# Appendix

## A Introduction on Baselines

### A.1 End-to-end Affordance Model

The main baselines we compare in this paper are end-to-end affordance prediction methods, including HOI [35], HAP [20], HOTSPOTS [44], VRB [4], and LOCATE [29]. We will briefly introduce these baselines: HOI [35] primarily predicts the trajectory of the hand and the area of the object that will be contacted in the future. HAP [20] focuses on observing the state of the hand to learn state-sensitive features and object affordances, including interaction region and the grasping pose. HOTSPOTS [44] propose to perform affordance grounding, which involves determining where on the current object can the interaction occur given a specific action. VRB [4] models an object's affordance as the contact area and the subsequent motion trajectory,then tries to predict them. LOCATE [29] focuses on images of human-object interaction to achieve affordance grounding and can transfer parts across different categories given a source image. The need for the source image makes it a "one-shot" method in contrast to our zero-shot approach.

Apart from LOCATE [29], all other baseline methods output heatmaps concerning the contact location. Particularly, both VRB [4] and HOI [35] predict both the contact area and the trajectory of the hand's movement. However, in this work, we solely focus on the object's contact area and do not consider the trajectory after the contact. For each baseline, we directly use the available pretrained models that were trained on EPIC-KITCHEN [10]. In the main text, we argue that for robot manipulation, the contact should be on points rather than regions. Therefore, for all heatmaps output from the baseline methods, we selected the top five points in the heatmap for subsequent evaluation.

When evaluating LOCATE [29], one label is needed for the action to be performed. For the objects selected from the original AGD-20K dataset, since they all belong to the "hold" action category of the seen setting, we organize these objects into this specific category. As for the newly collected door and drawer categories, we place them within the "open" action category of the unseen setting. For each input action label and corresponding image, LOCATE generates a localization map composed of normalized activation values, which serves as a representation of predictions for the affordance region. Consequently, we select the top five points with the highest activation values for comparison.

### A.2 Semantic Correspondence Methods

Semantic correspondence maps pixel location in the source image to the corresponding pixel location in the target image, such that the pair of pixels bear a similar semantic meaning. These methods can find the semantic correspondence beyond the category of objects, like the wings of a bird and the wings of an airplane, Recent advances find that foundation visual models have the ability to find semantic correspondence in a zero-shot manner, without additional training or finetuning.

We select two lines of works in zero-shot semantic correspondence as baselines, one is based on the idea of dense feature matching, and the other is based on special token optimization. The former one includes DIFT [53], SD-DINO [66], and DINO-ViT [6], while the latter one includes LDM-SC [24]. We use the official code of these methods to get the semantic correspondence results.

Methods based on feature mapping first extract the feature map of the source image and the target image via encoders of the visual model, then match the feature of the source pixel to its nearest neighbor in the target image. The matching is done by calculating the cosine similarity between the source feature and the target feature. Such a straightforward approach has proven to be effective in both DINO features [6] and DIFT features [53]. And SD-DINO [66] seeks to merge DINO and DIFT features to get better performance. These methods are easy to implement and don't need training or optimization of any kind, thus having a very fast inference speed (about 30s on an A40 GPU).

Methods based on special token optimization like LDM-SC [24] take a different approach. They try to find a special token in the language latent space of a text-to-image diffusion model that "describes" the semantic meaning of the source pixel, then find the pixel that matches the special token best in the target model. More specifically, they first calculate the cross-attention map between the special token and the source image, and optimize the special token to maximize the attention value at the source pixel. Then, they calculate the cross-attention map between the optimized special token and the target image, and identify the target pixel as the one with the highest attention valueassociated with the special token. Because each mapping needs to optimize a special token, this approach is slower than dense feature mapping methods (about 5 min on an A40 GPU).

We select DIFT as the semantic correspondence method in Robo-ABC, since it has the most balanced performance in accuracy and inference speed. Please refer to Section C.1 for a visualized comparison between these methods.

# B Implementation Details

**Table 6:** The updated success rate of baselines

| Sampled Points Number | HAP [20] | HOI [35] | HOTSPOTS [44] | LOCATE [29] | VRB [4] |
|---|---|---|---|---|---|
| 500 | 22.2 | 26.5 | 24.2 | 32.0 | 26.5 |
| 1000 | 21.8 | 26.2 | 24.0 | 29.8 | 26.9 |
| 2000 | 21.1 | 25.5 | 23.3 | 26.9 | 24.7 |

## B.1 Sampled points for Contact Points Estimation

All of our baseline models output probability distribution maps that are the same size as the image. To ensure that the sampled points are concentrated on the object, we select points with the top $k$ probability values, normalize their probability distributions, and sample five points from each for analysis. As shown in Table 6, this method mitigates the problem of over-concentration and improves the results on certain baselines. It is evident that the accuracy of our Locate [29] baseline is the highest when the number of sampling points is 500. Therefore, we choose to select the results of all baselines in the main text when taking the top 500 results.

### B.2 Selection of Affordance Buffer

We extract affordance information from the EpicKitchens-100 Video [10] dataset. We visualize all the videos, filtering out and removing those with poor lighting conditions, low video clarity, or the constant presence of objects obstructing the view. The extraction process is similar to that of VRB [4] and HOI [35]. In addition, we added a window for motion blur detection. In this process, we employ the Laplacian operator to detect and quantify motion blur, ensuring that we can crop and select the clearest frame near the frame where contact happens. We use skin segmentation [51] to detect the intersection of the bounding box (bbox) of the object $B_o$ and the hand $B_h$. We then randomly sample contact points within this area. Because some objects have a small surface area, and due to the errors in the mapping of the homography matrix between different frames, the sampled points may be outside the object. This is a significant challenge when it comes to semantic correspondence. To solve this problem, we took the average position of all the points that were mapped back. For most objects, the average position will be on the object. Then, we take this average position and randomly sample five points within a circle with a radius of four pixels around it. This procedure ensures that all the sampled contact points are on the object. The object categories we extracted are shown in the table below:

**Table 7:** List of all object categories in the affordance memory

| | |
|---|---|
| **All categories** | bottle, bowl, drawer, cup, door, fork, knife, cupboard, scissors, wine glass, banana, bread machine, tap, pot, lid, plate, fridge, bag, trash bin, kettle, oven, pizza, mug, cucumber, peeler, mouse, rice cooker, bag, spatula, slicer, computer keyboard, phone, container, hob, heater, onion, tray, melon, coffee maker, remote, dishwasher, spoon, processor, sponge, package, dough, meat, cheese, blender, button, tomato |

### B.3 Evaluation Dataset

Apart from the evaluation dataset from LOCATE, we collect two new types of objects using Labelme: door and drawer. For a cabinet, there may be multiple handles. When we mark the Ground Truth (GT), we mark all the possible interactive positions to ensure the fairness of the validation.

### B.4 The impact and selection of unseen categories.

The evaluation dataset's division into seen and unseen categories is based on whether these categories exist in our memory. However, as we only extracted $1/6$ of the EK100 categories and $1/20$ samples for the memory, all other baselines which we directly use the official checkpoints have seen more categories than us. For LOCATE, it has been trained on all categories in the evaluation dataset, except for "door" and "drawer". Yet, our method doubles the SR of the best baseline.

For "unseen" categories evaluation, we exclude each "seen" category from the memory separately and compare the results in 8. Robo-ABC keeps comparable performance even when the whole evaluation set is converted to an "unseen" setting. The results are shown in

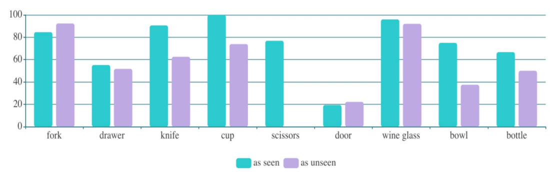


**Fig. 8: The ablation results of unseen objects.**

### B.5 Memory Retrieval

In the process of memory retrieval through visual and semantic similarity, we utilize different encoders to test the effectiveness of the retrieval results. Notably, we compare CLIP-B32, CLIP-B16 and LPIPS metrics.

For the CLIP-B32+LPIPS retriever, we use the CLIP-B32 encoder to retrieve the five images from the affordance memory that are most semantically similar. Subsequently, we employ a pre-trained VGG network to identify the one with the minimum LPIPS value among these five images, signifying the visually most similar image to the source.

### B.6 Affordance Mapping

During the extraction of affordance memory, we retrieved more than one contact point on the object, and experimented with two different ways of mapping the affordance memory to the target image. The first one is to map each source point to the target image separately, and then average the results. The second one is to first average the contact points, then map the average point to the target image. We found that while the second method is more effective, the first approach yields slightly better results, and we use this method in our experiments.

### B.7 Results for unseen and seen objects

The experiments of Table 2 are done on both seen and unseen objects, and the results are aggregated together. The seperate results are shown in 9. The success rate of each category in Table 2 is shown in 10.

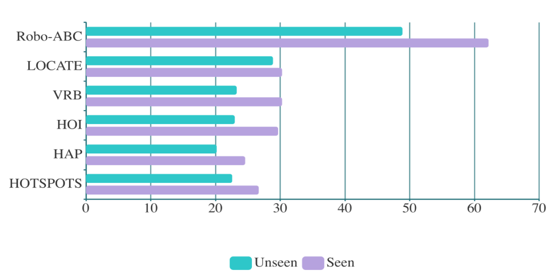


**Fig. 9: The seperate results of Table2**

### B.8 Robot Hardware Setup

For the real-world robot setup, we use the Franka Panda robot, and we set up a calibrated RealSense L515 camera externally on the robot for observations. Grasping flat objects is particularly challenging for a two-fingered gripper. We are following previous works like VRB to set up the real-world scene in a way that facilitates grasping and focusing on the prediction of contact points. All experiments are based on the same pre-trained AnyGrasp model for fair comparison.

## C Additional Results

### C.1 Correspondence

We visualize the correspondence results for different correspondence models and show them in Figure 12. DIFT-based methods yield the best correspondence results while having a faster inference speed.

### C.2 Additional Real Robot Results

The videos of real-world robot deployment results can be found in the supplementary material. We recommend watching the videos to have a better sense of Robo-ABC's ability. We report the success rate of our method for seven object categories along with VRB [4] in Table 8. For each experiment, we reposition the object and try to grasp for five times and report the overall average success rate.

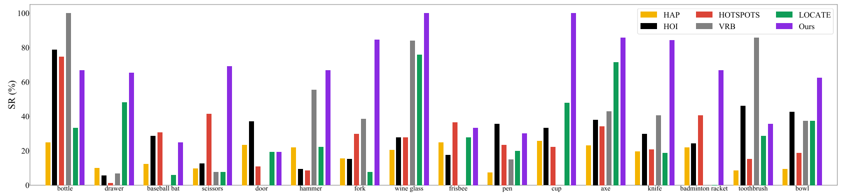


**Fig. 10: Success rate by category.** We demonstrate the performance of Robo-ABC and other baselines across various object categories within the entire evaluation dataset. As can be seen, in the vast majority of cases, Robo-ABC exhibits superior zero-shot generalization capabilities.

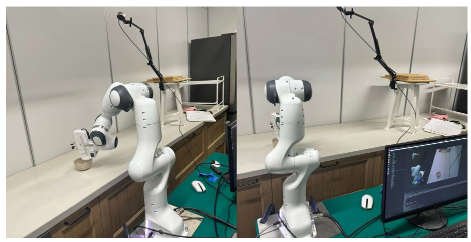

**Fig. 11: Our workspace.**

### C.3 Viewpoint changes and messy scenes

We have evaluated the grasping experiments in clustered and cross-view scenarios in Figure 13. These experimental results further demonstrate that Robo-ABC is indeed very robust to the viewpoint changes and messy scenes.

## D Future Works

Although most objects share similar contact regions across different actions, our pipeline has the potential to support a wider range of affordance types beyond just "grasping" by storing memories according to different actions. By categorizing memories based on action types and retrieving them accordingly, we can enable a broader variety of actions. Moreover, the low resolution of the original dataset and the unusability of some videos due to lighting problems can affect the accuracy of finding semantic correspondences in the subsequent stage.

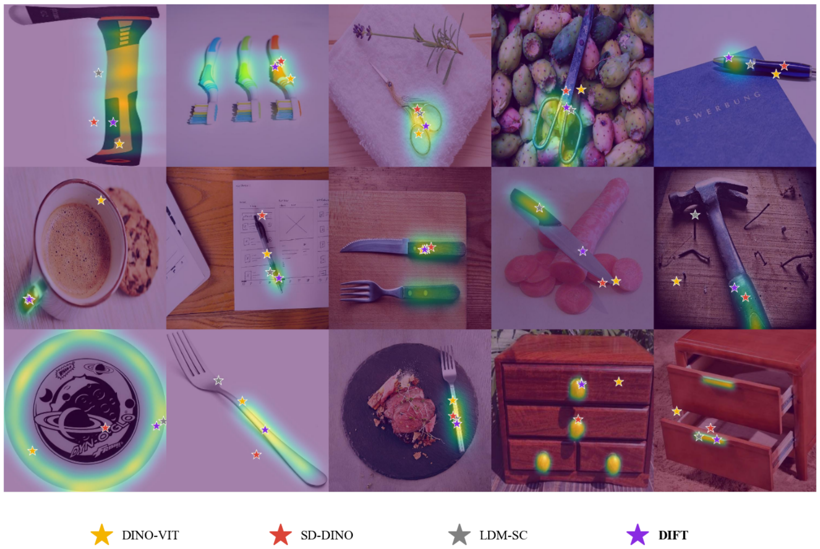


**Fig. 12: Visualization of different correspondence methods on the same source image.**

| **Object** | bowl | bottle | racket | scissor | knife | cup | glass | **overall** |
|---|---|---|---|---|---|---|---|---|
| VRB | 3/5 | 5/5 | 2/5 | 4/5 | 2/5 | 3/5 | 5/5 | 68.6% |
| Ours | 4/5 | 5/5 | 5/5 | 4/5 | 3/5 | 5/5 | 4/5 | **85.7%** |

**Table 8:** Real-World Success Rate

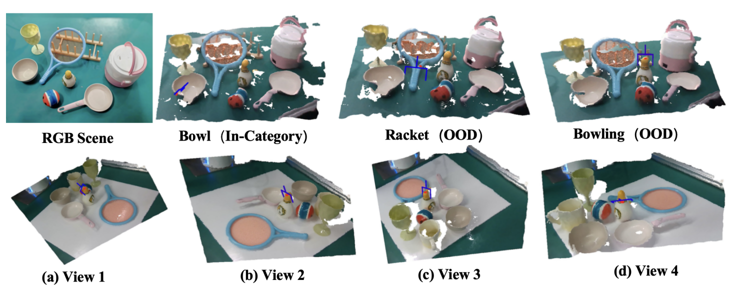


**Fig. 13: Cluster setting and Cross-view setting.** In the first row, Robo-ABC can infer the correct grasping pose for different objects with a complicated scene, while the second row demonstrates Robo-ABC's robustness against viewpoint changes.