3)$ at the first timestep of interaction, the current gripper pose $wp_i = (x_i, R_i) \in SE(3)$, the local positions for the gripper fingers $x_f \in \mathbb{R}^2$, the current contact point location $p_i \in \mathbb{R}^3$, a normalized direction of the articulated part joint axis $n_j \in \mathbb{R}^3$, the articulated part joint location $x_j \in \mathbb{R}^3$ (defined as the closest point on the joint axis to the contact point p), the closest distance from the contact point to the joint axis $d_{cj} \in \mathbb{R}$, and a directional vector $n_{cj} \in \mathbb{R}^3$ from the joint location to the contact point. We concatenate all the information together as a 33-dimensional state vector for feeding to the RL networks.

Action Space. At each timestep, we predict a residual gripper pose $wp_i - wp_{i-1} \in SE(3)$ to determine the next-step waypoint wp_i as the action output of the RL networks. We estimate a center offset $x_i - x_{i-1} \in \mathbb{R}^3$ and an euler angle difference $R_i - R_{i-1} \in SO(3)$.

Reward Design. There are two kinds of rewards: extrinsic task rewards and intrinsic curiosity feedbacks. For the extrinsic task rewards, we use: 1) a final-step success reward of 500 for a task completion when the current part pose reaches the target within 15% relative tolerance range, 2) a step-wise guidance reward of $300(|\theta - \Delta\theta_{i-1}| - |\theta - \Delta\theta_i|)$ encouraging the current part pose to get closer to the target than previous part pose, and 3) a distance penalty of $300 \cdot \mathbb{1}[d_{gc} > 0.1] + 150 d_{gc}$ to discourage the gripper from flying away from the intended contact point p, where d_{gc} denotes the l_2 distance from the contact point p to the current fingertip position and $\mathbb{1}[q]$ is a zero or one function indicating the boolean value of the predicate q. We will describe the curiosity rewards in Sec. 4.3.

Stopping Criterion. We stop an interaction trial until the task's success or after five maximal waypoint steps.

Implementation and Training. We implement the TD3 networks using MLPs and use a replay buffer of size 2048. To improve the positive data rates for efficient learning, we leverage Hindsight Experience Replay (Andrychowicz et al., 2017): an interaction trial may fail to accomplish the desired task θ, but it finally achieves the task of $\theta_k - \theta_0$. See Sec. A in appendix for more details.

4.2 THE PERCEPTION NETWORKS FOR ACTIONABLE VISUAL PRIORS

The perception system learns from the interaction trajectories collected by the RL exploration policy and predicts the desired *per-pixel actionable visual priors*. Besides several information encoding

modules, there are three decoding heads: 1) an actionability prediction module that outputs the actionability score $a_{p|O,T,\theta} \in [0,1]$, 2) a trajectory proposal module that models per-point distribution of diverse visual action trajectories $\tau_{p|O,T,\theta} = (wp_0, wp_1 - wp_0, \cdots, wp_k - wp_{k-1})$, and 3) a trajectory scoring module that rates the per-trajectory success likelihood $r_{\tau|O,p,T,\theta} \in [0,1]$. Fig. 2 (right) presents an overview of the system. Below we describe detailed designs and training strategies.

Input Encoders. The perception networks require four input entities: a partial object point cloud S_O, a contact point p, a trajectory τ, and a task θ. For the point cloud input S_O, we use a segmentation-version PointNet++ (Qi et al., 2017) to extract a per-point feature $f_s \in \mathbb{R}^{128}$. We employ three MLP networks that respectively encode the inputs p, τ, and θ into $f_p \in \mathbb{R}^{32}$, $f_\tau \in \mathbb{R}^{128}$, and $f_\theta \in \mathbb{R}^{32}$. We serialize each trajectory as a 30-dimensional vector after flattening all waypoint information. We augment the trajectories that are shorter than five waypoint steps simply by zero paddings.

Actionability Prediction Module. The actionability prediction network D_a, similar to Where2Act (Mo et al., 2021), is implemented as a simple MLP that consumes a feature concatenation of f_p, f_s, and f_θ, and predicts a per-point actionability score $a_{p|O,T,\theta} \in [0,1]$. Aggregating over all contact points p, one can obtain an actionability map $A_{O,T,\theta}$ over the input partial scan S_O, from which one can sample an interaction point at test time according to a normalized actionability distribution.

Trajectory Proposal Module. The trajectory proposal module is implemented as a conditional variational autoencoder (cVAE, Sohn et al. (2015)), composed of a trajectory encoder E_τ that maps the input trajectory τ into a Gaussian noise z and a trajectory decoder D_τ that reconstructs the trajectory input from the noise vector. Both networks take additional input features of f_p, f_s, and f_θ as conditions. We use MLPs to realize the two networks. We regularize the resultant noise vectors to get closer to a uniform Gaussian distribution so that one can sample diverse trajectory proposals by feeding random Gaussian noises to the decoder D_τ with the conditional features as inputs.

Trajectory Scoring Module. The trajectory scoring module D_s, implemented as another MLP, takes as inputs features of f_p, f_s, and f_θ, as well as the trajectory feature f_τ, and predicts the success likelihood $r_{\tau|O,p,T,\theta} \in [0,1]$. One can use a success threshold of 0.5 to obtain a binary decision.

Data Collection for Training. We collect interaction data from the RL exploration to supervise the training of the perception system. We randomly pick shapes, tasks, and starting part poses similar to the RL task initialization. For positive data, we sample 5000 successful interaction trajectories outputted by the RL. We sample the same amount of negative data, which are produced by offsetting the desired task θ_0 of a successful trajectory by a random value with $[0.1\theta_0, 45^\circ]$ for revolute parts and $[0.1\theta_0, 0.45]$ for prismatic parts. For the pulling experiments, we also consider another type of negative data that the first grasping attempt fails.

Implementation and Training. We find it beneficial to first train the trajectory scoring and the trajectory proposal modules before jointly training all three. We use the standard binary cross-entropy loss to train the trajectory scoring module D_s. To train the trajectory proposal cVAE of E_τ and D_τ, besides the KL divergence loss for regularizing Gaussian bottleneck noises, we use an L_1 loss to regress the trajectory waypoint positions and a 6D-rotation loss (Zhou et al., 2019) for training the waypoints orientations. For training the actionability prediction module D_a, we sample 100 random trajectories proposed by D_τ, estimate their success likelihood scores using D_s, and regress the prediction to the mean score of the top-5 rated trajectories with a L_1 loss. See Sec. B in appendix for more details.

4.3 Curiosity-driven Exploration

We build a bidirectional supervisory mechanism between the RL policy and the perception system. While the RL policy collects data to supervise the perception networks, we also add a curiosity-feedback (Pathak et al., 2017) from the perception networks to inversely affect the RL policy learning for exploring more diverse and novel interaction trajectories, which will eventually diversify the per-point trajectory distributions produced by the trajectory proposal decoder D_τ. The intuitive idea is to encourage the RL network to explore novel trajectories that the perception system currently gives low success scores. In our implementation, during the joint training of the RL and perception networks, we generate an additional intrinsic curiosity reward of $-500 r_{\tau|O,p,T,\theta}$ for a trajectory τ proposed by the RL policy and use the novel interaction data to continue supervising the perception system. We make sure to have training trajectory proposals generated by the RL network at different epochs in the buffer to avoid mode chasing of training the generative model D_τ.

Table 1: We report quantitative evaluations of the learned *actionable visual priors*. For each metric, we report numbers over test shapes from the training categories (before slash) and shapes from the test categories (after slash). Higher numbers indicate better results.

		Accuracy (%)	Precision (%)	Recall (%)	F-score (%)	Coverage (%)
door	pushing	82.24 / 72.44	81.28 / 72.83	85.22 / 73.86	82.76 / 72.54	82.00 / 70.54
	pulling	74.01 / 71.31	70.52 / 70.26	84.09 / 75.85	76.06 / 72.01	58.68 / 48.29
drawer	pushing	79.69 / 71.59	74.65 / 71.80	91.19 / 70.45	81.65 / 70.52	74.15 / 68.08
	pulling	78.41 / 71.88	74.54 / 72.29	87.50 / 72.44	80.23 / 71.71	81.15 / 64.31

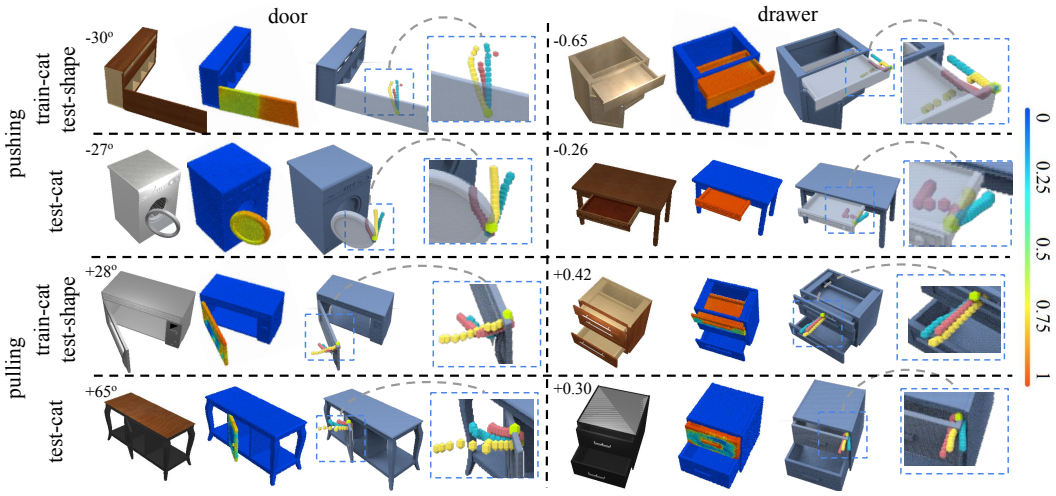

Figure 3: We show qualitative results of the actionability prediction and trajectory proposal modules. In each result block, from left to right, we present the input shape with the task, the predicted actionability heatmap, and three example trajectory proposals at a selected contact point.

5 Experiments

We perform our experiments using the SAPIEN simulator (Xiang et al., 2020) and PartNet-Mobility dataset (Chang et al., 2015; Mo et al., 2019). We evaluate the prediction quality of the learned visual priors and compare them to several baselines for downstream manipulation tasks. Qualitative results over novel shapes and real-world data show promising generalization capability of our approach.

Data and Settings. In total, we use 562 shapes from 7 categories in the PartNet-Mobility dataset. We conduct experiments over two commonly seen part articulation types: doors and drawers. For each experiment, we randomly split the applicable object categories into training and test categories. We further split the data from the training categories into training and test shapes. We train all methods over training shapes from the training categories and report performance over test shapes from the training categories and shapes from the test categories to test the generalization capabilities over novel shapes and unseen categories. We use the Panda flying gripper as the robot actuator and employ a velocity-based PID-controller to realize the actuation torques between consecutive trajectory waypoints. We use an RGB-D camera of resolution 448×448 and randomly sample viewpoints in front of the objects. See Sec. D E in appendix for more data statistics and implementation details.

5.1 Actionable Visual Priors

We present qualitative and quantitative evaluations of our learned actionable visual priors.

Metrics. For quantitatively evaluating the trajectory scoring module, we use standard metrics that are commonly used for evaluating binary classification problems: accuracy, precision, recall, and F-score. Since there is no ground-truth annotation of successful and failed interaction trajectories, we run our learned RL policy over test shapes and collect test interaction trajectories. In our implementation, we gather 350 positive and 350 negative trajectories for each experiment. To evaluate if the learned trajectory proposal module can propose diverse trajectories to cover the collected ground-truth ones, we compute a coverage score measuring the percentage of ground-truth trajectories that is similar enough to the closest predicted trajectory. See Sec. G in appendix for detailed metric definitions.

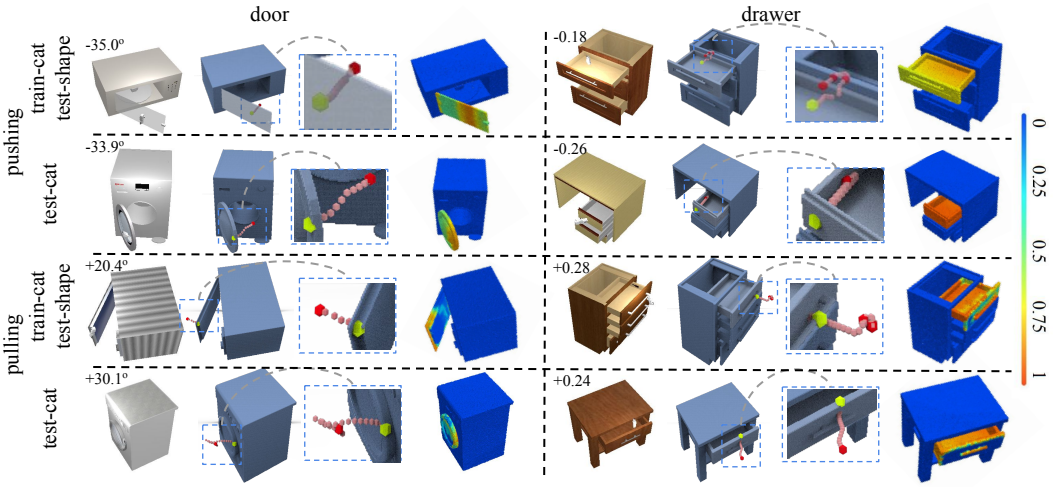

Figure 4: We present qualitative analysis of the learned trajectory scoring module. In each result block, from left to right, we show the input shape with the task, the input trajectory with its close-up view, and our network predictions of success likelihood applying the trajectory over all the points.

Results and Analysis. Table 1 presents quantitative results which demonstrate that we successfully learn the desired representations and that our learned model also generalizes to shapes from totally unseen object categories. Since we are the first to propose such actionable visual priors, there is no baseline to compare against. Fig. 3 presents qualitative results of the actionability prediction and trajectory proposal modules, from which we can observe that, to accomplish the desired tasks: 1) the actionability heatmap highlights *where* to interact (*e.g.*, to pull open a drawer, the gripper either grasp and pull open the handle from outside, or push outwards from inside), and 2) the trajectory proposals suggest diverse solutions for *how* to interact (*e.g.*, to push closed the door, various trajectories may succeed). In Fig. 4, we additionally illustrate the trajectory scoring predictions using heatmap visualization, where we observe interesting learned patterns indicating which points to interact with when executing the input trajectories. Fig. 5 (left) further visualizes the trajectory scoring predictions over the same input shape and observe different visual patterns given different trajectories and tasks.

5.2 Downstream Manipulation

We can easily use our learned actionable visual priors to accomplish the downstream manipulation of 3D articulated objects. To this end, we first sample a contact point according to the estimated actionability heatmap and then execute the top-rated trajectory among 100 random trajectory proposals.

Baselines. We compare to three baselines: 1) a naive TD3 RL that takes the shape point cloud together with the desired task as input and directly outputs trajectory waypoints for accomplishing the task; 2) a heuristic approach, in which we hand-engineer a set of rules for different tasks (*e.g.*, to pull open a drawer, we grasp the handle and pull straight backward). Note that we use ground-truth handle masks and joint parameters for the heuristic baseline; 3) a multi-step Where2Act (Mo et al., 2021) baseline that concatenates multiple (up to 15 steps) Where2Act-generated short-term pushing or pulling interactions that gradually accumulates the part pose changes for accomplishing the task. While the Where2Act work originally considers task-less short-term interactions, different from our long-term task-driven setting, we adapt it as a baseline by assuming an oracle part pose tracker is given. See Sec. H in appendix for more detailed descriptions of the baseline methods.

Metrics. We run interaction trials in simulation and report success rates for quantitative evaluation.

Results and Analysis. Table 2 presents the quantitative comparisons. Our method outperforms baselines on most comparisons. See below for detailed result analysis and more results in appendix.

For **the naive RL baseline**, it largely fails since we find it extremely difficult to train from scratch an end-to-end RL over highly diverse shapes for all tasks, which in fact implies the necessity of certain intermediate visual abstractions.

Table 2: We report task success rates (within 15% tolerance to tasks) of manipulating articulated objects comparing our method against three baseline methods.

	pushing door	pulling door	pushing drawer	pulling drawer
RL-baseline	1.43 / 3.72	0.0 / 0.0	0.43 / 0.43	0.0 / 0.0
Heuristic	25.74 / 25.23	8.82 / 3.73	40.25 / **57.99**	25.02 / 22.85
Where2Act	32.67 / 32.27	6.02 / 3.51	29.79 / 23.67	8.31 / 9.71
Ours	**55.14 / 38.30**	**12.04 / 14.33**	**56.02** / 41.67	**45.51 / 31.14**

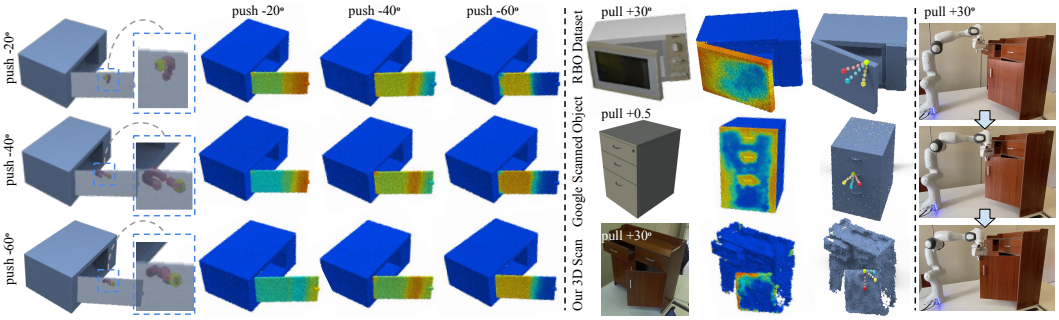

Figure 5: Left: qualitative analysis of the trajectory scoring prediction (each column shares the same task; every row uses the same trajectory); Middle: promising results testing on real-world data (from left to right: input, affordance prediction, trajectory proposals); Right: real-robot experiment.

For **the heuristic baseline**, we find that we win all comparisons except one task of drawer pushing. Knowing the ground-truth front board and prismatic joint axis, it is quite easy for the rule-based method to push a drawer along a straight-line trajectory, while our method does not take the ground-truth information and achieves worse results since we have to predict such information. However, our method achieves better performance for the other three experiments since we find that the heuristic method fails (see Fig. H.7) severely when 1) there are no handle parts for pulling, 2) some grasps over intricate handle geometry may slip and fail, or 3) the motion dynamics (*e.g.*, inertia) for non-prehensile manipulation (pushing) affects the accurate task accomplishment (*e.g.*, in case of an overshoot).

For **the Where2Act baseline**, it consistently performs worse than ours since it does not explicitly take the desired task as input and thus is more vulnerable of overshooting. Besides, the Where2Act baseline has the following issues: 1) it assumes an accurate part pose tracker, 2) it has to execute multiple segmented interactions and thus is empirically found 30 times slower, and 3) the gripper needs to be driven to multiple independent positions across different trajectory segments and thus it is very time-consuming and fragile if cases of motion planning failures.

5.3 Real-world Experiments

Fig. 5 (middle) presents qualitative results directly testing our model on real-world data: a microwave from the RBO dataset (Martín-Martín et al., 2019), one cabinet from Google Scanned Object (OpenRobotics), and one real-world 3D cabinet scan we capture using a ZED MINI RGB-D camera. We observe that our model trained on synthetic textureless data can generalize to real-world depth scans to some degree. We also show real-robot experiment in Fig. 5 (right) and supplementary video.

6 Conclusion

In this paper, we propose a novel perception-interaction handshaking point –*object-centric actionable visual priors*– for manipulating 3D articulated objects, which contains *dense* action affordance predictions and *diverse* visual trajectory proposals. We formulate a novel *interaction-for-perception* framework VAT-Mart to learn such representations. Experiments conducted on the large-scale PartNet-Mobility dataset and real-world data have proved the effectiveness of our approach.

Limitations and Future Works. First, our work is only a first attempt at learning such representations and future works can further improve the performance. Besides, the current open-loop trajectory prediction is based on a single-frame input. One may obtain better results considering multiple frames during an interaction. Lastly, future works may study more interaction and articulation types.

Ethics Statement. Our project helps build home-assistant robots to aid people with disabilities, or senior. Our proposed visual affordance and action trajectory predictions can be visualized and analyzed before use, which makes the AI system more explainable and thus less vulnerable to potential dangers. Training our system requires training data, which may introduce data bias, but this is a general concern for learning-based methods. We do not see any particular major harm or issue our work may raise up.

Reproducibility Statement. We will release our code, data, and pre-trained model weights publicly to the community upon paper acceptance.

REFERENCES

Ben Abbatematteo, Stefanie Tellex, and George Konidaris. Learning to generalize kinematic models to novel objects. In *Proceedings of the 3rd Conference on Robot Learning*, 2019.

Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, et al. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*, 2018.

Peter Anderson, Ayush Shrivastava, Joanne Truong, Arjun Majumdar, Devi Parikh, Dhruv Batra, and Stefan Lee. Sim-to-real transfer for vision-and-language navigation. *Conference on Robotic Learning (CoRL)*, 2020.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/453fadbd8a1a3af50a9df4df899537b5-Paper.pdf.

Miguel Arduengo, Carme Torras, and Luis Sentis. Robust and adaptive door operation with a mobile robot. *arXiv e-prints*, pp. arXiv–1902, 2019.

Jeannette Bohg, Karol Hausman, Bharath Sankaran, Oliver Brock, Danica Kragic, Stefan Schaal, and Gaurav S Sukhatme. Interactive perception: Leveraging action in perception and perception in action. *IEEE Transactions on Robotics*, 33(6):1273–1291, 2017.

Felix Burget, Armin Hornung, and Maren Bennewitz. Whole-body motion planning for manipulation of articulated objects. In *2013 IEEE International Conference on Robotics and Automation*, pp. 1656–1662. IEEE, 2013.

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.

Yevgen Chebotar, Ankur Handa, Viktor Makoviychuk, Miles Macklin, Jan Issac, Nathan D. Ratliff, and Dieter Fox. Closing the sim-to-real loop: Adapting simulation randomization with real world experience. *2019 International Conference on Robotics and Automation (ICRA)*, pp. 8973–8979, 2019.

Sachin Chitta, Benjamin Cohen, and Maxim Likhachev. Planning for autonomous door opening with a mobile manipulator. In *2010 IEEE International Conference on Robotics and Automation*, pp. 1799–1806. IEEE, 2010.

Sachin Chitta, Ioan Sucan, and Steve Cousins. Moveit! *IEEE Robotics & Automation Magazine*, 19 (1):18–19, 2012.

Enric Corona, Albert Pumarola, Guillem Alenya, Francesc Moreno-Noguer, and Grégory Rogez. Ganhand: Predicting human grasp affordances in multi-object scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5031–5041, 2020.

Karthik Desingh, Shiyang Lu, Anthony Opipari, and Odest Chadwicke Jenkins. Factored pose estimation of articulated objects using efficient nonparametric belief propagation. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 7221–7227. IEEE, 2019.

Thanh-Toan Do, Anh Nguyen, and Ian Reid. Affordancenet: An end-to-end deep learning approach for object affordance detection. In *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 5882–5889. IEEE, 2018.

Kuan Fang, Te-Lin Wu, Daniel Yang, Silvio Savarese, and Joseph J Lim. Demo2vec: Reasoning object affordances from online videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2139–2147, 2018.

Kuan Fang, Yuke Zhu, Animesh Garg, Andrey Kurenkov, Viraj Mehta, Li Fei-Fei, and Silvio Savarese. Learning task-oriented grasping for tool manipulation from simulated self-supervision. *The International Journal of Robotics Research*, 39(2-3):202–216, 2020.

Peter Florence, Lucas Manuelli, and Russ Tedrake. Self-supervised correspondence in visuomotor policy learning. *IEEE Robotics and Automation Letters*, 2019. ISSN 2377-3766.

Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pp. 1587–1596. PMLR, 2018.

Samir Yitzhak Gadre, Kiana Ehsani, and Shuran Song. Act the part: Learning interaction strategies for articulated object part discovery. In *International Conference on Computer Vision (ICCV)*, 2021.

Leni K Le Goff, Oussama Yaakoubi, Alexandre Coninx, and Stephane Doncieux. Building an affordances map with interactive perception. *arXiv preprint arXiv:1903.04413*, 2019.

Karol Hausman, Scott Niekum, Sarah Osentoski, and Gaurav S Sukhatme. Active articulation model estimation through interactive perception. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3305–3312. IEEE, 2015.

Stefan Hinterstoisser, Stefan Holzer, Cedric Cagniart, Slobodan Ilic, Kurt Konolige, Nassir Navab, and Vincent Lepetit. Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In *2011 international conference on computer vision*, pp. 858–865. IEEE, 2011.

Sebastian Höfer, Tobias Lang, and Oliver Brock. Extracting kinematic background knowledge from interactions using task-sensitive relational learning. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4342–4347. IEEE, 2014.

Xiaoxia Huang, Ian Walker, and Stan Birchfield. Occlusion-aware reconstruction and manipulation of 3d articulated objects. In *2012 IEEE International Conference on Robotics and Automation*, pp. 1365–1371. IEEE, 2012.

Andrew Hundt, Benjamin Killeen, Nicholas Greene, Hongtao Wu, Heeyeon Kwon, Chris Paxton, and Gregory D Hager. "good robot!": Efficient reinforcement learning for multi-step visual tasks with sim to real transfer. *IEEE Robotics and Automation Letters (RA-L)*, 2019.

Advait Jain and Charles C Kemp. Pulling open novel doors and drawers with equilibrium point control. In *2009 9th IEEE-RAS International Conference on Humanoid Robots*, pp. 498–505. IEEE, 2009.

Ajinkya Jain, Rudolf Lioutikov, Caleb Chuck, and Scott Niekum. Screwnet: Category-independent articulation model estimation from depth images using screw theory. *arXiv preprint arXiv:2008.10518*, 2020.

Stephen James, Paul Wohlhart, Mrinal Kalakrishnan, Dmitry Kalashnikov, Alex Irpan, Julian Ibarz, Sergey Levine, Raia Hadsell, and Konstantinos Bousmalis. Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 12627–12637, 2019.

Dinesh Jayaraman and Kristen Grauman. End-to-end policy learning for active visual categorization. *IEEE transactions on pattern analysis and machine intelligence*, 41(7):1601–1614, 2018.

Zhenyu Jiang, Yifeng Zhu, Maxwell Svetlik, Kuan Fang, and Yuke Zhu. Synergies between affordance and geometry: 6-dof grasp detection via implicit representations. *arXiv preprint arXiv:2104.01542*, 2021.

Abhishek Kadian, Joanne Truong, Aaron Gokaslan, Alexander Clegg, Erik Wijmans, Stefan Lee, Manolis Savva, Sonia Chernova, and Dhruv Batra. Sim2real predictivity: Does evaluation in simulation predict real-world performance? *IEEE Robotics and Automation Letters (RA-L)*, 2020.

Siddharth Karamcheti, Albert J. Zhai, Dylan P. Losey, and Dorsa Sadigh. Learning visually guided latent actions for assistive teleoperation. In *3rd Annual Learning for Dynamics & Control Conference (L4DC)*, 6 2021.

Dov Katz, Yuri Pyuro, and Oliver Brock. Learning to manipulate articulated objects in unstructured environments using a grounded relational representation. In *In Robotics: Science and Systems*. Citeseer, 2008.

Dov Katz, Moslem Kazemi, J Andrew Bagnell, and Anthony Stentz. Interactive segmentation, tracking, and kinematic modeling of unknown 3d articulated objects. In *2013 IEEE International Conference on Robotics and Automation*, pp. 5003–5010. IEEE, 2013.

Dov Katz, Andreas Orthey, and Oliver Brock. Interactive perception of articulated objects. In *Experimental Robotics*, pp. 301–315. Springer, 2014.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *The 3rd International Conference for Learning Representations*, 2015.

Hedvig Kjellström, Javier Romero, and Danica Kragić. Visual object-action recognition: Inferring object affordances from human demonstration. *Computer Vision and Image Understanding*, 115 (1):81–90, 2011.

Ellen Klingbeil, Ashutosh Saxena, and Andrew Y Ng. Learning to open new doors. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2751–2757. IEEE, 2010.

Mia Kokic, Danica Kragic, and Jeannette Bohg. Learning task-oriented grasping from human activity datasets. *IEEE Robotics and Automation Letters*, 5(2):3352–3359, 2020.

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.

Xiaolong Li, He Wang, Li Yi, Leonidas J Guibas, A Lynn Abbott, and Shuran Song. Category-level articulated object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3706–3715, 2020.

Jacky Liang, Saumya Saxena, and Oliver Kroemer. Learning active task-oriented exploration policies for bridging the sim-to-real gap. In *RSS 2020*, 2020.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pp. 740–755. Springer, 2014.

Qihao Liu, Weichao Qiu, Weiyao Wang, Gregory D Hager, and Alan L Yuille. Nothing but geometric constraints: A model-free method for articulated object pose estimation. *arXiv preprint arXiv:2012.00088*, 2020.

Martin Lohmann, Jordi Salvador, Aniruddha Kembhavi, and Roozbeh Mottaghi. Learning about objects by learning to interact with them. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

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. 2017.

Priyanka Mandikal and Kristen Grauman. Learning dexterous grasping with object-centric visual affordances. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

Roberto Martin Martin and Oliver Brock. Online interactive perception of articulated objects with multi-level recursive estimation based on task-specific priors. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2494–2501. IEEE, 2014.

Roberto Martın-Martın and Oliver Brock. Building kinematic and dynamic models of articulated objects with multi-modal interactive perception. In *AAAI Symposium on Interactive Multi-Sensory Object Perception for Embodied Agents, AAAI, Ed*, 2017.

Roberto Martín-Martín and Oliver Brock. Coupled recursive estimation for online interactive perception of articulated objects. *The International Journal of Robotics Research*, pp. 0278364919848850, 2019.

Roberto Martín-Martín, Sebastian Höfer, and Oliver Brock. An integrated approach to visual perception of articulated objects. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5091–5097. IEEE, 2016.

Roberto Martín-Martín, Clemens Eppner, and Oliver Brock. The rbo dataset of articulated objects and interactions. *The International Journal of Robotics Research*, 38(9):1013–1019, 2019.

Mayank Mittal, David Hoeller, Farbod Farshidian, Marco Hutter, and Animesh Garg. Articulated object interaction in unknown scenes with whole-body mobile manipulation. *arXiv preprint arXiv:2103.10534*, 2021.

Kaichun Mo, Shilin Zhu, Angel X Chang, Li Yi, Subarna Tripathi, Leonidas J Guibas, and Hao Su. Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 909–918, 2019.

Kaichun Mo, Leonidas Guibas, Mustafa Mukadam, Abhinav Gupta, and Shubham Tulsiani. Where2act: From pixels to actions for articulated 3d objects. In *International Conference on Computer Vision (ICCV)*, 2021.

Luis Montesano and Manuel Lopes. Learning grasping affordances from local visual descriptors. In *2009 IEEE 8th international conference on development and learning*, pp. 1–6. IEEE, 2009.

Jiteng Mu, Weichao Qiu, Adam Kortylewski, Alan Yuille, Nuno Vasconcelos, and Xiaolong Wang. A-sdf: Learning disentangled signed distance functions for articulated shape representation. *arXiv preprint arXiv: 2104.07645*, 2021.

Tushar Nagarajan and Kristen Grauman. Learning affordance landscapes for interaction exploration in 3d environments. In *NeurIPS*, 2020.

Tushar Nagarajan, Christoph Feichtenhofer, and Kristen Grauman. Grounded human-object interaction hotspots from video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8688–8697, 2019.

Tushar Nagarajan, Yanghao Li, Christoph Feichtenhofer, and Kristen Grauman. Ego-topo: Environment affordances from egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 163–172, 2020.

Urbano Miguel Nunes and Yiannis Demiris. Online unsupervised learning of the 3d kinematic structure of arbitrary rigid bodies. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3809–3817, 2019.

OpenRobotics. Drawer, June . URL https://fuel.ignitionrobotics.org/1.0/OpenRobotics/models/Drawer.

Antonio Paolillo, Anastasia Bolotnikova, Kévin Chappellet, and Abderrahmane Kheddar. Visual estimation of articulated objects configuration during manipulation with a humanoid. In *2017 IEEE/SICE International Symposium on System Integration (SII)*, pp. 330–335. IEEE, 2017.

Antonio Paolillo, Kevin Chappellet, Anastasia Bolotnikova, and Abderrahmane Kheddar. Interlinked visual tracking and robotic manipulation of articulated objects. *IEEE Robotics and Automation Letters*, 3(4):2746–2753, 2018.

Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning*, pp. 2778–2787. PMLR, 2017.

Deepak Pathak, Yide Shentu, Dian Chen, Pulkit Agrawal, Trevor Darrell, Sergey Levine, and Jitendra Malik. Learning instance segmentation by interaction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 2042–2045, 2018.

L Peterson, David Austin, and Danica Kragic. High-level control of a mobile manipulator for door opening. In *Proceedings. 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000)(Cat. No. 00CH37113)*, volume 3, pp. 2333–2338. IEEE, 2000.

Lerrel Pinto, Dhiraj Gandhi, Yuanfeng Han, Yong-Lae Park, and Abhinav Gupta. The curious robot: Learning visual representations via physical interactions. In *European Conference on Computer Vision*, pp. 3–18. Springer, 2016.

Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017.

Zengyi Qin, Kuan Fang, Yuke Zhu, Li Fei-Fei, and Silvio Savarese. Keto: Learning keypoint representations for tool manipulation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7278–7285. IEEE, 2020.

Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, Andrew Y Ng, et al. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, pp. 5. Kobe, Japan, 2009.

Santhosh K Ramakrishnan, Dinesh Jayaraman, and Kristen Grauman. An exploration of embodied visual exploration. *International Journal of Computer Vision*, pp. 1–34, 2021.

Kanishka Rao, Chris Harris, Alex Irpan, Sergey Levine, Julian Ibarz, and Mohi Khansari. Rl-cyclegan: Reinforcement learning aware simulation-to-real. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11157–11166, 2020.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

Tanner Schmidt, Richard A Newcombe, and Dieter Fox. Dart: Dense articulated real-time tracking. In *Robotics: Science and Systems*, volume 2. Berkeley, CA, 2014.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28:3483–3491, 2015.

Rafał Staszak, Milena Molska, Kamil Młodzikowski, Justyna Ataman, and Dominik Belter. Kinematic structures estimation on the rgb-d images. In *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, volume 1, pp. 675–681. IEEE, 2020.

Jürgen Sturm, Vijay Pradeep, Cyrill Stachniss, Christian Plagemann, Kurt Konolige, and Wolfram Burgard. Learning kinematic models for articulated objects. In *IJCAI*, pp. 1851–1856, 2009.

Jürgen Sturm, Cyrill Stachniss, and Wolfram Burgard. A probabilistic framework for learning kinematic models of articulated objects. *Journal of Artificial Intelligence Research*, 41:477–526, 2011.

Dimitrios Tzionas and Juergen Gall. Reconstructing articulated rigged models from rgb-d videos. In *European Conference on Computer Vision*, pp. 620–633. Springer, 2016.

Yusuke Urakami, Alec Hodgkinson, Casey Carlin, Randall Leu, Luca Rigazio, and Pieter Abbeel. Doorgym: A scalable door opening environment and baseline agent. *Deep RL workshop at NeurIPS 2019*, 2019.

Angelina Wang, Thanard Kurutach, Kara Liu, Pieter Abbeel, and Aviv Tamar. Learning robotic manipulation through visual planning and acting. In *Robotics: science and systems*, 2019a.

Jiayu Wang, Shize Lin, Chuxiong Hu, Yu Zhu, and Limin Zhu. Learning semantic keypoint representations for door opening manipulation. *IEEE Robotics and Automation Letters*, 5(4): 6980–6987, 2020.

Xiaogang Wang, Bin Zhou, Yahao Shi, Xiaowu Chen, Qinping Zhao, and Kai Xu. Shape2motion: Joint analysis of motion parts and attributes from 3d shapes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8876–8884, 2019b.

D Wilkes and JK Tsotsos. Active object recognition. In *Proceedings 1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 136–141. IEEE, 1992.

Jiajun Wu, Ilker Yildirim, Joseph J Lim, Bill Freeman, and Josh Tenenbaum. Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. *Advances in neural information processing systems*, 28:127–135, 2015a.

Jimmy Wu, Xingyuan Sun, Andy Zeng, Shuran Song, Johnny Lee, Szymon Rusinkiewicz, and Thomas Funkhouser. Spatial action maps for mobile manipulation. In *Proceedings of Robotics: Science and Systems (RSS)*, 2020.

Jimmy Wu, Xingyuan Sun, Andy Zeng, Shuran Song, Szymon Rusinkiewicz, and Thomas Funkhouser. Spatial intention maps for multi-agent mobile manipulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1912–1920, 2015b.

Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, et al. Sapien: A simulated part-based interactive environment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11097–11107, 2020.

Yu Xiang, Wonhui Kim, Wei Chen, Jingwei Ji, Christopher Choy, Hao Su, Roozbeh Mottaghi, Leonidas Guibas, and Silvio Savarese. Objectnet3d: A large scale database for 3d object recognition. In *European conference on computer vision*, pp. 160–176. Springer, 2016.

Danfei Xu, Ajay Mandlekar, Roberto Martín-Martín, Yuke Zhu, Silvio Savarese, and Li Fei-Fei. Deep affordance foresight: Planning through what can be done in the future. *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

Zhenjia Xu, Jiajun Wu, Andy Zeng, Joshua B Tenenbaum, and Shuran Song. Densephysnet: Learning dense physical object representations via multi-step dynamic interactions. In *Robotics: Science and Systems (RSS)*, 2019.

Jingyu Yan and Marc Pollefeys. Automatic kinematic chain building from feature trajectories of articulated objects. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pp. 712–719. IEEE, 2006.

Zhihao Yan, Ruizhen Hu, Xingguang Yan, Luanmin Chen, Oliver van Kaick, Hao Zhang, and Hui Huang. RPM-NET: Recurrent prediction of motion and parts from point cloud. *ACM Trans. on Graphics*, 38(6):Article 240, 2019.

Jianwei Yang, Zhile Ren, Mingze Xu, Xinlei Chen, David Crandall, Devi Parikh, and Dhruv Batra. Embodied visual recognition. *arXiv preprint arXiv:1904.04404*, 2019.

Lixin Yang, Xinyu Zhan, Kailin Li, Wenqiang Xu, Jiefeng Li, and Cewu Lu. Cpf: Learning a contact potential field to model the hand-object interaction. *arXiv preprint arXiv:2012.00924*, 2020.

Yifan You, Lin Shao, Toki Migimatsu, and Jeannette Bohg. Omnihang: Learning to hang arbitrary objects using contact point correspondences and neural collision estimation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021.

Vicky Zeng, Timothy E Lee, Jacky Liang, and Oliver Kroemer. Visual identification of articulated object parts. *arXiv preprint arXiv:2012.00284*, 2020.

Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5745–5753, 2019.

APPENDIX

A MORE DETAILS FOR THE RL POLICY (SEC. 4.1)

A.1 NETWORK ARCHITECTURE DETAILS

We leverage TD3 (Fujimoto et al., 2018) to train the RL policy. It consists of a policy network and a Q-value network, both implemented as a Multi-layer Perception (MLP). The policy network receives the state as input ($\mathbb{R}^{33}$), and predicts the residual gripper pose as action ($\mathbb{R}^{6}$). Its network architecture is implemented as a 4-layer MLP ($33 \rightarrow 512 \rightarrow 512 \rightarrow 512 \rightarrow 512$), followed by two separate fully connected layers ($512 \rightarrow 6$) that estimate the mean and variance of action probabilities respectively. The Q-value network receives both the state and action as input ($\mathbb{R}^{39}$), and predicts a single Q value ($\mathbb{R}^{1}$). Its network architecture is implemented as a 4-layer MLP ($39 \rightarrow 512 \rightarrow 512 \rightarrow 512 \rightarrow 1$).

A.2 MORE IMPLEMENTATION AND TRAINING DETAILS

We use the following hyper-parameters to train the RL policy: 2048 (buffer size); 512 (batch size); 0.0001 (learning rate of the Adam optimizer (Kingma & Ba, 2015) for both the policy and Q-value network); 0.1 (initial exploration range, decayed by 0.5 every 500 epochs.

B MORE DETAILS FOR THE PERCEPTION NETWORKS (SEC. 4.2)

B.1 NETWORK ARCHITECTURE DETAILS

The perception networks consist of four input encoders that extract high-dimensional features from the input partial point cloud S_O, contact point p, trajectory τ and task θ, and three parallel output modules that estimate the actionability $a_{p|O,T,\theta}$, trajectory proposal $\tau_{p|O,T,\theta}$ and scores $r_{\tau|O,p,T,\theta}$ from the extracted high-dimensional features of input encoders.

Regarding the four input encoders

- The input point cloud S_O with point-wise coordinate ($\mathbb{R}^3$) is fed through the PointNet++ (Qi et al., 2017) with the segmentation head and single-scale grouping (SSG) to extract the point-wise feature $f_s \in \mathbb{R}^{128}$.
- The input contact point $p \in \mathbb{R}^3$ is fed through a fully connected layer ($3 \rightarrow 32$) to extract a high-dimensional contact point feature $f_p \in \mathbb{R}^{32}$.
- The input trajectory $\tau \in \mathbb{R}^{30}$ is composed of five waypoints, each of which contains 3D rotation and 3D translation information. The input trajectory is further flattened to be fed through a 3-layer MLP ($30 \rightarrow 128 \rightarrow 128 \rightarrow 128$) to extract a high-dimensional trajectory feature $f_\tau \in \mathbb{R}^{128}$.
- The input task $\theta \in \mathbb{R}^1$ is fed through a fully connected layer ($1 \rightarrow 32$) to extract a high-dimensional task feature $f_\theta \in \mathbb{R}^{32}$.

Regarding the three output modules

- The actionability prediction module D_a concatenates the extracted point cloud f_s, contact point f_p and task f_θ features to form a high-dimensional vector ($\mathbb{R}^{192}$), which is fed through a 5-layer MLP ($192 \rightarrow 128 \rightarrow 128 \rightarrow 128 \rightarrow 128 \rightarrow 1$) to predict a per-point actionability score $a_{p|O,T,\theta}$.
- The trajectory proposal module is implemented as a conditional variational auto-encoder (cVAE), which learns to generate diverse trajectories from the sampled Gaussian value and given conditions (point cloud f_s, contact point f_p and task f_θ features). The encoder E_τ is

implemented as a 3-layer MLP ($320 \rightarrow 128 \rightarrow 128 \rightarrow 256$) that takes the concatenation of the trajectory feature f_τ and given conditions as input and estimates a Gaussian distribution ($\mu \in \mathbb{R}^{128}, \sigma \in \mathbb{R}^{128}$). The decoder D_τ is also implemented as a 3-layer MLP ($320 \rightarrow 512 \rightarrow 256 \rightarrow 30$) that takes the concatenation of the sampled Gaussian value $z \in \mathbb{R}^{128}$ and given conditions as input to recover the trajectory τ.

- The trajectory scoring module D_s concatenates the extracted point cloud f_s, contact point f_p, trajectory f_τ and task f_θ features to form a high-dimensional vector ($\mathbb{R}^{320}$), which is fed through a 2-layer MLP ($320 \rightarrow 128 \rightarrow 1$) to predict the success likelihood $r_{\tau|O,p,T,\theta}$.

B.2 More Implementation and Training Details

The interaction data provided by the RL policy provides the ground truth label to train the perception networks. We start by training the trajectory scoring module, followed by joint training of all the three modules.

To train the trajectory scoring module D_s, we firstly collect 5000 successful and 5000 unsuccessful interaction trajectories produced by the RL policy, then supervise the estimated success likelihood of the trajectory with binary cross entropy loss.

To train the trajectory proposal module, we supervise the output rotation and translation with the 6D-rotation loss (Zhou et al., 2019) and L_1 loss respectively from the 5000 successful collected interaction trajectories. We follow the common cVAE training process to add a KL divergence loss on the estimated Gaussian distribution for regularization.

To train the actionability prediction module D_a, we estimate the scores from 100 sampled trajectories with both the trajectory proposal and scoring modules, then calculate the mean value of top-5 rated scores as the ground truth label to supervise the actionability predictions. We adopt L_1 loss for supervision.

We use the following hyper-parameters to train the perception networks: 32 (batch size); 0.001 (learning rate of the Adam optimizer (Kingma & Ba, 2015) for all three modules).

C More Details for the Curiosity-driven Exploration (Sec. 4.3)

The curiosity-driven exploration alternates between training the RL policy and perception networks at odd and even epochs respectively. It aims for exploring more diverse trajectories with the RL policy that the perception networks are not confident with and are further used to diversify the trajectory proposals generated by the perception networks.

When training the RL policy, besides the extrinsic task rewards, it also awards the policy with the intrinsic curiosity reward. It is computed as $-500 r_{\tau|O,p,T,\theta}$, which is the weighted negative success likelihood estimated by the trajectory scoring module. It penalizes the learned trajectories that the trajectory scoring module is confident with and encourage more diverse trajectories generated by the RL policy.

When training the perception networks, we collect the equal number of successful and unsuccessful trajectories produced by the RL policy. To encourage the perception networks to learn more diverse trajectories, we sample the successful trajectories fifty-fifty with both high (¿0.5) and low (¡0.5) success likelihood estimated by the trajectory scoring module. Then we use these data to train the perception networks.

D More Detailed Data Statistics

Table D.3 summarizes the data statistics and splits.

Fig. D.6 presents a visualization of some example shapes from our dataset.

E SAPIEN Simulation Environment Settings

We follow exactly the virtual environment settings as in (Mo et al., 2021), except that we randomly sample camera viewpoints only in front of the target shape since most articulated parts are present

Train-Cats	All / Door	Cabinet	Door	Mirco	Fridge	All / Drawer	Cabinet
Train-Data	328	270	17	9	32	270	270
Test-Data	94	75	5	3	11	75	75
Test-Cats		Safe	Table	Washing			Table
Test-Data	140	29	95	16		95	95

Table D.3: We summarize the shape counts in our dataset. Here, *Micro* and *Washing* are short for microwave and washing machine.

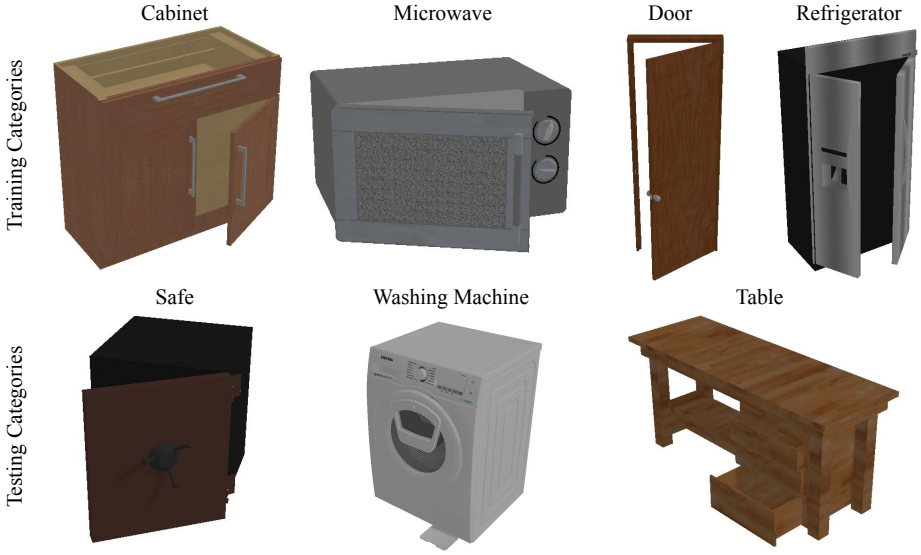

Figure D.6: Data Visualization. We show one example shape from each of the seven object categories we use in our paper.

in the front side of the PartNet-Mobility shapes and it is very unlikely that robot agents need to manipulate object parts from the back of objects. By default, the camera looks at the center of the target shape, and the location is sampled on a unit sphere centered at the origin. For the spherical sampling of the camera location, we set the range of the azimuth angle to [-90°, 90°], and the elevation angle to [30°, 60°]. Please refer to Section A and Section B in the appendix of (Mo et al., 2021) for more in-depth details of the virtual environment.

F Computational Costs and Timing

Training of the proposed system took around 3 days per experiment on a single Nvidia V100 GPU. More specifically, the initial RL policy training took around 18 hours, followed by a 15-hour training of the perception module using the RL-collected data. The subsequent curiosity-driven exploration stage took around 10 hours to jointly train the RL and the perception networks. At last, the three prediction branches in the perception module were further fine-tuned for around 20 hours.

During the inference time, a forward pass takes around 15 ms, 8 ms, and 8 ms for the actionability map prediction, diverse trajectory proposals, and success rate predictions, respectively. Our model only consumes 1,063 MB memory of the GPU in test time.

G More Metric Definition Details

G.1 Evaluating Actionable Visual Priors (Sec. 5.1)

We adopt the following evaluation metrics for quantitative evaluation: precision, recall and F-score for the positive data, and the average accuracy equally balancing the negative and positive recalls. Formally, let f_p, f_n, t_p, and t_n denote the data counts for false positives, false negatives, true positives, and true negatives, respectively. The F-score is the harmonic mean of precision and recall, and the

other metrics are defined as follow:

$$precision = \frac{t_p}{t_p + f_p},$$
$$recall = \frac{t_p}{t_p + f_n},$$
$$accuracy = \frac{1}{2}(\frac{t_p}{t_p + f_n} + \frac{t_n}{t_n + f_p}).$$

We also compute a coverage score for evaluation, which is calculated as the percentage of ground-truth trajectories that are similar enough to be matched in the predicted trajectories. To this end, we need to measure the distance $D(l_1, l_2)$ between two trajectories l_1 and l_2, which takes into account both the position difference and the orientation difference at every waypoint. Concretely, we calculate the L1 distance of the waypoint positions as the position distance D_{pos} between the two trajectories. Then, the orientation distance D_{ori} is measured as the 6D-rotation distance of the waypoint orientations. To balance the dimension of the quantities, $D(l_1, l_2)$ is calculated as: $D(l_1, l_2) = 5D_{pos}(l_1, l_2) + D_{ori}(l_1, l_2)$. We consider a ground truth trajectory to be covered if the distance between this ground truth trajectory and a predicted trajectory is lower than a threshold (10, in all our experiments), and then report the percentage of the ground truth trajectories that are covered by the predictions. To compensate the stochastic error, all reported quantitative results are averaged over 10 test runs.

G.2 Evaluating Downstream Manipulation (Sec. 5.2)

For each downstream manipulation task, we locate the contact point sample with the highest actionability score over the surface on each test shape, and then generate 100 diverse trajectories at this contact point, and pick the trajectory with the highest success rate for executing the downstream task. This results in 350 trajectories for each of the door pushing, door pulling, drawer pushing, and drawer pulling tasks.

In practice, generating trajectories that can perfectly fulfill the given task specification (*e.g.*, push to open a door by 30°) is very hard. Hence, we consider a trajectory can successfully accomplish the task if the specification is fulfilled within a tolerance threshold in percentage $\varepsilon = 15\%$. For example, if the task is to push a door open by 10°, a trajectory that can open the door between 8.5° and 11.5° is counted as a successful sample. We then report the success rate in percentage of all the generated trajectories on each task.

H More Details of Baselines in Sec. 5.2

In the following, we present more details about the baseline methods that we compare against in Section 5.2.

H.1 Naive RL Baseline

The naive RL baseline, which is comprised of two sub-RL modules denoted as sub-RL1 and sub-RL2 respectively. Sub-RL1 module takes as input the initial state of the target shape (*i.e.*, the partial scan of the initial shape, and the mask of the target part), and the task specification, and then outputs the initial position and orientation of the gripper (*i.e.*, the parameters of the first waypoint). Sub-RL2 module takes as input the state of the initial scene (*i.e.*, the partial scan of the shape, the mask of the target part, and the position and orientation of the gripper at the initial state), the state of the current scene (*i.e.*, the partial scan of the shape, the mask of the target part, and the position and orientation of the gripper at the current state), and the task specification, and then predicts the parameters of the next waypoint using the residual representation.

The network architectures of sub-RL1 and sub-RL2 are based on our RL networks as described in Section A.1, with minor adaptations for different input and output dimensionalities. We use a buffer size of 2048 and a batch size of 64 for training both sub-RL1 and sub-RL2 modules. We use the Adam optimizer and a learning rate 0.0001 for the policy and Q-value networks in both modules. We set the initial exploration range to 0.1, which is linearly decayed by a factor of 0.5 every 500 epochs during training.

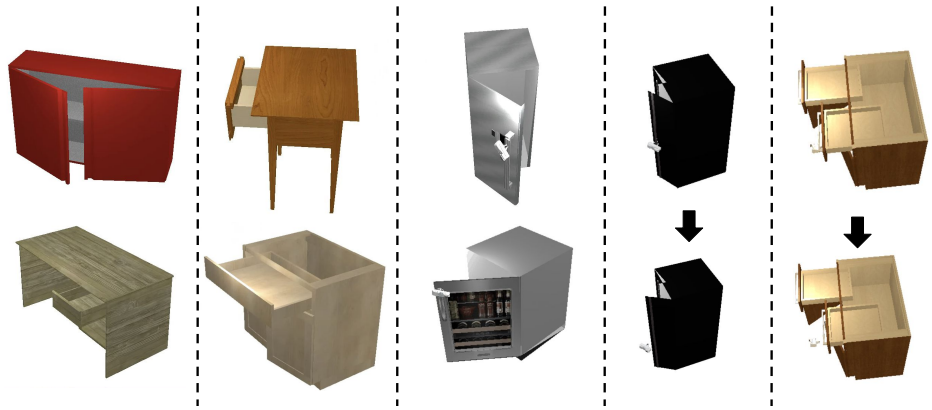

Figure H.7: We present some failure cases for the rule-based heuristic baseline to explain why such seemingly intuitive and easy heuristics often fail. Fist column: these two cabinets have no handle for pulling; Second column: from these viewpoints, these drawer front boards or door handles are occluded; Third column and fourth column: grasps over intricate handle geometry may fail or slip; Fifth column: dynamic factors (*e.g.*, inertia) affects the final achieved part pose, and the drawer moves farther than desired.

H.2 Rule-based Heuristic Baseline

For the rule-based heuristic baseline, we hand-craft a set of rules for different tasks. We describe the details as follows.

- **Door pushing:** we randomly sample a point p on the door surface, then initialize the gripper such that the gripper fingertips touch the point p and set the forward orientation along the negative direction of the surface normal at p. Let d denotes the distance between p and the rotation shaft of the door, to push the door by a degree θ, we push the door by a degree $\frac{\theta}{4}$ each time and repeat for four times. During each process, we simply move the gripper along the negative direction of the surface normal by a distance of $d \sin\left(\frac{\theta}{4}\right)$.

- **Door pulling:** we initialize the gripper at a point p sampled on the handle of the door and set the forward orientation along the negative direction of the surface normal at p. And then we pull the door by a degree θ by pulling $\frac{\theta}{4}$ each time as in door pushing. Different from door pushing, we also perform a grasping at contact point p for pulling.

- **Drawer pushing:** similarly, we randomly sample a point p on the front board of the drawer, and then initialize the gripper such that the gripper fingertips touch the point p and set the forward orientation along the negative direction of the normal at p. Let t denote the push distance in the task specification, we simply move the gripper by a distance of t along the *slide-to-close* direction of the drawer.

- **Drawer pulling:** the middle point of the handle is particularly sampled as the contact point. Note that we count the trial as a failure if the drawer does not have handles. Then we initialize the gripper as in drawer pushing, and move the gripper by a distance of t along the *slide-to-open* direction.

In Fig. H.7, we present some failure cases for the rule-based heuristic baseline to explain why such seemingly intuitive and easy heuristics often fail. See the figure caption for detailed explanations. Such rule-based heuristics require careful human hand-engineering given different task semantics. One needs to hand-design rules for different tasks and sometimes will find it difficult to enumerate all possible rules. Our system, instead, provides a unified system that automatically discovers useful knowledge for different tasks, without the need of spending human efforts, and learn a rich collection of data-driven priors from training over diverse shapes.

Table I.4: We compare our final approach to an ablated version that removes the curiosity-feedback channel and report quantitative evaluations of the visual actionable priors. For each metric, we report numbers over the test shapes from the training categories (before slash) and the shapes from the test categories (after slash). Higher numbers indicate better results.

	Curiosity	Accuracy (%)	Precision (%)	Recall (%)	F-score (%)	Coverage (%)
pushing door	w/o	77.70 / 71.73	77.83 / 72.62	76.99 / 69.89	77.55 / 70.80	**82.35 / 75.52**
	w	**82.24 / 72.44**	**81.28 / 72.83**	**85.22 / 73.86**	**82.76 / 72.54**	82.00 / 70.54
pulling door	w/o	68.89 / 67.90	64.62 / 65.29	**84.09 / 76.40**	72.62 / 69.94	50.17 / 43.52
	w	**74.01 / 71.31**	**70.52 / 70.26**	**84.09** / 75.85	**76.06 / 72.01**	**58.68 / 48.29**
pushing drawer	w/o	76.85 / 68.32	**78.82** / 70.67	73.86 / 61.65	75.45 / 64.89	63.25 / 60.08
	w	**79.69 / 71.59**	74.65 / **71.80**	**91.19 / 70.45**	**81.65 / 70.52**	**74.15 / 68.08**
pulling drawer	w/o	71.02 / 69.46	**74.79 / 73.20**	67.05 / 63.64	68.89 / 66.47	62.56 / 49.43
	w	**78.41 / 71.88**	74.54 / 72.29	**87.50 / 72.44**	**80.23 / 71.71**	**81.15 / 64.31**

H.3 WHERE2ACT BASELINE

For the Where2Act baseline, we use the pre-trained networks released by Mo et al. (2021) to generate multiple short-term trajectory segments, each of which samples a contact point and an action direction from Where2Act. We assume an oracle part pose tracker and choose the pushing or pulling Where2Act network to query at each time by comparing the ground-truth pose and the goal pose of the target part. Since Where2Act only executes short-term task-less interaction trajectories and does not explicitly consider task degree, we allow it to operate for up to 15 steps and record the closest distance it has ever reached to the desired target part pose. Concretely, in each step, this method

- uses Where2Act's actionability module to predict the affordance of each point, randomly selects a contact point within the estimated top-5 actionable points, and initializes the gripper in front of the selected contact point;
- initializes the gripper's initial 6D pose by running Where2Act's action proposal network and selecting the top-rated action;
- if pushing, directly moves the gripper 0.05 unit-length forward;
- if pulling, moves the gripper slightly forward, closes the gripper to grasp, and moves the gripper 0.05 unit-length backward when the gripper successfully grasps the target part;
- checks if the target part's pose achieves the goal pose, using the target part's ground-truth pose.

I ABLATION STUDY ON CURIOSITY FEEDBACK

We compare our final approach to an ablated version that removes the curiosity feedback (Sec. 5.3), in order to validate that the proposed curiosity-feedback mechanism is beneficial.

In Table I.4, we evaluate the prediction quality of the proposed actionable visual priors.

From this table, it is clear to see that the introduced curiosity feedback mechanism (Sec. 5.3) is beneficial as it improves the performance in most entries.

J MORE RESULTS AND ANALYSIS

In Fig. J.8, we show additional qualitative results of the actionability prediction and trajectory proposal modules to augment Fig. 3 in the main paper.

In Fig. J.9, we present an additional qualitative analysis of the learned trajectory scoring module to augment Fig. 4 in the main paper.

In addition, we provide another set of result analyses in Fig. J.10, where we show that to accomplish the same task, interacting at different contact points over the shape will give different trajectories.

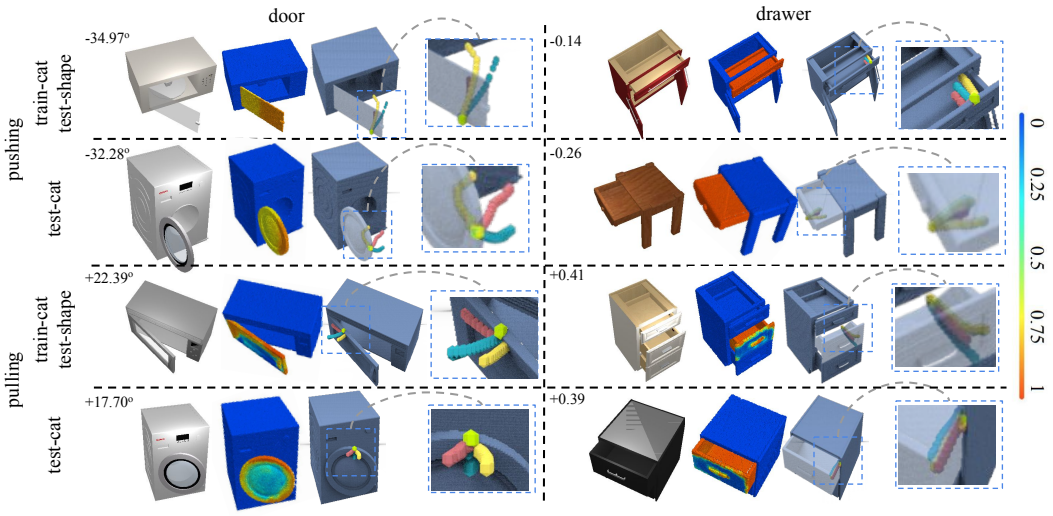

Figure J.8: We show additional qualitative results of the actionability prediction and trajectory proposal modules to augment Fig. 3 in the main paper. In each result block, from left to right, we present the input shape with the task, the predicted actionability heatmap, and three example trajectory proposals at a selected contact point.

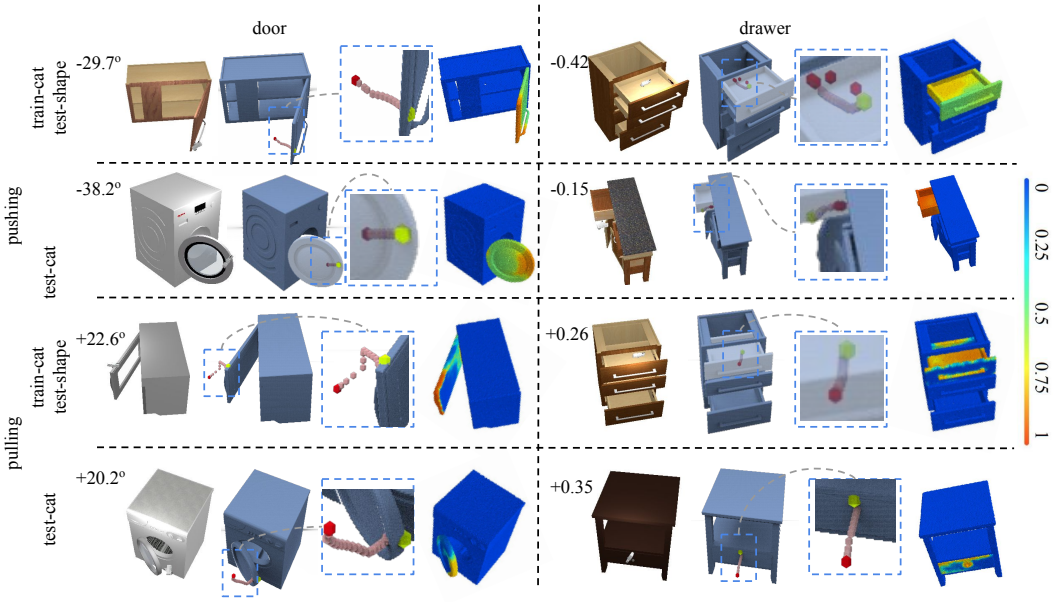

Figure J.9: We present an additional qualitative analysis of the learned trajectory scoring module to augment Fig. 4 in the main paper. In each result block, from left to right, we show the input shape with the task, the input trajectory with its close-up view, and our network predictions of success likelihood applying the trajectory over all the points.

K Real-robot Settings and Experiments

For real-robot experiments, we use one real cabinet and set up one Franka Panda robot facing the front of the cabinet. One ZED MINI RGB-D camera is mounted at the front right of the cabinet. The camera captures 3D point cloud data as inputs to our learned models.

We control the robot using Robot Operating System (ROS) (Quigley et al., 2009). The robot is programmed to execute each waypoint in the predicted trajectory step by step. We use MoveIt! (Chitta et al., 2012) for the motion planning between every adjacent waypoint pair.

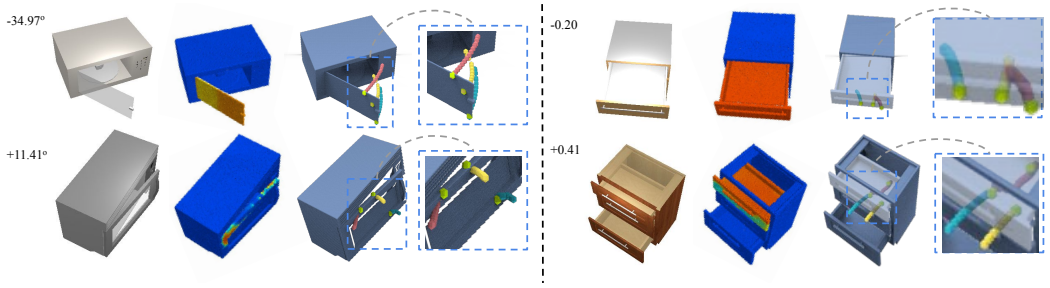

Figure J.10: We use some examples to show that, to accomplish the same task, interacting at different contact points over the shape will give different trajectories. In each result block, from left to right, we show the input shape with the task, the predicted actionability heatmap, and three example trajectory proposals at three different contact points.

We demonstrate various tasks on the cabinet, including pulling open the door at the edge or handle, pushing closed the door at different contact points, pulling open the drawer by grasping the edge or handle. Please check our results in the supplementary video.