https://doi.org/10.26599/AIR.2024.9150032 CAAI Artificial Intelligence Research

Object-to-Manipulation Graph for Affordance Navigation

Xinhang Song[1,2], Bohan Wang[1,2], Liye Dong[1,2], Gongwei Chen[1,2], Xinyun Hu[1,2], and Shuqiang Jiang[1,2] ✉

ABSTRACT

Object navigation, whose goal is to let the agent to reach some places (or objects), has been a popular topic in embodied Artificial Intelligence (AI) researches. However, in our real-world applications, it is more practical to find the targets with particular goals, raising the new requirements of finding the places to achieve the particular functions. In this paper, we define a new task of affordance navigation, whose goal is to find possible places to accomplish the required functions, achieving some particular effects. We first introduce a new dataset for affordance navigation, collected by the proposed affordance algorithm. In order to avoid the high cost of labor, the groundtruth of each episode which is annotated with the interaction data provided by the AI2-THOR simulator. In addition, we also propose an affordance navigation framework, where an Object-to-Manipulation Graph (OMG) is constructed and optimized to emphasize the corresponding nodes (including object nodes and manipulation nodes). Finally, a navigation policy is implemented (trained by reinforcement learning) to guide the navigation to the target places. Experimental results on AI2-THOR simulator illustrate the effectiveness of the proposed approach, which achieves significant gains of 14.0% and 11.7% (on success rate and Success weighted by Path Length (SPL), respectively) over the baseline model.

KEYWORDS

navigation; affordance; manipulation; graph neural network

Navigation in three-dimensional environments is an essential capability of mobile intelligent systems that function properly in the physical world. In current works, intelligent agents are trained to perform navigation to target objects or positions with reinforcement learning model. However, instead of reaching a specific position or getting near an object, the essential goal of navigation is usually accomplishing particular jobs such as "find something to heat the bread". In most cases, finding the proper location to accomplish a comprehensive task such as "making the bread hot" may be more important to finding the particular tools for the job, such as oven, toaster, or cooker. Such requirements lead to the task of affordance navigation, whose goal is to find environments capable of facilitating the completion of specified tasks. Affordance is first introduced to Human-Computer Interaction (HCI) area in Ref. [1] as follows: "Affordance refers to the perceived and actual properties of the thing, primarily those fundamental properties that determine just how the thing could possibly be used." Different from object-goal[2–5] or placegoal[6–9] navigation works, in affordance navigation, we offer a description of a task instead of direct descriptions of targets. Affordance navigation is a more challenging task due to the abstractness and diversity of target descriptions.

Affordance researches[10–14] have been first investigated with images or videos. Recently, several works[15–17] have investigated to recognize or detect affordances in the scenario of "Internet AI", with images or videos that are mainly collected from internet. Ego-Topo[15] infers the affordances of kitchen environments from the egocentric videos. Action Maps[16] estimates manipulation labels for grid cells. Ade-Affordance[17] predicts the possess manipulation and the corresponding consequence for images in pixel-level. Although the data of those works can be collected from internet, large labor cost is also required for label annotation. Moreover, since affordance of different abilities can be accomplished with various objects (in various positions), analyzing affordance on those pre-captured images or videos may limit the abundance of affordance, which seems not practical to the real-world scenario.

In alternative to "Internet AI", a more recent work explore the affordance landscapes in the 3D environments[18]. It mainly focuses on the potential ability of the detected objects (through the manipulation), such as openable, sliceable, and pickupable. These abilities are the useful references to address the requirements of the users, however, there is still gap between such abilities and the essential goal of the users. For instance, while oven is openable and able to turn on, these abilities alone may not be sufficient for the agent to complete the job of "heat the bread" with oven. Focusing on the users' primary objectives, we explore the affordance navigation task by considering both direct manipulation and its subsequent effects. In particular, our affordance is oriented to a "target object" (which is given as the input), and the goal is to manipulate this target with the "tools" to achieve the desired effects. Thus, there are two challenges coming up: (1) how to define all possible affordances (also can be used as annotation for training episodes); (2) how to get close to the places of possible "tools" (may be an object or a place with a couple of objects) for manipulation.

To address the first problem, we propose a novel affordance mining algorithm to collect all possible affordances by manipulating the target with the "tools", and effects are judged by the attribute change of the target. For instance, after manipulating bread with oven, if the temperature of the bread gets higher, indicating the oven can heat the bread. An intuitive way to

1 Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS), Institute of Computing Technology, CAS, Beijing 100190, China
2 University of Chinese Academy of Sciences, Beijing 100049, China
Address correspondence to Shuqiang Jiang, sqjiang@ict.ac.cn

address the second problem is the visual navigation to the places of the "tools". Compared to the previous object navigation task, the main differences are in the following two aspects: (1) the "tools" may rely on multiple types of objects or the places with a combination of objects, which results in complex visual contents in the "tools" area; (2) each type of affordance involves many types of objects (for both targets and tools), leading to large diversity.

In this paper, we propose an Object-to-Manipulation Graph (OMG) framework affordance navigation, which exploits various types of relations between affordance and objects (including target and tools). Specifically, the OMG consists of three heterogeneous nodes: the manipulation node (e.g., heat), target object node (e.g., bread), and the tool node (e.g., toaster, microwave, etc.). The edges (relations) between nodes are constructed based on dataset or interaction ability of simulation environment, and explicitly reflect inherent characteristics of affordances (involving activities and objects) to solve the challenges of complex visual and semantic associations. Moreover, we introduce a benchmark with multiple types of affordance based on AI2-THOR[19] simulator, where all groundtruth of successful affordance places is annotated with automatic interaction with the environment, avoiding labor cost for annotation. Experimental results on the benchmark demonstrates the effectiveness of the proposed approach in affordance navigation. Remarkably, the proposed method achieves significant gains of 14.0% and 11.7% (on success rate and Success weighted by Path Length (SPL), respectively) over the baseline model.

1 Related Work

1.1 Visual navigation in 3D environments

Vision navigation in 3D simulators[8, 20–22] requires the agent to move intelligently in a static environment to reach a goal[3, 23], which can be divided into Point-goal and Object-goal. Point-goal navigation[23] refers to the problem where an agent starts from a randomly chosen pose and learns to navigate to a specific target point, usually specified in terms of 2D/3D coordinates relative to the agent. Object-goal navigation[3, 23] refers to the problem where the agent instead learns to navigate to a specified target object while successfully avoiding obstacles.

Although the proposed affordance navigation task also requires an agent to navigate to a target place, it aims to promoting the agent's understanding of possible activities in a dynamic environment. The possible activities focus on the organic combination of various objects instead of a specified object, which differs from existing object navigation tasks.

1.2 Visual affordance

Visual affordances has been studied extensively in the computer vision literature: inferring how people might use a space[11, 16] or tool[24], predicting where to grasp an object from images and video[10–14], and likely human body poses that would occur[25–28]. Although prior work explores affordances in various forms, learning affordance in a 3D dynamic environment is less studied. More closely related to our work, INTEXP[29] learns to leverages affordance landscape for interaction exploration, where the affordance only reflect the property of single object. In contrast, the affordances considered in our work are not strongly tied to the properties of a single object. Instead, we introduce an object-to-manipulation graph to learn the affordances associated with combinations of objects. This approach seeks to emulate the advanced human ability of interpreting the environment for an embodied agent.

1.3 Indoor navigation

As for indoor navigation, popular solutions include wireless navigation and visual navigation. However, while offering many advantages for indoor navigation, wireless navigation has several potential drawbacks and limitations which make it unsuitable for the task of affordance navigation. Wireless navigation, such as WiFi routers[30] or Bluetooth beacons[31], usually do not have the ability to identify multiple types of objects in the room. In contrast, with the implementation of Convolutional Neural Networks (CNN), visual navigation can classify objects easily. Moreover, most wireless navigation methods rely on the deployment of specific devices or beacons, while visual navigation mainly requires the sensors within the robots, not requiring external signal emission source. Thus, particularly for affordance navigation, we investigate our research based on visual navigation.

2 Affordance Navigation

2.1 Task definition

As shown in Fig. 1, our task requires an agent to navigate to the target places which provide the specified affordance (see samples in Fig. 2) in unseen environments when given a brief description of the task. For example, if the agent receive affordance "heat bread", it should navigate to places where the functions that allow the agent to make the bread hot (resulting in an increase in temperature) are provided.

At the beginning of each episode, the agent is given a target affordance (with a target object and manipulation) and spawned at a random location in the environment, the agent need to make a sequence of navigation actions to reach the target place. During navigation, RGB images in an egocentric view are the only available visual information for an agent. In all the environments, the action space of an agent consists of navigation actions $\mathscr{A}$ ={MoveAhead, RotateLeft, RotateRight, LookUp, LookDown, Done}. Note that the agent in our task is required to navigate to the places, but is not required to complete any manipulation tasks (such as putting the apple).

2.2 Evaluation metrics

We evaluate the models based on two metrics: Success Rate (SR) and the SPL metric proposed by Ref. [23], which are adopted by other target-driven visual navigation tasks[8, 32]. One successful

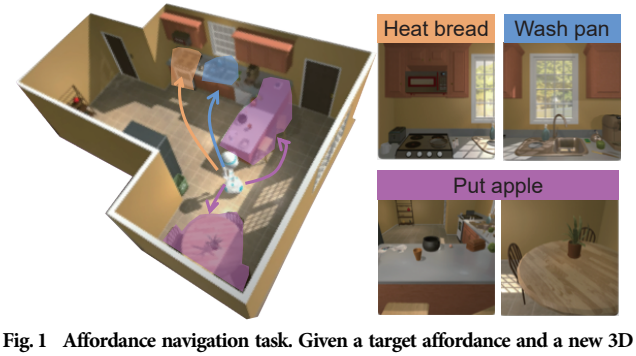


Fig. 1 Affordance navigation task. Given a target affordance and a new 3D environment, the agent is required to navigate to the places which provide the target affordance. This task aims at promoting the agent's understanding of possible interactions in a dynamic environment. The resulting navigation policy prepares the agent for downstream tasks that involve multiple object interactions.

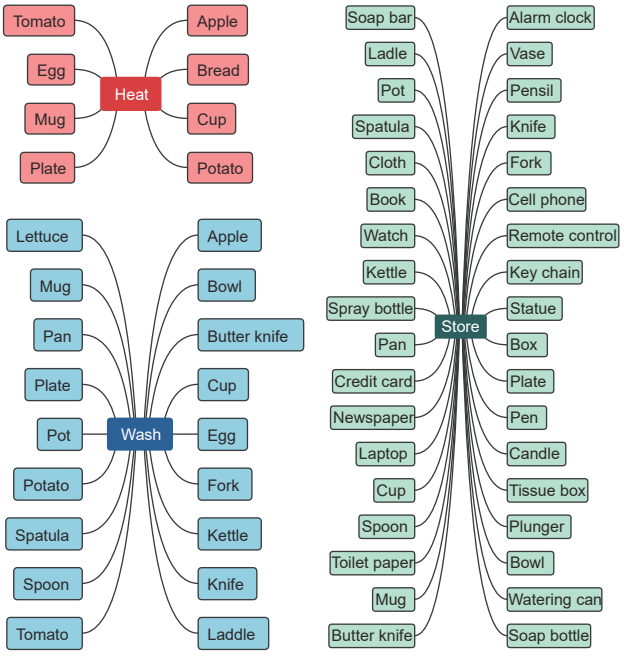


Fig. 2 Affordances examples from our dataset. Here we treat affordances as (manipulation, target objects) pairs.

episode is defined as: an agent operates the termination action "Done" when the center of agent mass is within the target places of the target affordance. If a termination action is executed at any other time, the agent fails and the episode ends.

SR is defined as $\frac{1}{N}\sum_{i=1}^{N} S_i$, where N is the total number of episodes, and S_i is a binary indicator of success in episode i.

SPL is defined as $\frac{1}{N}\sum_{i=1}^{N} S_i \frac{L_i}{\max(P_i, L_i)}$, where P_i represents path length, and L_i is the optimal trajectory length to any target affordance in a given environment from initial state (provided by the environment for evaluation) in episode i. It is important to reiterate that one affordance might has more than one target places in test environments (e.g., the places near the empty "CounterTop" or "DinningTable" all afford "put apple"). Thus, we carefully re-formulate the shortest path length $L_i = \min_{c=1\to M}(L_i^c)$, where L_i^c denotes the shortest path to the c-th target place.

2.3 Data collection

Affordances. Our goal is to collect high-level households affordances that may be assigned to a home robot in future. Thus, we choose AI2-THOR (The House of inteRactions)[19] environment as our platform for affordance navigation tasks. AI2-THOR is a challenging simulation environment, containing 120 FloorPlans categorized into four different room layouts: Kitchen, Living room, Bedroom, and Bathroom. Each room is unique in terms of furniture placement and item types, which supports various affordances. We collect $A = 200$ affordances across 5 different types (Heat, Cool, Wash, Put, and Store). Figure 2 gives the examples of affordances in our dataset.

Target places. Contrasting point/object navigation task[3, 32, 33] where the target places can directly come from the distance between the target point/object, affordance relyies on the object state changes according to target affordance. For example, the target places of "heat bread" should afford the agent to rise the bread temperature. Thus, we introduce Algorithm 1 to obtain the target places of affordance in the environment. First given target affordance a, we define specific target object goal state S_{goal}, e.g., {object: bread; temperature: hot} for "heat bread". Then we traverse all the reachable places to collect the target places. For one place, the decision process encodes all the visible objects, as well as high-level environment dynamics at current place, into PDDL[34] planer. The place is successful if the PDDL planer could generate a reasonable plan to change the target object state to S_{goal}. Here the plan is a sequence of interactions specifically. For example, given affordance "heat bread", one reasonable plan near the toaster is {put bread in toaster, toggle-on toaster}.

Algorithm 1 Affordance episode collection

Input: Target affordance $d = (\text{manipulation}, \text{object})$
Input: Object initial state S_{init}
Input: Planning Domain Definition Language (PDDL) planer
Define target object state S_{goal} for d
Create targe places set $G = \{\}$
Get all reachable location set P
for p in P **do**
  Extract visible object set O in p
  Plans $L = \text{PDDL}(O, S_{\text{init}}, S_{\text{goal}})$
  for l in L **do**
    if Execute l successfully **then**
      Add p to G; **break**
    end if
  end for
end for
Return: Places G of target affordance

3 Affordance Navigation Framework

In this paper, we propose an effective method with reinforcement learning. Specifically, our model includes two parts: an object-to-manipulation graph that incorporates the relations between affordances and objects to learn an informative visual representation from RGB observations, and a policy network that decides the navigation action to take for the next step.

3.1 Object-to-manipulation graph

We explicitly build the relations between manipulations and objects based on the dataset. Specifically, we first treat affordances as 〈manipulation, target objects〉 pairs, and combine tool objects to obtain three kinds of concepts. Considering the three kinds of semantic concepts and the relations between them, it is intuitive to construct a graph to explicitly capture and encode the relation with their representations. To this end, we propose an OMG with heterogeneous relations to produce effective embedding for affordance navigation.

In the following parts, we first present the graph architecture that integrates the relations between affordances and objects (Section 3.1). Then we introduce how to obtain the representation of context node from RGB images (Section 3.1). Finally, we delve into the details of how we incorporate Graph Convolution Networks (GCNs) for the task of affordance navigation and how it helps the generalization to unseen scenes (Section 3.1).

(1) Creating the object-to-manipulation graph

We denote our graph by $G = (V, E)$, where V and E denote the nodes and the edges between nodes, respectively. As shown in Fig. 2, the affordance consists of (manipulation, object) pair. For

clarification, we denote the object in affordance as target object. Meanwhile, other objects, which provide the affordance, are denoted as tool object. For example, the tool objects of "heat bread" are {Toaster, Microwave}. Specifically, the node $V = \{V_M, V_F, V_T\}$ means manipulation, tool object, and target object nodes, respectively. There are two types of edges, (manipulation, tool object) and (target object, tool object) in the graph.

In the following, we introduce two kinds of relations which are used to form the mentioned edges:

● Co-occurrence relation between manipulation and tool object. The relation can connect the manipulation with different tool objects, which facilitates the agent learning of the situation where an affordance needs more than one tool objects. Specifically, based on the constructed affordance dataset, target place of each affordance will consist of several objects. We collect these objects and count the number of times that the tool object and manipulation co-occur. Finally, we normalize this co-occurrence matrix according to the total number of occurrences of activities.

● Interactive relation between target object and tool object. Although co-occurrence relation can help to identify affordance in a way, there are still some uncertainties when classifying the fine-grained affordances with same manipulation. To this end, we exploit the realistic operability of affordance with the help of interaction ability of simulation environment. Specifically, if the target object (e.g., "bread") can successfully interact with a tool object (e.g., put in "fridge"), we consider that these two objects with interaction relation and the edge between them will be set to value 1; otherwise, 0.

In alternative to previous works that introduce external knowledge[32], our agent infers the relations through interacting with the environment. Figure 3 illustrates some examples of relationships.

(2) Context node representation

Our goal is to incorporate affordance graph in the context of current environment when planning the path. Thus, it is important to encode the current environment state into the node representations. As indicated in Fig. 4, our graph has three types

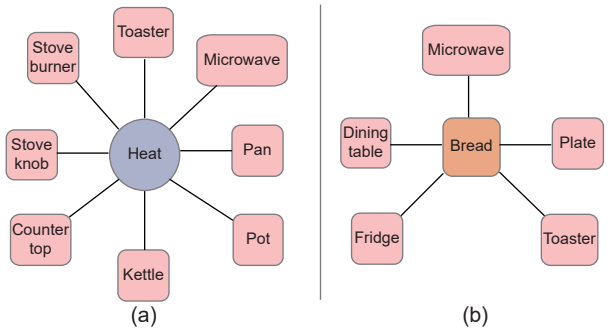


Fig. 3 Illustration of relationships. (a) Co-occurrence relations between the manipulation and tool objects. (b) Interactive relation between target object and tool objects.

of nodes: manipulation, tool object, and target object nodes, respectively. In each step, all nodes are initialized with current observation.

Manipulation node. Our core idea is that the manipulation node should help the agent decide which regions to visit (e.g., activities provided by the area around the wall is few, so better to avoid them) and help extrapolate the termination state in unseen environments (e.g., wash afforded by the area around {sink, faucet} suggests that this area supports other wash-related affordance), leading to more efficient navigation policy. Thus, we implement a ResNet-18[35] as an manipulation classification model to transform an input RGB image into a N_d-dim logits P, where each dim indicates that the probability of the corresponding manipulation is likely to conduct. Training samples for this model comes from the observation in agent navigation process. For example, if it successfully move to the target place which affords "heat bread", the current view (an RGB image) of agent is labeled as heat, and these labels are propagated to all frames near the successful place, such affordance will also be recognizable even from a little further away. The initialization node feature vector X_{A_i} is the concatenation of the manipulation visual vector and semantic embedding. The visual vector $A_i = \hat{A} \cdot P_i$, where $\hat{A}$ is the input feature of the last classification layer of manipulation

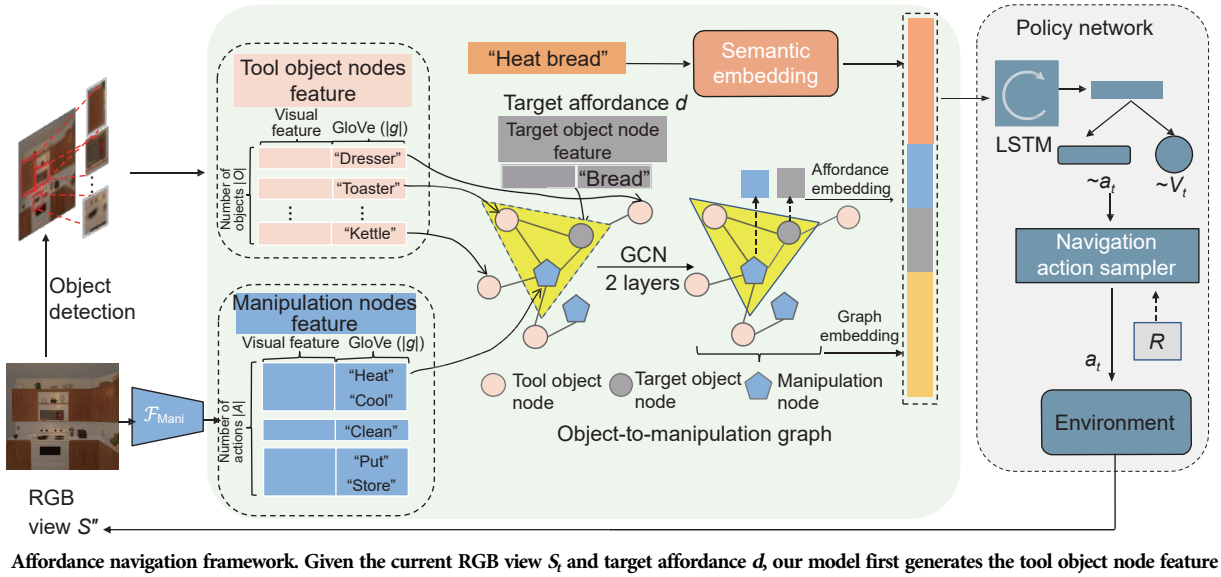


Fig. 4 Affordance navigation framework. Given the current RGB view S_t and target affordance d, our model first generates the tool object node feature from object detection and manipulation node feature by F_{Mani}. Then the node feature is updated by the object-to-manipulation graph, which encodes the relationships between affordance and objects. The graph consists of three nodes: manipulation node, target object node, and tool node. Finally, the policy network takes the semantic embedding of d, the affordance embedding of d, and the graph embedding as input to train a policy π to maximize the reward in Eq. (3).

network, and P_i is the probability of the corresponding manipulation category and used to re-weight the common feature $\hat{A}$. The semantic vector is the pre-trained GloVe word embedding[36].

Tool object node. In order to find all tool objects in current frame, we train an object detector, i.e., Faster R-CNN[37] based on the pre-defined tool object categories. Given an input image, we first perform object detection to localize all the objects of interest. If there are multiple instances of an object class, we choose the one with the highest confidence score. We record the bounding box positions and object appearance feature and then concatenate them as the visual vector. When the tool object is not detected, its visual vector will be set as a vector with all zeros. Finally, the initialization feature X_F of tool object node is the concatenation of the visual vector and semantic embedding (GloVe word embedding) of the corresponding object.

Target object node. We do not detect the target objects when using trained object detector, which means that the visual vectors of these nodes are unavailable. The main reason about missing target objects is that we want to eliminate the possible inappropriate bias caused by the connection between occurrence of target objects and target affordance. For example, in dataset, the area with bread may often provide the affordance "heat bread", which is inappropriate connection for learning because this area may correspond to the affordance "put bread" actually. To this end, we set the visual vector of each target object to a vector with all zeros. The initialization node feature X_M is also the concatenation of the visual vector and the corresponding semantic embedding.

(3) Optimizing with graph convolution network

As mentioned above, we initialize the nodes with different features and also use two kinds of relations to form the edges in graph. Here, we try to model the relation between these nodes and embed them into the node features by using Graph Convolution Network (GCN)[38]. GCN works in a similar way to convolution and operate on the neighboring nodes defined by the adjacency matrix. Given a graph $G=(V,E)$, the initialization node features are $X=[x_1,x_2,\ldots,x_{|V|}]$. Based on edges (manipulation A, tool object F) and (tool object F, target object M), we construct a binary adjacency matrix B. Following the definition in Ref. [38], we have

$$H^{l+1}=f\left(\widehat{B}H^lW^l\right) \tag{1}$$

where $\widehat{B}$ is a normalized version of B, W^l is learnable parameter in GCN layer l, and $f(\cdot)$ means non-linear function Rectified Linear Unit (ReLU). We set $H^0=X$ and take H^L as the last node outputs.

As illustrated in Fig. 4, we use three layers of GCN, the first two layers output 1024 dimensional latent features and the output dimension of last layers is 512. Finally, we will generate two kinds of outputs.

First, we extract the output features manipulation and target nodes as the affordance embedding. For example, as illustrated in Fig. 4, we use the heat node and bread node features to generate embedding for affordance "heat bread". Second, we aggregate the last layer outputs of each node into a single value which results in a $|V|$ dimension graph embedding. This feature vector is basically an encoding of relations in the context of the current scene and environment.

In addition to graph learning, we construct semantic branch to embed the input phrase (target affordance) into a vector. This semantic branch takes as input GloVe word embedding and feeds it into a fully-connected layer to produce a 512-D vector. Finally, we concatenate the affordance embedding, graph embedding, and the features generated from the semantic branches. As illustrated in Fig. 4, the joint feature is further fed into the policy network for navigation action prediction and sampling.

3.2 Policy network

We model the affordance navigation agent with a deep reinforcement learning framework. Given a target affordance d, the agent receives a visual input s_t (i.e., the egocentric RGB image from the current location and orientation) at the time step t and infers an navigation action a_t from the set of possible actions $\mathscr{A}$ according to its policy π. We approximate the policy by a deep policy network $\pi(\cdot;\theta)$:

$$a_t\sim\pi(s_t,d;\theta) \tag{2}$$

where θ is the parameter for the network.

We employ the Asynchronous Advantage Actor-Critic (A3C)[39] model to predict the policy at each time step. The A3C model generates two outputs, i.e., the policy and the value. We sample the action from the predicted policy. We consider a reward to minimize the trajectory length to the target affordance area: If the termination action "Done" is sampled and the agent's center of mass is within the nearest specified location, the agent receives a large positive reward 5.0. If not the nearest, the reward will decrease to 4.0. Otherwise, we penalize each step with a small penalty -0.01.

$$R(s,a)=\begin{cases}5, & \text{if } c_s\in E^*;\\ 4, & \text{if } c_s\in E;\\ -0.01, & \text{otherwise}\end{cases} \tag{3}$$

where c_s denotes the agent's center of mass, E is the target places of the environment, and E^* means the nearest target places.

4 Experiment

4.1 Implementation detail

During training, we train our model with 12 asynchronous agents for 3 million episodes on the training FloorPlans of AI2-THOR environment. We use the Adam optimizer[40] to update parameters with a learning rate 10^{-4}. Faster R-CNN[37] is leveraged as detector in our model. We first train the detector on COCO, then finetune it on 3×10^4 images (collected from training FloorPlans of AI2-THOR). Our implementation of the A3C model consists of three layers: input, hidden layer, and outputs. The hidden layer is a fully connected layer followed by the ReLU activation layer which maps the fused input into a 512-D latent space. Then the $|\mathscr{A}|$ dimensional policy and the value are generated by two branches of network, as shown in Fig. 2.

For evaluation, we perform navigation for 1000 different episodes (5 for each affordance). We randomize the test environment (object positions and states) and states (start location and camera viewpoint) when sampling episodes. All models are evaluated using the same set. For each run, we select the model that performs the best on the validation set in terms of success.

4.2 Result

Since we propose a new task, affordance navigation, there is a lack of tailored methods. We consider some previous methods that are proposed in similar tasks, like object navigation, as comparisons, and make some appropriate changes to suit the proposed task.

● RANDOM samples an navigation action from the navigation action space randomly at each step.

● BASELINE closely resembles that of Ref. [41], as it comprises of the current observation (encoded with a frozen ResNet-18[35] CNN, where we take the output of the final convolution layer to preserve spatial information necessary for grounding specific affordance in the visual frame. We embed this output using two more 1×1 convolution layers and a fully-connected layer to generate a 512-D feature specifically.) and the target information (in the form of GloVe embedding of the target affordance).

● OBJNAVIGATION[32] uses the prior object relations in the form of a knowledge graph for object navigation. We augment this baseline by replacing target information and supervision reward.

● INTEXP[29] uses affordance landscape (learned by a segmentation network) to explicitly facilitate interaction exploration task. We also use the segmentation network to generate affordance landscape as complementary to the image embedding of BASELINE.

In general, RANDOM determines if a learnable policy is required at all, given small and easy to navigate environments. BASELINE and OBJNAVIGATION evaluate whether the intelligent affordance navigation policies fall out naturally from traditional navigation methods.

Table 1 indicates that our method outperforms the BASELINE by a large margin on both success rate (+14.0%) and SPL (+11.7%). BASELINE only resorts to the manipulation feature and global feature for navigation. The relations among manipulation and tool objects, and the association between the visual observations and affordance are not exploited. This comparison suggests that our method leads to informative visual representation for navigation and thus remarkably improves the effectiveness and efficiency of our navigation system. As indicated in Table 1, we observe that our method significantly outperforms OBJNAVIGATION[32] and INTEXP[29]. It can be seen from Fig. 5a that the testing SR of our method rapidly increases within the first 0.5 million episodes itself, before saturating. The manipulation-wise results (of Fig. 5a) are illustrated in Fig. 5b. This shows that our models learn to correctly find the places of target affordance much faster than others.

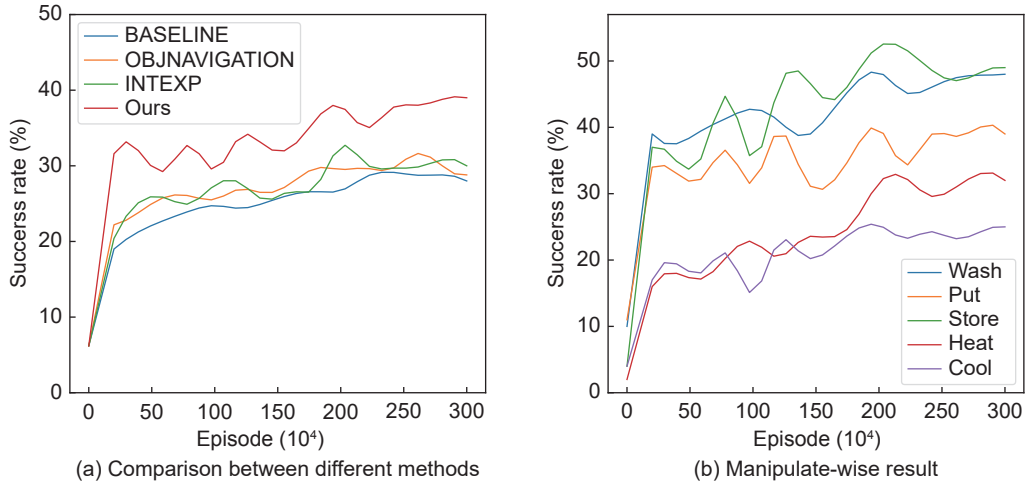
Those related works directly employ the relations between objects to generate visual representation, or introduce segmentation network to generate object-centric affordance landscape from pre-training. However, those methods may not shrink the gap between objects and affordance explicitly. In contrast, our method leverage the relations between affordance and objects with object-to-manipulation graph. Thus our method achieves expressive representations and expedites the policy learning for affordance navigation. As shown in Table 2, our method works better on challenging affordances which need more than one tool objects (e.g., heat), or changes the tool object due to the target object (e.g., store) specifically.

Meanwhile, we visualize the trajectories in testing environments in Fig. 6. The Baseline and INTEXP both issues "Done" in a location of stove, which failed to afford "heat bread" due to the missing of pan. Although the OBJNAVIGATION reaches the tool object (i.e., Microwave), it suffers from longer path compared to our method.

Table 1 Performance comparisons on affordance navigation. We report the average SR and SPL. $L \geqslant 5$ represents the episodes which require at least 5 steps.

(%)

Model	ALL		$L \geqslant 5$	
	SR	SPL	SR	SPL
RANDOM	6.2	1.9	0.3	0.1
BASELINE[41]	28.0	16.1	11.5	7.9
OBJNAVIGATION[32]	28.8	16.9	13.7	10.5
INTEXP[29]	29.8	18.0	12.6	8.2
Ours	**42.0**	**27.8**	**28.6**	**19.6**



(a) Comparison between different methods (b) Manipulate-wise result

Fig. 5 Success rate on unseen environments vs. training episodes. (a) Compared with current state-of-the-art methods, our agents significantly outperform on average success rate and convergence quickly. (b) In the analysis, we calculate the success rate of our agents per manipulation. Results are averaged from three training runs.

Table 2 Performance evaluation on manipulation-wise basis. We report the average SR and SPL. (%)

Model	Heat		Cool		Wash		Put		Store		Average	
	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL
RANDOM	2.0	0.83	4.0	1.0	10.0	3.2	11.0	3.6	4.0	1.2	6.2	1.9
BASELINE[41]	16.0	9.2	18.0	7.0	46.0	25.0	23.0	12.8	37.0	26.5	28.0	16.1
OBJNAVIGATION[32]	17.0	9.9	20.0	10.0	43.0	25.4	28.0	14.3	36.0	24.9	28.8	16.9
INTEXP[29]	22.0	13.5	20.0	9.0	41.0	26.3	36.0	20.2	30.0	20.8	29.8	18.0
Ours	**35.0**	**21.2**	**24.0**	**15.2**	**53.0**	**34.0**	**44.0**	**28.8**	**54.0**	**39.9**	**42.0**	**27.8**

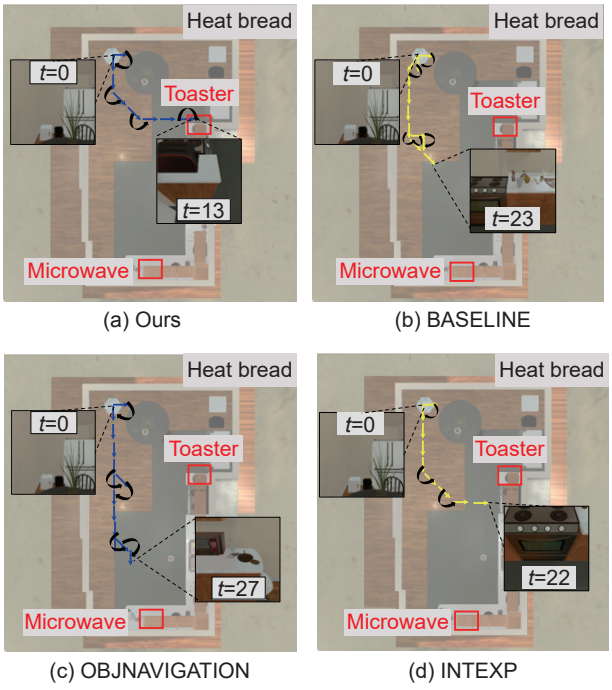


Fig. 6 Qualitative examples of different models in testing environments. The target affordance is "heat bread", and "tools" are highlighted by the red bounding boxes. Black arrows represent rotation. Blue and yellow arrows represent success and failure cases, respectively.

4.3 Ablation analysis

Several variants of the OMG are considered as follows:

- Ours (RGB) uses only the egocentric RGB frames to learn the policy, which is the same as BASELINE.
- Ours (AFFOR) uses RGB plus affordance embedding from the OMG.
- Ours (GRAPH) uses RGB plus graph embedding from the OMG.
- Ours (ALL) uses RGB plus affordance and graph embedding from the OMG.

All variants are rewarded for navigation to the target places (Eq. (3)) and use the same architecture (Section 3.2). As can be seen in Table 3, ours (AFFOR) embedding improves the performance of navigation systems compared with the BASELINE model, especially in SPL (+10.6%). Note that the considerable SPL improvements demonstrate that proposed affordance visual states explicitly is helpful. Meanwhile, ours (GRAPH) improves the navigation policy compared to the model only use RGB frames (+9.0%). This indicates that involving OMG improves the efficiency of navigation systems. Furthermore, when we feed both affordance and graph embedding to the policy network, the SR increases with (+14.0%). The improvements imply that these two features are complementary and also verify the effectiveness of our proposed framework.

4.4 Real-world experiment

To further verify the feasibility of the proposed method, experiments are conducted in real-world environment. A simplified indoor environment which mimicked the layout of a household is designed for the experiments. The indoor environment consists of three different rooms including dining room, kitchen, and living room. A varied selection of items are placed in each room in a sensible manner. The navigation task is split into two phases. First, the robot searches for an instance of the specified type of objects. Then in the second phase, the robot navigates to the target affordance. In the experiments, four types of affordances are tested, and each type of task is repeated 25 times with different objects. We consider it as a successful navigation when the robot identifies the object, and is within a proximity of 1.5 m from the object. The navigation is considered to fail when the robot collides with the wall, fails to reach the destination, or is unable to locate the object. We use LoCoBot[42] as the robot in our experiments.

As shown in Table 4, our model achieved an average success rate of 72.0% in Phase 1 and yielded a success rate of 37% in Phase 2.

5 Conclusion

In this paper, we introduce the task of affordance navigation, whose goal is to find the target places to finish the required manipulations, also may achieve the desired effects. To evaluate the models for affordance navigation task, we propose an

Table 3 Ablation results. $L \geqslant 5$ represents the episodes which require at least 5 steps. (%)

Model	ALL		$L \geqslant 5$	
	SR	SPL	SR	SPL
Ours (RGB)	28.0	16.1	11.5	7.9
Ours (GRAPH)	37.0	24.3	22.3	15.8
Ours (AFFOR)	38.8	26.7	25.3	18.3
Ours (ALL)	**42.0**	**27.8**	**28.6**	**19.6**

Table 4 Real-world results.

Task	Success rate (%)	
	Phase 1: Target navigation	Phase 2: Affordance navigation
Heat	68.0	36.0
Wash	72.0	24.0
Put	68.0	48.0
Store	80.0	40.0
Mean	72.0	37.0

affordance collecting algorithm, which can annotate the episodes automatically according to the feedback of simulators, avoiding the high cost labor. Since the affordance in the proposed task involves multiple types of components, we propose an OMG to exploit the contextual relations between them, whose graph representation is further fed to a navigation policy network for action prediction. Note that in this work, we mainly focus on the scenario where the target affordance (target object + manipulation) is given, limited to the affordance about target object, which is one limitation of our work. It is also worth researching on affordance of more open scenarios in future work.

Acknowledgment

This work was supported by the Beijing Natural Science Foundation (No. JQ22012), and in part by the National Natural Science Foundation of China (Nos. 62125207, 62032022, 62272443, and U23B2012).

Article History

Received: 29 May 2023; Revised: 26 December 2023; Accepted: 6 February 2024

References

[1] D. A. Norman, *The Psychology of Everyday Things*. New York, NY, USA: Basic Books, 1988.

[2] W. Yang, X. L. Wang, A. Farhadi, A. Gupta, and R. Mottaghi, Visual semantic navigation using scene priors, presented at 7th Int. Conf. Learning Representations (ICLR 2019), New Orleans, LA, USA, 2019.

[3] M. Wortsman, K. Ehsani, M. Rastegari, A. Farhadi, and R. Mottaghi, Learning to learn how to learn: Self-adaptive visual navigation using meta-learning, in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition* (*CVPR*), Long Beach, CA, USA, 2019, pp. 6743–6752.

[4] H. Du, X. Yu, and L. Zheng, Learning object relation graph and tentative policy for visual navigation, in *Proc. 16th European Conf. Computer Vision* (*ECCV*), Glasgow, UK, 2020, pp. 19–34.

[5] S. Zhang, X. Song, Y. Bai, W. Li, Y. Chu, and S. Jiang, Hierarchical object-to-zone graph for object navigation, arXiv preprint arXiv: 2109.02066, 2021.

[6] S. Gupta, J. Davidson, S. Levine, R. Sukthankar, and J. Malik, Cognitive mapping and planning for visual navigation, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition* (*CVPR*), Honolulu, HI, USA, 2017, pp. 7272–7281.

[7] D. S. Chaplot, D. Gandhi, S. Gupta, A. Gupta, and R. Salakhutdinov, Learning to explore using active neural SLAM, presented at 8th Int. Conf. Learning Representations (ICLR 2020), Addis Ababa, Ethiopia, 2020.

[8] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, et al., Habitat: A platform for embodied AI research, in *Proc. IEEE/CVF Int. Conf. Computer Vision* (*ICCV*), Seoul, Republic of Korea, 2019, pp. 9338–9346.

[9] E. Wijmans, A. Kadian, A. Morcos, S. Lee, I. Essa, D. Parikh, M. Savva, and D. Batra, DD-PPO: Learning near-perfect pointgoal navigators from 2.5 billion frames, presented at 8th Int. Conf. Learning Representations (ICLR 2020), Addis Ababa, Ethiopia, 2020.

[10] H. S. Koppula and A. Saxena, Anticipating human activities using object affordances for reactive robotic response, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 1, pp. 14–29, 2016.

[11] T. Nagarajan, C. Feichtenhofer, and K. Grauman, Grounded human-object interaction hotspots from video, arXiv preprint arXiv: 1812.04558, 2018.

[12] Y. Zhu, C. Jiang, Y. Zhao, D. Terzopoulos, and S. C. Zhu. Inferring forces and learning human utilities from videos, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition* (*CVPR*), Las Vegas, NV, USA, 2016, pp. 3823–3833.

[13] J. B. Alayrac, J. Sivic, I. Laptev, and S. Lacoste-Julien, Joint discovery of object states and manipulation actions, in *Proc. IEEE Int. Conf. Computer Vision* (*ICCV*), Venice, Italy, 2017, pp. 2146–2155.

[14] K. Fang, T. L. Wu, D. Yang, S. Savarese, and J. J. Lim, Demo2Vec: Reasoning object affordances from online videos, in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition* (*CVPR*), Salt Lake City, UT, USA, 2018, pp. 2139–2147.

[15] T. Nagarajan, Y. Li, C. Feichtenhofer, and K. Grauman, Ego-Topo: Environment affordances from egocentric video, in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition* (*CVPR*), Seattle, WA, USA, 2020, pp. 160–169.

[16] N. Rhinehart and K. M. Kitani, Learning action maps of large environments via first-person vision, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition* (*CVPR*), Las Vegas, NV, USA, 2016, pp. 580–588.

[17] C. Y. Chuang, J. Li, A. Torralba, and S. Fidler, Learning to act properly: Predicting and explaining affordances from images, in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition* (*CVPR*), Salt Lake City, UT, USA, 2018, pp. 975–983.

[18] T. Nagarajan and K. Grauman, Learning affordance landscapes for interaction exploration in 3D environments, presented at NeurIPS 2020: 34th Annual Conf. Neural Information Processing Systems, Virtual Event, 2020.

[19] E. Kolve, R. Mottaghi, W. Han, E. VanderBilt, L. Weihs, A. Herrasti, M. Deitke, K. Ehsani, D. Gordon, Y. Zhu, et al., AI2-THOR: An interactive 3D environment for visual AI, arXiv preprint arXiv: 1712.05474, 2017.

[20] J. Straub, T. Whelan, L. Ma, Y. Chen, E. Wijmans, S. Green, J. J. Engel, R. Mur-Artal, C. Ren, S. Verma, et al., The Replica dataset: A digital replica of indoor spaces, arXiv preprint arXiv: 1906.05797, 2019.

[21] F. Xia, A. R. Zamir, Z. He, A. Sax, J. Malik, and S. Savarese, Gibson env: Real-world perception for embodied agents, in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition* (*CVPR*), Salt Lake City, UT, USA, 2018, pp. 9068–9079.

[22] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Nießner, M. Savva, S. Song, A. Zeng, and Y. Zhang, Matterport3D: Learning from RGB-D data in indoor environments, arXiv preprint arXiv: 1709.06158, 2017.

[23] P. Anderson, A. Chang, D. S. Chaplot, A. Dosovitskiy, S. Gupta, V.

Koltun, J. Kosecka, J. Malik, R. Mottaghi, M. Savva, et al., On evaluation of embodied navigation agents, arXiv preprint arXiv: 1807.06757, 2018.

[24] Y. Zhu, Y. Zhao, and S. C. Zhu, Understanding tools: Task-oriented object modeling, learning and recognition, in *Proc. 2015 IEEE Conf. Computer Vision and Pattern Recognition* (*CVPR*), Boston, MA, USA, 2015, pp. 2855–2864.

[25] A. Gupta, A. Kembhavi, and L. S. Davis, Observing human-object interactions: Using spatial and functional compatibility for recognition, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 10, pp. 1775–1789, 2009.

[26] M. Savva, A. X. Chang, P. Hanrahan, M. Fisher, and M. Nießner, SceneGrok, *ACM Trans. Graph.*, vol. 33, no. 6, pp. 1–10, 2014.

[27] X. Wang, R. Girdhar, and A. Gupta, Binge watching: Scaling affordance learning from sitcoms, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition* (*CVPR*), Honolulu, HI, USA, 2017, pp. 3366–3375.

[28] V. Delaitre, D. F. Fouhey, I. Laptev, J. Sivic, A. Gupta, and A. A. Efros, Scene semantics from long-term observation of people, in *Proc. 12th Eur. Conf. Computer Vision*, Florence, Italy, 2012, pp. 284–298.

[29] T. Nagarajan and K. Grauman, Learning affordance landscapes for interaction exploration in 3D environments, arXiv preprint arXiv: 2008.09241, 2020.

[30] M. Kotaru, K. Joshi, D. Bharadia, and S. Katti, Spotfi: Decimeter level localization using WiFi, in *Proc. 2015 ACM Conf. Special Interest Group on Data Communication*, London, UK, 2015, pp. 269–282.

[31] Y. Zhuang, C. Y. Zhang, J. Z. Huai, Y. Li, L. Chen, and R. Z. Chen, Bluetooth localization technology: Principles, applications, and future trends, *IEEE Internet Thing J.*, vol. 9, no. 23, pp. 23506–23524, 2022.

[32] W. Yang, X. Wang, A. Farhadi, A. Gupta, and R. Mottaghi, Visual semantic navigation using scene priors, arXiv preprint arXiv: 1810.06543, 2018.

[33] H. Du, X. Yu, and L. Zheng, Learning object relation graph and tentative policy for visual navigation, arXiv preprint arXiv: 2007.11018, 2020.

[34] D. McDermott, M. Ghallab, A. Howe, C. A. Knoblock, A. Ram, M. Veloso, D. S. Weld, and D. Wilkins, PDDL—The planning domain definition language, Tech. Rep. CVC TR-98-003/DCS TR-1165, AIPS-98 Planning Competition Committee, New Haven, CT, USA, 1998.

[35] K. He, X. Zhang, S. Ren, and J. Sun, Deep residual learning for image recognition, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition* (*CVPR*), Las Vegas, NV, USA, 2016, pp. 770–778.

[36] J. Pennington, R. Socher, and C. Manning, GloVe: Global vectors for word representation, in *Proc. 2014 Conf. Empirical Methods in Natural Language Processing* (*EMNLP*), Doha, Qatar, 2014, pp. 1532–1543.

[37] S. Ren, K. He, R. Girshick, and J. Sun, Faster R-CNN: Towards real-time object detection with region proposal networks, arXiv preprint arXiv: 1506.01497, 2015.

[38] T. N. Kipf and M. Welling, Semi-supervised classification with graph convolutional networks, arXiv preprint arXiv: 1609.02907, 2016.

[39] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Harley, T. P. Lillicrap, D. Silver, and K. Kavukcuoglu, Asynchronous methods for deep reinforcement learning, in *Proc. ICML'16: 33rd Int. Conf. Int. Conf. Machine Learning*, New York, NY, USA, 2016, pp. 1928–1937.

[40] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv: 1412.6980, 2014.

[41] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, F. F. Li, and A. Farhadi, Target-driven visual navigation in indoor scenes using deep reinforcement learning, in *Proc. IEEE Int. Conf. Robotics and Automation* (*ICRA*), Singapore, 2017, pp. 3357–3364.

[42] Trossen Robotics, LoCoBot, https://www.trossenrobotics.com/locobot-overview.aspx, 2024.