The following publication S. Liu, C. Huang and H. Huang, "Behavior Cloning-Based Active Scene Recognition via Generated Expert Data With Revision and Prediction for Domestic Robots," in IEEE Transactions on Robotics, vol. 41, pp. 4180-4194, 2025 is available at https://doi.org/10.1109/TRO.2025.3582814.

Behavior Cloning-based Active Scene Recognition via Generated Expert Data with Revision and Prediction for Domestic Robots

Shaopeng Liu, *Member, IEEE*, Chao Huang, *Senior Member, IEEE*, and Hailong Huang, *Senior Member, IEEE*

***Abstract*—Given the limitations of current methods in terms of accuracy and efficiency for robot scene recognition (SR) in domestic environments, this paper proposes an active scene recognition approach (ASR) that allows the robot to recognize scenes correctly using less images, even when the robot's position and observation direction are uncertain. ASR includes a behavior cloning-based action classification model, which can adjust the robot view actively to capture beneficial images for scene recognition. To address the lack of essential expert data for training the action model, we introduce an expert data generation method that avoids time-consuming and inefficient manual data collection. Additionally, we present a multi-view scene recognition method to handle the multiple images resulting from view changes. This method includes a scene recognition model that scores each image and a revision and prediction method to mitigate the compounding error introduced by behavior cloning as well as output the finial recognition result. We conducted numerous comparative experiments and an ablation study in various domestic environments using a publicly simulated platform to validate our ASR method. The experimental results demonstrate that our proposed approach outperforms state-of-the-art methods in terms of both accuracy and efficiency for scene recognition. Furthermore, our method, trained in simulated environments, demonstrates excellent generalization capabilities, allowing it to be directly transferred to the real world without the need for fine-tuning. When deployed on a TurtleBot 4 robot, it achieves precise and efficient scene recognition in diverse real-world environments. The supplements of our method is available at https://sites.google.com/view/asrbc.**

***Index Terms*—Scene recognition, behavior cloning, data generation, robot active vision, domestic robot.**

I. Introduction

FOR domestic robots operating in indoor environments, understanding their surroundings is essential. One crucial task is to assign semantic labels (e.g., "kitchen," "livingroom") to the scenes they encounter. This semantic understanding enhances robots' ability to perform tasks such as semantic mapping [1], [2], [3], [4], object searching [5], [6], [7], [8], and more. Scene recognition (SR), which involves labeling scenes, has thus become a major research focus in both computer vision [9], [10] and robotics [11], [12], [13], [14], [15].

This work was supported in part by the Departmental General Research Fund under Grant P0040253, in part by PolyU Postdoc Matching Fund Scheme under Grand P0046637, and in part by The Hong Kong Polytechnic University. (*Corresponding author: Chao Huang.*)

Shaopeng Liu, and Chao Huang are with Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Hong Kong, China (e-mail: shaopeng.liu@polyu.edu.hk; hchao.huang@polyu.edu.hk).

Hailong Huang is with Department of Aeronautical and Aviation Engineering, The Hong Kong Polytechnic University, Hong Kong, China(e-mail: hailong.huang@polyu.edu.hk)

The common methods for scene recognition often treat it as an image classification problem, relying on deep learning-based models trained to recognize a single scene image [9]. However, this approach can be inadequate in complex family environments. For example, if a robot is tasked with finding milk in a kitchen [6], the robot needs to accurately determine if it is in the kitchen by recognizing various scenes. During this process, the captured image may provide an incomplete observation field due to an unsuitable observation direction or position. Furthermore, the observed scene might differ from the robot's actual location due to scene openness. These issues result in low-quality scene images and scene recognition errors. Some methods that utilize multiple images to achieve a comprehensive observation field can enhance the accuracy of scene recognition [4], [16]. However, these methods depend on a large number of images and additional laser data, compromising efficiency.

In this paper, our focus centers on achieving $accurate$ and $efficient$ robot scene recognition in domestic environments using vision images. By $accurate$, we mean the robot can correctly recognize scenes even when its position and observation direction are uncertain. Simultaneously, we aim for $efficienct$ scene recognition by minimizing the number of images required, thereby reducing the computational load. To address these goals, inspired by active vision [17], we propose a scene recognition method named active scene recognition (ASR), which combines active view adjustment with multi-view scene image recognition. The view adjustment process, akin to an action decision mechanism, aims to capture beneficial observation images for scene recognition. We treat this as an action classification problem, leveraging behavior cloning to enhance accuracy and efficiency. Since ready-made data for training the action classification model is scarce, we introduce an expert data generation method to generate image-action pairs efficiently. We employ ResNet [18] to design the action classification model to ensure that the robot can obtain high-quality multi-view images for scene recognition. To recognize multi-view images and mitigate errors introduced by behavior cloning, we develop a multi-view scene recognition method. This method includes a scene recognition model based on vision transformer [19], which provides scene scores for the multi-view images. Building upon this foundation, a designed revision and prediction method actively adjusts actions based on the observation context, yielding accurate and efficient recognition results. The major contributions of our work are summarized as follows.

1) A novel ASR method is proposed that ensures accurate and efficient scene recognition despite incomplete observations caused by unsuitable robot positions or directions.
2) A behavior cloning-based action classification model is developed that dynamically adjusts the robot's viewpoint for accurate scene recognition, and an expert data production approach is devised to automate training data creation, eliminating the need for manual building.
3) A revision and prediction method is introduced to recognize the multi-view scene images, which can reduce the compounding error and enhance the accuracy and efficiency of scene recognition.
4) Comparative experiments and ablation study are conducted on various domestic environments from a public simulated platform [20]. The results confirm the superior scene recognition performance of our ASR method compared to the state-of-the-art approaches. The ASR method, trained in the simulated environments, can be directly used in the real-world domestic environments without fine-tuning. Real-world experiments validate the effectiveness and transferability of our ASR method.

The remainder is organized as follows. Section II reviews related work. Section III formulates the problem of ASR. Section IV details our proposed methodology. Section V describes the experimental procedures and analyzes the results. Finally, Section VI presents the conclusions.

II. Related work

A. Scene recognition

SR plays a significant role in the field of computer vision [9], [10], [21], [22], [23], [24], [25]. Early scene recognition research relied on manual feature extraction [26], [27]. With the advent and development of deep neural networks such as VGG [28], ResNet [18], densenet [29], etc., learning-based methods have become powerful tools for solving image classification and scene recognition problems. For instance, Mandar $et\ al.$ repurposed a convolutional neural network (CNN) trained to recognize objects for scene recognition [10]. Similarly, Shaopeng $et\ al.$ utilized ResNet-based transfer learning and data augmentation to recognize scene [21]. Strategies based on multi-feature fusion [22] have also been shown to improve the accuracy of scene recognition . Additionally, methods based on metric learning [23] have been developed to address the challenges of scene recognition. Furthermore, advancements in network structures, such as spatial pyramid pooling [24], scale attentive network [25], etc. have further improve the performance of scene recognition.

The progress in scene recognition within the field of computer vision has significantly advanced both outdoor and indoor scene recognition for robots. Outdoor scene recognition, also known as place recognition, is a binary classification problem in which the robot determines if it has previously visited a location based on a captured image [11], [30], [31], [32]. Indoor scene recognition, the focus of this paper, is a multi-class classification problem that aims to label different functional areas (e.g., kitchen, bathroom) using semantic information. In [33], the scene image were represented by region proposals encoded using a CNN-based extractor for scene recognition. Ricardo $et\ al.$ proposed a semantic inter-object relationship method for scene recognition based on the distance relationships between recognized objects within a scene image [34]. Bayesian reasoning has also been employed to achieve scene recognition based on object relationships [12], [14]. Similarly, an Object-to-Scene approach [35] was presented to extract object features and learn object relations for scene recognition. Bo $et\ al.$ constructed an attention mechanism based on global and object attribute fusion to recognize indoor scene [15]. These methods primarily utilized single image to address indoor scene recognition. Another approach involved using multiple images for indoor scene recognition, often combined with semantic mapping [3]. Niko $et\ al.$ employed CNN to recognize abundant images captured during mapping and used a Bayesian filtering over the class labels of these images for scene recognition [16]. Based on the Bayesian fusion, Jose $et\ al.$ designed an approach to select the informative views for scene recognition, requiring not only plenty of images but also the extra information (geometric knowledge, semantic knowledge, and laser data) [4].

Overall, due to low-quality images caused by suboptimal observation positions or directions, most scene recognition methods relying on a single image in complex domestic environments suffer from accuracy issues. These images provide insufficient information to accurately distinguish the correct scene category. Although methods based on multiple images are more likely to obtain valuable information for scene recognition, they require an excessive number of images, leading to a reduced efficiency and unnecessary use of the robot's computing resources. Therefore, we propose an ASR method to enhance the accuracy and efficiency of scene recognition.

B. Robot active vision & behavior cloning

Active vision involves the deliberate control of a robot or sensor to perform specific tasks [17]. For instance, active visual categorization entails selecting optimal viewpoints to recognize objects or scenes [36], [37], [38]. Xiaodong $et\ al.$ proposed a method also named active scene recognition, however, which mainly focused on single image recognition and needed extra high level knowledge [36]. Dinesh $et\ al.$ developed a view-selection strategy among spherical panoramas [37], [38], trained on a panoramic image dataset SUN360 [39]. However, since most scenes in SUN360 are outdoor environments, they significantly differ from indoor domestic scenes. The action policy learned in SUN360 is not directly applicable to solving the domestic scene recognition problem. Therefore, an active vision-based method that can be directly applied to robot scene recognition in domestic environments needs to be developed.

Furthermore, the majority of the aforementioned studies have concentrated on reinforcement learning [37], [38]. In this approach, an agent is trained within a simulated environment by experimenting with various actions and receiving either positive or negative feedback. This method necessitates extensive exploration of actions to develop an appropriate strategy,

leading to a time-consuming and inefficient training process. In contrast, behavior cloning, where desired behaviors are learned by imitating an expert's demonstrations [40], offers advantages in training efficiency. It has been successfully applied in autonomous driving [41], [42], robotic manipulation [43], [44], [45], and so on. Thus, it is a promising approach to utilize behavior cloning to address the problem of ASR.

Despite its advantages, the behavior cloning method has two main drawbacks: (1) the need for high-quality expert data [46], and (2) prone to the compounding error caused by the covariate shift problem [47]. Due to the lack of available expert data for the ASR method and the inefficiency of manually generating sufficient demonstrations, efficiently generating expert data is essential. To address the compounding error, one common solution is dataset aggregation (Dagger) [48], which involves querying an expert and iterative training. However, this approach can reduce training efficiency.

Overall, the limitations mentioned above motivate us to develop an ASR method based on behavior cloning. We propose a solution to automatically generate abundant high-quality expert data, leveraging the advantages of behavior cloning. Additionally, we introduce a revision and prediction method to reduce the compounding error without requiring extra query or training, thereby improving the efficiency and accuracy.

III. Problem Formulation

The process of ASR is represented by $(S, A, \pi^a, T, I, \pi^s)$. Each element is defined as follows.

- $S = \{s_1, s_2, ..., s_t\}$ refers to the state space. The robot captures a scene image i_t at the state s_t during a scene recognition task, where t expresses different times.
- $A = \{a_1, a_2, ..., a_k\}$ is the action space, including k types of actions. Actions with similar ranges, such as clockwise rotations of 10° and 15°, result in minimal perspective changes. Therefore, A is discretized into k types of actions, such as moving forward 30cm, clockwise rotation 30°, etc.
- π^a, termed the action strategy

$$a_t = \pi^a(s_t), \tag{1}$$

  guides the robot to take an action a_t from s_t to s_{t+1} to capture a more beneficial observation for scene recognition.
- T is a state transition function defined by

$$s_{t+1} = T(s_t, a_t) = T(s_t, \pi^a(s_t)), \tag{2}$$

  where the robot executes an action a_t from A in the current state s_t to the next state s_{t+1}.
- $I = \{i_1, i_2, ..., i_t\}$ is scene image sequence in a scene recognition task. Given Eq. (2), I can be expressed as

$$I = \{s_t \rightarrow i_t | T(s_{t-1}, \pi^a(s_{t-1}))\}. \tag{3}$$

- π^s denotes the scene recognition strategy, mapping from I to scene labels Y, defined by

$$Y = \pi^s(I) = \pi^s(\{s_t \rightarrow i_t | T(s_{t-1}, \pi^a(s_{t-1}))\}). \tag{4}$$

The goal of ASR is to get high accuracy with fewer actions (as few images as possible), which can be accomplished by designing effective strategies π^a and π^s.

Following behavior cloning, π^a can be treated as a classification problem solved by an action classification model based on deep neural network f^a_θ as a nonlinear approximation

$$a_t = f^a_\theta(s_t) \rightarrow \pi^a(s_t), \tag{5}$$

where θ represents the parameters trained using expert data $\mathbb{D} = \{(s_1, a_1), (s_2, a_2), ..., (s_H, a_H)\}$. $\mathbb{D}$ includes abundant data pairs $(s_H, a_H) \rightarrow (i_H, a_H)$.

Since the number of actions (defined as l) required to complete a scene recognition task is not fixed, the length of I varies. We use a deep neural network f^s_β, termed scene recognition model, to recognize each scene image i_t in I by

$$y_t = f^s_\beta(i_t), \tag{6}$$

where β denotes the training parameters. The recognition results y_{i_t} of each i_t are then processed by a designed prediction algorithm F. Consequently, F combined with f^s_β as a nonlinear approximation of $\pi^s(I)$ can be expressed as

$$Y = F(\{y_{i_t} | f^s_\beta(i_t)\}) \rightarrow \pi^s(I). \tag{7}$$

The compounding error is inevitable when develop behavior cloning-based methods, which increase l and reduce scene recognition accuracy. Therefore, we address the compounding error in two ways. First, f^a_θ can be optimized using a proposed action revision strategy R^a, which outputs a more appropriate action a^*_t according to $f^a_\theta(s_t)$ and the previous action a_{t-1}

$$a^*_t = R^a(f^a_\theta(s_t), a_{t-1}). \tag{8}$$

Second, when recognizing i_t, the recognition results of the last scene image $f^s_\beta(i_{t-1})$ are considered to improve f^s_β by a designed scene recognition revision method R^s expressed as

$$y^*_t = R^s(f^s_\beta(i_t), f^s_\beta(i_{t-1})). \tag{9}$$

Ultimately, this paper aims to enhance the accuracy of robot scene recognition and minimize the number of actions by addressing three key issues: 1) efficiently generating $\mathbb{D}$, 2) designing f^a_θ, f^s_β, and F, 3) proposing R^a and R^s.

IV. Proposed Approach

A. Expert Data Generation Method

AVDB dataset [49] collects dense scene images scanned from various domestic environments. It is highly suitable for extracting data pairs due to the action relationships between the images. As shown in Fig. 1, we propose an expert data generation method to efficiently create abundant data pairs from AVDB for training f^a_θ.

For any image in AVDB, we need to determine whether it is a well-observed image, meaning the angle and position of the image can support accurate scene recognition. As shown in Fig. 1 (step 1), to avoid inefficient and extensive manual judgments, we design a feature extraction model to make decisions by learning features of the well-observed images

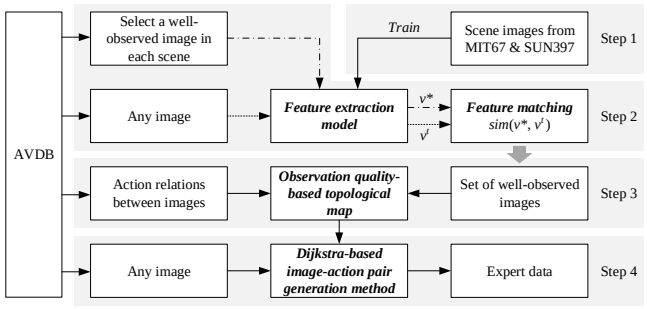

Fig. 1. The flow of expert data generation method including four steps.

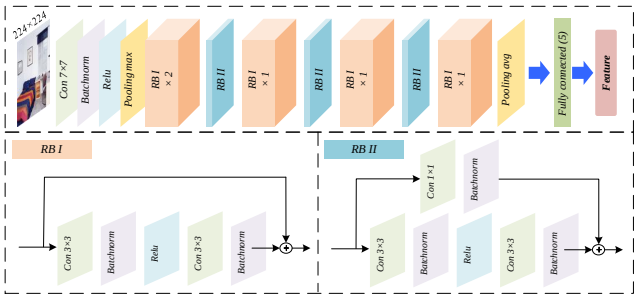

Fig. 2. The architecture of feature extraction model based on two kinds of residual blocks (RB).

from public scene datasets. As demonstrated in Fig. 2, the architecture of the feature extraction model is based on an 18-layer ResNet [18] (ResNet-18), containing two types of residual blocks (RB). ResNet-18 is pre-trained on ImageNet dataset [50] with a top-1 accuracy of 69.758%. Since AVDB contains five types of domestic scenes (bathroom, bedroom, kitchen, livingroom, and dining room), the number of neurons in the fully connected layer is set to five. These five classes of scene images from MIT-67 [51] and SUN397 [52] are used to train the feature extraction model. After training, the output of the fully connected layer is taken as the feature vector.

Step 2 in Fig. 1 is to generate a set of well-observed images. First, the images collected in a domestic environment are divided according to the scene where the images are located. For the images in a scene, we only need to manually select one well-observed image that includes the whole vision information of this scene. This image, as the matching standard, is input into the feature extraction model to get a feature vector v^*. Then, input any image from this scene into the feature extraction model to obtain a feature vector v^t. Cosine similarity

$$sim(v^*, v^t) = \frac{v^* \cdot v^t}{||v^*||||v^t||}, \tag{10}$$

is used to match features. If $sim > 0.99$, the input image is considered well-observed. Else, the input image is abandoned. Repeat these processes to obtain a set of well-observed images.

Step 3 in Fig. 1 is to build an observation quality-based topological map of each scene. As shown in Fig. 3(a), annotations in AVDB clarify the six types of action relations between images. For example, the robot can take a $rotate\_ccw$ action from i_1 to i_2. Based on the annotations, a topological map of each scene can be constructed, with images serving as nodes and action relations as edges. The color of each node [see

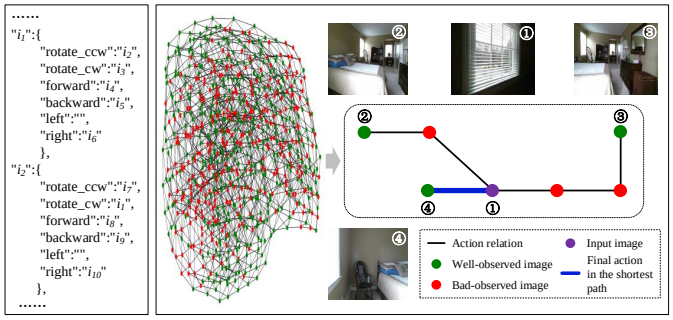

Fig. 3. The six types of action relations between images in annotations of AVDB: counterclockwise rotation 30° ($rotate\_ccw$), clockwise rotation 30° ($rotate\_cw$), forward 30cm, backward 30cm, left 30cm, and right 30cm (a), and an example of image-action pair generation using the topological map based on observation quality (b).

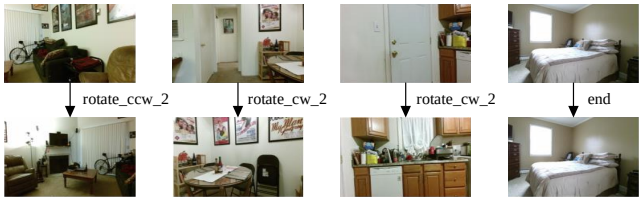

Fig. 4. Some visualized results of generated image-action pairs: first line is i_s, and bottom line is well-observed image after taking the middle $a^*_{i_s}$.

in Fig. 3(b)], derived from the set of well-observed images, indicates the quality of observation.

In step 4 in Fig. 1, for a scene image, we present a Dijkstra-based method (outlined in Algorithm 1) to determine its image-action pair. To improve the action efficiency, two additional action types $rotate\_cw\_2$ and $rotate\_ccw\_2$ are added to shorten the path (see lines 8-10 in Algorithm 1). Considering that the visual changes produced by actions $right$ and $left$ are similar to those of actions $rotate\_cw$ and $rotate\_ccw$, the number of action categories is reduced by using $rotate\_cw$ and $rotate\_ccw$ instead of $right$ and $left$ (see lines 18-23 in Algorithm 1). An example of image-action pair generation is provided in Fig. 3(b). For the input image ①, firstly use Algorithm 1 to obtain the paths between it and the well-observed images in the observation quality-based topological map. Then select the shortest one from these paths, which is the path ① - ④. The action relation ($rotate\_cw\_2$) between image ① and image ④ is to form the image-action pair $< image$ ①$, rotate_cw_2 >$. Finally, the expert data generated by Algorithm 1 contains 6,933 image-action pairs, some of which are visualized in Fig. 4.

B. Action Classification Model

An 18-layer ResNet [18] is utilized to build the action classification model (ACM), as illustrated in Fig.5. The purpose of ACM is to output appropriate actions to adjust the view for beneficial observation by learning from the expert data. The input of ACM is a 224×224 RGB image from the image-action pair. The architecture of ACM is similar to the feature extraction model, containing two types of residual blocks. Given that there are seven classes of actions in the image-action pairs: $forward$ $(30cm)$,

Algorithm 1 Image-action pair generation method

Input: Any image in current scene i_s, set of well-observed images in current scene I_s^*, annotations of current scene An, observation quality based topological map of current scene M_s.

Output: The image-action pair $(i_s, a_{i_s}^*)$ of i_s.

if i_s in I_s^* **then**
  $a_{i_s}^* = end$
else
  $l_p = 10$ // initialize variate of last path length
  for i_s^t in I_s^* **do**
    Using Dijkstra algorithm in M_s to get the shortest path $P_i(i_s, i_s^t) = [i_s, i_s^1, i_s^2, ..., i_s^t]$ from i_s to i_s^t.
    According to An to find the action path $P_a(i_s, i_s^t) = [a_{i_s}, a_{i_s^1}, a_{i_s^2}, ..., a_{i_s^t}]$ of $P_i(i_s, i_s^t)$.
    if adjacent two $rotate\_cw$ or $rotate\_ccw$ in $P_a(i_s, i_s^t)$ **then**
      Define two $rotate\_cw$ as $rotate\_cw\_2$ and $rotate\_ccw$ as $rotate\_ccw\_2$ to shorten $P_a(i_s, i_s^t)$
    end if
    Get the length l_{P_a} of $P_a(i_s, i_s^t)$.
    if $l_{P_a} < l_p$ **then**
      $l_p = l_{P_a}$
      The first action of $P_a(i_s, i_s^t)$ is $a_{i_s}^*$
    end if
  end for
end if
if $a_{i_s}^*$ is $right$ **then**
  $a_{i_s}^* = rotate\_cw$
end if
if $a_{i_s}^*$ is $left$ **then**
  $a_{i_s}^* = rotate\_ccw$
end if
Return $(i_s, a_{i_s}^*)$

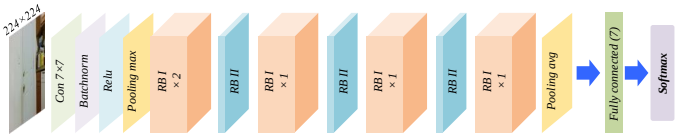

Fig. 5. The architecture of action classification model based on two kinds of residual blocks (RB).

backward $(30cm)$, *clockwise rotation* $(rotate\_cw)$ $(30°)$, *counterclockwise rotation* $(rotate\_ccw)$ $(30°)$, *clockwise rotation* $(rotate\_cw\_2)$ $(60°)$, *counterclockwise rotation* $(rotate\_ccw\_2)$ $(60°)$, and *end*, the fully connected layer in ACM has seven neurons. The weights of the layers before the fully connected layer in ACM are transferred from a pre-trained ResNet on the ImageNet dataset [50] with a top-1 accuracy of 69.758%. Finally, a softmax layer is added to predict actions.

The generated image-action pairs are used to train ACM with a loss function *Cross Entropy Loss*:

$$loss(x, |C|) = -log(\frac{e^{x_{|C|}}}{\sum_j e^{x_j}}), \tag{11}$$

where x is the predicted value of ACM and $|C|$ is the class number. The predicted results of ACM are scores of each action category.

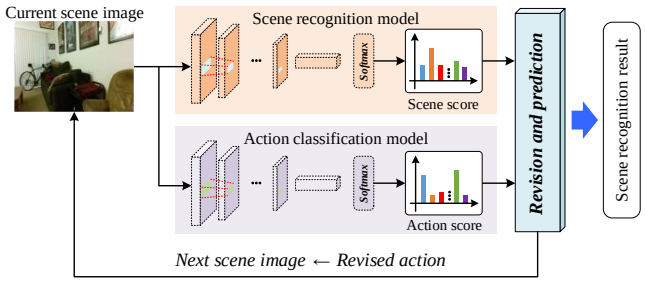

Fig. 6. The pipeline of multi-view scene recognition method, including scene recognition model, action classification model, and revision and prediction method.

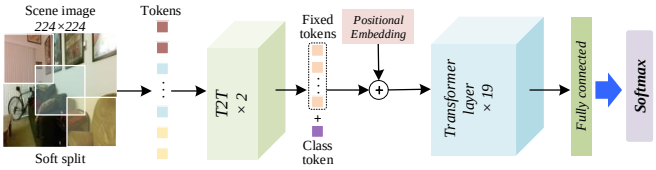

Fig. 7. The architecture of scene recognition model based on T2T-ViT-19, containing two tokens-to-token modules (T2T) and 19 transformer layers.

C. Multi-view scene recognition method

As illustrated in Fig. 6, the multi-view scene recognition method is proposed to handle scene images from various views to predict the scene category. This method includes three parts: ACM, scene recognition model (SRM), and revision and prediction method (RPM). During the recognition process, each scene image is input into both SRM and ACM to generate a scene score and an action score, respectively. Based on these scores, RPM determines whether to continue the action for next scene image or stop and output the recognition result.

Given the excellent performance of Tokens-to-Token Vision Transformer (T2T-ViT) [19] on image recognition, we utilize a T2T-ViT model with 19 transformer layers (T2T-ViT-19) to build SRM. The architecture of SRM is shown in Fig. 7. The input of SRM is a 224×224 RGB scene image, which is softly split into patches. These patches are unfolded as a list of tokens, which are then reduced by two tokens-to-token modules (T2T) to form fixed tokens. The fixed tokens, along with an added class token and position embedding, are extracted features by 19 transformer layers and a fully connected layer. Finally a softmax layer outputs the scene score. During the training, the weights of the layers before the fully connected layer in SRM are transferred from a pre-trained T2T-ViT-19 on the ImageNet dataset [50] with a top-1 accuracy of 81.9%. The loss function is *Cross Entropy Loss* defined in Eq. (11). The output of SRM is the scene score.

RPM, outlined in Algorithm 2, is introduced to mitigate the compounding error of ACM and predict the scene recognition result more accurately and efficiently. Firstly, the revision strategy in RPM consists of two parts: (1) revision based on action score (refer to lines 7-18 in Algorithm 2), and (2) revision based on scene score (see lines 26-29 in Algorithm 2). When ACM predicts an action for the first time ($step = 1$), if the action with the highest score is the opposite of the action

TABLE I
THE RELATIONSHIP BETWEEN ACTION AND ITS OPPOSITE ACTION

Action		Opposite action
$forward$	$\leftrightarrows$	$backward$
$rotate\_cw, rotate\_cw\_2$	$\leftrightarrows$	$rotate\_ccw, rotate\_ccw\_2$

with the second highest score and the two scores are similar (judged by line 10 in Algorithm 2), the output action of ACM will be end. When $step > 1$, the current action is revised by the last action. If the current action is opposite to the last action, the current action is changed to end. In addition, the scene recognition results of the adjacent two images can also revise the action. As list in lines 26-29 in Algorithm 2, if the scene recognition results of the current image and last image are same and their scene scores are greater than 0.99, the action will be halted.

For scene prediction (refer to lines 21-25 in Algorithm 2), the scene scores of the adjacent two images are fused according to the same scene category. The max value in each scene category is retained to form a list of max scene scores, from which the final result is obtained (see in line 35 in Algorithm 2). The action score threshold 0.01 and the scene score threshold 0.99 in Algorithm 2 are empirical values.

V. EXPERIMENTS

The proposed method is compared with different kinds of scene recognition methods in some simulated domestic environments to prove its state-of-the-art performance. Subsequently, we apply our method to a TurtleBot 4 in real-world experiments to showcase its practical effectiveness. For validation purposes, we provide detailed descriptions of the experimental setup and result analysis in this section.

A. Simulated Environments and Experimental Setup

We conduct simulated experiments on a computer equipped with an Intel Core i7-13700K CPU, 64 GB of memory, 1 TB SSD, and a NVIDIA GeForce RTX 4060 Ti GPU. The operating system is Ubuntu 18.04. Our programs and algorithms are implemented using Python 3.9 and the PyTorch 2.2.1 deep learning framework

The simulated environments is derived from Habitat-Matterport 3D Research Dataset (HM3D) [20], which comprises high-resolution 3D scans of real-world indoor spaces of domestic environments [shown in Fig. 8(a)]. HM3D can simulate a robot to move and rotate to capture images for scene recognition. Fig. 8(b) illustrates our placement of dense scene recognition points in one of the simulated environments. At each scene recognition point, we define four scene recognition tasks from different directions, with a 90-degree angle between the adjacent tasks. This deliberate task arrangement allows us to robustly evaluate the scene recognition method, when the robot's observation position and direction are uncertain. We meticulously label 10,036 scene recognition tasks across 15 simulated domestic environments in HM3D to assess various scene recognition methods. Refer to Fig. 9 for the task distribution in each environment. Crucially, the testing environments are different from those used to generate the expert data, providing compelling evidence for validating the generalization of our proposed method.

Algorithm 2 RPM: The revision and prediction method

Input: The action classification model $f^a_\theta(i_t)$, the scene recognition model $f^s_\beta(i_t)$, and the current scene image i_t.
Output: The scene recognition result y_{i_t}.

1: $step = 0$ // action count
2: $f^s_\beta(i_t) = [y^{c^s_1}_{i_t}, y^{c^s_2}_{i_t}, ..., y^{|c^s|}_{i_t}]$ // the scores of various scene labels c^s in descending order.
3: $c^s_{i_t} = argmax(f^s_\beta(i_t))$
4: **while** a_{i_t} is not end **do**
5: $step = step + 1$
6: $f^a_\theta(i_t) = [x^{c^a_1}_{i_t}, x^{c^a_2}_{i_t}, ..., x^{|c_a|}_{i_t}]$ // the scores of various action labels c_a in descending order.
7: **if** $step = 1$ **then**
8: $d_a = (x^{c^a_1}_{i_t} - x^{c^a_2}_{i_t})/x^{c^a_2}_{i_t}$
9: get the opposite action $c^{a\prime}_2$ of c^a_2 from Table I.
10: **if** c^a_1 is $c^{a\prime}_2$ and $d_a \leqslant 0.01$ **then**
11: $c^a_1 = end$
12: **end if**
13: **else**
14: Get the opposite action a' of the last action a^* from Table I.
15: **if** c^a_1 is a' **then**
16: $c^a_1 = end$
17: **end if**
18: **end if**
19: $a_{i_t} = c^a_1$ // the output action at i_t.
20: $i_{t+1} = T(i_t, a_{i_t})$ // perform a_{i_t} to obtain next scene image i_{t+1}.
21: $f^s_\beta(i_{t+1}) = [y^{c^s_1}_{i_{t+1}}, y^{c^s_2}_{i_{t+1}}, ..., y^{|c^s|}_{i_{t+1}}]$ // the scores of various scene labels c^s in descending order.
22: $c^s_{i_{t+1}} = argmax(f^s_\beta(i_{t+1}))$
23: $[(y^{c^s_1}_{i_t}, y^{c^s_1}_{i_{t+1}}), (y^{c^s_2}_{i_t}, y^{c^s_2}_{i_{t+1}}), ..., (y^{|c^s|}_{i_t}, y^{|c^s|}_{i_{t+1}})]$ // fuse $f^s_\beta(i_t)$ and $f^s_\beta(i_{t+1})$ according the same scene category
24: $y^{|c^s|}_{max} = max(y^{|c^s|}_{i_t}, y^{|c^s|}_{i_{t+1}})$ // the max value of each scene category
25: $F(i_t, i_{t+1}) = [y^{c^s_1}_{max}, y^{c^s_2}_{max}, ..., y^{|c^s|}_{max}]$
26: **if** $step > 1$ and $c^s_{i_t} = c^s_{i_{t+1}}$ and $y^{c^s_1}_{i_t} > 0.99$ and $y^{c^s_1}_{i_{t+1}} > 0.99$ **then**
27: $a_{i_t} = end$
28: $F(i_t, i_{t+1}) = f^s_\beta(i_{t+1})$
29: **end if**
30: $a^* = a_{i_t}, i_t = i_{t+1}$
31: **end while**
32: **if** $step = 1$ **then**
33: $y_{i_t} = c^s_{i_t}$
34: **else**
35: $y_{i_t} = argmax(F(i_t, i_{t+1}))$
36: **end if**
37: Return y_{s_t}

We employ two evaluation metrics to assess scene recognition performance: commonly used accuracy (acc) [4], [16] and average action number (aan) that indicates the efficiency of scene recognition. The acc is calculated as

$$acc = \frac{n_r}{n_{all}}, \tag{12}$$

where n_r represents the number of tasks with correct recogni-

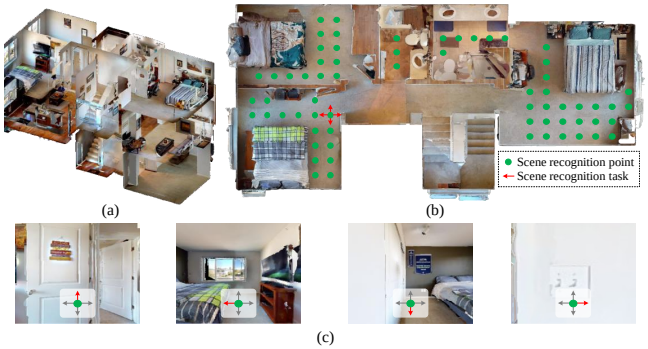

Fig. 8. An example of the simulated domestic environments: (a) a dollhouse view of the scanned 3D environment from HM3D, (b) a topdown view of one floor labeled with dense scene recognition points and tasks, and (c) four scene images from different directions [red arrows in (b)] in the same scene recognition point representing various scene recognition tasks.

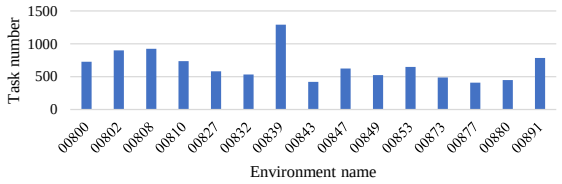

Fig. 9. The number of scene recognition tasks of each simulated domestic environment.

tion results, and n_{all} is the total number of scene recognition tasks. The aan is defined as

$$aan = \frac{\sum_{i=1}^{n_r} Num(T_i^{sr})}{n_r}, \tag{13}$$

where $Num(T_i^{sr})$ denotes the action count for a correct scene recognition task T_i^{sr}. A smaller aan indicates a more efficient scene recognition method that requires fewer images to achieve accurate results.

B. Baseline Methods and Implementation Details

We employ three classes of scene recognition methods as baselines for comparative assessment against our method.

- Single image-based method (SIM): This common approach for scene recognition involves training deep neural network models on scene datasets to recognize scenes. The input of SIM is a single RGB scene image. We use four CNN-based scene recognition models (VGG [28], ResNet [18], densenet121 [29], and mobilenet [29]) and two vision transformer based classification networks (ViT [53] and T2TViT [19]) as SIM for scene recognition tasks. Those models are pre-trained on the ImageNet dataset [50] and then fine-tuned on 8,711 domestic scene images from two public scene datasets MIT-67 [51] and SUN397 [52], using transfer learning. The training parameters are detailed in Table II. During training, each model is saved once after every epoch. Ultimately, each SIM produces eleven well-trained models. We test all these models on the scene recognition tasks and chose the one with the highest accuracy to contrast with our method.

TABLE II
THE TRAINING PARAMETERS OF SIM AND ASR.

Model	SIM	ASR	
		ACM	SRM
Image size	224×224		
Batch size	32	32	32
Learning rate	3^{-5}	3^{-5}	3^{-5}
Optimizer	$Adam$		

- Multi-image-based method (MIM): We consider two state-of-the-art methods (PC [16] and PCS [4]) that utilize multiple images for scene recognition. In PC, the fixed interval images captured during the robot moving are recognized using an AlexNet-based recognition network. The recognition result of each image is then processed by a Bayesian filter to determine the scene category. For a fair comparison, we replace the original AlexNet in PC with the same SRM proposed in our method. Based on PC, PCS adds a view selection strategy that allows the robot to change its view during movement for scene recognition. Note that the original view selection strategy in [4] requires extra laser data. To maintain consistency in input data, we use our ACM as an alternative. We set various quantitative limitations n of actions in each task for PC and PCS, expressed as PC_n and PCS_n. For example, PC_n means that PC utilizes $n+1$ images to finish a scene recognition task.
- Vision language model based zero-shot method (VLM): The well-trained VLMs can obtain the semantic information of the image including the implicit scene information. So we leverage two VLM-based zero-shot methods (VLFM [54] and LFG [55]) to finish the scene recognition tasks. In VLFM, a pre-trained VLM BLIP-2 [56] with the input of the observed RGB scene image and a text prompt of a scene label (e.g. "kitchen") generates a cosine similarity score. The scene label with the largest score is the result of scene recognition. LFG uses language guide to recognize scene. First, the observed RGB scene image with a query text `"Question: Which action can be chosen from clockwise rotation, counterclockwise rotation, forward, backward, and stop to adjust the viewing angle for better recognition of the scene category in this image? Answer:"` is input in a pre-trained BLIP-2 [56] to output an action to change view. Then, the new observed RGB scene image with a text prompt `"Question: Which of the following scenes does this picture belong to? Bathroom, bedroom, dining_room, kitchen, or livingroom? Answer:"` is fed in the BLIP-2 to output the scene recognition result.
- Our proposed method (Ours): ASR comprises two trained models: ACM and SRM. ACM is trained on the generated expert data, while SRM is trained on the same scene images as those used for SIM. The training parameters of ACM and SRM are specified in Table II. We train ACM five times with different random seeds (40, 41, 42, 43, and 44), and the testing curve for each

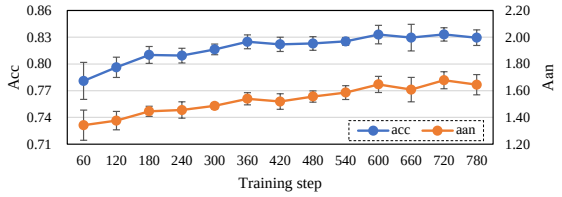

Fig. 10. The testing curve of the saved ACM using various random seeds during training, where the lines denote the average values and the error bar refers to the standard deviation.

saved model is shown in Fig. 10. As the number of training step increases, the acc exhibits an upward trend, indicating that ACM gradually learns an action strategy to obtain beneficial observations for scene recognition. Additionally, there is a positive correlation between aan and acc. The aan eventually stabilizes around 1.7. When using a training seed of 42, we select the model saved at step 660, which achieved the highest acc (0.8560) and a competitive aan (1.7346), as the well-trained ACM tested in the experiments.

C. Quantitative Results and Analysis

The testing results of the baseline methods and ours are provided in Table III. Among SIM, CNN-based models achieve an accuracy of less than 0.61, which is lower than that of vision transformer-based models. T2TViT, with the highest accuracy in SIM, demonstrates its strong recognition ability for single scene image. This observation supports the rationale behind designing our multi-view scene recognition model based on T2TViT. The lower accuracy of SIM compared to MIM suggests that obtaining accurate scene recognition in a complex home environment with only one image is challenging, especially when the robot's observation location and direction are uncertain.

Using multiple images improves scene recognition performance (MIM accuracy exceeds 0.7). PCS significantly improves upon PC, highlighting the importance of view selection for better scene recognition performance. As a result, PCS_6 achieves a leading acc of 0.7980 among the baseline methods. It can be found in Table III that there is a weak positive correlation between the number of recognized images and accuracy. From two images to six images, both PC and PCS show slight improvements in accuracy. However, simply increasing the number of the recognized images does not significantly enhance the accuracy and but can increase computational load.

Different from SIM and MIM, VLFM can achieve scene recognition with 0.6614 acc without training. LFG makes use of the inference ability of the VLM for view selection, but only obtains a low acc 0.4934. This shows that VLM can not be directly applied to view selection for accurate scene recognition. Therefore, the zero-shot based VLM methods cannot obtain the scene recognition results similar to ours.

Compared to ours, the state-of-the-art methods have lower accuracy and efficiency. Ours achieves the highest accuracy of 0.8560 and a small aan (only 1.7346), which indicates that the proposed method enables accurate scene recognition with fewer images and actions, avoiding computational overhead from redundant image recognition and saving energy consumption by reducing the number of actions.

TABLE III
THE acc AND aan OF THE BASELINE METHODS AND OURS IN THE SIMULATED ENVIRONMENTS.

Type	Model	acc	aan
Single image-based method (SIM)	VGG11 [28]	0.5545	-
	ResNet18 [18]	0.6097	-
	densenet121 [29]	0.5987	-
	mobilenet [57]	0.5812	-
	ViT [53]	0.6635	-
	T2TViT [19]	0.6836	-
Multi-image-based method (MIM)	PC_2 [16]	0.7060	2
	PC_4 [16]	0.7094	4
	PC_6 [16]	0.7105	6
	PCS_2 [4]	0.7928	2
	PCS_4 [4]	0.7929	4
	PCS_6 [4]	0.7980	6
Vision language model based zero-shot method (VLM)	VLFM [54]	0.6614	-
	LFG [55]	0.4934	0.5979
	Ours	**0.8560**	1.7346

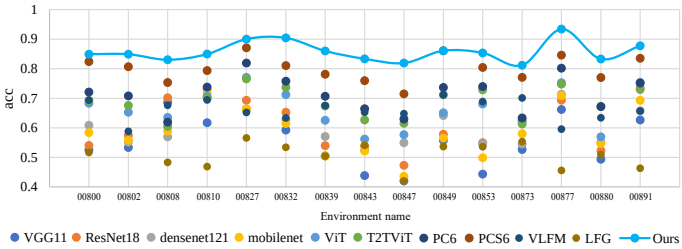

Fig. 11. The acc of baseline methods and ours in each testing environment.

To compare scene recognition performance across different environments, the acc of each testing environment using both baseline methods and ours is recorded in Fig. 11. It can be seen that ours consistently achieves an acc greater than 0.8 in each environment, surpassing the baseline methods. In contrast, the acc of the comparison methods exhibits significant fluctuations across different environments. Specifically, for the SIM approach, T2TViT achieves an acc close to 0.8 in environment 00827 but only around 0.6 in environment 00847. Meanwhile, MIM shows overall improvements in acc across environments, but substantial variations persist. For example, the second-ranked PC_6 has 0.88 acc in environment 00827, whereas it drops to approximately 0.7 acc in environment 00847. Therefore, our proposed method consistently outperforms the baseline approach in diverse domestic environments, showing its well adaptive capacity of scene recognition across different domestic environments.

D. Qualitative Results and Analysis

To comprehensively demonstrate the superiority of our method over the baseline approaches, we present detailed visualizations of six scene recognition tasks. In Fig. 12, we depict the scene recognition locations and observation directions of these tasks. The right side of Fig. 12 showcases the process of accomplishing these tasks using our method, including the observed images, output actions, and scene scores. We display the performed actions and observed images related to PC_6 and PCS_6 in Fig. 13. The scene scores of each task, obtained using both baseline methods (SIM and MIM) and our proposed approach, are presented in Fig. 14. The scene category with the highest score represents the recognition result. The scene

Fig. 12. Six scene recognition tasks from the various environments. In every task, the scene recognition location (green point) and the observation direction (red arrow) are labeled on the topdown view of floor (left). During the ASR process, the observed images (the first image is the initial image), output actions, and scene score of each captured image by our method are shown in the right.

recognition results of VLFM and LFG in the six tasks are demonstrated in Fig. 15.

The ground truth of task ① is kitchen. Initially, the observation direction does not point toward the kitchen, resulting in the captured image lacking kitchen-related visual information. All SIM and VLFM produce the incorrect recognition results due to this limitation. To solve this challenging task, our method applies the $rotate\_cw\_2$ action three times to change the observation direction to capture representative scene images of kitchen (see in Fig. 12) and then recognizes the scene correctly based on those multi-view images. Notably, the moving actions $left$ do not provide useful observation information or yield correct recognition results for PC_6 and PCS_6 (see Fig. 13 and 14). Similarly, an inappropriate action $forward$ does not help LFG get the correct result in Fig. 15.

The difficulty of scene recognition increases when the observations are obscured. For example, at the beginning of task ②, the robot's proximity to the kitchen boundary and the obstructed view (due to the refrigerator) pose the errors of SIM in Fig. 14 and the failure of VLFM in Fig. 15. In this case, as demonstrated in Fig. 12 and Fig. 14, our method adjusts the view by two $rotate\_cw\_2$ actions to capture a complete observation of the kitchen and outputs a correct result. Similarly, under the guidance of reasonable actions ($right$) in Fig. 13, PC_6 and PCS_6 also obtain six beneficial images to correctly identify the scene. In this task, LFG outputs a $rotate\_cw$ action and obtains the right result (see in Fig. 15).

In task ③, the observed scene (livingroom) differs from the actual scene (dining room) where the robot is located, casing the scene recognition errors of SIM, PC_6, PCS_6, VLFM, and LFG. By comparison, our method guides the robot to capture the images of the dining room by $rotate\_cw\_2$ and $rotate\_cw$ in Fig. 12 and gives a correct result (dining room) in Fig. 14.

In certain scenes, identical objects may appear. For example, a table lamp and a green plant can be found in both livingroom and bedroom. As exhibited in Fig. 12 and Fig. 14, when the observed images only contain these common objects, SIM and PC_6 yield incorrect recognition results in task ④ and task ⑤. PCS_6 and our method can change the view to obtain additional visual information, allowing them to accurately determine the scene category. Notably, our method achieves correct recognition results with small number of required actions and images in task ④ and task ⑤. The performance of VLFM and LFG is unstable, leading to failure in task ④ and success in task ⑤.

Our approach does not always necessitate multi-image for recognition. In cases where the initial observation suffices—such as in task ⑥—our method directly recognizes the scene from the initial image using a stopping action end. However, PC_6 and PCS_6 still rely on redundant actions and images for scene recognition, highlighting their inefficiency.

In summary, scene recognition tasks in domestic environments pose complexity and difficulty. When observations are suboptimal, relying solely on a single image often fails to yield accurate results. While PC improves recognition performance to some extent, it overly depends on the moving actions. The key to enhancing scene recognition lies in selecting beneficial images through guided actions. Although PCS demonstrates perspective selection capabilities, it still suffers from redundancy in actions and images, resulting in inefficient recognition. In addition, VLM have unstable scene recognition performance on different tasks, further demonstrating the strength and necessity of our proposed method. In contrast, our method intelligently determines view adjustments based on the observation context, achieving accurate scene recognition with fewer actions and images and outperforming the baselines.

E. Ablation Study

We conduct an ablation study to discuss the rationality of the proposed ACM, RPM, and model architecture.

1) ACM: ACM is designed to output actions to adjust the view for well-observed scene images. To assess the efficacy of ACM, two action strategies (no action and random action) are used to compare with ACM. 'No action' means that the ASR method recognizes the scene only using the initial image, without any view-changing actions. 'Random action' signifies

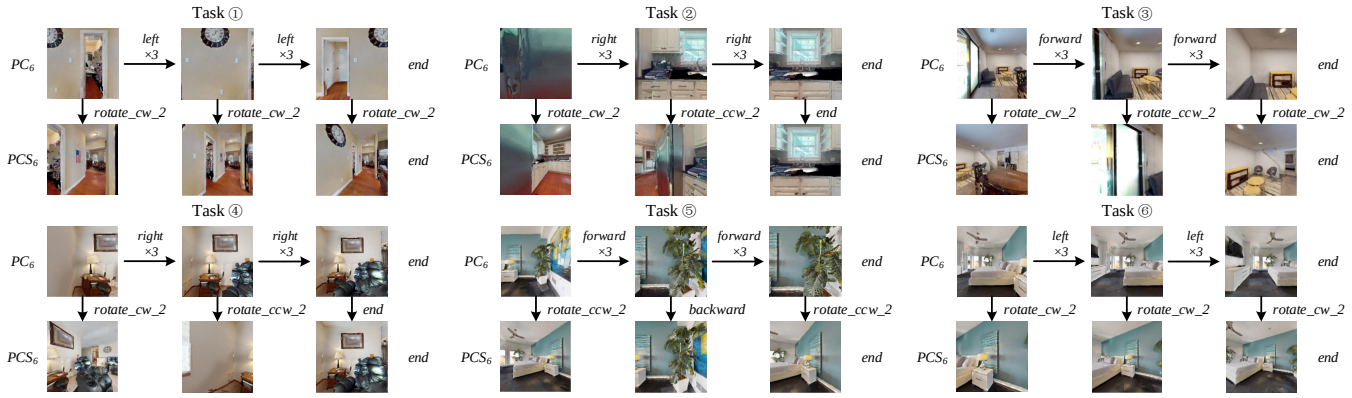

Fig. 13. The performed actions and observed images of PC_6 and PCS_6 in each task. The actions in PC_6 guides the robot to obtain multi-image for scene recognition. On this basis, PCS_6 outputs an extra action by ACM to adjust the observation direction after a guided action.

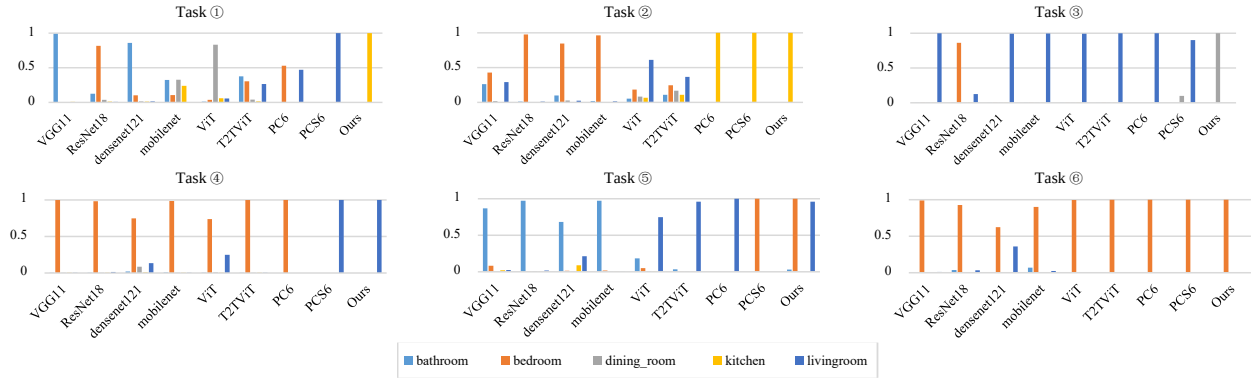

Fig. 14. The scene scores of the baseline methods (SIM and MIM) and our proposed method in the six tasks.

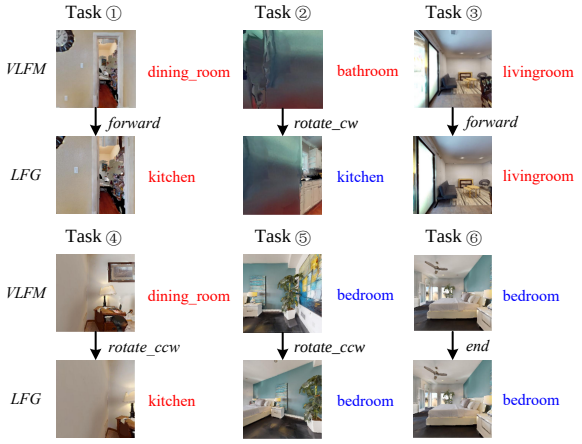

Fig. 15. The scene recognition results of VLFM and LFG in the six tasks, where the blue indicates success while the red denotes failure.

TABLE IV
THE acc OF ASR METHOD BASED ON VARIOUS ACTION STRATEGIES.

Action strategy	acc
No action	0.6836
Random action	0.7250
ACM	**0.8560**

that an action is selected randomly from the seven classes of actions as same as ACM, with the constraint that the number of performed actions is less than five. The testing accuracy of these action strategies are listed in Table IV.

As can be seen from Table IV, when 'random action' is introduced to capture multi-view scene images, the acc of the ASR method is improved compared with 'no action'. However, this aimless strategy has limitations—it focuses solely on accuracy enhancement. In contrast, ACM, trained on the expert data, achieves the highest acc, validating the rationality of ACM and the quality of the generated expert data.

Additionally, we explore an alternative approach: training an action strategy using reinforcement learning. Here, an action model based on deep Q-learning learning [58], named DQN-A, is used to compare with our method. DQN-A shares the same network structure as ACM. The reward function in DQN-A is based on the feature extraction model used in expert data generation. If the category of the image captured in the next state matches the ground truth and its score exceeds 0.99, the reward is positive 2. Otherwise, the reward is 0. DQN-A is trained in an environment identical to the one used for generating expert data. We train DQN-A five times with different random seeds (40, 41, 42, 43, and 44) using the following parameters: image size 224×224, batch size 32, learning rate 2.5^{-4}, memory size 1000, replace target iteration 100, and Adam optimizer. One model is saved every 10 learning step during training. After training, the saved DQN-A models replace ACM in the ASR method to finish the scene recognition tasks. The testing curves are drawn in Fig. 16, contrasting DQN-A with ACM.

As shown in Fig. 16, DQN-A can learn an action strategy to

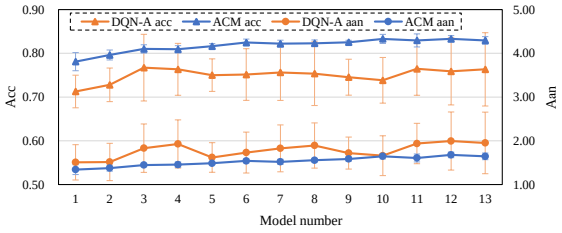

Fig. 16. The testing curve of the saved DQN-A and ACM during training, where the lines denote the average values and the error bar refers to the standard deviation.

TABLE V
THE TESTING RESULTS OF m_i, DAGGER (m_e), AND RPM.

Method	acc	aan
m_i	0.8574	3.8094
Dagger (m_e) [48]	0.8601	3.4121
RPM	0.8560	**1.7346**

adjust the view for improved scene recognition, achieving an average acc of approximately 0.76. However, during DQN-A's training process, the agent's exploration of actions is random, leading to significant fluctuations in accuracy at the same training stage. On the contrary, our ACM, trained using high-quality expert data, exhibits a more stable improvement in accuracy during training, resulting in a higher average acc. Moreover, the aan line of ACM lies below that of DQN-A, highlighting DQN-A's efficiency shortcomings. The above results prove the superiority of ACM.

2) RPM: RPM in our ASR method is specifically designed to remedy the compounding error caused by ACM. To evaluate the performance of RPM, we compare RPM with a well-known algorithm for reducing compounding error in imitation learning: Dagger [48]. Following the idea of Dagger, firstly ACM is trained by the raw expert data $\mathbb{D}$ to generate a well-trained model m_i. Then m_i is tested by the scene recognition tasks. We manually select 200 wrong cases from the testing results and give the correct action labels to form a new dataset $\mathbb{D}_n$. The extended data $\mathbb{D}_e \leftarrow \mathbb{D} \cup \mathbb{D}_n$ is used to fine-tune m_i for an improved action model m_e. The final testing results of m_i, Dagger (m_e), and RPM are shown in Table V.

The analysis of Table V highlights that the compounding error primarily impacts the efficiency of ASR. When the compounding error is left unaddressed, m_i tends to produce unnecessary actions, even when the observed images alone suffice for accurate recognition. Consequently, m_i exhibits the biggest aan with the lowest efficiency. Although Dagger partially mitigates this issue, its efficiency improvement remains limited, and it relies on manual labels. In comparison, the proposed RPM significantly reduces aan while achieving higher scene recognition efficiency without extra expert consultation or model retraining.

RPM comprises two parts aimed at reducing the compounding error: (1) action score-based revision (AR), and scene score-based revision (SR). To explore the impact of AR and SR individually, we conduct separate tests. The comparative results are summarized in Table VI, where both AR and SR contribute to improve efficiency with the reduced aan. This indicates both AR and SR effectively address the compounding error. Combining AR with SR in RPM ($AR+SR$)

TABLE VI
THE TESTING RESULTS OF DIFFERENT REVISION METHODS IN RPM.

Revision method	acc	aan
No revision	0.8574	3.8094
AR	0.8567	2.2469
SR	0.8570	2.5432
$AR+SR$	0.8560	**1.7346**

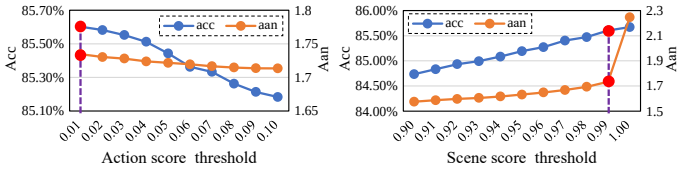

Fig. 17. The testing curves of RPM using various values of action score threshold and scene score threshold.

yields the best scene recognition efficiency. Notably, the acc of AR, SR, and $AR+SR$ remains similar. Thus, the choice of incorporating both AR and SR within RPM is well-founded and leads to efficient and accurate scene recognition.

In RPM, two empirical values play a crucial role: the action score threshold (set at 0.01) and the scene score threshold (set at 0.99). The rationale behind selecting these values is illustrated in Fig. 17. Varying the action score threshold significantly impacts acc but has a minor effect on aan. To maximize accuracy, we choose the value of 0.01 as the action score threshold. By contrast, the value of scene score threshold affects both acc and aan. Especially, the aan experiences a substantial increase when transitioning from 0.99 to 1.00. Balancing accuracy and efficiency, we opt for a scene score threshold of 0.99.

RPM fuses the scene scores of the adjacent images by retaining the maximum of each scene category, named Max. By contrast, here we discuss the results that using other two fusion strategies Avg and $Bayes$ to replace Max. Avg computes the average value of scene scores for the same category across the adjacent images. $Bayes$ means utilizing a Bayesian filter [16] to process the scene scores. It can been seen from the results in Table VII that Max gets the highest acc and the second-lowest aan. Although Avg minimizes aan, it also results in the lowest acc. In summary, our choice of Max as the fusion strategy within RPM ensures accurate scene recognition while maintaining a good efficiency.

3) model architecture: As shown in Fig. 6, the proposed ASR method includes two independent models (ACM and SRM) to output the action score and the scene score of an image. If a merged model can output both the action score and the scene score, the model size will be reduced. So, the merged model is designed by adding a fully connected layer and a softmax layer to output the scene score following the last average pooling layer of ACM. As the merged model deals with two kinds of tasks, its training process is divided into two parts. Firstly, fix the parameters in the fully connected layer of scene score and use the generate expert data to train the merged model for action scoring using the parameters of ACM in Table II. Then, fix all the parameters except those in the fully connected layer of scene score and utilize the scene data to train the merged model for scene scoring by the parameters of SRM in Table II.

TABLE VII
THE TESTING RESULTS OF RPM BASED ON THE DIFFERENT FUSION STRATEGIES OF THE SCENE SCORES.

Fusion strategy	acc	aan
Avg	0.8399	**1.7106**
$Bayes$	0.8548	1.7352
Max	**0.8560**	1.7346

After training, the merged model is evaluated by the same scene recognition tasks, achieving 0.5949 acc and 2.1931 aan. The acc of the merged model is far below the acc (0.8560) of the ASR method. Meanwhile, the efficiency of the merged model is lower than that of the ASR method (1.7346 aan). Therefore, designing two independent models to finish the two tasks in the ASR method is reasonable.

F. Expanding Scene Classes

In the proposed ASR method, both ACM and SRM are trained on the expert data and scene data, respectively. These datasets include five common domestic scene classes (bathroom, bedroom, dining room, kitchen, and livingroom). However, in practical scenarios, robots encounter scenes beyond these five categories—such as office environments. To explore the method's recognition performance for a broader range of scenes, we conduct an expanding experiment. We use all classes of images from MIT-67 dataset [51] to train SRM so that the ASR method could recognize more indoor scenes. After training, the ASR method with the expanding SRM is tested in a new scene (meeting room) using some dense scene recognition tasks shown in Fig. 18.

The testing results are 0.80 acc and 2.56 aan, visualized in Fig. 18. As shown in Fig. 18 (a), most of tasks are successfully completed, demonstrating that our method can recognize new scene classes through SRM expansion. It can be found in Fig. 18 (b) and (c) that ACM can output appropriate actions to change the views for correct recognition, proving that new scene dose not affect the performance of ACM. Therefore, our ASR method owns the expanding ability of recognizing more scene classes. However, too many scene classes may lead to SRM misidentification (e.g., task ④ in Fig. 18 (b)). To address this, we recommend restricting SRM's output categories based on practical application needs. For instance, in an office building, relevant classes (e.g., meeting room, office) can be retained, while less relevant ones (e.g., prison cell, hair salon) can be filtered out. This strategy improves testing results to 0.98 acc and 2.28 aan.

G. Using ASR during robot exploration

The proposed ASR method, packaged as an independent API (Application Programming Interface), can be invoked to assist robots in various tasks, including semantic mapping, localization, and so on. To show its capabilities, some demonstration videos recorded in Habitat environments are provided [1]. In these videos, the robot employs the ASR method to accurately recognize scenes, facilitating semantic localization and environment understanding during its exploration in the domestic environments.

[1]https://sites.google.com/view/asrbc

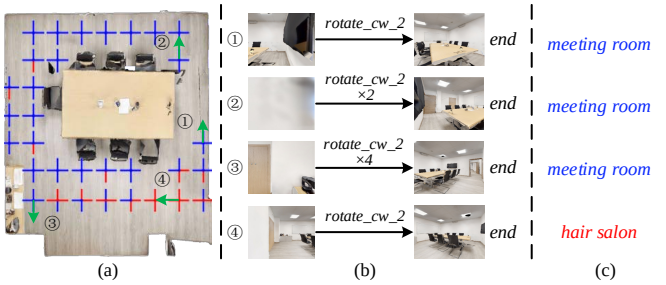

Fig. 18. The testing results of the ASR method with the expanding SRM in a new scene (meeting room): (a) scene recognition tasks, where the blue line indicates success while the red line denotes failure, and (b) some instances of scene recognition tasks with the outputting scene categories (c).

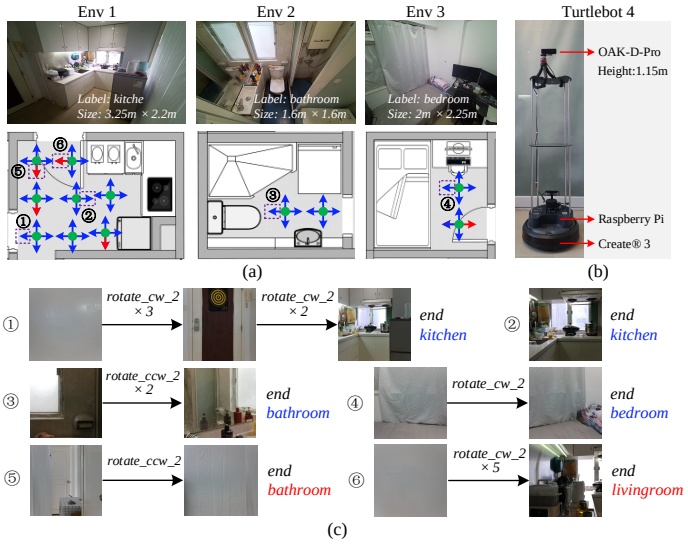

Fig. 19. (a) three real-world testing environments (env 1, env 2, and env 3) with different scene labels, including plenty of scene recognition points (green points) and tasks (arrows) marked in the floor plans. The blue arrow indicates success while the red arrow denotes failure. (b) the mobile robot TurtleBot 4. (c) some results labeled in (a) among the scene recognition tasks.

H. Real-world Experiments

We conduct real-world experiments using a mobile robot to validate the practical performance of our ASR method.

1) Testing Environment Setting: As demonstrated in Fig. 19 (a), the experiments involve dense scene recognition tasks across three different domestic environments: kitchen, bathroom, and bedroom. At each scene recognition point, four tasks are defined in different directions, resulting in a total of 48 scene recognition tasks for assessment.

2) Robot & Software Setting: TurtleBot 4, depicted in Fig. 19(b), is utilized for testing. TurtleBot 4 contains an onboard computer (Raspberry Pi 4B) running the Robot Operating System (ROS 2 Humble)[2], a moving chassis (Create® 3), and an RGB-D camera (OAK-D-Pro). The OAK-D-Pro camera's height is 1.15 meters, and it captures RGB images at a resolution of 224×224 pixels. Additionally, we employ a laptop equipped with an Intel i7-13620H CPU, 32GB memory, 1 TB SSD, and a Nvidia RTX 4060 GPU (8GB). This laptop executes Python scripts for both ACM and SRM based on Ubuntu 22.04 and ROS 2 Humble.

[2]https://www.ros.org/

TABLE VIII
THE MAPPING BETWEEN ACM AND ROBOT ACTION TYPE AND RANGE.

ACM output	Robot action type	Robot action range
forward	forward	30cm
backward	backward	30cm
rotate_cw	clockwise rotation	15°
rotate_cw_2	clockwise rotation	30°
rotate_ccw	counterclockwise rotation	15°
rotate_ccw_2	counterclockwise rotation	30°
end	stop	0

TABLE IX
THE COMPARISON RESULTS OF PCS_2, PCS_4, AND THE ASR METHOD IN REAL-WORLD ENVIRONMENTS.

Method	acc	aan
PCS_2 [16]	0.7292	2.00
PCS_4 [16]	0.7500	4.00
Ours	**0.8958**	2.42

To establish communication between TurtleBot 4 and the laptop, we set up a local area network using a wireless router. Our software framework consists of three nodes: image node, model node, and action node. The image node captures RGB images and publishes them via ROS topics. The model node subscribes to the image data and performs scene recognition by the same ACM and SRM used in the simulated experiments. It outputs actions and scene recognition results. The action node controls the robot's view adjustments based on the action types received from the model node. The action mapping between ACM and action node is provided in Table VIII. The linear and angular velocities of TurtleBot 4 are set to 0.2 m/s and 1.0 rad/s, respectively, within the action node. In addition, we use the strongest baseline method PCS as a comparison. Considering the small size of the experimental environments, only PCS_2 and PCS_4 are selected for testing.

3) Results and Analysis: The testing results of our ASR method across three different environments are depicted in Fig. 19 (a). Some instances of these results are illustrated in Fig. 19 (c). Additionally, the comparison results of PCS_2, PCS_4, and the ASR method are provided in Table IX.

What stands out from Table IX is that the proposed ASR method achieves the best accuracy of 0.8958, much better than PCS_2 and PCS_4. This verifies that our method has better performance on scene recognition than previous methods in the real-world environments. The test environments, which consist of compact apartments [see in Fig. 19 (a)], significantly differ from the large-area villa environments used in the simulation experiments. Consequently, the aan experiences a slight increase compared to the simulated results. In certain scene recognition tasks involving remote locations or challenging observation directions [e.g., Fig. 19 (c) task ①], the ASR method outputs additional actions to capture beneficial images for accurate recognition. Conversely, when initial observations suffice for scene recognition [e.g., Fig. 19 (c) task ②], the ASR method avoids unnecessary actions and directly recognizes the scene. The proposed ASR method successfully completes the vast majority of the scene recognition tasks. It is inevitable that there are a small number of failures. The reasons of failure are mainly caused by the ACM error and SRM error. Task ⑤ in Fig. 19 (c) is a failure caused by ACM outputting the inappropriate actions. Task ⑥ in Fig. 19 (c) shows an example of SRM error. In this task, although the image containing the visual information of the kitchen is obtained after appropriate actions, the SRM gives a wrong score (score of $livingroom$ is 1.00), resulting in the misidentification.

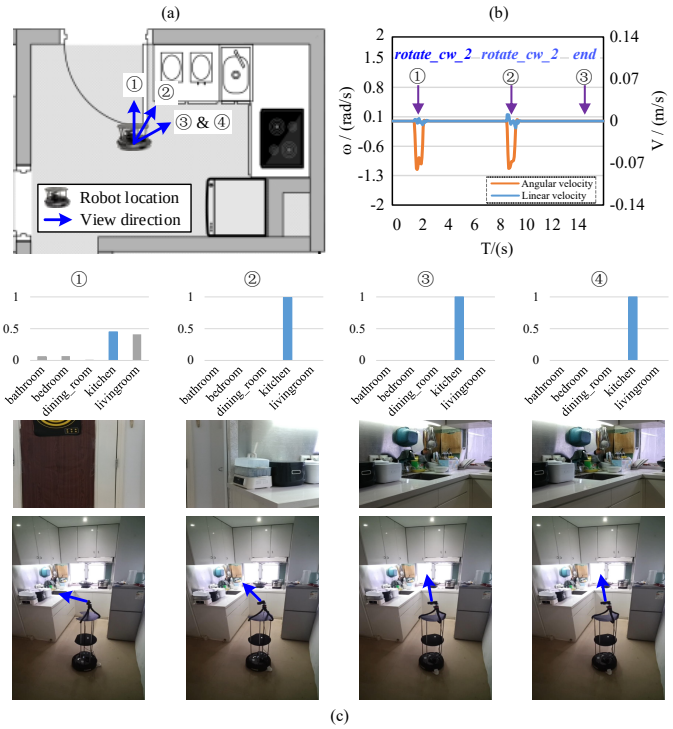

Fig. 20. A demo that Turtlebot 4 uses the ASR method to finish a scene recognition task: (a) robot location and view direction of different stages, (b) the performed actions expressed by the recorded liner velocity and angular velocity (negative value indicates backward and clockwise rotation), and (c) the scene score of the captured image in each stage with the third person perspective of TurtleBot 4 labeled by the view direction (blue arrows).

To provide a vivid example of how Turtlebot 4 accomplishes a scene recognition task using our proposed ASR method, consider the demo shown in Fig. 20. In the initial stage ① , Turtlebot 4 is positioned in the kitchen, with its observation direction aimed at a door [see in Fig. 20 (c) ①]. Given the low scene score for the correct label (kitchen), ACM outputs a $rotate\_cw\_2$ action to adjust the robot's view clockwise. This action allows the robot to capture relevant visual information, such as an electric cooker (see in Fig. 20 (c) ②). To obtain more relevant visual information, ACM continues to output a $rotate\_cw\_2$ action. In stage ③ , the robot sees more kitchen-related items (e.g., sink, bowls), which guarantees a high scene score of kitchen. As the scene scores of kitchen in stage ② and ③ are both larger than 0.99, our method outputs a $stop$ action according to RPM and generates the correct result (kitchen) of this task.

Overall, our ASR method, initially trained in a simulation environment, seamlessly transitions to real-world scenarios without fine-tuning. Its robust performance in domestic environments underscores its practical applicability and portability.

VI. Conclusion

In this work, we proposed an ASR method for domestic robots, aiming to enhance the accuracy and efficiency of scene recognition when faced with uncertain robot positions and observation directions. We presented an expert data generation method to create abundant data automatically for training the designed ACM based on behavior cloning. ACM could adjust the robot's view to capture beneficial images for subsequent scene recognition. The multi-view scene recognition method was introduced to deal with these images by SRM and RPM. RPM could reduce the compounding error without expert consultation or model retraining. It outputted the final recognition result, enabling accurate scene recognition with fewer images. Our extensive evaluations, including comparative experiments and an ablation study, demonstrated the advantages of our method in terms of accuracy and efficiency compared to the state-of-the-art approaches. Notably, the proposed method, initially trained in simulated environments, seamlessly transitioned to real-world domestic environments without fine-tuning. We validated its portability and practical applicability by deploying it on a TurtleBot 4 robot.

Looking ahead, we plan to integrate our ASR method with SLAM (Simultaneous Localization and Mapping) techniques to achieve efficient semantic mapping. In addition, to further improve the performance of the ASR model, we plan to optimize the ACM by combining behavior cloning and reinforcement learning and try to design an on-line learning mechanism to revise the wrong recognition cases by man-machine interaction for SRM, improving the scene recognition accuracy in the current environment.

References

[1] N. Sünderhauf, F. Dayoub, S. Mcmahon, B. Talbot, R. Schulz, P. Corke, G. Wyeth, B. Upcroft, and M. Milford, "Place categorization and semantic mapping on a mobile robot," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2016.

[2] L. Bernreiter, A. Gawel, H. Sommer, J. Nieto, R. Siegwart, and C. Cadena, "Multiple hypothesis semantic mapping for robust data association," *IEEE Robot. Autom. Lett.*, vol. 4, no. 4, pp. 3255–3262, 2019.

[3] Y. C. Sousa and H. F. Bassani, "Topological semantic mapping by consolidation of deep visual features," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 4110–4117, 2022.

[4] J. L. Matez-Bandera, J. Monroy, and J. Gonzalez-Jimenez, "Efficient semantic place categorization by a robot through active line-of-sight selection," *Knowl.-Based Syst.*, vol. 240, pp. 1–13, 2022.

[5] A. C. Hernandez, E. Derner, C. Gomez, R. Barber, and R. Babuka, "Efficient object search through probability-based viewpoint selection," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2020, pp. 6172–6179.

[6] S. Liu, G. Tian, Y. Zhang, M. Zhang, and S. Liu, "Service planning oriented efficient object search: A knowledge-based framework for home service robot," *Expert Syst. Appl.*, vol. 187, p. 115853, 2022.

[7] Y. Li, Y. Ma, X. Huo, and X. Wu, "Remote object navigation for service robots using hierarchical knowledge graph in human-centered environments," *Intell. Serv. Robot.*, pp. 1–15, 2022.

[8] Y. Zhang, G. Tian, X. Shao, S. Liu, M. Zhang, and P. Duan, "Building metric-topological map to efficient object search for mobile robot," *IEEE Trans. Ind. Electron.*, vol. 69, no. 7, pp. 7076–7087, 2022.

[9] L. Xie, F. Lee, L. Liu, K. Kotani, and Q. Chen, "Scene recognition: A comprehensive survey," *Pattern Recognit.*, vol. 102, pp. 1–18, 2020.

[10] M. Dixit, Y. Li, and N. Vasconcelos, "Semantic fisher scores for task transfer: Using objects to classify scenes," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 42, no. 12, pp. 3102–3118, 2020.

[11] S. Lowry, N. Sünderhauf, P. Newman, J. J. Leonard, D. Cox, P. Corke, and M. J. Milford, "Visual place recognition: A survey," *IEEE Trans. Robot.*, vol. 32, no. 1, pp. 1–19, 2016.

[12] L. Zhou, J. Cen, X. Wang, Z. Sun, T. L. Lam, and Y. Xu, "Borm: Bayesian object relation model for indoor scene recognition," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.* IEEE, 2021, pp. 39–46.

[13] S. Liu, G. Tian, Y. Zhang, and P. Duan, "Scene recognition mechanism for service robot adapting various families: A cnn-based approach using multi-type cameras," *IEEE Trans. Multimedia*, vol. 24, pp. 2392–2406, 2021.

[14] S. Choe, H. Seong, and E. Kim, "Indoor place category recognition for a cleaning robot by fusing a probabilistic approach and deep learning," *IEEE Trans. Cybern.*, vol. 52, no. 8, pp. 7265–7276, 2022.

[15] B. Zhu, X. Fan, X. Gao, G. Xu, and J. Xie, "A heterogeneous attention fusion mechanism for the cross-environment scene classification of the home service robot," *Robot. Autom. Syst.*, p. 104619, 2024.

[16] N. Sünderhauf, F. Dayoub, S. McMahon, B. Talbot, R. Schulz, P. Corke, G. Wyeth, B. Upcroft, and M. Milford, "Place categorization and semantic mapping on a mobile robot," in *Proc. IEEE Int. Conf. Robot. Autom.* IEEE, 2016, pp. 5729–5736.

[17] R. Bajcsy, Y. Aloimonos, and J. K. Tsotsos, "Revisiting active perception," *Autom. Robot.*, vol. 42, pp. 177–196, 2018.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[19] L. Yuan, Y. Chen, T. Wang, W. Yu, Y. Shi, Z.-H. Jiang, F. E. Tay, J. Feng, and S. Yan, "Tokens-to-token vit: Training vision transformers from scratch on imagenet," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021, pp. 558–567.

[20] S. K. Ramakrishnan, A. Gokaslan, E. Wijmans, O. Maksymets, A. Clegg, J. M. Turner, E. Undersander, W. Galuba, A. Westbury, A. X. Chang, M. Savva, Y. Zhao, and D. Batra, "Habitat-matterport 3d dataset (HM3d): 1000 large-scale 3d environments for embodied AI," *arXiv*, 2021. [Online]. Available: https://arxiv.org/abs/2109.08238

[21] S. Liu, G. Tian, and Y. Xu, "A novel scene classification model combining resnet based transfer learning and data augmentation with a filter," *Neurocomputing*, vol. 338, pp. 191–206, 2019.

[22] Z. Zou, W. Liu, and W. Xing, "Adanff: A new method for adaptive non-negative multi-feature fusion to scene classification," *Pattern Recognit.*, vol. 123, p. 108402, 2022.

[23] C. Wang, G. Peng, and B. De Baets, "Embedding metric learning into an extreme learning machine for scene recognition," *Expert Syst. Appl.*, vol. 203, p. 117505, 2022.

[24] P. S. Yee, K. M. Lim, and C. P. Lee, "Deepscene: Scene classification via convolutional neural network with spatial pyramid pooling," *Expert Syst. Appl.*, vol. 193, p. 116382, 2022.

[25] X. Yuan, Z. Qiao, and A. Meyarian, "Scale attentive network for scene recognition," *Neurocomputing*, vol. 492, pp. 612–623, 2022.

[26] E. Fazl-Ersi and J. K. Tsotsos, "Histogram of oriented uniform patterns for robust place recognition and categorization," *Int. J. of Robot. Res.*, vol. 31, no. 4, pp. 468–483, 2012.

[27] J. Yang, S. Zhang, G. Wang, and M. Li, "Scene and place recognition using a hierarchical latent topic model," *Neurocomputing*, vol. 148, pp. 578–586, 2015.

[28] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv*, 2014. [Online]. Available: https://doi.org/10.48550/arXiv.1409.1556

[29] G. Huang, Z. Liu, V. Laurens, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2261–2269.

[30] A. Khaliq, S. Ehsan, Z. Chen, M. Milford, and K. McDonald-Maier, "A holistic visual place recognition approach using lightweight cnns for significant viewpoint and appearance changes," *IEEE Trans. Robot.*, vol. 36, no. 2, pp. 561–569, 2020.

[31] B. Ferrarini, M. J. Milford, K. D. McDonald-Maier, and S. Ehsan, "Binary neural networks for memory-efficient and effective visual place recognition in changing environments," *IEEE Trans. Robot.*, vol. 38, no. 4, pp. 2617–2631, 2022.

[32] P. Yin, A. Abuduweili, S. Zhao, L. Xu, C. Liu, and S. Scherer, "Bioslam: A bioinspired lifelong memory system for general place recognition," *IEEE Trans. Robot.*, vol. 39, no. 6, pp. 4855–4874, 2023.

[33] P. Uršič, A. Leonardis, M. Kristan *et al.*, "Part-based room categorization for household service robots," in *Proc. IEEE Int. Conf. Robot. Autom.* IEEE, 2016, pp. 2287–2294.

[34] R. Pereira, L. Garrote, T. Barros, A. Lopes, and U. J. Nunes, "A deep learning-based indoor scene classification approach enhanced with inter-object distance semantic features," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.* IEEE, 2021, pp. 32–38.

[35] B. Miao, L. Zhou, A. S. Mian, T. L. Lam, and Y. Xu, "Object-to-scene: Learning to transfer object knowledge to indoor scene recognition," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.* IEEE, 2021, pp. 2069–2075.
[36] X. Yu, C. Fermüller, C. L. Teo, Y. Yang, and Y. Aloimonos, "Active scene recognition with vision and language," in *Proc. IEEE Int. Conf. Comput. Vis.* IEEE, 2011, pp. 810–817.
[37] D. Jayaraman and K. Grauman, "End-to-end policy learning for active visual categorization," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 41, no. 7, pp. 1601–1614, 2019.
[38] S. K. Ramakrishnan, D. Jayaraman, and K. Grauman, "Emergence of exploratory look-around behaviors through active observation completion," *Sci. Robot.*, vol. 4, no. 30, pp. 1–12, 2019.
[39] J. Xiao, K. A. Ehinger, A. Oliva, and A. Torralba, "Recognizing scene viewpoint using panoramic place representation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 2695–2702.
[40] M. Zare, P. M. Kebria, A. Khosravi, and S. Nahavandi, "A survey of imitation learning: Algorithms, recent developments, and challenges," *IEEE Trans. Cybern.*, 2024.
[41] J. Zhang and K. Cho, "Query-efficient imitation learning for end-to-end simulated driving," in *Proc. AAAI Conf. Artif. Intell.*, vol. 31, no. 1, 2017.
[42] M. Kelly, C. Sidrane, K. Driggs-Campbell, and M. J. Kochenderfer, "Hg-dagger: Interactive imitation learning with human experts," in *Proc. IEEE Int. Conf. Robot. Autom.* IEEE, 2019, pp. 8077–8083.
[43] J. Wong, A. Tung, A. Kurenkov, A. Mandlekar, L. Fei-Fei, S. Savarese, and R. Martín-Martín, "Error-aware imitation learning from teleoperation data for mobile manipulation," in *Proc. Conf. Robot Learn.* PMLR, 2022, pp. 1367–1378.
[44] P. Florence, C. Lynch, A. Zeng, O. A. Ramirez, A. Wahid, L. Downs, A. Wong, J. Lee, I. Mordatch, and J. Tompson, "Implicit behavioral cloning," in *Proc. Conf. Robot Learn.* PMLR, 2022, pp. 158–168.
[45] M. Beliaev, A. Shih, S. Ermon, D. Sadigh, and R. Pedarsani, "Imitation learning by estimating expertise of demonstrators," in *Proc. Int. Conf. Mach. Learn.* PMLR, 2022, pp. 1732–1748.
[46] S. Belkhale, Y. Cui, and D. Sadigh, "Data quality in imitation learning," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 36, 2024.
[47] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 1, 1988.
[48] S. Ross, G. J. Gordon, and J. A. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Int. Conf. Artif. Intell. Stat.*, 2011, p. 627–635.
[49] P. Ammirato, P. Poirson, E. Park, J. Košecká, and A. C. Berg, "A dataset for developing and benchmarking active vision," in *Proc. IEEE Int. Conf. Robot. Autom.* IEEE, 2017, pp. 1378–1385.
[50] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, pp. 211–252, 2015.
[51] A. Quattoni and A. Torralba, "Recognizing indoor scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 413–420.
[52] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, "Sun database: Large-scale scene recognition from abbey to zoo," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2010, pp. 3485–3492.
[53] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv*, 2020. [Online]. Available: /https://arxiv.org/abs/2010.11929
[54] N. Yokoyama, S. Ha, D. Batra, J. Wang, and B. Bucher, "Vlfm: Vision-language frontier maps for zero-shot semantic navigation," in *Proc. IEEE Int. Conf. Robot. Autom.* IEEE, 2024, pp. 42–48.
[55] D. Shah, M. R. Equi, B. Osiński, F. Xia, B. Ichter, and S. Levine, "Navigation with large language models: Semantic guesswork as a heuristic for planning," in *Proc. Conf. Robot Learn.* PMLR, 2023, pp. 2683–2699.
[56] J. Li, D. Li, S. Savarese, and S. Hoi, "Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models," in *Proc. Int. Conf. Machine Learn.* PMLR, 2023, pp. 19 730–19 742.
[57] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv*, 2017. [Online]. Available: https://doi.org/10.48550/arXiv.1704.04861
[58] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

Shaopeng Liu (Member, IEEE) received the B.S. degree in measurement and control technology and instrument from Qingdao University of science and technology, Qingdao, China, in 2016, the M.S. degree in control engineering, and Ph.D. degree in control theory and control engineering from Shandong University, Jinan, China, in 2019 and 2023, respectively. He is currently a Postdoctoral Fellow in the Department of Industrial and Systems Engineering, at The Hong Kong Polytechnic University. His current research interests include robot cognition, scene recognition, object searching, deep learning, and service robot.

Chao Huang (Senior Member, IEEE) received the Ph.D. degree in control engineering from the University of Wollongong, Wollongong, NSW, Australia, in 2018. She is currently a Research Assistant Professor with the Department of Industrial and System Engineering, The Hong Kong Polytechnic University (PolyU), Hong Kong. Her research interests include human–machine collaboration, fault-tolerant control, mobile robot (EV, UAV), and path planning and control. Dr. Huang is an Associate Editor for the *IEEE Transactions on Transportation Electrification* and *IEEE Transactions on Intelligent Vehicles*, *Engineering Applications of Artificial Intelligence*.

Hailong Huang (Senior Member, IEEE) received the Ph.D degree in Systems and Control from the University of New South Wales, Sydney, Australia, in 2018. He was a postdoctoral research fellow at the School of Electrical Engineering and Telecommunications, University of New South Wales, Sydney, Australia. He is now an Assistant Professor at the Department of Aeronautical and Aviation Engineering, the Hong Kong Polytechnic University, Hong Kong. His current research interests include guidance, navigation, and control of UAVs and mobile robots. Dr. Huang is an Associate Editor for the *IEEE Transactions on Vehicular Technology* and *IEEE Transactions on Intelligent Vehicles*.