Generalizing Object Manipulation through End-to-End Visual Affordance Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Learning to manipulate 3D articulated objects in an interactive environment has been challenging in reinforcement learning (RL) studies. It is hard to train a policy that can generalize over different objects with vast semantic categories, diverse shape geometry, and versatile functionality. Visual affordance provides object-centric information priors that offer actionable semantics for objects with movable parts. For example, an effective policy should know the pulling force on the handle to open a door. Nevertheless, how to learn affordance in an end-to-end fashion within the RL process is unknown. In this study, we fill such a research gap by designing algorithms that can automatically learn affordance semantics through a *contact prediction* process. The contact predictor allows the agent to learn the affordance information (*i.e.*, where to act for the robotic arm on the object) from previous manipulation experience, and such affordance semantics then helps the agent learn effective policies through RL updates. We use our framework on several downstream tasks. The experimental result and analysis demonstrate the effectiveness of end-to-end affordance learning.

1 Introduction

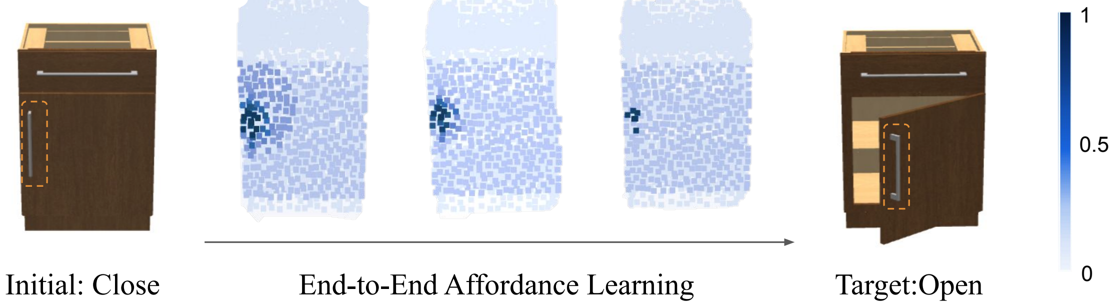

Figure 1: **Overview of our method.** From left to right we show the process of end-to-end Affordance learning. As RL continues to update the policy, the affordance map is constantly being adapted to focus on a few points.

Learning to manipulate objects is a fundamental problem in RL and the robotic field. A generalizable learning approach can explore the reach range of future intelligent robotics. A primary concern is that the agents have to manipulate objects they have never seen before. Recently, researchers have shown an increased interest in visual affordance [2, 13, 14, 20, 21], *i.e.* a prior representation of an object that is specific to a task. Such representations provide the agents with information on unseen objects, allowing the generalization of previously learned policy.

The existing affordance methods for manipulation have two stages [14, 21]. VAT-Mart [21] first trains the affordance map with data collected by an RL agent driven by curiosity, then fine-tunes the affordance map and the RL agent together. In Where2act [14], the affordance is associated with a corresponding primitive action for each task, such as pushing and pulling. The two-stage approach, which trains the affordance map first and proposes action sequence with the learned affordance, has a drawback that the success rate of interaction is highly related to the accuracy of the learned affordance. Any imperfect affordance predictions will significantly reduce the task performance.

In this paper, we investigate learning affordance in an end-to-end fashion, in which the affordance does not associate with a specific primitive action but associate with the contact information predicted by a module trained with past manipulation experiences. In our method, a RL algorithm learns to utilize visual affordance to find the most suitable position to interact with, while visual affordance is generated from contact information. We also incorporate visual affordance in the reward signals to encourage the RL agent to focus on points with higher affordance. The advantage of end-to-end affordance learning has two folds: 1) affordance can awaken the agent where to act as an additional observation and be incorporated into reward signals to encourage the agent to focus on the point with high affordance; 2) automatically learn affordance semantics and manipulation policy through an RL pipeline without human demonstration or a module dedicated to collecting data. The affordance is treated as input and processed by a neural network to obtain the features and inform the decision at the feature level rather than generating action sequences directly based on the weights of affordance[21] that needs an accurate affordance prediction. We conduct experiments on four representative robot tasks, including opening and closing doors and drawers with diverse shapes. The performance of our proposed method outperforms the baseline, including RLs and the current two-stage affordance methods. In summary, our main contributions are:

- (1) We propose an end-to-end affordance learning method, improved object level generalization ability with a large margin.
- (2) We establish an object-level large-scale benchmark and a high-efficiency training platform for object-level generalization tasks.

2 Related Work

2.1 Generalizable Manipulation Policy Learning

Generalization is essential in most real world manipulation tasks. The recent simulators and benchmarks have boosted the development of object-level generalizable manipulation policy learning methods [22, 15, 26]. For rigid object manipulation, there are already robust algorithms handling tasks such as grasping [4, 3] and planar pushing [11, 25]. However, generalization on articulated objects with multiple parts is still challenging. Many studies have attempted to tackle this problem from different perspectives. UMPNet [23] utilizes visual observation to directly propose action sequence, while [1, 9] achieves robust and adaptive control through model prediction.

2.2 Visual Actionable Affordance Learning

Up to now, several studies have demonstrated the generalizability of affordance representation on manipulation [14, 21], grasping [13, 10], scene classification [6, 27], scene understanding [8, 24] and object detection [7]. The semantic information in affordance is instructive for manipulation. Some prior affordance learning methods for manipulation have two stages, such as Where2Act [14], VAT-Mart [21] and AdaAfford [20]. They need to collect interacting data to pretrain the affordance in the first stage, and train the policy depending on the affordance in the second stage. Others [2, 13, 16] utilize demonstration collected from human to learn the affordance. Unlike those studies, ours can automatically learn affordance semantics through a contact predictor process in an end-to-end fashion within the RL process.

3 Method

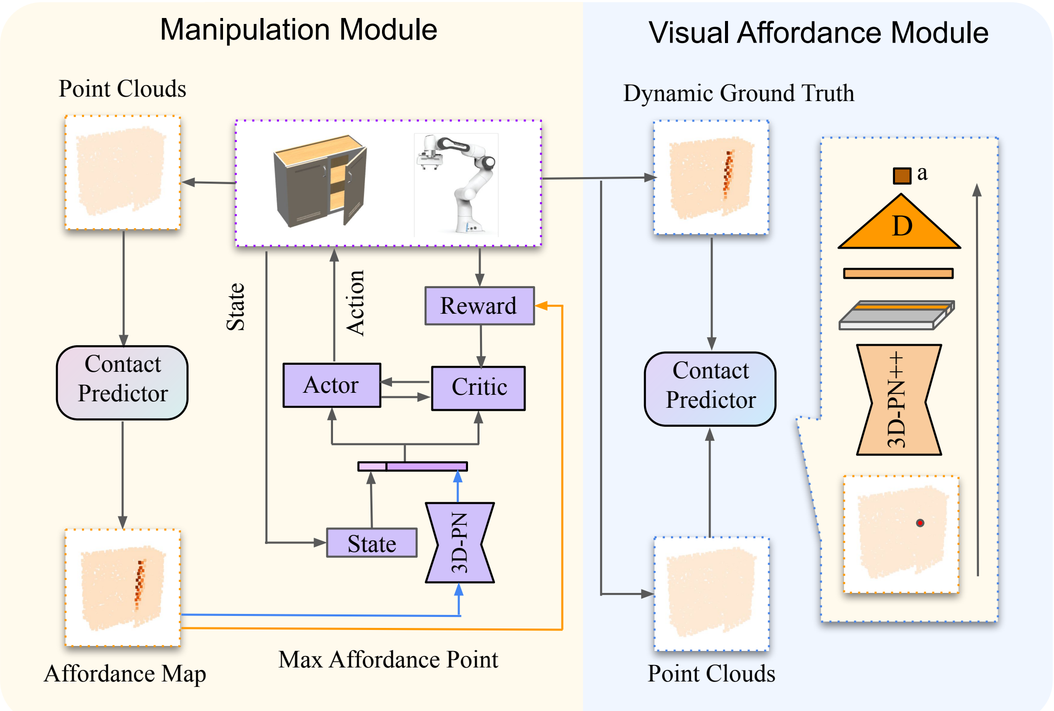

Figure 2: **Training pipeline.** Our pipeline contains two main modules: *Manipulation Module* (MA Module) generating interaction trajectories and *Visual Affordance Module* (VA Module) learning to predict affordance map. The *Contact Predictor* (CP), shared across two modules, serves as a bridge between the two: 1) MA Module uses the affordance map predicted by the CP as a reward feedback (indicated by the orange arrow) and as a part of the input observation; 2) VA Module uses the dynamic ground truth that is produced from data generated by MA Module as the target for training CP.

3.1 Method Overview

In manipulation settings, contact is the fundamental way human interacts with a object. We believe that human contact positions during interactions reflect the understanding of crucial semantic information about the object *i.e.*, affordance. For example, humans grasp a handle to open a door because the handle provides the position to apply force.

We propose an end-to-end affordance learning framework for training a generalizable policy for manipulating 3D articulated objects. As shown in Figure 2, our framework is comprised of 1) *Manipulation Module* (MA Module), which uses the affordance map predicted by the *Contact Predictor* (CP) as a reward feedback and as a part of the input observation; 2)*Visual Affordance Module* (VA Module), which uses the dynamic ground truth that is produced from data generated by MA Module as the target for training CP.

Concretely, at time step t, the MA Module proposes an action a_t , utilizing the affordance map M_t predicted by the VA Module. The affordance map 1) provides additional *observation* for the agent to propose actions a_t; 2) is incorporated in the reward to encourage the agent to approach points with higher affordance. The contact information produced by the action is added to the *Contact Buffer* (CB). Every k time steps, we calculate the *Dynamic Ground Truth* (GT) and fit the CP to GT.

3.2 Visual Affordance Module: *Affordance as Guidance*

When human beings try to interact with articulated objects (*e.g.*, doors), we have the intuition of where to interact (*e.g.*, door handle). *Visual Affordance Module* provides the MA Module with such intuition in terms of visual affordance. Visual affordance is highly localized to each point on the object and thus can generalize to different shapes [21].

VA Module learns from the previous preference of the MA Module to provide information on how preferable is each position. Here we use *frequency* to measure how preferable is each position on the object.

Input. We represent the environment with a point cloud $\mathcal{P}$ and additional masks m . For each point on $\mathcal{P}$, the additional masks take value in $\{00, 01, 10, 11\}$. The first digit indicates whether the point belongs to the movable part of the object, the second digit indicates whether the point belongs to the end effector of the robotic arm.

Output. The output of the VA Module is a per-point affordance map *M*, which expresses the preference of the agent for the points on $\mathcal{P}$.

Module Architecture. The main part of VA Module is the *Contact Predictor*. In the predictor, We use a PointNet++ [18] to extract a per-point feature $f \in \mathbb{R}^{128}$ from point cloud of objects, which is fed through a Multi-layer Perceptron (MLP) to evaluate per-point actionable affordance. Another part of VA Module is the *Contact Buffer*, which keeps record of history contact information and generates the GT for training CP .

Dynamic Ground Truth. In order to train the CP, we obtain training data by keeping a record of preferred interaction positions on the object. For object of type i, we maintain a fixed-length array *Contact Buffer* CB^i to record the position of the end effector in the coordinate system of the object at the time of mutual contact. The buffer randomly evicts a record whenever a new record of contact event is inserted. At time step t, for every point p in the point cloud $\mathcal{P}^i_t$, we calculate the number of contacts within radius r from p recorded in the *Contact Buffer* CB^i_t, and apply normalization to get dynamic ground truth GT^i_t .

$$GT^i_t(p) = \frac{\sum_{q \in CB^i_t} I(|p-q|_2 < b)}{\max_p \sum_{q \in CB^i_t} I(|p-q|_2 < b) + \epsilon}. \tag{1}$$

The network of the CP is updated with GT^i_t .

Training. We use supervised learning to train CP .

$$CP^*_t = \underset{CP}{\arg\min} \sum_i s^i_t \left\| \sum_{p \in p^i_t} CP(p|c^i) - GT^i_t(p) \right\|_2 \tag{2}$$

where s^i_t is the current success rate of tasks with object i and CP^*_t is the optimal *CP* .

3.3 Manipulation Module: *Affordance as Reward Feedback*

Manipulation Module (MA Module) is a reinforcement learning framework, learning to manipulate articulate objects from scratch. Different from previous RL methods, our MA Module takes advantage from both the reward and observation generated by the VA Module.

To improve sample efficiency, we deploy k different objects in the simulator. Each object is replicated n times and given to an robotic arm. Hence, there are a total of kn environments, each with a robotic arm interacting with an object.

Input. The input for MA Module includes, for each environment, 1) a point cloud $\mathcal{P}$ describing the environment; 2) additional masks m for the end effector and the movable part of the object; 3) an affordance map M generated by VA Module; 4) the state w of the robotic arm. The additional masks are the same as in the MA Module. The state w consists of position, velocity and angle of each joint of the robotic arm.

Output. The output of the MA Module is an action a , which is then excecuted by the robotic arm.

Module Architecture. The policy of the MA Module is a neural network π_θ with learnable parameter θ. The network consists of a PointNet [17] and a Multi-layer Perceptron (MLP) . The PointNet extracts feature $f \in \mathbb{R}^{128}$ from the pointcloud $\mathcal{P}$, affordance map M and additional masks m . f is then concatenated with w and fed to the MLP to obtain actions.

Training. We use Proximal Policy Optimization algorithm [19] to train the MA Module.

4 Experiment

4.1 Task Description

We design four manipulation tasks to evaluate our method. Our task settings are highly simplified compared to the real world but still challenging for the agent to learn a generalizable manipulation policy. Concretely, in each task, a robotic arm acts as an agent to complete a specific manipulation task. Our goal is to have the agent reach the success criterion of the task on different shapes of objects in a stable way. In order to let the agent better adapt to the environment, we add a movable base to the arm, which allows the arm to move horizontally within a specific range. Since there are objects with multiple identical parts, and it is impossible for agents to determine which part is the target, we add additional masks to the point clouds of the target part to differentiate it from other parts. In all of the four tasks, we provide dense reward for the agents. The following describes other details of each task:

Close Door. The door motion is constrained by a revolute joint attached to the cabinet body and the target door on each object is initially open. The success criterion is to close the door completely. Since closing doors is a relatively simple task, we apply an additional force on the door attempting to keep the door at the initial position and double the friction of the hinge to increase the difficulty of the task.

Open Door. The target door on each object is initially closed. The success criterion is to open the door to a specific angle. The robot arm has to learn to leverage key parts like the handle to open the door, which is challenging.

Push Drawer. The drawer motion is constrained by a prismatic joint attached to the cabinet body. Similar to *close door*, the success criterion is to close the drawer on a cabinet completely.

Pull Drawer. Similar to *open door*, the success criterion is to open the drawer to a certain distance.

More details are described in our supplementaries.

4.2 Dataset and Simulator

We perform our experiments using the Isaac Gym simulator [12]. Isaac Gym offers a high performance platform to train policies directly on GPU, which allows us to train our models on multiple environments simultaneously. We use a single NVIDIA A100 graphics card, two Intel Xeon Silver 4110 CPU and 128 gigabytes memory to run our experiments. For all tasks, we used Franka Panda robot 3D model as the agent. Our training and testing data are the subset of the PartNet-Mobility dataset [5].

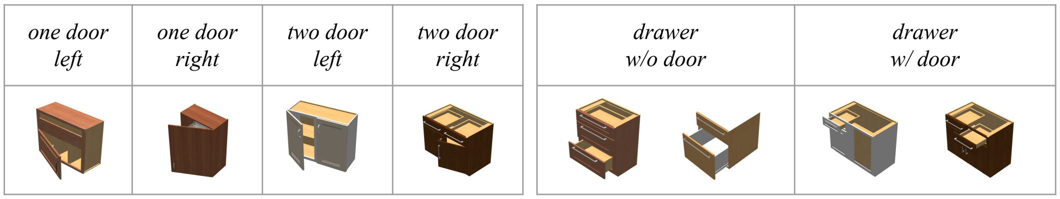

Figure 3: **Object examples of door and drawer datasets.**

For *close door* and *open door* task, we divide the objects with door handles in the StorageFurniture category into 4 subcategories: *one door left*, *one door right*, *two door left* and *two door right*, as shown in Figure 3. For each subcategory, we randomly select 10-12 objects for our training dataset (44 objects in total) and 3 objects for the testing dataset (12 objects in total).

For *pull drawer* and *push drawer*, we divide the objects with door handles in the StorageFurniture category into 2 subcategories: *drawer without door* and *drawer with door*, as shown in Figure 3. For each subcategory, we randomly select 22 objects for our training dataset (44 objects in total) and 8 objects for the testing dataset (16 objects in total). Reference the supplementaries for whole datasets.

4.3 Baselines

We compare our method with five baselines:

- 1) Where2act [14]: Where2act generate short-term interaction proposals, so in our tasks, we implemented a multi-step Where2act baseline (up to 5 steps). Pushing or pulling interactions generated by Where2act gradually changed the part pose until the task is finished or it reaches the maximum number of steps. what's more, due to the nature of short-term and task-less, we assume a oracle pose tracker for this baseline.
- 2) Where2act + RL: We retain the MA Module and replace the Contact Predictor with a pretrained Where2act model that can output a per-point actionable score similar to our affordance map. The parameters in the Where2act model is frozen.
- 3) Multi-Task PPO [19]: We adapt PPO to the multi-task setting by providing the one-hot task ID as input. To make this method comparable on the test set, both the test set and the training set are used in training process. So this is an oracle baseline.
- 4) VAT-Mart [21]: For VAT-Mart baseline, we implemented the algorithm in our tasks. Similar to the original settings in VAT-Mart, we use only a gripper without robotic arm to finsh our tasks in this baseline.
- 5) RL: We trained a point cloud-based RL baseline. The only difference between this baseline and our approach is that there is no *Contact Predictor* to provide additional observations and rewards.

See supplementary for more implementation details about baselines.

4.4 Quantitative Evaluation

To evaluate an algorithm, for each task, we train the algorithm on the training set and saves a checkpoint every 3200 time steps within 64000 total time steps. After training, we choose the checkpoint with the largest average success rate on training set for comparison.

We design two types of success rate to measure learning effectiveness and generalizability:

- 1) Average success rate (AR): The average success rate of an algorithm on the training/testing dataset is the average of the algorithm's success rate on the objects in the dataset. This metric measures the effectiveness of the algorithm. In particular, AR on testing datasets measures generalizability of the algorithm;
- 2) Master percentage (MP): We believe that a policy is stable on an object if it has a success rate of more than 50% on that object. The master percentage of an algorithm on the training/testing dataset is the percentage of objects which the algorithm can success with a probability greater than or equal to 50%. Therefore, this metric measures how stable the algorithm performs.

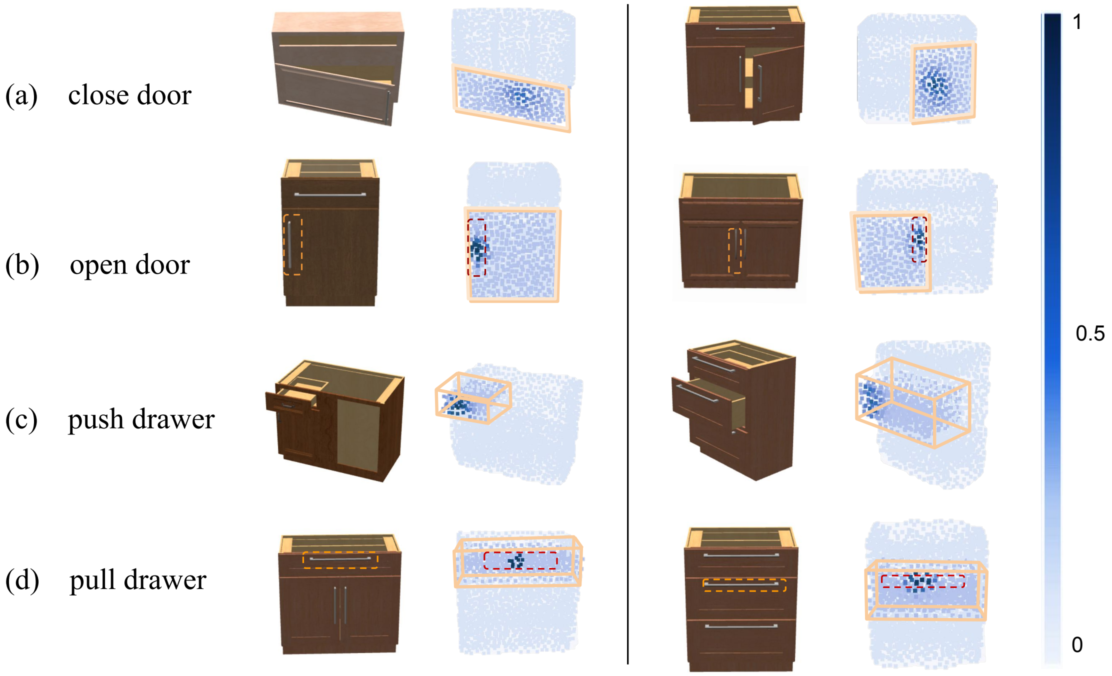

Figure 4: **Affordance Map Visualisation.** For each task, we visualize the final affordance maps generated from two different objects. The darker the color, the higher the Affordance. We also mark the target part and the handle.

Table 1: **Average success rate (AR).**

datasets methods	close door(%)		open door(%)		push drawer(%)		pull drawer(%)	
	train	test	train	test	train	test	train	test
Where2act	11.6	1.6	22.8	14.1	5.2	1.6	19.0	12.9
Where2act+RL	89.2	86.0	13.1	12.4	82.0	86.7	1.5	1.1
VAT-Mart	49.9	33.6	23.2	21.9	52.3	36.3	5.5	5.1
Multi-task PPO	77.4	76.5	4.7	4.0	68.2	69.3	3.3	3.4
RL	73.0	71.0	10.3	4.7	85.8	82.7	9.9	7.6
Ours	**97.2**	**94.3**	**51.9**	**34.4**	**97.2**	**94.9**	**22.4**	**19.4**

Table 2: **Master percentage (MP).**

datasets methods	close door(%)		open door(%)		push drawer(%)		pull drawer(%)	
	train	test	train	test	train	test	train	test
Where2act	0.0	0.0	6.8	8.3	0.0	0.0	2.3	0.0
Where2act+RL	96.8	91.7	7.1	9.4	85.2	91.4	1.4	1.6
VAT-Mart	45.5	41.2	31.8	33.3	52.3	43.8	0.0	0.0
Multi-task PPO	75.0	66.7	0.0	0.0	68.2	68.8	0.0	0.0
RL	75.0	66.7	6.8	0.0	86.1	81.3	0.0	0.0
Ours	**100.0**	**100.0**	**59.1**	**41.7**	**100.0**	**100.0**	**15.9**	**6.3**

Table 1 shows the average success rate (AR) of our method outperforming all baselines on all tasks over both training and testing sets, especially on *open door* and *pull drawer* task. Table 2 shows the master percentage (MP) of our method and baselines, which demonstrates the greater stability of our algorithm. In order to better analyze the experimental results, we visualized the affordance map of different tasks, as shown in Figure 4. Combining the experimental results and the affordance map, we can observe the following phenomena:

- 1) From Figure 4 we can see our affordance map is consistent with human experience.
- 2) In *open door* and *pull drawer*, the handle is important in completing the task. Hence an affordance map which focuses on the handle is beneficial. According to the result of our experiments, Where2act fails to produce an affordance map that concentrates on the handle, while our algorithm succeeds. This further explains the significant advantage of our algorithm in both tasks.
- 3) The MP scores indicate that our method performs stably on more objects than other methods. As shown in row (b) and (d) of Figure 4, the agent understands the key point of the task, implying its cognition in this type of objects.

4.5 Ablation Study and Analysis

Table 3: **Ablation study.**

datasets	close door(%)		open door(%)		push drawer(%)		pull drawer(%)	
methods	train	test	train	test	train	test	train	test
Ours w/o MPR	84.7	91.8	21.8	10.3	94.0	94.5	13.9	5.3
Ours w/o MPO	84.8	83.7	1.1	1.4	80.3	82.9	14.5	9.0
Two Stage	95.9	91.8	27.5	11.5	91.9	**95.2**	11.9	10.4
Ours	**97.2**	**94.3**	**51.9**	**34.4**	**97.2**	94.9	**22.4**	**19.4**

To further evaluate the importance of different components of our method, we conduct ablation study by comparing our method with:

- 1) Ours w/o MPR: our method without the max point reward;
- 2) Ours w/o MPO: our method without the max point obversation;
- 3) Two Stage: This ablation turns our method into a two-stage method. First we train the RL policy without VA Module, then we use the trained RL policy to train CP , and finally we fine-tune the RL policy on the freezed CP to obtain final success rate.

As we can see from Table 3, each part of our method is indispensable. To further analyze whether end-to-end training can learn a better affordance, we compare the affordance map between Two Stage method and our method. As shown in Figure 5, our method produces more concentrated affordance than the Two Stage method. This fully demonstrates the mechanism of the MA Module's feedback to the VA Module. With the feedback from RL, VA Module can continuously adjust CP according to the updating of RL policy and generate better affordance map, which in turn helps RL to learn a better policy. The result shows that our end-to-end method provides a bond for both modules to continuously producing feedbacks to each other, which makes both modules perform better than if they are trained separately.

5 Conclusion

To the best of our knowledge, we are the first to propose an end-to-end affordance learning framework. In the end-to-end fashion, affordance can awaken the agent by providing additional observation and being incorporated into reward signals. Our framework can automatically learn affordance semantics and manipulation policy through an RL pipeline without human demonstration or a module dedicated to collecting data. The simplicity of our method combined with the superior performance over baselines demonstrates the effectiveness of learning end-to-end visual affordance with RL. We believe this method could potentially be a plug-and-play technique for future RL manipulation tasks.

Limitation and Future Work. In this work, we only consider one-stage tasks that there is no need for the agent to think about the relationship and sequence in different stages. Future work may

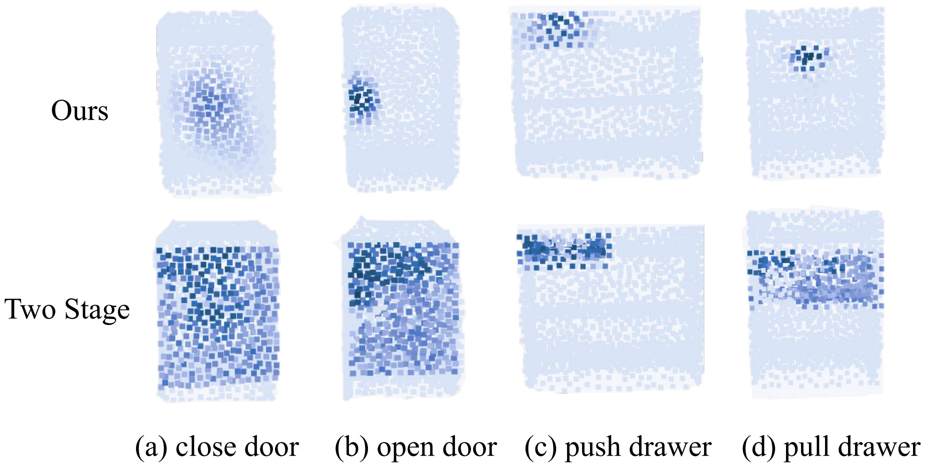

Figure 5: **Affordance map of our method and two-stage ablation.** Our method produces more concentrated affordance. The affordance map shows that with the feedback from RL, VA Module can continuously adjust CP according to the updating of RL policy and generate better affordance map.

extend our framework to multi-stage tasks like *pick up the hammer and hit the nail*. Also, a multi-agent setting may be a promising direction to extend our framework, in which each agent learns a local affordance under the supervision of a global affordance.

Ethics Statement. Our work has the potential to help build robots assisting human. Our visual affordance can be visualized, which makes the system more explainable, thus reduce the risk to be attacked. We train our system in simulators on public third-party dataset, which may introduce data bias. However, this is a general concern of similar methods. We do not foresee any other possible major harm or issue.

References

[1] Ben Abbatematteo, Stefanie Tellex, and George Konidaris. Learning to generalize kinematic models to novel objects. In Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura, editors, *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pages 1289–1299. PMLR, 30 Oct–01 Nov 2020. URL `https://proceedings.mlr.press/v100/abbatematteo20a.html`.

[2] Jessica Borja-Diaz, Oier Mees, Gabriel Kalweit, Lukas Hermann, Joschka Boedecker, and Wolfram Burgard. Affordance learning from play for sample-efficient policy learning. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Philadelphia, USA, 2022.

[3] Michel Breyer, Jen Jen Chung, Lionel Ott, Siegwart Roland, and Nieto Juan. Volumetric grasping network: Real-time 6 dof grasp detection in clutter. In *Conference on Robot Learning*, 2020.

[4] Hanwen Cao, Hao-Shu Fang, Wenhai Liu, and Cewu Lu. Suctionnet-1billion: A large-scale benchmark for suction grasping. *IEEE Robotics and Automation Letters*, 6(4):8718–8725, 2021. doi: 10.1109/LRA.2021.3115406.

[5] 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.

[6] Mandar Dixit, Si Chen, Dashan Gao, Nikhil Rasiwasia, and Nuno Vasconcelos. Scene classification with semantic fisher vectors. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2974–2983, 2015.

[7] 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)*, pages 5882–5889. IEEE, 2018.

[8] Sam Fowler, Hansung Kim, and Adrian Hilton. Human-centric scene understanding from single view 360 video. In *2018 International Conference on 3D Vision (3DV)*, pages 334–342. IEEE, 2018.

[9] Ajinkya Jain and Scott Niekum. Learning hybrid object kinematics for efficient hierarchical planning under uncertainty. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5253–5260. IEEE, 2020.

[10] 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.

[11] Jueya Li, Wee Sun Lee, and David Hsu. Push-net: Deep planar pushing for objects with unknown physical properties. *Robotics: Science and Systems XIV*, 2018.

[12] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, and Gavriel State. Isaac gym: High performance GPU based physics simulation for robot learning. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

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

[14] Kaichun Mo, Leonidas J. Guibas, Mustafa Mukadam, Abhinav Gupta, and Shubham Tulsiani. Where2act: From pixels to actions for articulated 3d objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6813–6823, October 2021.

[15] Tongzhou Mu, Zhan Ling, Fanbo Xiang, Derek Cathera Yang, Xuanlin Li, Stone Tao, Zhiao Huang, Zhiwei Jia, and Hao Su. Maniskill: Generalizable manipulation skill benchmark with large-scale demonstrations. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

[16] 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*, pages 8688–8697, 2019.

[17] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[18] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

[19] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[20] Yian Wang, Ruihai Wu, Kaichun Mo, Jiaqi Ke, Qingnan Fan, Leonidas Guibas, and Hao Dong. Adaafford: Learning to adapt manipulation affordance for 3d articulated objects via few-shot interactions. *arXiv preprint arXiv:2112.00246*, 2021.

[21] Ruihai Wu, Yan Zhao, Kaichun Mo, Zizheng Guo, Yian Wang, Tianhao Wu, Qingnan Fan, Xuelin Chen, Leonidas Guibas, and Hao Dong. VAT-Mart: Learning visual action trajectory proposals for manipulating 3d articulated objects. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=iEx3PiooLy`.

[22] 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*, pages 11097–11107, 2020.

[23] Zhenjia Xu, He Zhanpeng, and Shuran Song. Umpnet: Universal manipulation policy network for articulated objects. *IEEE Robotics and Automation Letters*, 2022.

[24] Chengxi Ye, Yezhou Yang, Ren Mao, Cornelia Fermüller, and Yiannis Aloimonos. What can i do around here? deep functional scene understanding for cognitive robots. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4604–4611. IEEE, 2017.

[25] Kuan-Ting Yu, Maria Bauza, Nima Fazeli, and Alberto Rodriguez. More than a million ways to be pushed. a high-fidelity experimental dataset of planar pushing. In *2016 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 30–37. IEEE, 2016.

[26] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning*, pages 1094–1100. PMLR, 2020.

[27] Lei Zhang, Xiantong Zhen, and Ling Shao. Learning object-to-class kernels for scene classification. *IEEE Transactions on image processing*, 23(8):3241–3253, 2014.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] See Section 2.
   (b) Did you describe the limitations of your work? [Yes] See Section 5.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See 5.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes] See 5.
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] We will open our source code after rearrangment.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [No] Not for all. Data splits are described in 4.2, hyperparameters are included in our source code which will be oecome open source soon.

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] We only ran our experiments once.

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See subsection 4.2.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

   (a) If your work uses existing assets, did you cite the creators? [Yes] See subsection 4.2.

   (b) Did you mention the license of the assets? [No]

   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]

   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes]

   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No]

5. If you used crowdsourcing or conducted research with human subjects...

   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]