Long-Term Active Object Detection for Service Robots: Using Generative Adversarial Imitation Learning With Contextualized Memory Graph

Ning Yang, Fei Lu, Guohui Tian, *Member, IEEE*, and Jun Liu, *Senior Member, IEEE*

***Abstract*—Active object detection (AOD) is a crucial task in embodied artificial intelligence within robotics. Previous works mainly address this challenge through deep reinforcement learning (DRL), characterized by prolonged training cycles and model convergence difficulties. Moreover, they often emphasize whether a single AOD task can be completed, overlooking the reality that robots perform long-term AOD tasks. To this end, this article introduces a new AOD solution utilizing a graph based on generative adversarial imitation learning (GAIL). A new expert strategy is devised using the active vision dataset benchmark (AVDB), generating high-quality expert trajectories. Meanwhile, a new AOD model based on GAIL is proposed to predict the robot's execution actions. Moreover, a contextualized memory graph (CMG) is constructed, providing partial state information for the GAIL model and enabling the robot to directly make decisions based on the humanlike memory function. Experimental validation against existing methods in AVDB demonstrates superior results, achieving an 88.8% action prediction accuracy, reducing average path length (APL) to 12.182 steps, and shortening single-step action prediction time to 0.133 s. The proposed method is further evaluated in a real-world home scene, affirming its efficacy and generalization capabilities.**

***Index Terms*—Active vision, contextualized memory graph (CMG), generative adversarial imitation learning (GAIL), long-term active object detection (AOD), service robot.**

I. Introduction

In recent years, embodied artificial intelligence (AI) has garnered significant attention as a mechanism enabling an agent to interact with its environment and actively learn intelligent behaviors [1]. Within service robotics, as a key embodied AI task [2], active object detection (AOD) can guide a robot to autonomously make decisions based on environmental information, assisting it in completing home service tasks related to objects.

How to define AOD? In computer vision, object detection takes an image as input and utilizes a detector, such as Faster R-CNN [3], SSD [4], and YOLO [5], to recognize objects. For the static images, the technology has achieved very significant results, but this passive detection method is unsuitable for robotics [6]. When a robot performs service tasks, it involves many object-related operations, such as grasping [7] and moving [8]. However, the initial distance between the robot and the object typically lies beyond the operational range. Consequently, the robot must not only detect the object but also steadily approach it [9], ultimately observing it from the optimal perspective, as shown in Fig. 1.

Considering the importance of AOD, many research has focused on how to address this problem. Ammirato et al. built an active vision dataset benchmark (AVDB), and the REINFORCE algorithm was employed [10]. Subsequent studies [2], [6], [9], [11] also treated AOD as a Markov decision process, leveraging deep reinforcement learning (DRL) to achieve it. Then, Liu et al. approached AOD as a simple action classification problem and addressed it using behavior cloning (BC) [12]. Ding et al. utilized an integrated framework combining online decision transforms methodologies [13], constructing an AOD model via a blend of offline pretraining and online fine-tuning [2]. Although learning-based approaches have effectively addressed the AOD problem, there are still several issues that need to be resolved.

1) *Effectiveness and Generalization*: When using DRL to address AOD, the agent gathers numerous erroneous experiences, and it is difficult to design an optimal reward function. These will lead to inefficiencies in model training and performance limitations [12].
2) *State Space*: The previous studies [2], [6], [9], [10], [12] took RGB images or bounding boxes as the state space, but they only used the robot's sensory information. Later, long-short term memory (LSTM) was employed in [14], while it enriched the model's state information on the temporal level. No research has considered incorporating home structural characteristics into state information from a spatial perspective.

Received 13 February 2024; revised 7 June 2024, 15 July 2024, and 24 August 2024; accepted 19 September 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 61973187 and Grant 62273203; and in part by the Research Grant Council (RGC) of Hong Kong under Grant 11212321 and Grant 11217922. *(Corresponding authors: Fei Lu; Jun Liu.)*

Ning Yang, Fei Lu, and Guohui Tian are with the School of Control Science and Engineering, Shandong University, Jinan 250061, China (e-mail: 202234938@mail.sdu.edu.cn; lawyerlf@sdu.edu.cn; g.h.tian@sdu.edu.cn).

Jun Liu is with the Department of Data and Systems Engineering, The University of Hong Kong, Hong Kong 999077, China (e-mail: djliu@hku.hk).

Digital Object Identifier 10.1109/TIE.2024.3468636

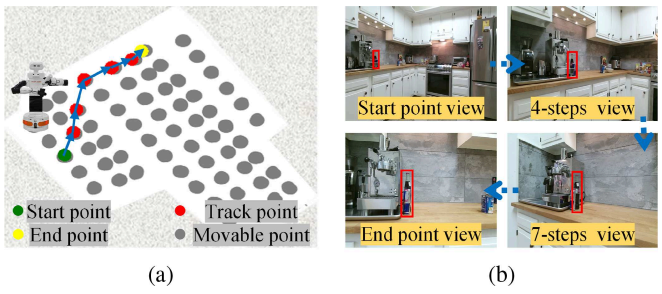


Fig. 1. Illustration of an AOD task. (a) AOD process in a 2-D plan view. (b) First-person view of the service robot.

3) *Long-Term AOD*: Existing researches focus on whether individual AOD tasks can be completed accurately and efficiently, overlooking the reality of multiple executions AOD tasks in the same scene. In long-term AOD, robots need not only the ability to make decisions in the face of new states but also the memory function of historical states, allowing the accumulation of historical tasks to provide experience for later AOD tasks.

To address these challenges, this article introduces a novel method centered around the generative adversarial imitation learning (GAIL) [15] framework. Since training an agent based on GAIL requires a large amount of expert data, there is currently no public expert dataset for the AOD task. To address it, leveraging the original AVDB and utilizing NetworkX [16] along with the A^* algorithm [17], we introduce a novel expert strategy that can automatically and efficiently generate high-quality expert data. In consideration of real-world applications where a service robot executes AOD multiple times for a long time within a home scene, we encapsulate the scene information encountered by the robot during task execution in a graph called contextualized memory graph (CMG). To improve the performance of the GAIL model, we incorporate CMG as a component of the generator input. Additionally, although the structure of CMG is simple, its memory function can help the robot to participate in action decision-making, significantly improving the efficiency of long-term AOD tasks. The main contributions are concluded as follows.

1) A new AOD method based on the GAIL is proposed to help service robots efficiently to complete long-term AOD tasks in complex and dynamic home environments.
2) A novel expert strategy is proposed to generate the expert data for training the GAIL model. This strategy can ensure that the trajectories in expert data are relatively short while including the target object in each view.
3) Considering the long-term nature of the AOD task, a simple yet effective CMG is constructed. The home spatial structure feature obtained from CMG can be used as part of the input of the GAIL generator. Meanwhile, the CMG can also directly participate in AOD action decision-making with its memory function.
4) Comparative experiments against other methods and ablation study are conducted using AVDB to demonstrate the efficiency of our method. Meanwhile, real-environment AOD task testing further validates our method.

The rest of this article is organized as follows. We first outline the related work in Section II. Second, the proposed method is described in detail in Section III. Then, the experiments in AVDB and the real-world scenario are implemented in Section IV. Finally, Section V concludes this article with conclusions and future work.

II. Related Work

According to the traditional robot navigation paradigm, we need to create a map in an environment and use path-planning algorithm to realize AOD [8], [18]. However, when faced with unseen scenes, these methods need to rebuild the map, which is inconvenient in practice. Due to the advancements in DRL, recent research on AOD has predominantly adopted a learning-based approach. Specifically, Ammirato et al. collected visual data such as RGB images from multiple real home scenes to build an AVDB, and the baseline algorithm REINFORCE was employed [10]. Based on it, Han et al. designed a multistep action prediction model to enhance the accuracy and efficiency of AOD tasks [6]. To leverage historical trajectory information, an AOD model incorporating an LSTM module was proposed [14]. To further utilize the trajectory information, Ding et al. proposed an integrated framework combining online Decision Transforms methodologies [13]. Furthermore, to bring the robot closer to the target object for a better perspective, Liu et al. devise a new DRL reward function based on the target object's bounding box [9]. However, when using DRL to address AOD, the early collection of a substantial amount of negative reward experiences by the agent during training can significantly impact model training negatively. Additionally, designing an optimal reward function poses a challenge. Sparse rewards [19], [20] or relying solely on local state determinations [9] can lead to model instability and the agent getting stuck in suboptimal loops, thus impacting task accuracy. Instead of maximizing a reward, Kotar and Mottaghi [21] and Wortsman et al. [22] trained an AOD model on a sequence of images to maximize the detection results of the first frame and used a meta-learning framework. Considering the slow training of DRL models, Liu et al. approached AOD as a simple action classification problem and addressed it using BC [12]. However, this article does not provide a detailed elucidation of how the optimal expert path with the shortest length and the best object observation was generated. Moreover, some studies [23], [24], [25] have employed a graph to establish the relationship between objects, assisting robot in indoor navigation, localization, and predicting scene change. Diverging from these conventional methods, this article utilizes the CMG to represent the spatial structure of home scenes, with the aim of enhancing the cognitive capabilities of agents regarding their surroundings.

III. Proposed Method

A. Problem Statement

Imagine a scenario where a homeowner spots an apple on a nearby table. To quickly reach the apple, the person would take the shortest path to the table while keeping their eyes on the

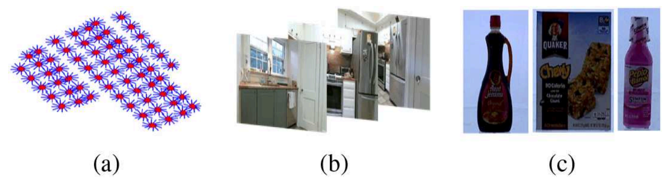


Fig. 2. Composition of AVDB. (a) Image collection point map (M_{ic}). (b) RGB image dataset (I_{rgb}). (c) Target objects (O_s).

apple. We want robots to learn this behavior, so AOD needs to meet the following two perspectives.

1) *Efficiency*: The robot should complete an AOD task with as few steps as possible.
2) *Accuracy*: The target object should remain within the robot's field of view.

AOD requires a robot to not only detect the target object but also continuously navigate toward the object and observe it from a relatively optimal view. Given the need to detect objects, AOD assumes that the target object is already present in the robot's initial view. When executing an AOD task, the robot primarily uses visual information received from its camera as input. In this article, to enrich the input data, we also incorporate spatial structure information about the environment. Based on the input, the robot predicts the most accurate action from seven possible discrete actions: *forward, backward, left, right, rotate_ccw (counterclockwise rotation), rotate_cw (clockwise rotation), done*. The *done* action allows the robot to automatically terminate the task. An AOD task is considered successful if the following conditions are met: the robot avoids collisions with obstacles during task execution, maintains the target object within its field of view, and has the target object within 1 m when the *done* action is executed.

B. Expert Data Generation for AOD

The new AOD expert dataset is generated by leveraging existing AVDB. As shown in Fig. 2, each scene in AVDB primarily consists of M_{ic}, I_{rgb}, and O_s. Additionally, the annotation files F_{at} in AVDB document the bounding boxes of objects, along with action connection between images. Specifically, the generating process of an expert trajectory will be illustrated using a *hunts sauce* in the scene $Home\_005\_2$ as an example.

1) *Constructing a Scene Topology Network (STN)*: The STN is constructed by using the F_{at} and a network analysis tool NetworkX, as shown in Fig. 3. In the STN, cyan nodes represent RGB images collected from I_{rgb}, while gray edges signify the action between two images.
2) *Generating an Initial Trajectory*: Utilizing the STN, existing path-planning algorithm A* can determine the initial shortest trajectory $E_t = [I_1, I_2, \ldots, I_e]$ between a start state I_1 and an end state I_e. However, not all nodes within the trajectory contain the target object, as shown in the grayscale images in steps 6 and 7 of Fig. 3. So, further optimization of the initial trajectory is necessary.
3) *Optimizing the Initial Trajectory E_t*: Traversing all trajectory points I_i ($i \in [1, e]$) in E_t sequentially, if *hunts sauce* appears in the RGB image corresponding to I_i, it is retained in E_t in its original order. Otherwise, it needs to return to I_{i-1} and reselect image points from the $i-1$ step. Specifically, based on F_{at}, identify all image points $P = [p_1, p_2, \ldots, p_6]$ connected to I_{i-1}. Then, successively use the images in P to determine the shortest trajectories $E'_{t-j}(j \in [1, 6])$ of every new initial state to the end state I_e. Subsequently, calculate the view departure length for each trajectory
$$l_j = L_j - \text{Num}(E'_{t-j}) \quad (1)$$
where L_j represents the length of the new subtrajectory, and $\text{Num}(E'_{t-j})$ denotes the number of images containing the *hunts sauce*. The smaller the l_j is, the better the quality of the new subtrajectory is. Select E'_{t-j} corresponding to the minimum value and use it to replace the trajectory from I_i to I_e in the initial trajectory E_t. Then, traverse the new trajectory points in E'_{t-j} and repeat the aforementioned process. Eventually, an optimized trajectory $\dot{E}_t$ is obtained. The optimized trajectory, illustrated as the red trajectory in Fig. 3, is visually represented by RGB images labeled with numbers.

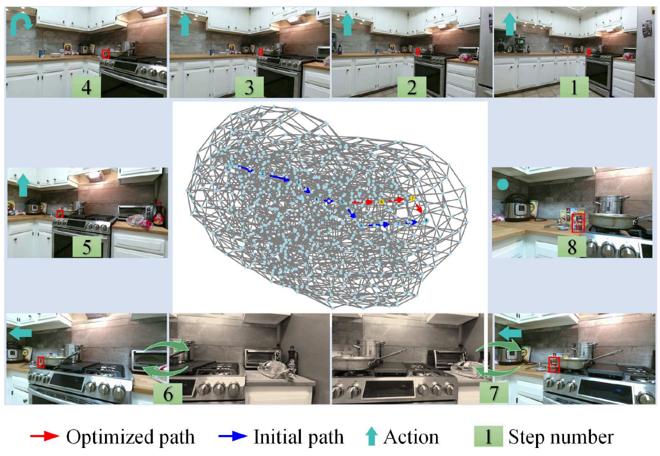


Fig. 3. STN is centrally positioned. The blue path depicts the initial trajectory, while the red path illustrates the optimized expert trajectory. Distributed around are first-person views of the robot at different times.

The expert trajectory consists of binary trajectory points in the form of (s,a). After obtaining the optimal path $\dot{E}_t$, the connection action a between image points can be determined based on F_{at}, then the expert trajectory $T = [(s_1, a_1), (s_2, a_2), \ldots, (s_e, a_e)]$ can be obtained. Finally, we generate a total of 53 452 expert trajectories.

C. GAIL-AOD Model

The overall framework of the GAIL-based AOD model (GAIL-AOD) is shown in Fig. 4. The network structure of the discriminator is composed of multi-layer perception (MLP). The generator is the core part of GAIL-AOD, which consists of the state feature extraction network and the asynchronous advantage actor–critic (A3C) network. The state feature extraction network comprises the observation stream and the CMG (see Section II-C for constructing details) stream. The former

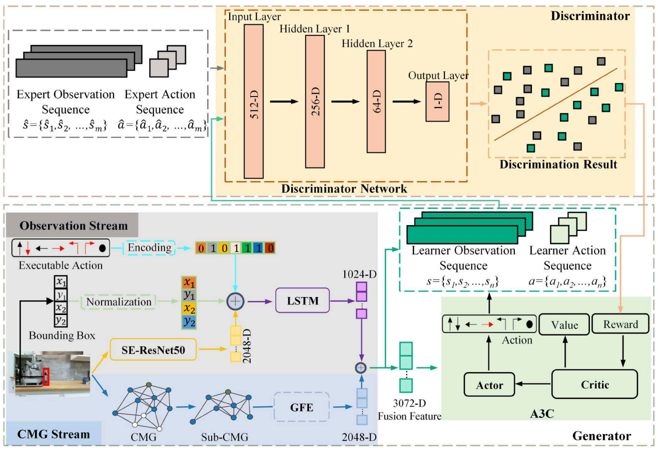


Fig. 4. Overall framework of GAIL-AOD model.

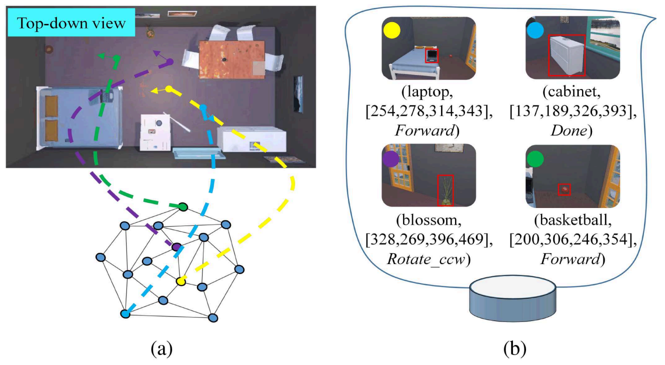

Top-down view
(laptop, [254,278,314,343], *Forward*)
(cabinet, [137,189,326,393], *Done*)
(blossom, [328,269,396,469], *Rotate_ccw*)
(basketball, [200,306,246,354], *Forward*)
(a)
(b)

Fig. 5. (a) Example for establishing the CMG. (b) Local memory carried by the CMG.

mainly comprises an RGB image, the bounding box of the target object, and action sequence A. Initially, we employ the SE-ResNet50 [26] to extract the feature from the RGB image. Regarding sequence A, we subject it to encoding processing: actions feasible in the current state are encoded as 1, whereas others are encoded as 0. After synthesizing the three feature vectors, they are fed into the LSTM module, resulting in a 1024 dimensions output feature, denoted as N_1.

In the CMG stream, we select all nodes within a *n-step* distance from the robot's current node to construct a sub-CMG. Subsequently, the graph feature extraction (GFE) is performed on the subnetwork. This module draws on the principles of graph convolution network: assuming the adjacency matrix, degree matrix, and feature matrix of S-CMG are denoted as $A^{N\times N}$, $D^{N\times N}$, and $H^{N\times D}$, respectively, where N represents the number of nodes and D represents the node feature dimension. Then, we can extract the feature N_2 of S-CMG

$$N_2 = \sigma[W \times (\tilde{D}^{-\frac{1}{2}} \times \tilde{A} \times \tilde{D}^{-\frac{1}{2}} \times H)] \tag{2}$$

where σ represents the activation function ReLu, and $W^{1\times N}$ is a weight matrix. If the current time step is t, and the establishment time step of each node in S-CMG is t_k (k = 1, 2, ..., T), then

$$W = [e^{-(t-t_1)}, e^{-(t-t_2)}, \ldots, e^{-(t-t_{\mathrm{T}})}] \tag{3}$$

where $e^{-(t-t_k)}$ is defined as the time decay factor. By concatenating the output features N_1 and N_2, we can obtain N

$$N = \text{Concatenate}(N_1, N_2). \tag{4}$$

Upon inputting the concatenated N into the A3C network, the actor network can derive the predicted action, while the critic network can assess and score the execution of the predicted action in the current state.

D. CMG

The simple but efficient CMG represents the spatial structure of the home scene where a robot is located in the form of a graph, as shown in Fig. 5(a). We denote the CMG as $G = (V, E)$, where V and E represent nodes and connecting edges between nodes, respectively. Each node $v \in V$ is recorded as a triplet of position and orientation:

$$v = (x, y, \beta) \tag{5}$$

where (x, y) represents the 2-D coordinates of the robot in the home space, and β represents the orientation. Each node in CMG stores the feature information of the corresponding RGB image. Assuming that v_t is (x_t, y_t, β_t) at time t, then v_{t+1} is as follows:

$$(x_t + d \times \cos(\beta_t + \alpha), y_t + d \times \sin(\beta_t + \alpha), \beta_t + \alpha) \tag{6}$$

where d and α represent the distance and angle of the robot movement. If the robot takes the action *Rotate_ccw*, α takes a positive value, otherwise it takes a negative value. Each edge in E represents the action relationship between two nodes. For example, from (0, 0, 0) to (0, 0, 30), the action between these two nodes is *Rotate_ccw*.

In addition, CMG also carries a local memory (LM), as shown in Fig. 5(b), which is used to record the name of the target object, the bounding box, and the historical action. It is worth noting that to maximize the auxiliary performance of CMG for the long-term AOD task, this article does not store action into the LM after each execution, only when the AOD task is successfully completed, and the actions will be stored in the corresponding nodes. Since each scene and the objects within it are different, the CMG and its LM are also unique. When performing tasks in a new scene, the agent needs to recreate and continuously update the CMG.

E. Overall Method of the Long-Term AOD

The overall execution process of a long-term AOD task is as shown in Fig. 6. If the current state s_t has been stored in the nodes of CMG, the robot will enter the memory search stage; otherwise, it will directly perform single-step action prediction using the GAIL-AOD model. In the memory search stage, the robot first tries to use the CMG and A* algorithm to plan the shortest path *Ts* directly from s_t to the stored end state s_e. If possible, the robot will execute the actions $[a_t, a_{t+1}, \ldots, a_e]$ obtained from *Ts* in sequence until it encounters the termination

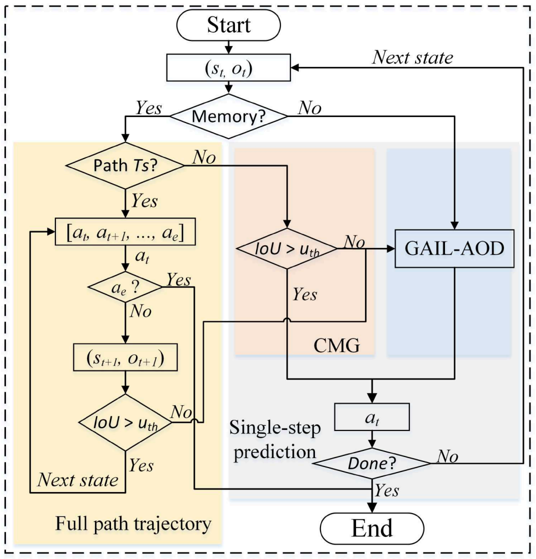


Fig. 6. Entire AOD process using the GAIL-AOD model with a CMG.

Algorithm 1: The Process of Each Step in the Long-Term AOD.

```
Require: Current state $s_t$, target object $o_t$, the CMG and the
    learned GAIL-AOD model.
Ensure: An optimal planned path $Ts$ or a predicted action $a_t$.
1: if $Ts$ is generated by the A$^*$ algorithm using CMG then
2:     Return $Ts$
3: else if $O_t$ exists in the CMG and $o$ satisfies condition (7)
   then
4:     Get the stored historical action $a_t$
5:     Return $a_t$
6: else
7:     Use $O_t$ to predict $a_t$ via the GAIL-AOD model
8:     Return $a_t$
9: end if
```

action a_e. If Ts cannot be obtained due to incompleteness of the CMG, the action will be obtained with single-step action prediction. During this process, if the historical action execution information corresponding to the target object in the current state can be queried in the CMG and satisfy (7) compared with the historical record value, the robot will directly execute the action a_t.

$$\mathrm{IoU} > u_{\mathrm{th}} \tag{7}$$

where IoU represents the intersection over the union of the current and historical bounding boxes. u_{th} is a threshold belonging to the range 0–1. If the above condition is not met, the robot needs to use the state information of s_t to predict the action in real time. Repeat the above steps until the *done* action is raised. In short, at each time step, based on the current state information, the robot either directly obtains the shortest planned path or uses the GAIL-AOD model or CMG to perform single-step action prediction to obtain an executable action, as shown in Algorithm 1.

TABLE I
DATASET SPLIT IN OUR EXPERIMENT

Train Scenes	Test Scenes	Split
Home_001_1, Home_003_1, Home_004_1, Home_005_1, Home_007_1, Home_010_1, Home_011_1, Home_014_1, Home_015_1, Home_016_1	Home_001_2, Home_005_2, Home_014_2	Split1 (KE)
	Home_002_1, Home_006_1, Home_008_1	Split2 (UE)

TABLE II
PRIMARY PARAMETERS FOR GAIL-AOD MODEL

Parameter	Value	Parameter	Value
Image size	224 × 224	Agent numbers	4
Discount factor	0.99	Epoch	5000
Batch size	128	Tau	0.95

IV. Experiments

A. Dataset and Setup

AVDB simulates the activity process of a service robot in real indoor scenes. Currently, the dataset includes 17 household scenes and two office scenes, with over 30 000 RGB-D images and over 70 000 2-D bounding boxes of objects. We divide AVDB into two splits: *Split*1 and *Split*2, as shown in Table I. *Split*1 is mainly used to test the generalization of models when the environment is known (KE), but the position of target objects is changed. Correspondingly, *Split*2 tests in a completely unknown environment (UE).

All experiments are conducted on a single GeForce RTX 3090 GPU. The GAIL-AOD model is trained by an Adam optimizer with a learning rate of 0.0003 to optimize the MLP and the A3C network. Table II provides other training parameters. To assess the performance of different methods, three metrics are selected: success rate (SR), APL, and average decision time (ADT). SR represents the effectiveness of various methods

$$\mathrm{SR} = \frac{\mathrm{count}(T_d)}{\mathrm{count}(T_{all})} \tag{8}$$

where count(T_d) and count(T_{all}), respectively, represent the number of successful and total AOD tasks. APL evaluates task execution efficiency from the perspective of path length

$$\mathrm{APL} = \frac{\sum_{z\in T_d}(\mathrm{PL})_z}{\mathrm{count}(T_d)} \tag{9}$$

where PL represents the path length of a successful task z. ADT accesses the robot's decision-making speed from a time perspective, denoted as

$$\mathrm{ADT} = \frac{\sum_{z\in T_d}\sum_{d\in z}(\mathrm{DT})_d}{\sum_{z\in T_d}(\mathrm{PL})_z} \tag{10}$$

where DT represents the time it takes to make a decision *d*.

Additionally, to evaluate the computational burden of each method, metrics such as floating point operations (FLOPs), memory usage, GPU utilization (GPU-Util), and the minimum

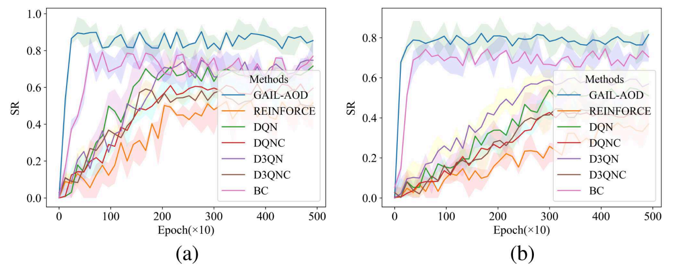


Fig. 7. Comparison of the accuracy of different AOD models, including average and deviation values. (a) Test in KE. (b) Test in UE.

TABLE III
TEST RESULTS OF DIFFERENT METHODS

Method	KE			UE			Average		
	SR	APL	ADT	SR	APL	ADT	SR	APL	ADT
Random	0.039	5.706	0.007	0.073	8.659	0.007	0.056	7.183	0.007
Forward	0.005	3.170	0	0.012	6.324	0	0.009	4.747	0
REINFORCE [10]	0.541	8.565	0.341	0.384	8.480	0.353	0.463	8.523	0.347
DQN [9]	0.765	13.130	0.547	0.576	14.780	0.576	0.671	13.955	0.562
DQN-C [6]	0.645	18.410	0.501	0.454	21.880	0.511	0.550	20.145	0.506
D3QN	0.760	11.320	0.645	0.620	15.250	0.631	0.690	13.285	0.638
D3QN-C	0.605	13.360	0.636	0.470	15.581	0.603	0.538	14.471	0.620
BC [12]	0.814	12.476	0.551	0.774	18.434	0.495	0.794	15.455	0.523
GAIL-AOD	0.868	16.516	0.702	0.818	17.436	0.554	0.843	16.976	0.628
BC-G	0.852	**10.993**	**0.137**	0.809	15.817	0.215	0.831	13.405	0.176
Ours	**0.907**	11.989	0.144	**0.869**	**12.375**	**0.121**	**0.888**	**12.182**	**0.133**

Note: The bold entries indicate the best result for each metric.

number of epochs (Min-Epochs) required for model convergence are considered.

B. Results and Analysis

To demonstrate the advantages of our method, ten comparative experiments are conducted.

1) *Random*: Randomly select an action at each step.
2) *Forward*: Only *forward* action is performed.
3) *REINFORCE*: The baseline policy used in [10].
4) *DQN*: A deep Q-learning model using prioritized experience replay in [9].
5) *DQN-C*: The reward function is determined based on the instance classification value [6], [10].
6) *D3QN*: Upon the foundation of DQN, a double Q-learning training strategy is incorporated alongside the introduction of a dueling network structure.
7) *D3QN-C*: Similar to D3QN, but the reward function is the same as DQN-C.
8) *BC*: The network structure is an MLP, and an auxiliary revision method is designed [12].
9) *GAIL-AOD*: Using only our proposed GAIL-AOD model.
10) *BC-G*: The BC algorithm combined with our proposed CMG memory function.
11) *Ours*: The proposed overall method involves the GAIL-AOD model and incorporating the CMG memory function.

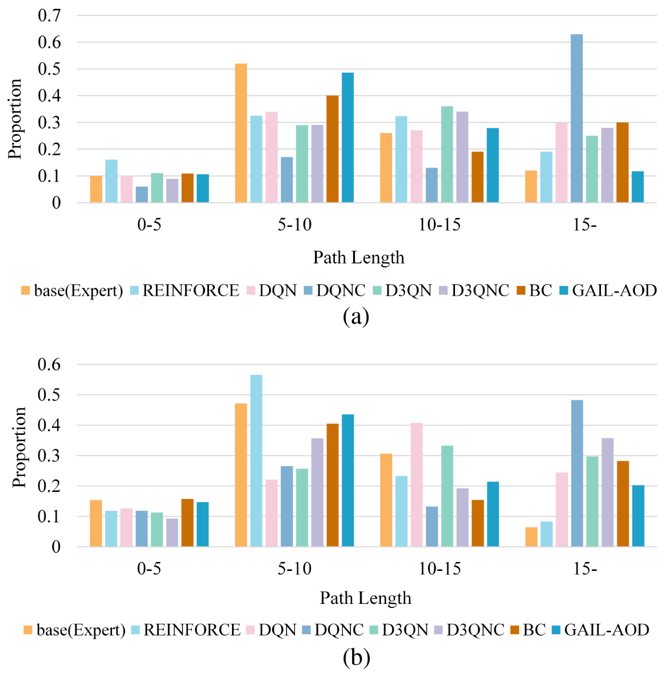


Fig. 8. Proportion of path length ranges for successful AOD tasks in different models. (a) KE. (b) UE.

TABLE IV
COMPARISON OF COMPUTATIONAL BURDEN AMONG DIFFERENT METHODS

Method	GFLOPs	ADT	Memory Usage (GB)	GPU-Util (Average/Max)	Min-Epochs
REINFORCE	1.342	0.347	1.589	0.201/0.422	395
DQN	3.671	0.562	2.972	0.262/0.507	276
DQN-C	3.671	0.506	2.970	0.257/0.462	283
D3QN	3.721	0.638	3.237	0.318/0.580	251
D3QN-C	3.721	0.620	3.237	0.325/0.610	280
BC	7.837	0.523	4.651	0.425/0.720	37
GAIL-AOD	4.155	0.628	4.178	0.316/0.660	**18**
BC-G	7.837	0.176	3.953	0.307/0.620	—
Ours	**4.155**	**0.133**	**3.537**	**0.235/0.680**	—

Note: The bold entries indicate the best result for each metric.

To avoid the incidental nature of experimental outcomes, each set of experiments is conducted thrice. During the evaluation, the average and deviation values of the accuracy for different learning-based models are illustrated in Fig. 7. The test results are recorded in Table III. Fig. 8 shows the proportion of successful AOD tasks between different step intervals. Table IV compares the computational burden of different methods using metrics such as FLOPs, ADT, and GPU-Util. In Fig. 9, an AOD task execution process based on different methods is visualized. Combining all the above experimental results, the following conclusions can be drawn.

1) Figs. 7 and 8 show that, whether in KE or UE, our model achieves convergence with fewer training iterations and exhibits optimal performance. The distribution of step numbers is also closest to the expert policy. The quick model convergence is due to the presence of expert data, eliminating the need for inefficient exploration. The expert

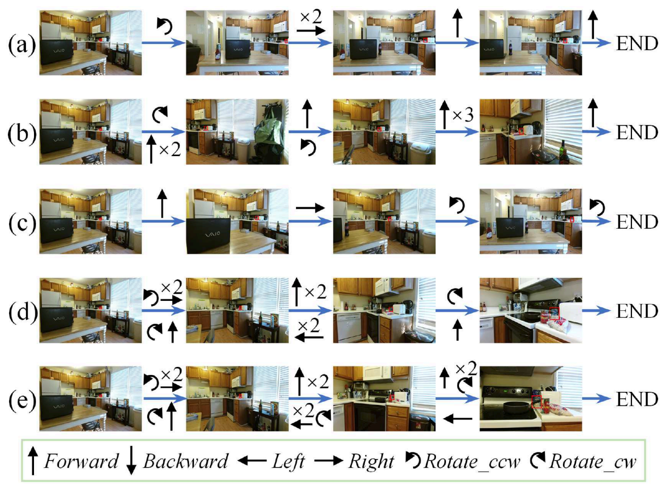


Fig. 9. Visualizing the execution process of an AOD task: a demonstration showcasing the detection of "canned clam soup," where the red bounding box indicates the recognized object. The different methods are as follows: (a) REINFORCE; (b) DQN; (c) D3QN-C; (d) BC; and (e) Ours.

TABLE V
TEST RESULTS USING DIFFERENT EXPERT STRATEGIES

Expert Strategy	P-L	T-O	KE		UE		Average	
			SR	APL	SR	APL	SR	APL
Heuristic strategy	–	✓	0.676	19.254	0.651	18.382	0.664	18.818
NetworkX + A*	✓	–	0.377	**10.582**	0.320	14.357	0.349	12.470
Ours	✓	✓	**0.907**	11.989	**0.869**	**12.375**	**0.888**	**12.182**

Note: The bold entries indicate the best result for each metric.

TABLE VI
TEST RESULTS WITH DIFFERENT APPLICATIONS OF CMG

State Input	Action Decision	KE			UE			Average		
		SR	APL	ADT	SR	APL	ADT	SR	APL	ADT
–	–	0.621	21.299	0.601	0.576	25.975	0.527	0.599	23.137	0.564
✓	–	0.868	16.516	0.702	0.818	17.436	0.554	0.843	16.976	0.628
–	✓	0.728	15.475	0.179	0.687	12.403	0.154	0.708	13.939	0.167
✓	✓	**0.907**	**11.989**	**0.144**	**0.869**	**12.375**	**0.121**	**0.888**	**12.182**	**0.133**

Note: The bold entries indicate the best result for each metric.

data constructed in this article are of high quality and efficiency. The "high quality" enables the agent to learn correct behaviors, improving action prediction accuracy compared to [12]. The "high efficiency" ensures that the agent completes AOD tasks with the fewest possible steps. These two factors together enable the agent to accomplish various difficulty levels in AOD tasks.

2) Table III shows that our proposed GAIL-AOD model has the highest accuracy. Specifically, it achieves a 6.2% performance improvement over the latest method [12]. This improvement is mainly due to considering the local spatial information of the robot in the model input, enhancing the robot's spatial structure cognition. The APL value also indicates that the GAIL-AOD model can complete relatively difficult long-distance AOD tasks. When the CMG memory mechanism is added, the overall method achieves a performance boost, with an average accuracy reaching 88.8%, a 4.5% increase over the standalone GAIL-AOD model. This improvement is mainly due to the CMG memory mechanism, which allows the robot to directly execute the correct action stored in the CMG when facing the same state, rather than relying solely on the GAIL-AOD model for action prediction. The assistance of CMG enhances the execution efficiency of AOD tasks, especially achieving an ADT value of 0.133 s, proving that our method is more suitable for long-term AOD task execution in the same home scenario.

3) According to IV, the FLOPs value of our GAIL-AOD model is larger than that of RL-based models but 47% lower than the latest method [12]. This reduction is mainly due to replacing the complex residual network used for deep information extraction with a simple but efficient CMG spatial representation and its special feature extraction method, GFE. Although the GAIL-AOD model has relatively higher computational complexity and resource requirements, it requires fewer training iterations, significantly improving training efficiency. Additionally, for long-term AOD tasks, our model's average action prediction speed is faster. Especially compared to BC, our overall method requires fewer computational resources, making it more suitable for practical deployment, considering performance, efficiency, and resource conservation, provided the hardware allows.

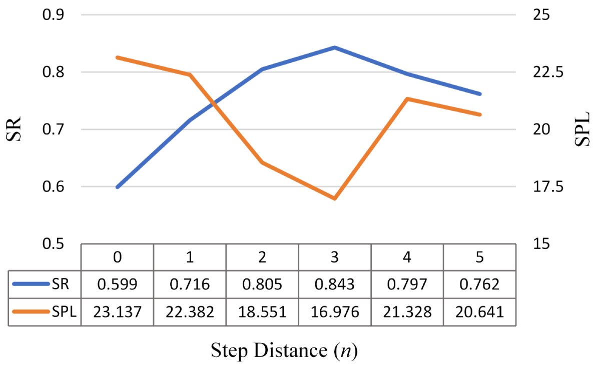
4) Fig. 9 visualizes a challenging AOD task characterized by long distances and an obstacle. It indicates that RL-based methods fail the task either due to the robot colliding with obstacles or losing the target object from view after executing the final action. The task execution process of the BC-based method appears successful but actually fails because the agent prematurely terminates the AOD task, resulting in a suboptimal final observation view of the target object. Our method guides the robot to gradually approach the target object, avoid obstacles, and keep the object in view, ultimately successfully terminating the task in an optimal observation state.

C. Ablation Study

We conduct the experiments of ablation study about the expert strategy, the CMG, the *n-step* about ensuring sub-CMG, and the threshold u_{th} in (7) and analyze their respective roles in the AOD task.

1) Expert Strategy: In the AOD task, there are two requirements for the expert strategy: the path length is as short as possible (P-L), and the target object cannot leave the robot's field of view (T-O). To explore the necessity of choosing the two points above as indispensable conditions for the expert strategy, three different expert strategies are selected for comparison as follows.

a) *Heuristic strategy*: Based on the reward function outlined in [9], a method to determine the optimal action for each state is employed as a heuristic action selection strategy. This strategy satisfies T-O but not P-L.



	0	1	2	3	4	5
SR	0.599	0.716	0.805	0.843	0.797	0.762
SPL	23.137	22.382	18.551	16.976	21.328	20.641

Fig. 10. Test results at different step distance *n*.

TABLE VII
TEST RESULTS USING DIFFERENT u_{th} VALUES

u_{th}	SR	APL	ADT
0.25	0.751	14.561	0.152
0.5	**0.888**	**12.182**	**0.133**
0.75	0.869	16.357	0.343
1.0	0.857	17.674	0.558

Note: The bold entries indicate the best result for each metric.

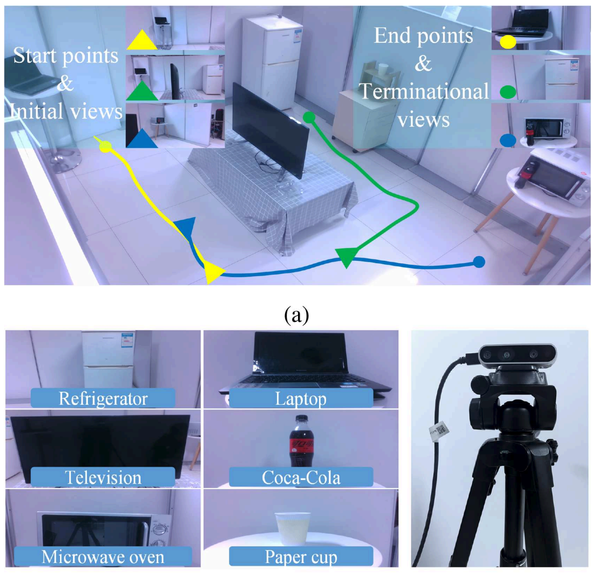


Fig. 11. (a) Experimental setup in a real environment and an example of three different levels of AOD tasks. (b) Different types of target objects. (c) Intel RealSense D435 depth camera.

b) *NetworkX* $+A^*$: Using the A^* algorithm in the STN allows for the direct determination of the expert path, which satisfies P-L but not T-O.

c) *Ours*: The expert strategy proposed in this article, which can satisfy the above two requirements.

We conduct model training using expert data generated by the three methods above, the outcomes of which are presented

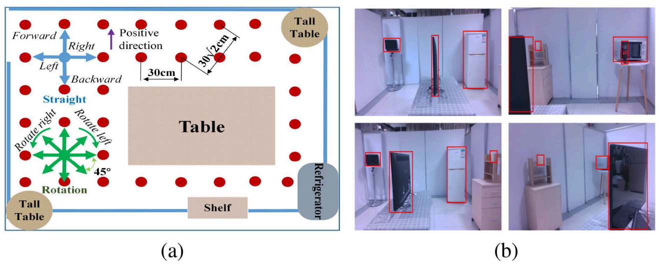


Fig. 12. (a) Two-dimensional layout of the real environment. (b) Partial RGB images and distribution of target objects in the real-world scene dataset.

TABLE VIII
TEST RESULTS IN THE REAL ENVIRONMENT

Method	Group	SR	APL	ADT
MC_{rgb}	Simple	1.0	6.758	0.306
	Complex	1.0	8.054	0.529
	Average	1.0	7.406	0.418
Ours	Simple	**1.0**	**5.353**	**0.206**
	Complex	**1.0**	**6.146**	**0.189**
	Average	**1.0**	**5.750**	**0.198**

Note: The bold entries indicate the best result for each metric.

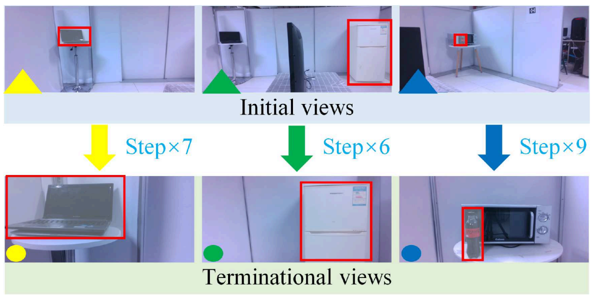


Fig. 13. Visualization of three different AOD Tasks, showing only initial and end states before and after task execution.

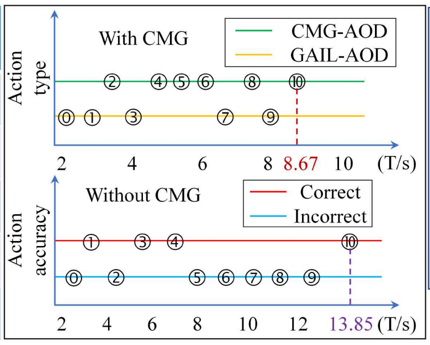
in Table V. Clearly, our expert strategy demonstrates better performance, achieving an accuracy of 88.8% and maintaining an APL of only 12.182. Therefore, it is verified that when designing expert policies, P-L and T-O are crucial considerations. Neglecting the former affects task execution efficiency (APL: 18.818), while poor design of the latter significantly reduces task completion accuracy (SR: 34.9).

2) CMG: In this article, CMG serves two primary functions: a) *state input*: the extracted spatial structure feature from CMG can be used as part of the input of the GAIL-AOD model; and b) *action decision*: utilize the memory information to assist the robot in quick action decision-making.

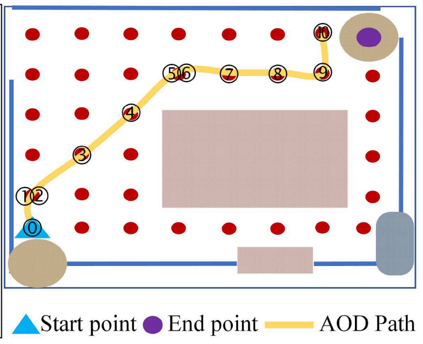
To investigate how the presence of CMG aids robots in solving the AOD task, the ablation experiments are divided into four groups, and the findings are presented in Table VI. The results indicate that using the local spatial features represented by the CMG as state input significantly enhances the SR. In the first two sets and the latter two sets of experiments, the accuracy increase by 24.4% and 18.0% when *state input* is

Step Transfer	Action	Step Transfer	Action
⓪→①	*Left*	⑤→⑥	*Rotate_cw*
①→②	*Rotate_ccw*	⑥→⑦	*Forward*
②→③	*Forward*	⑦→⑧	*Forward*
③→④	*Forward*	⑧→⑨	*Forward*
④→⑤	*Forward*	⑨→⑩	*Left*

(a)



(b)



(c)

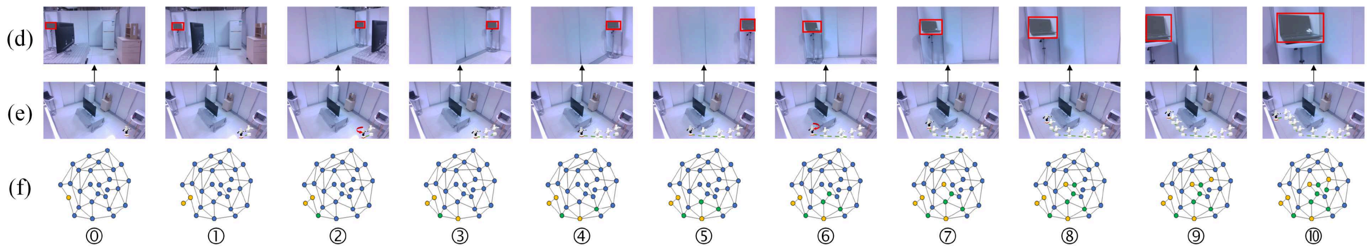


Fig. 14. Demo of AOD task. (a) Action type between two transition steps. (b) Impact of whether to use the CMG on the action decision-making. The robot's moving path is demonstrated in (c). (d) Laptop in the first view. The simulated movement trajectory of the robot in the real scene is shown in (e), and the red arrow represents the rotation action. Meanwhile, the updating process of CMG is presented in (f). The yellow nodes represent actions corresponding to the node's state predicted by the GAIL-AOD model, while green nodes signify actions obtained directly from the CMG.

present compared to when it is absent. When considering the *action decision*, a comparison of the results between the first and third groups, as well as the second and fourth groups, reveals a decrease in APL values by 9.198 and 4.794, respectively. Additionally, the single-step decision-making time is reduced by 0.397 and 0.495 s. It is evident that despite the simplicity of the CMG structure, its application significantly enhances the efficiency of the AOD tasks.

3) n-Step in Sub-CMG: To select the sub-CMG that best represents the local spatial structure around a certain point from the CMG, we tested different values of n in KE and UE, with the results shown in Fig. 10. From the results, it can be observed that the performance is suboptimal when n is less than 3, mainly because the sub-CMG fails to sufficiently represent the local structure, limiting the information obtained by the robot. Conversely, when n is greater than 3, performance declines because the sub-CMG structure becomes overly complex, leading to redundant representations of the spatial structure, from which the robot cannot effectively extract useful information. Therefore, considering both the SR and APL metrics, the optimal performance for the robot in executing AOD tasks is achieved when n is set to 3.

4) Threshold u_{th}: Table VII indicates that when $u_{\text{th}} = 0.5$, the CMG's decision-support function is most effectively utilized, helping the robot to complete AOD tasks with higher accuracy and efficiency. When $u_{\text{th}} < 0.5$, the condition is too lenient, leading to incorrect usage of historical information due to untimely updates of memory information, which degrades task performance. Conversely, when $u_{\text{th}} > 0.5$, the condition is too stringent, preventing the CMG from fully supporting decision-making. As a result, action decisions overly rely on the GAIL-AOD model, significantly reducing task efficiency. Therefore, $u_{\text{th}} = 0.5$ is the relatively optimal value.

D. Experiments in Real-World Scene

A real home scene [see Fig. 11(a)] is built to validate the proposed method, and the robot can engage in three varying levels of difficulty for AOD tasks: easy, moderate, and hard, as shown by the yellow, green, and blue paths, respectively. In this scene, there are six different target objects [see Fig. 11(b)]. The real-world scene is discretized, and the Intel RealSense D435 depth camera [see Fig. 11(c)] is used to collect images. Fig. 12(a) illustrates the 2-D layout of the constructed scene encompassing 35 discrete points. At each point, an RGB image is captured by rotating 45°. Ultimately, 280 images are collected, each potentially containing multiple target objects [see Fig. 12(b)]. For target object, we use the pretrained Yolo-v3 [27].

1) Quantitative Analysis: To emphasize the performance of the proposed method, an additional method is introduced: MC_{rgb}: manually selecting the action based on the size and position of the target object in the RGB image; and *Ours*: the proposed GAIL-AOD method with the CMG. During the test, target objects are divided into two groups according to two attributes (size and dynamics): simple group: refrigerator, television, and microwave oven; and complex group: laptop, coca-cola, and paper cup. The experimental result is shown in Table VIII. In Fig. 13, the execution process of three different difficult AOD tasks mentioned in Fig. 11 is simply visualized. According to these, *Ours* reaches the same level as MC_{rgb} in terms of accuracy, and the APL and ADT values are lower, indicating that our method can complete the AOD task and achieve humanlike performance in the real environment. Fig. 13 shows that given an initial state containing a target object, the robot can leverage our method to predict the correct sequence of actions and terminate the task in the suitable state.

2) Qualitative Analysis: To qualitatively analyze the robot's execution performance, we select an AOD task targeting a laptop as a demonstration example, as illustrated in Fig. 14. Initially, the laptop is at the edge of the RGB image [Fig. 14(d)⓪], so the robot performs the $left$ and $rotate\_ccw$ actions to keep the laptop in the view, reaching state ②. Subsequently, to minimize the distance, the robot consecutively performs three *forward* at states: ②, ③, and ④. At state ⑤, the laptop is about to move out of view, then the robot executes $rotate\_cw$ to reposition it back to the center of view, which demonstrates the robustness of the proposed method. Then, the robot continuously performs three *forward* actions and a *left* action to approach the laptop and attains the best observation viewpoint steadily. Finally, at state ⑩, the robot proactively terminates this AOD task. Furthermore, when using CMG, the robot completes the AOD task in only 8.67 s, which is 5.18 s faster than without CMG (13.85 s) [see Fig. 14(b)]. This acceleration is due to specific actions (②, ④, ⑤, ⑥, ⑧, ⑩) directly obtained from CMG, significantly reducing task execution time. When using CMG, all action predictions are accurate, whereas without CMG, there are four incorrect action predictions (①, ③, ④, ⑩). Thus, using CMG improves time efficiency and enhances task completion accuracy.

V. Conclusion

In summary, this article introduces a novel AOD framework for the service robot based on GAIL and considers the long-term nature of AOD tasks. We used the NetworkX and A* algorithm to automatically generate expert data. Considering task accuracy and efficiency, an expert trajectory optimization strategy is proposed. To enhance the robot's perception of spatial information, we develop a simple yet efficient CMG to represent spatial structure and extract the local spatial feature. Moreover, the CMG is endowed with a human-like memory mechanism, enabling it to participate directly in the action decision process. Experimental results in AVDB demonstrate that the GAIL-AOD model achieves higher accuracy and efficiency. The introduction of the CMG not only further improves task accuracy but also significantly reduces the ADT and decreases computational resource consumption. This is particularly beneficial for long-term AOD tasks, effectively reducing the consumption of power resource. Furthermore, we also establish a real-world home scene to verify the proposed method. In the future, we will attempt to apply our method to more challenging object navigation or search tasks and establish a larger outdoor test scenario.

References

[1] J. Duan, S. Yu, H. L. Tan, H. Zhu, and C. Tan, "A survey of embodied AI: From simulators to research tasks," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 6, no. 2, pp. 230–244, Jan. 2022.

[2] W. Ding et al., "Learning to view: Decision transformers for active object detection," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2023, pp. 7140–7146.

[3] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[4] W. Liu et al., "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2016, pp. 6517–6525.

[5] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 779–788.

[6] X. Han, H. Liu, F. Sun, and X. Zhang, "Active object detection with multistep action prediction using deep Q-network," *IEEE Trans. Ind. Inf.*, vol. 15, no. 6, pp. 3723–3731, Jun. 2019.

[7] Y. Hou, J. Li, and I.-M. Chen, "Self-supervised antipodal grasp learning with fine-grained grasp quality feedback in clutter," *IEEE Trans. Ind. Electron.*, vol. 71, no. 4, pp. 3853–3861, Apr. 2024.

[8] D. Huang, B. Li, Y. Li, and C. Yang, "Cooperative manipulation of deformable objects by single-leader-dual-follower teleoperation," *IEEE Trans. Ind. Electron.*, vol. 69, no. 12, pp. 13162–13170, Dec. 2022.

[9] S. Liu, G. Tian, Y. Zhang, M. Zhang, and S. Liu, "Active object detection based on a novel deep Q-learning network and long-term learning strategy for the service robot," *IEEE Trans. Ind. Electron.*, vol. 69, no. 6, pp. 5984–5993, Jun. 2022.

[10] P. Ammirato, P. Poirson, E. Park, J. Kosecka, and A. C. Berg, "A dataset for developing and benchmarking active vision," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2017, pp. 1378–1385.

[11] X. Han, H. Liu, F. Sun, and D. Yang, "Active object detection using double DQN and prioritized experience replay," in *Proc. Int. Jt. Conf. Neural Netw.*, 2018, pp. 1–7.

[12] S. Liu, G. Tian, X. Shao, and S. Liu, "Behavior cloning-based robot active object detection with automatically generated data and revision method," *IEEE Trans. Robot.*, vol. 39, no. 1, pp. 665–680, Feb. 2023.

[13] Q. Zheng, A. Zhang, and A. Grover, "Online decision transformer," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 27042–27059.

[14] N. Ye, R. Wang, and N. Li, "A novel active object detection network based on historical scenes and movements," *Int. J. Comput. Theory Eng.*, vol. 13, no. 3, pp. 79–83, 2021.

[15] J. Ho and S. Ermon, "Generative adversarial imitation learning," in *Proc. 30th Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 4572–4580.

[16] A. A. Hagberg, D. A. Schult, and P. J. Swart, "Exploring network structure, dynamics, and function using NetworkX," in *Proc. 7th Python Sci. Conf.*, 2008, pp. 11–15.

[17] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cyber.*, vol. 4, no. 2, pp. 100–107, Jul. 1968.

[18] C. Cadena et al., "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, Dec. 2016.

[19] Y. Zhu et al., "Target-driven visual navigation in indoor scenes using deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2017, pp. 3357–3364.

[20] N. Yang, F. Lu, B. Yu, F. Yao, D. Zhang, and G. Tian, "Service robot active object detection based on spatial exploration using deep recurrent q-learning network," in *Proc. IEEE Int. Conf. Robot. Biomimetics*, 2023, pp. 1–6.

[21] K. Kotar and R. Mottaghi, "Interactron: Embodied adaptive object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 14840–14849.

[22] 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. Comput. Vis. Pattern Recognit.*, 2019, pp. 6743–6752.

[23] V. K. Nikolay Savinov and Alexey Dosovitskiy, "Semi-parametric topological memory for navigation," in *Proc. 6th Int. Conf. Learn. Represent.*, 2018, pp. 1–12.

[24] W. Yang, X. Wang, A. Farhadi, A. Gupta, and R. Mottaghi, "Visual semantic navigation using scene priors," in *Proc. 7th Int. Conf. Learn. Represent.*, 2019, pp. 1–14.

[25] X. Ye and Y. Yang, "Hierarchical and partially observable goal-driven policy learning with goals relational graph," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 14096–14105.

[26] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7132–7141.

[27] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*.

Ning Yang received the B.S. degree in automation from Qingdao University, Qingdao, China, in 2022. He is currently working toward the M.S. degree in control engineering with Shandong University, Jinan, China.

His research interests include embodied artificial intelligence, machine learning, and service robot.

Fei Lu received the M.S. degree in system engineering from the Northeastern University, Shenyang, China, in 1997, and the Ph.D. degree in control theory from Shandong University, Jinan, China, in 2007.

She is currently a Professor with the School of Control Science and Engineering, Shandong University. Her research interests include service robots, intelligent space, robot cognition, deep learning, and computer vision.

Guohui Tian (Member, IEEE) received the B.S. degree in control science from the Department of Mathematics and the M.S. degree in automation from the Department of Automation, Shandong University, Jinan, China, in 1990 and 1993, respectively, and the Ph.D. degree in automatic control theory and application from the School of Automation, Northeastern University, Shenyang, China, in 1997.

He is currently a Professor with the School of Control Science and Engineering, Shandong University. His research interests include service robot, intelligent space, cloud robotics, and brain-inspired intelligent robotics.

Jun Liu (Senior Member, IEEE) received the B.Eng. degree in automation and the B.A.Sc. degree in economics from Shandong University, Jinan, China, in 2008, and the Ph.D. degree in mechanical engineering from the University of Toronto, Toronto, Canada, in 2016.

He is currently an Associate Professor with the Department of Data and System Engineering, The University of Hong Kong, Hong Kong, China. He was a Postdoctoral Fellow with Cornell University, New York, USA, from 2017 to 2019, and an Assistant Professor with City University of Hong Kong, Hong Kong, China, from 2019 to 2024. His research interests include micro/nanorobotics, service robotics, and advanced manufacturing.