

# Learning Foresightful Dense Visual Affordance for Deformable Object Manipulation

Ruihai Wu<sup>1,3,4\*</sup> Chuanruo Ning<sup>2,1\*</sup> Hao Dong<sup>1,3,4†</sup>

<sup>1</sup>CFCS, School of CS, PKU <sup>2</sup>School of EECS, PKU <sup>3</sup>BAAI

<sup>4</sup>National Key Laboratory for Multimedia Information Processing, School of CS, PKU

## Abstract

Understanding and manipulating deformable objects (e.g., ropes and fabrics) is an essential yet challenging task with broad applications. Difficulties come from complex states and dynamics, diverse configurations and high-dimensional action space of deformable objects. Besides, the manipulation tasks usually require multiple steps to accomplish, and greedy policies may easily lead to local optimal states. Existing studies usually tackle this problem using reinforcement learning or imitating expert demonstrations, with limitations in modeling complex states or requiring hand-crafted expert policies. In this paper, we study deformable object manipulation using dense visual affordance, with generalization towards diverse states, and propose a novel kind of foresightful dense affordance, which avoids local optima by estimating states' values for long-term manipulation. We propose a framework for learning this representation, with novel designs such as multi-stage stable learning and efficient self-supervised data collection without experts. Experiments demonstrate the superiority of our proposed foresightful dense affordance. Project page: <https://hyperplane-lab.github.io/DeformableAffordance>

## 1. Introduction

Many kinds of deformable objects, such as ropes and fabrics, exist everywhere in our daily life. Perceiving and manipulating deformable objects plays a significant role and paves the way for future home-assistant robots.

Unlike rigid or articulated objects, due to the complex dynamics, high-dimensional and nearly infinite degrees of freedom, large action space, and severe self-occlusion, deformable objects pose much more challenges to manipulate. Moreover, unlike tasks for rigid objects (like grasping) or articulated objects (like pushing a door) that require one or only a few steps to accomplish, deformable object manipulation tasks usually require many steps to accomplish,

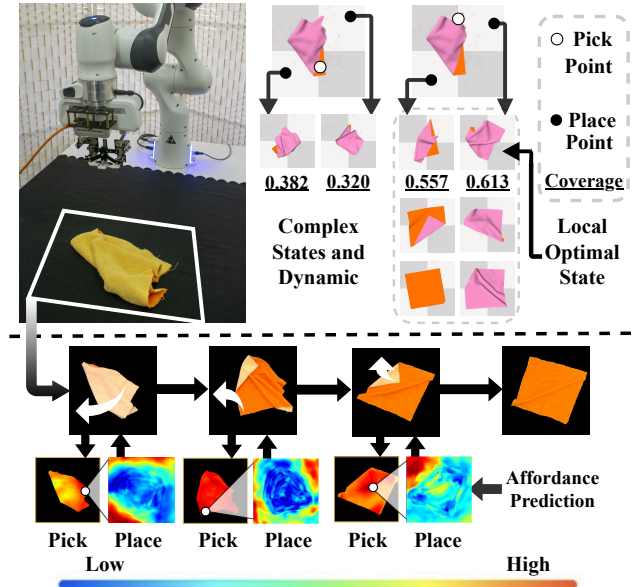


Figure 1. **Deformable Object Manipulation** has many difficulties. 1) It requires **multiple steps** to complete. 2) Most actions can hardly facilitate tasks, for the **exceptionally complex states and dynamics**. 3) Many **local optimal states** are temporarily closer to the target, but making following actions harder to coordinate for the whole task. We propose to learn **Foresightful Dense Visual Affordance** aware of future actions to avoid local optima for deformable object manipulation, with real-world implementations.

laying much focus on relationships and influences between actions in a sequence, as an action leading to local optimal states may not eventually complete the task.

Specifically, as shown in Figure 1, unfolding crumpled cloth requires a sequence of actions (pick-and-place). Because of the exceptionally complex states and dynamics, and large action space, most actions fail to facilitate the task. Moreover, although some cloth in local optimal states temporarily have larger coverage areas than others, following actions face difficulties in smoothly completing the task.

Proposed by Gibson [13] and aimed at providing indicative information for agents to execute actions (e.g., el-

\*Equal contribution, order determined by coin flip.

†Corresponding author

elementary actions such as picking and pulling) and thus facilitating downstream tasks, visual affordance is arousing much attention in vision and robotics. Recent works have demonstrated its efficiency in a large range of tasks like grasping [28, 32, 5, 21, 20, 48], manipulating articulated objects [30, 44, 41, 10] and assisting robots in a scene [34, 35, 14]. Among them, point-level dense affordance [44, 30, 31, 41] learns whether an action on each point of the object could facilitate the task. Compared with Reinforcement Learning (RL) approaches, dense affordance is stably supervised and has better generalization ability towards objects with diverse shapes.

The above dense affordance is suitable for representing deformable objects with complex states, capable of indicating whether diverse actions could help complete the task.

While most previous works only study dense affordance for short-term manipulation on rigid [49] or articulated objects [30, 44], to tackle the local optima problem in multi-step manipulation, we move a step towards equipping dense affordance with foresightfulness for future states.

Inspired by Dynamic Programming with Bellman Equation [1] and Q-Learning [42], estimating a state’s ‘value’ (expected return in the long term, instead of only the current performance) for future actions to coordinate and eventually complete the task will help avoid local optimal states and boost the smoothness and quickness of multi-step manipulation. Dense affordance is suitable for such ‘value’ estimation, because such ‘value’ requires understanding and aggregating a large number of diverse following actions on complex states and their corresponding results.

With such state ‘value’s (instead of only the current performance) in supervisions, dense affordance would gain foresightfulness for the future.

We propose to learn dense visual affordance for manipulating deformable objects, and further estimate state ‘value’s by aggregating such affordance to avoid local optima and smoothly accomplish multi-step tasks. As shown in Figure 1 (Down), the task can be accomplished smoothly using our proposed dense affordance in the real world. To learn such representations, we propose a novel framework generic to diverse tasks with many novel designs, such as the stage-by-stage stable training and *Fold to Unfold* efficient multi-stage data collection. Thus the proposed affordance could be learned stably, efficiently, and self-supervisedly without hand-crafted policies for different tasks. Experiments on representative benchmark tasks demonstrate our framework’s impressive performance.

In summary, our contributions are:

- We propose to use dense visual affordance for manipulating deformable objects with complex states and dynamics, using such representation to estimate state ‘value’s for future actions to avoid local optima and smoothly accomplish multi-step manipulation tasks;

- We propose a self-supervised framework with novel designs such as multi-stage training and efficient data collection to learn the proposed affordance stably;
- Qualitative and quantitative results on representative benchmarks and real-world experiments demonstrate the superiority of our proposed dense visual affordance and learning framework for deformable objects.

## 2. Related Work

### 2.1. Perceive and Manipulate Deformable Objects

For deformable object perception, previous works learn and leverage appearance and geometric information [36, 33, 40], softness [3], visual-tactile sensation [7], shape completion [4], emotional perception [24], optimal perception [8] and the continuity of perception [29, 9, 22]. The manipulation is challenging for its high complexity. Typical methods utilize Reinforcement Learning (RL) or Imitation Learning. State-based RL methods [17] consume object states to perform objects manipulation, while others [45, 25] utilize visual RL to manipulate rope and cloth with limited states or targets. Transporter [47, 38] exploits imitation learning to perform different tasks, requiring human-designed expert policy and demonstrations for each task, limiting the generalization over tasks and objects. Vision-based methods [11, 18] use visual feedback or correspondence to perform manipulations. Flow-based methods [39, 43] learn forward dynamics for manipulation, requiring much time in planning. Instead of learning flow-based dynamics or from demonstrations, our work builds a bridge between perception and manipulation, taking advantage of the generalization ability of affordance and estimating state ‘value’ for efficient planning.

### 2.2. Visual Affordance for Robotic Manipulation

Learning visual affordance [13] aims to learn representations of objects or scenes that indicate possible ways for robots to interact and complete tasks. Many recent works learn affordance for grasping [28, 32, 5, 21, 20, 48], articulated object manipulation [30, 44, 41, 10, 12], object-object interaction [31], collaboration [49] and interaction in a scene [34, 35, 14]. Most tasks can be achieved in a single step (*e.g.*, grasping objects, pulling drawers), and learned affordance only contains actionable information for single-step manipulation. However, deformable objects with complex states and dynamics require many steps to manipulate. We move a step towards proposing dense affordance for deformable objects, which not only indicates complex states and dynamics but also takes the subsequent actions of a certain state into consideration for long-term tasks to avoid local optima. FlingBot [15] learns affordance for unfolding cloth with flinging actions, while our proposed affordance is more generic for large action space and diverse tasks.

### 3. Problem Formulation

In this section, we describe how we formulate learning **policy for multi-step deformable object manipulation** into learning **policies for picking and placing**.

Following current benchmarks *DeformableRavens* [38] and *SoftGym* [26], we formulate the problem as learning a policy  $\pi$  that outputs robot action  $a_t$ , given the visual observation  $o_t$  (denoted as  $o$ ) at time  $t$ . We use pick-and-place as primitive action, *i.e.*,  $a_t = (a_{pick}, a_{place})$ , with  $a_{pick}$  and  $a_{place}$  denoting picker pose for picking and placing. As explained in [38], the picker can complete the tasks without rotation, so  $a_{pick}$  and  $a_{place}$  can be denoted as picking point  $p_{pick}$  and placing point  $p_{place}$ .

The composite of  $p_{pick}$  and  $p_{place}$  makes up a large combinatorial action space hard for a network to learn directly and simultaneously. For the underlying nature that  $p_{place}$  highly depends on  $p_{pick}$ , we follow [45] and disentangle learning the composite of  $p_{pick}$  and  $p_{place}$  into respectively learning  $p_{pick}$  and  $p_{place|p_{pick}}$  (denoted as  $p_{place}$ ).

Therefore, we formulate the problem into learning picking and placing policies. In Method Section 4, we describe how we learn the policies using **foresightful dense visual affordance** with each state’s ‘value’ to avoid local optima.

### 4. Method

#### 4.1. Overview

As shown in Figure 2, our framework is composed of two main parts: (1) we propose to use dense affordance representing manipulation policy (4.2), estimate state ‘value’ using dense affordance and incorporate ‘value’ into dense affordance to avoid local optima for multi-step manipulation (4.3), break the picking-placing dependency cycle and stably learn affordance stage by stage (4.4); (2) to tackle the difficulty in collecting multi-stage and successful interactions, we propose a method (named *Fold to Unfold*) generic to many tasks to efficiently collect data in the reversed task completion order (*e.g.*, collecting unfolding data by folding cloth) (4.5). Besides, we propose Integrated Systematic Training to further integrate the proposed affordance into a whole system (4.6). Finally, we describe network architectures and loss function (4.7).

#### 4.2. Dense Visual Affordance Representing Policy

This section introduces how to use dense visual affordance to represent manipulation policy. For simplicity, we first discuss affordance for a greedy policy.

Described in Section 3, we formulate the manipulation problem into learning picking and placing policies, *i.e.*, the picking point  $p_{pick}$  and placing point  $p_{place}$  given the observation  $o$  (with the size of  $m \times n$  points). As the action space is all points on the object for picking, and all points in the space for placing, it comes naturally to use dense picking affordance map  $A_o^{pick}$  (size  $m \times n$ ) indicating how picking

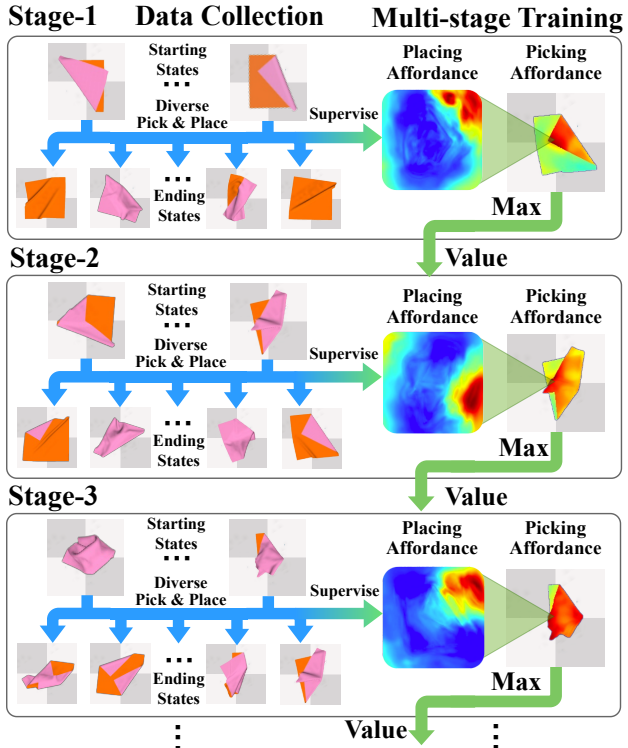


Figure 2. **Our proposed framework** learns dense picking and placing affordance for deformable object manipulation (*e.g.*, **Unfolding Cloth**). We collect multi-stage interaction data efficiently (Left) and learn proposed affordance stably in a multi-stage schema (Right) in the reversed task accomplishment order, from states close to the target to complex states.

each point will facilitate the task, and dense placing affordance map  $A_{o|p_{pick}}^{place}$  (size  $m \times n$ ) indicating how placing the picking point  $p_{pick}$  on each point will facilitate the task.

Like other dense affordance studies [30, 44, 49], a **greedy** way to supervise  $A_{o|p_{pick}}^{place}$  is directly using the distance between **the target**  $T$  and **the new object state**  $o'$  after picking  $p_{pick}$  and placing on  $p_{place}$  in  $o$ . For example, we use  $1 - dist(o, T)$ , *i.e.*, the cloth coverage area, to supervise  $A_{o|p_{pick}}^{place}$  for unfolding. So we can estimate placing affordance score  $g_{o, p_{place|p_{pick}}}^{place}$  on  $p_{place}$  as (Figure 3, Middle, temporarily dismiss ‘value’ in the Figure):

$$g_{o, p_{place|p_{pick}}}^{place} = 1 - dist(o', T) \quad (1)$$

Given a picking point  $p_{pick}$ , the placing policy will select  $p_{place}$  with the highest affordance, so the picking affordance score  $g_{o, p_{pick}}^{pick}$  on  $p_{pick}$  can be estimated using the affordance score of the best placing point (Figure 3, Left):

$$g_{o, p_{pick}}^{pick} = \max_i g_{o, p_i|p_{pick}}^{place}, i \in \{1, \dots, m \times n\} \quad (2)$$

We use two networks  $\mathcal{M}_{pick}$  and  $\mathcal{M}_{place}$  to respectively learn  $A_o^{pick}$  and  $A_{o|p_{pick}}^{place}$  (architectures in Section 4.7).

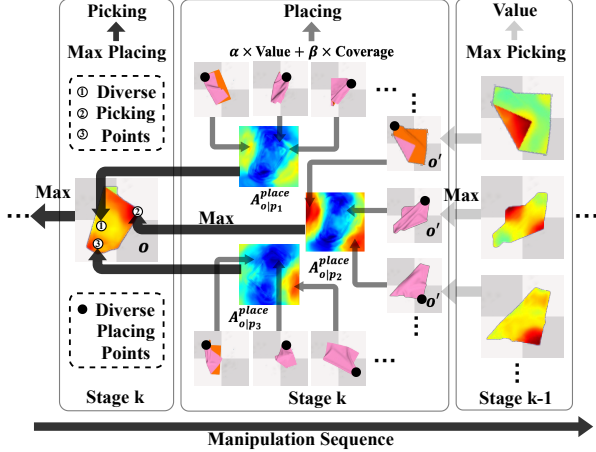


Figure 3. Learning placing and picking affordance with state ‘value’ for the future. Left to Right: The bottom black arrow indicates the manipulation (inference) order. Right to Left: Arrow flows show dependencies among placing affordance, picking affordance and ‘value’s. Given observation  $o$ , we select 3 picking points  $p_1 p_2 p_3$ , and show how to supervise corresponding placing affordance  $A_{o|p_1}^{place} A_{o|p_2}^{place} A_{o|p_3}^{place}$ , and how to supervise  $A_o^{pick}$  on  $p_1 p_2 p_3$  using computed corresponding placing affordance.

During inference, in each step, the greedy policy first selects  $p_{pick}$  with the highest affordance score in  $A_o^{pick}$  given  $o$ , and then selects  $p_{place}$  with the highest affordance score in  $A_{o|p_{pick}}^{place}$  given  $o$  and  $p_{pick}$ , resulting in the **temporary best state** after the pick-and-place action.

### 4.3. Estimating State Values and Learning Foresightful Affordance

As shown in Figure 4, for multi-step manipulation, only evaluating the direct distance between the current state and the target (greedy method described above) may result in many local optimal states that are temporarily closer to target but harder for future actions to complete the whole task.

Dynamic Programming (DP) [1] and Q-Learning [42] tackle this local optima problem by estimating the ‘value’ of a state that indicates whether a state is beneficial to the task in the long term (instead of the current performance). Inspired by them, we can add such state ‘value’ (formally formulated in Equation 4) to the estimation of  $A_{o|p_{pick}}^{place}$ :

$$g_{o, p_{place}|p_{pick}}^{place} = \alpha \times value_{o'} + \beta \times (1 - dist(o', T)) \quad (3)$$

where  $\alpha + \beta = 1$

With such  $A_{o|p_{pick}}^{place}$  that is foresightful for long-term tasks,  $A_o^{pick}$ , which is the aggregation of  $A_{o|p_{pick}}^{place}$ , will therefore get such foresightfulness spontaneously.

Estimating the ‘value’ in a state requires understanding all possible actions and their corresponding future results, and then selecting the best for the estimation. As  $A_o^{pick}$  estimates the action result on each point, state ‘value’ can

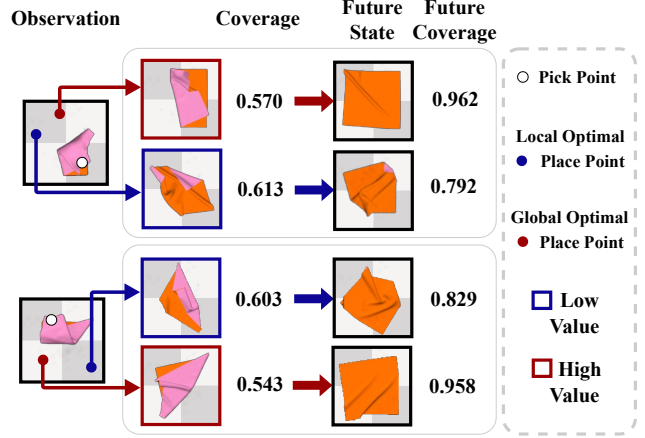


Figure 4. Local Optima v.s. Global Optima. Many local optimal states are temporarily closer to target (e.g., having larger coverage area in unfolding task), but making future actions hard to coordinate to accomplish the whole task. We propose to use ‘value’ to indicate whether a state is suitable for future actions, with which the policy can avoid those local optimal states in multi-step tasks.

be estimated by selecting the  $p_{pick}$  with the highest score in  $A_o^{pick}$  (Figure 3, Left):

$$value_o = \max_i g_{o, p_i}^{pick}, i \in \{1, \dots, m \times n\} \quad (4)$$

As  $value_o$  could be estimated by  $A_o^{pick}$ ,  $A_{o|p_{pick}}^{place}$  could be reformulated using both  $A_o^{pick}$  and the direct distance:

$$g_{o, p_{place}|p_{pick}}^{place} = \alpha \times \max_i g_{o', p_i}^{pick} + \beta \times (1 - dist(o', T)) \quad (5)$$

where  $\alpha + \beta = 1$

### 4.4. Break the Cycle and Cut into Stages: Learning Foresightful Affordance Stably Stage by Stage

The above-formulated picking and placing affordance forms a chicken-egg dependency cycle: picking affordance is dependent on placing affordance, while placing affordance is dependent on picking affordance.

To break the dependency cycle, **states close to the target** (e.g., cloth almost fully unfolded) become the key. As the task is almost accomplished in these states, their values and direct distances to the target are nearly the same. So for an interaction where the ending state  $o'$  is close to the target, we directly use their distances to the target (instead of both distances and ‘value’s) to supervise the corresponding placing affordance of the starting state  $o$ :

$$g_{o, p_{place}|p_{pick}}^{place} = 1 - dist(o', T) \quad (6)$$

when  $dist(o', T)$  is close to 0 i.e., the last step

According to Equations 2 5 6, the proposed picking and placing affordance could be estimated without the dependency cycle. As the dependency cycle breaks in states close to the target, we learn dense affordance from these simple

states to more complex states (reversed order of inference) as shown in Figure 2 (Right). Specifically, we divide the learning procedure into multiple stages. In the first stage, we learn affordance for states that can reach states close to the target within one step, using the direct distances of the following states as supervisions. In the  $i$ -th ( $i > 1$ ) stage, we learn affordance for states that can reach states in the  $(i-1)$ -th stage within one step, using both the direct distances and the ‘value’s of states in the  $(i-1)$ -th stage as supervisions. In each stage, we first train  $\mathcal{M}_{place}$  using stable ‘value’s provided from the trained  $\mathcal{M}_{pick}$  in the previous stage, and then train  $\mathcal{M}_{pick}$  with stable supervisions (max of placing affordance) provided from the trained  $\mathcal{M}_{place}$  in this stage.

In this way, during the whole training,  $A_{o|p_{pick}}^{place}$  and  $A_o^{pick}$  both have stable supervisions from the former stage, and can provide stable supervisions for the latter stage.

This stage-by-stage learning schema empowers our method with superiority over RL methods. Although RL also estimates states and actions using Bellman Equation, it simultaneously estimates and updates values across all states (offline RL) or trajectories of states (online RL), which is difficult to efficiently and stably learn the values, as RL struggles in iteratively updating state ‘value’s, especially when considering the prohibitively large state and action space of deformable objects. In contrast, like other dense affordance works [30, 49, 44], we stably learn affordance with ‘value’ using supervised learning, with stable supervisions provided in the previous training stage. Experiments demonstrate our superiority over RL in Section 5.4.

#### 4.5. Fold to Unfold: Efficient Multi-stage Data Collection for Learning Foresightful Affordance

The above-described training schema requires multi-stage data for training. Specifically, in the first stage, the starting states are one-step to states close to the target. In the  $i$ -th ( $i > 1$ ) stage, the starting states are one-step to states in the  $(i-1)$ -th stage. Each starting state’s actions and corresponding ending states should be diverse, as we need to learn the affordance representing dense distributions.

However, due to the complexity of states and dynamics, data collection methods used by previous dense affordance works (random policies [30, 31] or state-based RL [44, 49]) could hardly collect such data. On the other hand, designing expert policies [38] or hand-crafting demonstrations are difficult and time-consuming for different tasks.

Therefore, we propose a novel self-supervised method, named *Fold to Unfold*, using reversed actions of tasks to efficiently collect multi-stage data. This method is generic to many kinds of deformable object manipulation tasks, with no need for human-designed expert policies or annotations.

Similar to the training procedure, we collect data from states close to the target, to more complex states.

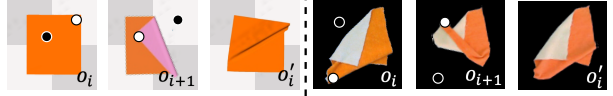


Figure 5. *Fold to Unfold* collection in simulator and real world.

Specifically, as shown in Figure 5, from a state  $o_i$  in stage  $i$ , we select a picking point  $p_{pick}$ , put it on a placing point  $p_{place}$  and get  $o_{i+1}$ . Then, from  $o_{i+1}$ , we execute the reversed action, pick  $p_{place}$ , place on  $p_{pick}$  and get  $o'_i$ . If  $o'_i$  is similar to  $o_i$ , we choose  $o_{i+1}$  as a starting state in stage  $i+1$ , and sample diverse actions on  $o_{i+1}$  with different corresponding results to train dense affordance in the  $(i+1)$ -th stage (shown in Figure 2).

Through a few stages of data collection, object states become complex and diverse, empowering trained affordance networks with generalization towards diverse novel states.

**Note that**, although reversed actions cannot fully recover previous states, *i.e.*,  $o'_i$  are not the same as  $o_i$ , chances are that  $o'_i$  and  $o_i$  are similar, and thus this method still greatly improves sample efficiency. Also, proposed affordance is **not dependent on** this data collection method, as it can be trained on data collected by any method.

#### 4.6. Integrated Systematic Training

Although the above designs and training procedure enable the affordance learning for multi-step tasks,  $\mathcal{M}_{pick}$  and  $\mathcal{M}_{place}$  are trained using only offline collected data in different stages, without considering the actual execution performance of the policies provided by the two modules. During actual manipulation procedures, the policy is the composite policy of  $\mathcal{M}_{pick}$  and  $\mathcal{M}_{place}$ , and pick-and-place actions are executed one by one sequentially as a whole system. Therefore, we propose the Integrated Systematic Training (IST) procedure to adapt  $\mathcal{M}_{pick}$  and  $\mathcal{M}_{place}$  using online data.

In this procedure, with offline trained  $\mathcal{M}_{place}$  and  $\mathcal{M}_{pick}$ , we randomly sample object initial states, use  $\mathcal{M}_{place}$  and  $\mathcal{M}_{pick}$  as the policy to select  $p_{pick}$  and  $p_{place}$ , execute pick-and-place step by step, and use actual results to simultaneously update  $\mathcal{M}_{place}$  and  $\mathcal{M}_{pick}$ . Through this procedure, the two modules are constantly adapted by consecutively online-sampled and actually-executed data, and thus are gradually integrated into a whole system.

#### 4.7. Network Architectures and Loss Function

For architectures of  $\mathcal{M}_{pick}$  and  $\mathcal{M}_{place}$ , we use Fully Convolutional Networks (FCNs) [27] same in Transposer [47, 38] with extra skip-connections as backbone per-point feature extractor. For  $\mathcal{M}_{pick}$ , we directly use  $p_{pick}$  feature to predict picking affordance score on  $p_{pick}$ . For  $\mathcal{M}_{place}$ , we use feature concatenation of  $p_{pick}$  and  $p_{place}$  to predict placing affordance score on  $p_{place}$ . To train both  $\mathcal{M}_{pick}$  and  $\mathcal{M}_{place}$ , we use Mean Absolute Error (MAE) between predictions and ground truth as the loss function.

## 5. Experiments

### 5.1. Tasks, Settings and Metrics

**Tasks.** To demonstrate the superiority of our framework, we select 2 representative tasks from *DeformableRavens* benchmark: (1) **cable-ring**: manipulating a ring to a given green circle, (2) **cable-ring-notarget**: manipulating a ring to any circle, as well as 2 representative tasks from *SoftGym*: (3) **SpreadCloth**: spreading crumpled cloth to be flat, (4) **RopeConfiguration**: manipulating a rope from a random pose to a target pose (we use the shape ‘S’ as the target). Among them, the first two are relatively easier. They can be accomplished without considering future actions and states, and we conduct them to show our dense affordance’s superiority over methods imitating expert demonstrations. The last two are much harder and would be better accomplished considering future actions and states to avoid local optima.

**Settings.** For all tasks, in both training and testing, we set different random seeds for each episode, producing unseen and diverse initial poses of objects. To compare the generalization ability between our proposed dense affordance and imitation-based methods, for cables in *DeformableRavens*, we directly test the model trained on cables with 32 beads over novel cable configurations with 24, 28 or 36 beads.

**Metrics.** For cable-ring and cable-ring-notarget, we follow *DeformableRavens* and use the manipulation successful rate as the metric. For SpreadCloth and RopeConfiguration, we follow *SoftGym* and use the normalized score as the metric. For all tasks, higher scores indicate better performance.

### 5.2. Baselines

For two cable-ring related tasks, we compare our method with baselines with or without expert demonstrations:

- **Transporter** [47, 38] is commonly used for robotic manipulation by learning visual correlation for picking and placing points. In *DeformableRavens* [38] it is trained by cloning expert demonstrations and achieves SOTA performance over relevant tasks.
- **GT-State** receives ground truth (GT) pose of the target object, and regresses  $p_{pick}$  and  $p_{place}$  with MLP.
- **GT-State 2-Step** first regresses  $p_{pick}$  and then  $p_{place}$  using  $p_{pick}$  and GT pose concatenation, both via MLP.

For SpreadCloth and RopeConfiguration, as object states and dynamics are too complex and the tasks are too difficult to hand-engineer expert policies, we compare our method with baselines focused on multi-step planning:

- **CURL-SAC** [23] that uses a model-free RL approach with contrastive unsupervised representations.
- **DrQ**[46] applies augmentation, regularization to RL.
- **PlaNet** [16] learns state space dynamics for planning.
- **MVP** [45] learns pick-and-place policy with model-free RL designed for deformable object manipulation.

### 5.3. Qualitative Results and Analysis

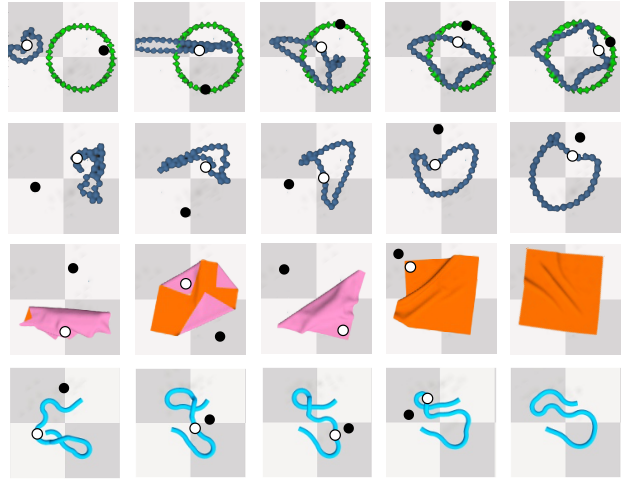


Figure 6. **Example action sequences** for cable-ring, cable-ring-notarget, SpreadCloth and RopeConfiguration. **White point denotes picking and black point denotes placing.**

Figure 6 shows examples of manipulation trajectories for diverse tasks using our proposed affordance. It is worth mentioning that, in the second state of SpreadCloth (Row 3), though it is intuitive to place the picking point (white) to the top-left position, the model places it to the bottom-right position (black), as the corresponding next state has **low coverage but high ‘value’**, requiring only one following pick-and-place action to almost fully unfold the cloth.

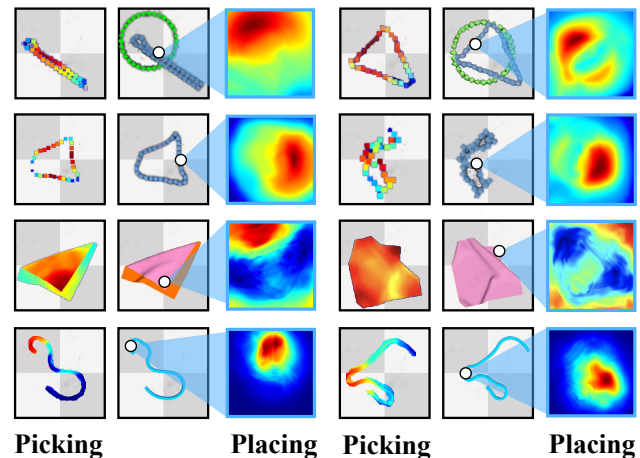


Figure 7. **Picking and placing affordance.** Each row contains two (picking affordance, observation with  $p_{pick}$ , placing affordance) tuples for a task.  $p_{pick}$  is selected by picking affordance. **Higher color temperature means higher affordance.**

Figure 7 visualizes picking and placing affordance, clearly showing that the learned affordance represents deformable objects with complex states and dynamics and facilitates selecting picking and placing points for manipulation. Figure 8 visualizes ‘value’s of states.

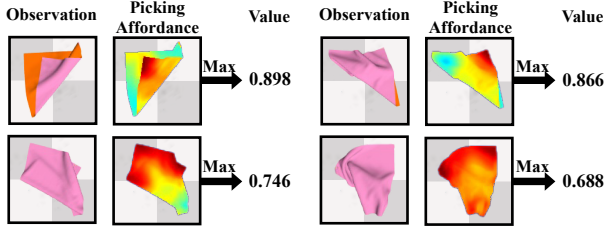


Figure 8. **Visualization of ‘value’** shows that some states with closer distances to the target (e.g., larger area) may not have higher ‘value’, as these states are hard for future actions to fulfill the task.

#### 5.4. Quantitative Results and Analysis

| Method          | cable-ring  | cable-ring-notarget |
|-----------------|-------------|---------------------|
| GT-State        | 0.0         | 5.0                 |
| GT-State 2-Step | 0.0         | 1.7                 |
| Transporter     | 68.3        | 70.0                |
| Ours            | <b>81.7</b> | <b>95.0</b>         |

Table 1. **Quantitative results in *DeformableRavens*.**

| Method   | SpreadCloth  | RopeConfiguration |
|----------|--------------|-------------------|
| CURL-SAC | 0.195        | 0.348             |
| PlaNet   | 0.387        | 0.236             |
| DrQ      | 0.275        | 0.154             |
| MVP      | 0.372        | 0.258             |
| Ours     | <b>0.758</b> | <b>0.529</b>      |

Table 2. **Quantitative results in *SoftGym*.**

Table 1 shows our framework outperforms all baselines in *DeformableRavens*. Specifically, for **GT State** and **GT State 2-Step**, GT states can only provide part of the necessary information, and it is difficult to acquire the precise GT states of deformable objects in the real world. We outperform **Transporter** that learns visual correlation for the matching between picking and placing points, even though it directly clones successful demonstrations from hand-crafted expert policies. Two possible reasons are that: 1) Dense affordance is more suitable for deformable objects as it represents the results of diverse actions on complex states, while visual correlation in **Transporter** is suitable for matching like assembling-kits; 2) training on expert demonstrations and cloning expert policies will limit the model’s generalization toward diverse situations during inference. To further evaluate the **generalization ability of the dense affordance**, we produce different novel object configurations, using 24, 28, and 36 as the bead number instead of the initial 32. Our method’s slighter decrease in performance over novel object configurations shown in Ta-

ble 3 also demonstrates its generalization ability. Besides, expert policies need to be elaborately hand-designed for different tasks, while our method can apply to any task without modifications.

Table 2 shows our framework outperforms all baselines in *SoftGym*. As described in 4.4, compared with those RL methods, our framework learns representations of complex states for multi-step manipulation in a stable way.

| Task           | cable-ring                | cable-ring-notarget       |
|----------------|---------------------------|---------------------------|
| configurations | 24 / 28 / 36              | 24 / 28 / 36              |
| Transporter    | 33.3 / 58.3 / 32.7        | 60.0 / 71.7 / 31.7        |
| Ours           | <b>61.6 / 86.7 / 58.3</b> | <b>81.7 / 96.7 / 78.3</b> |

Table 3. **Manipulation scores on novel configurations in *DeformableRavens*** showing our method’s generalization capability.

#### 5.5. Ablation Studies and Analysis

To demonstrate necessities of our framework’s different components, we conduct ablation experiments by comparing our method with: 1) **Ours RandPick**: our method with the picking policy replaced by a random policy; 2) **Ours ExpertPick**: our method with the picking policy replaced by Transporter’s expert; 3) **Ours w/o IST**: our method without Integrated Systematic Training (IST);

For SpreadCloth and RopeConfiguration that require strongly related sequential actions, we additionally compare 1) ablated versions using different stages of data, 2) **Ours only dist** directly and **greedily** trained on all collected data instead of stage-by-stage considering ‘value’s.

| Method          | cable-ring  | cable-ring-notarget |
|-----------------|-------------|---------------------|
| Ours RandPick   | 11.7        | 58.3                |
| Ours ExpertPick | 76.7        | 41.7                |
| Ours w/o IST    | 78.3        | 91.7                |
| Ours            | <b>81.7</b> | <b>95.0</b>         |

Table 4. **Ablation studies in *DeformableRavens*.**

| Method         | stage1       | stage2       | stage3       | stage4       | stage5       |
|----------------|--------------|--------------|--------------|--------------|--------------|
| Ours RandPick  | 0.241        | 0.211        | 0.304        | 0.185        | 0.190        |
| Ours w/o IST   | 0.526        | 0.586        | 0.621        | 0.624        | 0.612        |
| Ours only dist | <b>0.701</b> | <b>0.701</b> | 0.701        | 0.701        | 0.701        |
| Ours           | 0.589        | 0.695        | <b>0.752</b> | <b>0.754</b> | <b>0.758</b> |

Table 5. **Ablation studies in *SpreadCloth*.**

Table 4, 5 and 6 show quantitative results of ablation experiments. **Ours RandPick** and **Ours ExpertPick** show that, with the same placing affordance, our proposed picking affordance helps the framework perform the best.

| Method         | stage1       | stage2       | stage3       | stage4       | stage5       |
|----------------|--------------|--------------|--------------|--------------|--------------|
| Ours RandPick  | 0.329        | 0.302        | 0.332        | 0.334        | 0.322        |
| Ours w/o IST   | 0.359        | 0.418        | 0.437        | 0.479        | 0.474        |
| Ours only dist | <b>0.460</b> | 0.460        | 0.460        | 0.460        | 0.460        |
| Ours           | 0.441        | <b>0.503</b> | <b>0.518</b> | <b>0.527</b> | <b>0.529</b> |

Table 6. Ablation studies in *RopeConfiguration*.

**Ours only dist** in Table 5 and 6 show that, directly training on all the diverse data without estimating state ‘value’ limits the performance compared with our proposed framework. Besides, as shown in Figure 9, **Ours only dist** will propose actions leading to local optimal states, while our foresightful affordance will help to avoid that.

Table 5 and 6 show that, a series of steps of data empower affordance with generalization to diverse states.

**Ours w/o IST** in Table 4, 5 and 6, and adjusted affordance and ‘value’s in Figure 10 demonstrate IST helps integrating picking and placing modules into a whole, generating more precise perception of affordance and ‘value’s.

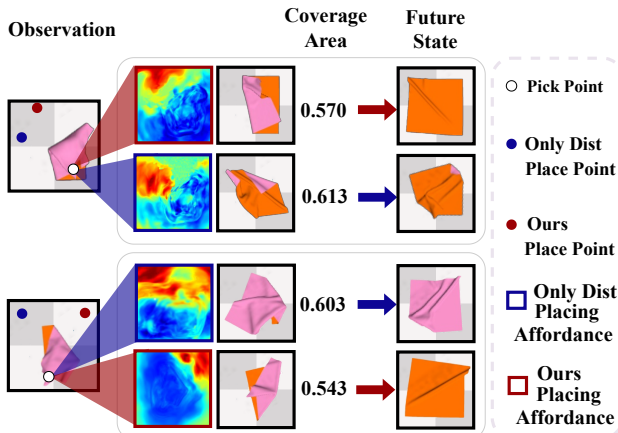


Figure 9. Placing affordance trained using ‘value’ supervision (red) and only using the greedy direct distance (blue).

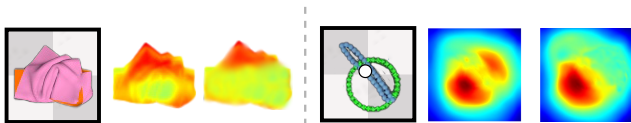


Figure 10. Picking and placing affordance before (middle) and after (right) IST of the observation (left). White: pick point.

## 5.6. Real-world Experiments

To bridge the sim2real gap and implement our method in the real world, similar to [45, 15], we use domain randomization to train affordance models in simulation and fine-tune them in real world. Specifically, we collect real-world data using *Fold to Unfold*, fine-tune trained-in-simulation  $M_{pick}$  and  $M_{place}$  stage by stage using the collected data.

For evaluations, we randomly lift and drop the objects for five times to get the initial state, and then run the models for ten pick-and-place actions to perform the tasks. The manipulation score is the average normalized score (computed the same as in *SoftGym*) of sixty trajectories.

Shown in Figure 11, given real-world observations with textures and physics different from objects in simulation, our method predicts reasonable picking and placing affordance and selects actions for tasks.

We compare our method with MVP [45], as it also uses pick-and-place as action primitive and thus could do both cloth and rope related tasks, and provides real world experiments. As shown in Table 7, our method outperforms MVP as explained in 5.4.

See the supplementary for more implementation details.

| Method | SpreadCloth  | RopeConfiguration |
|--------|--------------|-------------------|
| MVP    | 0.307        | 0.227             |
| Ours   | <b>0.683</b> | <b>0.461</b>      |

Table 7. Manipulation scores in the real world.

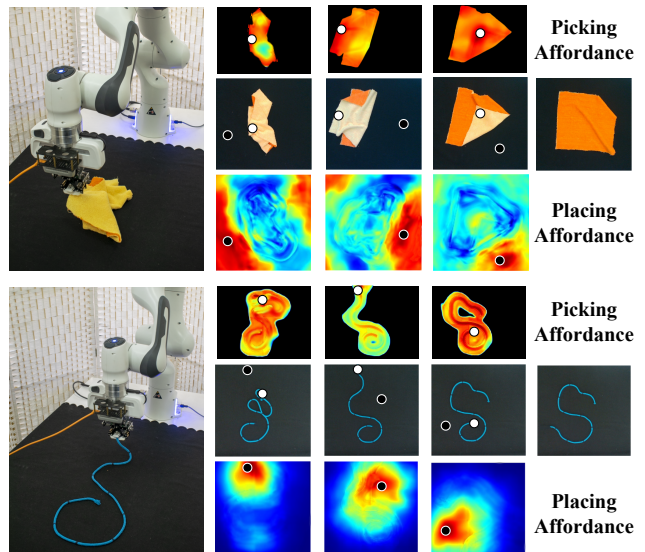


Figure 11. Examples of real-world manipulation trajectories guided by picking and placing affordance.

## 6. Conclusion

We propose to use dense visual affordance for manipulating deformable objects with complex states and dynamics. For tasks that require a series of strongly related actions, we further empower the proposed affordance with the awareness of a certain action’s influence on subsequent actions. We propose a self-supervised framework with novel designs to efficiently collect multi-stage interaction data and stably learn this representation. Experiments on representative tasks and in the real world show the superiority of our proposed dense affordance and the learning framework.



## References

- [1] Richard Bellman. Dynamic programming. *science*, 153(3731):34–37, 1966. 2, 4
- [2] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016. 11
- [3] Cristiano Cellini, Lukas Kaim, and Knut Drewing. Visual and haptic integration in the estimation of softness of deformable objects. *i-Perception*, 4(8):516–531, 2013. 2
- [4] Cheng Chi and Shuran Song. Garmentnets: Category-level pose estimation for garments via canonical space shape completion. In *The IEEE International Conference on Computer Vision (ICCV)*, 2021. 2
- [5] 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*, pages 5031–5041, 2020. 2
- [6] Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. 2016. 11
- [7] Shaowei Cui, Rui Wang, Junhang Wei, Fanrong Li, and Shuo Wang. Grasp state assessment of deformable objects using visual-tactile fusion perception. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 538–544. IEEE, 2020. 2
- [8] Ignacio Cuiral-Zuero and Gonzalo López-Nicolás. Rgb-d tracking and optimal perception of deformable objects. *IEEE Access*, 8:136884–136897, 2020. 2
- [9] Peter R Florence, Lucas Manuelli, and Russ Tedrake. Dense object nets: Learning dense visual object descriptors by and for robotic manipulation. *arXiv preprint arXiv:1806.08756*, 2018. 2
- [10] Samir Yitzhak Gadre, Kiana Ehsani, and Shuran Song. Act the part: Learning interaction strategies for articulated object part discovery. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 15752–15761, October 2021. 2
- [11] Aditya Ganapathi, Priya Sundareshan, Brijen Thananjeyan, Ashwin Balakrishna, Daniel Seita, Jennifer Grannen, Minh Hwang, Ryan Hoque, Joseph E Gonzalez, Nawid Jamali, et al. Learning dense visual correspondences in simulation to smooth and fold real fabrics. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11515–11522. IEEE, 2021. 2
- [12] Yiran Geng, Boshi An, Haoran Geng, Yuanpei Chen, Yaodong Yang, and Hao Dong. End-to-end affordance learning for robotic manipulation. *arXiv preprint arXiv:2209.12941*, 2022. 2
- [13] James J Gibson. The theory of affordances. *Hilldale, USA*, 1(2):67–82, 1977. 1, 2
- [14] Leni K Le Goff, Oussama Yaakoubi, Alexandre Coninx, and Stephane Doncieux. Building an affordances map with interactive perception. *arXiv preprint arXiv:1903.04413*, 2019. 2
- [15] Huy Ha and Shuran Song. Flingbot: The unreasonable effectiveness of dynamic manipulation for cloth unfolding. In *CoRL*, 2021. 2, 8
- [16] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pages 2555–2565. PMLR, 2019. 6
- [17] Rishabh Jangir, Guillem Alenya, and Carme Torras. Dynamic cloth manipulation with deep reinforcement learning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4630–4636. IEEE, 2020. 2
- [18] Biao Jia, Zhe Hu, Jia Pan, and Dinesh Manocha. Manipulating highly deformable materials using a visual feedback dictionary. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 239–246. IEEE, 2018. 2
- [19] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *The 3rd International Conference for Learning Representations*, 2015. 11
- [20] 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. 2
- [21] Mia Kokic, Johannes A. Stork, Joshua A. Haustein, and Danica Kragic. Affordance detection for task-specific grasping using deep learning. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pages 91–98, 2017. 2
- [22] Priyadarshini Kumari and Subhasis Chaudhuri. Haptic rendering of thin, deformable objects with spatially varying stiffness. In *International Conference on Human Haptic Sensing and Touch Enabled Computer Applications*, pages 483–492. Springer, 2016. 2
- [23] Michael Laskin, Aravind Srinivas, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. *Proceedings of the 37th International Conference on Machine Learning, Vienna, Austria, PMLR 119*, 2020. arXiv:2004.04136. 6
- [24] Jung Min Lee, Jongsoo Baek, and Da Young Ju. Anthropomorphic design: Emotional perception for deformable object. *Frontiers in psychology*, page 1829, 2018. 2
- [25] Robert Lee, Daniel Ward, Akansel Cosgun, Vibhavari Dasagi, Peter Corke, and Jurgen Leitner. Learning arbitrary-goal fabric folding with one hour of real robot experience. *arXiv preprint arXiv:2010.03209*, 2020. 2
- [26] Xingyu Lin, Yufei Wang, Jake Olkin, and David Held. Softgym: Benchmarking deep reinforcement learning for deformable object manipulation. In *CoRL*, 2020. 3
- [27] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. 5
- [28] Priyanka Mandikal and Kristen Grauman. Learning dexterous grasping with object-centric visual affordances. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2021. 2
- [29] Luz Martínez, Javier Ruiz-del Solar, Li Sun, J Paul Siebert, and Gerardo Aragon-Camarasa. Continuous perception for deformable objects understanding. *Robotics and Autonomous Systems*, 118:220–230, 2019. 2
- [30] 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. 2, 3, 5
- [31] Kaichun Mo, Yuzhe Qin, Fanbo Xiang, Hao Su, and Leonidas Guibas. O2O-Afford: Annotation-free large-scale object-object affordance learning. In *Conference on Robot Learning (CoRL)*, 2021. 2, 5
- [32] Luis Montesano and Manuel Lopes. Learning grasping affordances from local visual descriptors. In *2009 IEEE 8th international conference on development and learning*, pages 1–6. IEEE, 2009. 2
- [33] Francesc Moreno-Noguer. Deformation and illumination invariant feature point descriptor. In *CVPR 2011*, pages 1593–1600. IEEE, 2011. 2
- [34] Tushar Nagarajan and Kristen Grauman. Learning affordance landscapes for interaction exploration in 3d environments. In *NeurIPS*, 2020. 2
- [35] 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*, pages 163–172, 2020. 2
- [36] Erickson R Nascimento, Guilherme Potje, Renato Martins, Felipe Cadar, Mario FM Campos, and Ruzena Bajcsy. Geobit: A geodesic-based binary descriptor invariant to non-rigid deformations for rgb-d images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10004–10012, 2019. 2
- [37] 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*, number 3.2, page 5. Kobe, Japan, 2009. 12
- [38] Daniel Seita, Pete Florence, Jonathan Tompson, Erwin Coumans, Vikas Sindhwani, Ken Goldberg, and Andy Zeng. Learning to rearrange deformable cables, fabrics, and bags with goal-conditioned transporter networks. In *ICRA*, 2021. 2, 3, 5, 6
- [39] Bokui Shen, Zhenyu Jiang, Christopher Choy, Leonidas J. Guibas, Silvio Savarese, Anima Anandkumar, and Yuke Zhu. Acid: Action-conditional implicit visual dynamics for deformable object manipulation. *Robotics: Science and Systems (RSS)*, 2022. 2
- [40] Edgar Simo-Serra, Carme Torras, and Francesc Moreno-Noguer. Dali: deformation and light invariant descriptor. *International journal of computer vision*, 115(2):136–154, 2015. 2
- [41] 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. *European conference on computer vision (ECCV 2022)*, 2022. 2
- [42] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8:279–292, 1992. 2, 4
- [43] Thomas Weng, Sujay Man Bajracharya, Yufei Wang, Khush Agrawal, and David Held. Fabricflownet: Bimanual cloth manipulation with a flow-based policy. In *Conference on Robot Learning*, pages 192–202. PMLR, 2022. 2
- [44] 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. 2, 3, 5
- [45] Yilin Wu, Wilson Yan, Thanard Kurutach, Lerrel Pinto, and Pieter Abbeel. Learning to manipulate deformable objects without demonstrations. *Robotics: Science and Systems (RSS)*, 2020. 2, 3, 6, 8
- [46] Denis Yarats, Ilya Kostrikov, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International Conference on Learning Representations*, 2021. 6
- [47] Andy Zeng, Pete Florence, Jonathan Tompson, Stefan Welker, Jonathan Chien, Maria Attarian, Travis Armstrong, Ivan Krasin, Dan Duong, Vikas Sindhwani, et al. Transporter networks: Rearranging the visual world for robotic manipulation. In *Conference on Robot Learning*, pages 726–747. PMLR, 2021. 2, 5, 6
- [48] Andy Zeng, Shuran Song, Kuan-Ting Yu, Elliott Donlon, Francois R Hogan, Maria Bauza, Daolin Ma, Orion Taylor, Melody Liu, Eudald Romo, et al. Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 3750–3757. IEEE, 2018. 2
- [49] Yan Zhao, Ruihai Wu, Zhehuan Chen, Yourong Zhang, Qingnan Fan, Kaichun Mo, and Hao Dong. Dualafford: Learning collaborative visual affordance for dual-gripper object manipulation. *International Conference on Learning Representations (ICLR)*, 2023. 2, 3, 5

## A. Additional Details of Tasks, Settings and Metrics

### A.1. Tasks

We select 2 representative tasks from *Deformable Ravens* benchmark: **cable-ring** and **cable-ring-notarget**, as well as 2 harder tasks from *SoftGym*: **SpreadCloth** and **RopeConfiguration** (we use the shape ‘S’ as the target).

- (1) For **cable-ring**, it has a ring-shaped cable with 32 beads. The goal of the robot is to manipulate the cable towards a target zone denoted by a green circular ring in the observation. The maximum convex hull area of the cable-ring and the target cable-ring are the same.
- (2) For **cable-ring-notarget**, the setting is the same as **cable-ring**, except that there is no visible target zone in the observation, so that the goal is to manipulate the cable to a circular ring anywhere on the workbench.
- (3) For **SpreadCloth**, we use a square cloth. The cloth is randomly perturbed to a crumpled state. The goal of the robot is to manipulate the crumpled cloth into flat state.

- (4) For **RopeConfiguration**, the rope is randomly perturbed to a crumpled state. The goal of the robot is to manipulate the rope into the shape of letter ‘S’.

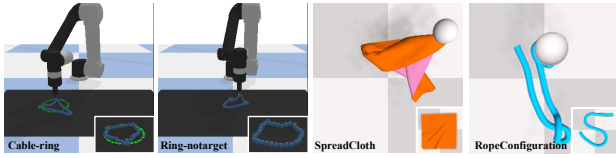


Figure 12. **Demonstrations of the selected tasks.** Each image shows an observation and a successful state (the cropped sub-images to the bottom right).

## A.2. Simulation

For *DeformableRavens* benchmark, we use the suite of simulated manipulation tasks using PyBullet [6] physics engine and OpenAI GYM [2] interfaces. For *SoftGym* benchmark, we use Nvidia FleX physics simulator and the Python interface for a better simulation of cloth and rope.

## A.3. Metrics

For **cable-ring** and **cable-ring-notarget**, we follow *DeformableRavens* benchmark, when the convex hull area of the ring beads exceeds a thresh  $\beta$ , the manipulation is judged as a success. We set  $\beta$  to be 0.75 for these two tasks, as established in *DeformableRavens*. We use the success rate as manipulation score, which is number of successful manipulation trajectories divided by total number of manipulation trajectories.

For the **SpreadCloth** and **RopeConfiguration**, following *SoftGym* benchmark, we choose the normalized score as manipulation score, which is  $\frac{metric_{final} - metric_{initial}}{metric_{goal} - metric_{initial}}$ , where  $metric_{final}$  means the score of final state,  $metric_{initial}$  means the score of initial state and  $metric_{goal}$  means the score of target state. Specifically, for **SpreadCloth**, we use the coverage area of cloth as the measurement, and  $metric_{goal}$  is 1.00. For **RopeConfiguration**, we use the negative bipartite graph matching distance as the metric, and  $metric_{goal}$  is  $-0.04$ .

During **testing**, we randomly select 60 (the same number of trials as in *DeformableRavens*) random seeds representing different initial configurations of the objects, conduct experiments and report manipulation score on these different initial states.

## B. Additional Details of Experiments

### B.1. Data Collection

We collect 5,000 interactions in cable-related tasks, and 40000 interactions in **SpreadCloth** and **RopeConfiguration** for each step. The training of the **SpreadCloth** and

**RopeConfiguration** needs more data, for the reason that, the states, kinematics and dynamics of these objects are much more complex.

From a starting state, we collect both successful interaction data using the proposed *Fold to Unfold* data collection method, and failure interaction data using a random policy. Therefore, the trained dense affordance could represent the distribution of diverse results of diverse actions. Each interaction data contains the actions (picking point and placing point) and results after the action (e.g. cloth coverage area for **SpreadCloth**).

### B.2. Hyper-parameters

We set batch size to be 20, and use Adam Optimizer [19] with 0.0001 as the initial learning rate. During **Integrated Systematic Training (IST)** procedure, we set learning rate to be 0.00005, as the affordance modules have been trained before, and are only adapted and integrated into a system in this procedure.

### B.3. Computing Resources

We use TensorFlow as our Deep Learning framework. Each experiment is conducted on an RTX 3090 GPU, and consumes about 20 GB GPU Memory for training. It takes about 12 hours and 6 hours to respectively train the Placing Module and the Picking Module for one step. Besides, the Integrated Systematic Training procedure consumes 6 hours.

## C. Additional Details of Network Architectures

The Picking Module and the Placing Module both employ Fully Convolutional Networks (FCNs) with the same structure to extract point-level features. Through the FCNs, the feature of the  $W \times H \times C$  dimension input sequentially transforms to  $W \times H \times 64$ ,  $W \times H \times 64$ ,  $W/2 \times H/2 \times 128$ ,  $W/4 \times H/4 \times 256$ ,  $W/8 \times H/8 \times 512$ ,  $W/16 \times H/16 \times 512$  (bottleneck of the net work, where global feature is extracted),  $W/8 \times H/8 \times 512$ ,  $W/8 \times H/8 \times 256$ ,  $W/4 \times H/4 \times 256$ ,  $W/4 \times H/4 \times 128$ ,  $W/2 \times H/2 \times 128$ ,  $W/2 \times H/2 \times 256$ ,  $W \times H \times 256$ ,  $W \times H \times 256$ .

Afterwards, the Picking Module uses MLPs with hidden sizes to be (256→256, 256→1) to predict picking affordance, and the Placing Module uses MLPs with hidden sizes to be (1024→256, 256→1) to predict placing affordance. Here, 256 denotes the feature dimension of each (picking or placing) point, and 1024 denotes the dimension of the concatenation of the picking point feature (256), the placing point feature (256), and the global feature (512).

## D. Additional Details of Real-world Experiments

### D.1. Real-robot Settings

For real-robot experiments, we set up one Franka Panda robot on the workbench, with a RealSense camera mounted on the robot gripper to take observations. We use Robot Operating System (ROS) [37] to control the robot to execute actions.

Additionally, as shown in Figure 13, as the original fingers of Franka Panda is wide and coarse, to ensure that the gripper can pick only one layer of cloth instead of two layers at a time, we design two fine-grained fingers mounted on the fingertips of the original fingers.

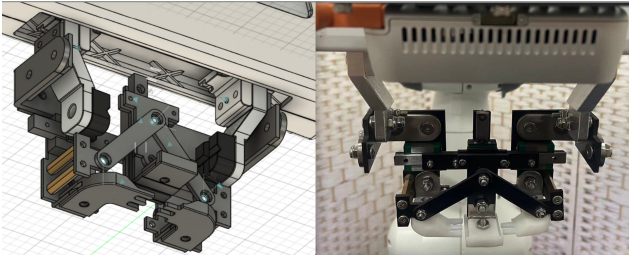


Figure 13. **Our Designed Fine-grained Fingers** to better pick deformable objects.

### D.2. Real-world Data Collection and Fine-tuning

As the configurations, kinematics and dynamics of deformable objects (like cloth and ropes) in the real world are different from those in simulation, we fine-tune the trained-in-simulation affordance using real-world collected interactions.

Specifically, following Section 4.5 (**Fold to Unfold: Efficient Multi-stage Data Collection for Learning Foresightful Affordance**) in the main paper, we collect real-world interactions using the *Fold-to-Unfold* method in different stages (demonstrations shown in Figure 14). For fine-tuning, we tune the learned picking and placing affordance stage-by-stage using the above real-world collected data.

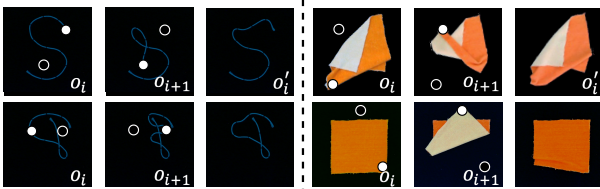


Figure 14. **Demonstrations of real-world collected data.** State  $o_{i+1}$  shows the starting state and state  $o'_i$  show the ending state of interaction data for training.

### D.3. Metrics

For **SpreadCloth**, we use the same metric as in A.3, as it is easy to compute the coverage area of the cloth in the real world. For **RopeConfiguration**, we mark a black dot on the rope every 10cm, and compute the distances between these black dots and their ideal locations as the bipartite graph matching distance for further evaluation.

### D.4. Video Records of Real-world Manipulations

Please see the **the videos in our project page** for video records of real-world manipulations for both **SpreadCloth** and **RopeConfiguration** tasks.

## E. Assets

We use the cable assets in *DeformableRavens* benchmark as well as cloth and rope assets in *SoftGym* benchmark, following their licenses. Our proposed assets with novel configurations can be generated using our code.