A Category Agnostic Model for Visual Rearrangment

Yuyi Liu[1,2], Xinhang Song[1,2], Weijie Li[1,2], Xiaohan Wang[1], Shuqiang Jiang[1,2]
[1]Key Lab of Intelligent Information Processing Laboratory of the Chinese Academy of Sciences (CAS),
Institute of Computing Technology, Beijing [2]University of Chinese Academy of Sciences, Beijing
{yuyi.liu, xinhang.song, weijie.li, xiaohan.wang}@vipl.ict.ac.cn
sqjiang@ict.ac.cn

Abstract

This paper presents a novel category agnostic model for visual rearrangement task, which can help an embodied agent to physically recover the shuffled scene configuration without any category concepts to the goal configuration. Previous methods usually follow a similar architecture, completing the rearrangement task by aligning the scene changes of the goal and shuffled configuration, according to the semantic scene graphs. However, constructing scene graphs requires the inference of category labels, which not only causes the accuracy drop of the entire task but also limits the application in real world scenario. In this paper, we delve deep into the essence of visual rearrangement task and focus on the two most essential issues, scene change detection and scene change matching. We utilize the movement and the protrusion of point cloud to accurately identify the scene changes and match these changes depending on the similarity of category agnostic appearance feature. Moreover, to assist the agent to explore the environment more efficiently and comprehensively, we propose a closer-aligned-retrace exploration policy, aiming to observe more details of the scene at a closer distance. We conduct extensive experiments on AI2THOR Rearrangement Challenge based on RoomR dataset and a new multi-room multi-instance dataset MrMiR collected by us. The experimental results demonstrate the effectiveness of our proposed method.

1. Introduction

Rearrangement task remains a practical challenge for embodied agents that assist humans in real life, whose goal is to bring a given physical environment into the goal state with a goal specification [2]. In this paper, we focus on a branch of the general rearrangement task based on ExperienceGoal, i.e., visual rearrangement task[45], which requires an agent to recover the scene configuration after it was shuffled randomly. Due to the excessive complexity of the state space, the end-to-end deep reinforcement learning methods previously used for navigation struggle to cover this task, resulting in performance only marginally above chance [18, 45]. Recent works demonstrate that the modular methods, such as MaSS [41] and TIDEE [38], effectively reduce the complexity of the rearrangement task by dividing the task into several modules. These methods use a pre-trained detector to assign category labels to each object and infer the rearrangement goals through matching the semantic scene graphs of both the goal and shuffled configuration.

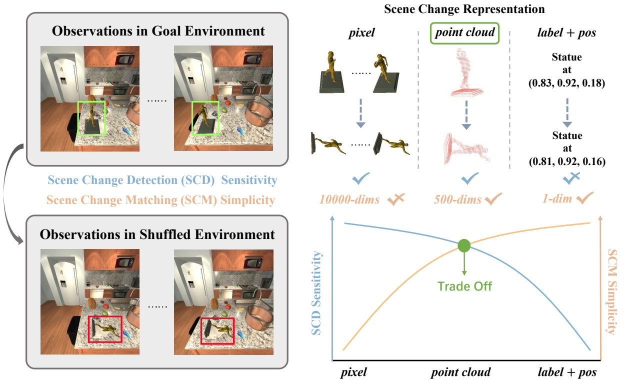

Figure 1. **Influence of different scene change representations on scene change detection sensitivity and scene change matching simplicity**. To strike a balance between these two issues, we select point cloud as our scene change representation.

However, the introduction of category information may not be that necessary as the essential goal of visual rearrangement task is equal to “make it like what it was before”. Even without the category labels, we can still perform the task by memorizing the appearance characteristics and the state information of objects in the scene. Besides, due to the limited accuracy of the detector, the transition from visual input to category information will inevitably lead to errors, which can accumulate and propagate to subsequent modules, thereby causing the accuracy drop of the entire task. Previous works achieve large gains with ground-truth semantic segmentation[18, 41]. Moreover, there are inherent limitations of the methodologies based on category information. Once the detector is trained, these methods are restricted to a fixed set of categories and powerless against the object categories not previously observed in training environment. Using zero-shot methods, such as SAM [22] combined with CLIP [33], can considerably expand the known categories, but they are still within a limited set. It is impractical to retrain the model every time a new object category emerges due to the extensive resources required.

To address the above problems, our motivation is to identify all scene changes in the room and restore them, regardless of any category. Previous methods use semantic labels for SCM because these labels provide a high-level representation of objects and make SCM straightforward. However, the scene changes can be represented in numerous ways, ranging from pixel to point cloud, and up to label combined with positional information. As shown in Fig. 1, there is an inherent trade-off: while simpler representations minimize information loss during conversion and enhance the sensitivity of SCD, they simultaneously complicate the process of SCM.

Point cloud can serve as an appropriate representation of scene change, as it captures rich geometric, positional, and scale information of objects and remains robust against varied observation angles and obstructions from other objects. Leveraging point cloud facilitates efficient SCD and also provides richer appearance information for SCM. However, due to the inherent unordered nature and rotational invariance of point cloud, it is difficult to match the point cloud directly. We need to extract high-dimensional appearance features from point cloud for SCM.

Based on these observations, we propose a category agnostic model for visual rearrangement task called CAVR, to the best of our knowledge, this is the first attempt for visual rearrangement without category inferring. By utilizing point cloud as the scene change representation, CAVR can recover the scene configuration to its goal state without any category concepts. In CAVR, we introduce a closer-aligned-retrace exploration policy to help agent conduct exploration effectively for SCD. Meanwhile, we maintain a diff-cloud, which consists of two components, one for the point cloud moved and another for the point cloud protruding in the shuffled scene configuration, compared to the goal configuration. The diff-cloud precisely captures the variations occurring throughout the scene. After exploration, we utilize the pre-trained appearance feature extractor to embed the diff-cloud and then match the scene changes across various locations based on the similarity of appearance feature, resulting in a series of rearrangement goals. Then we use a planning-based policy to restore them to their goal states in succession.

We conduct experiments on AI2THOR Rearrangement Challenge based on the RoomR dataset[45] and shows improvements on both the success rate and the portion of successfully fixed objects. To cater to more practical demands, we introduce a multi-room multi-instance rearrangement dataset MrMiR based on ProcTHOR simulator[12]. The experimental results on MrMiR dataset fully demonstrate the effectiveness of our method in the complex multi-room environment.

2. Related Works

Rearrangement The general rearrangement problem [2] aims to transform the environment from an initial state to a goal state through interaction. We focus on an instantiation of the rearrangement problem[45], in which the goal state is specified by immersing the agent in the goal environment and allowing the agent to explore autonomously. Prior works can be classified into two categories, end-to-end reinforcement learning and modular methods. The end-to-end methods [18, 45] perform poorly mainly due to the large action space and complex stages in the task. Comparatively, the modular methods [38, 41] have shown surprising progress in improving the success rate. In detail, Mass[41] proposes a semantic policy with a voxel-based semantic map to find and match the changed objects. TIDEE[38] utilizes the spatial relationships between objects to determine the changed objects. Motivated by prior works, we also propose a modular method, while our model can perform the task without any category information.

Visual exploration Visual exploration refers to the process in which an agent collects information about the 3D environment through motion and perception [14, 29, 30, 35]. For visual exploration, efficiency is of utmost significance, involving how to access a broader range of regions [3, 6, 17, 39], observe more objects [16] and obtain a larger volume of environmental information relevant to downstream tasks (such as navigation) [25, 43, 44, 46–49] within a certain budget.

To improve the efficiency of exploration, several methods have employed ideas like curiosity [5, 7, 30, 31], coverage [6, 11] and reconstruction [21, 34]. Most related to ours is the coverage-based works, which try to maximize the area seen in the environment [6, 11]. In our exploration policy, both the area explored and the observation distance are considered simultaneously to accurately observe more details of the scene.

Scene Change Detection Scene change detection (SCD) refers to the task of identifying and localizing changes of a scene captured at different times[9, 26, 27, 36, 37, 40, 42]. Depending on the types of scene representation, methods are classified into two categories, respectively, 2D domain and 3D domain[37]. The first one devises specific neural networks to process the image pair taken at different times and generate a pixel-level prediction, namely, each pixel is classified into a category of change[1, 4, 9, 10, 19, 36, 37, 42]. Some studies focus on the scene change detection in 3D domain. They aim to reconstruct a time-varying 3D model from images taken from multiple viewpoints at different times and represent the temporal scene changes over several decades[26, 27, 40]. In our task, we not only focus on identifying changes in the 3D environment, but also emphasize the importance of matching changes across various locations, which is crucial for enabling the agent to accurately recover the scene configuration.

3. Method

Given the intrinsic complexity of visual rearrangement task, in this section, we present a modular approach to tackle the task, decomposing it into manageable subtasks including visual exploration, scene change detection and scene change matching. Our pipline is illustrated in Fig. 2. We start this section by giving the definition of visual rearrangement task in Sec. 3.1. Then we describe the three modules separately. The visual exploration module (Sec. 3.2) requires the agent to explore the environment efficiently and comprehensively while retaining memory of the environment. Subsequently, the scene change detection module (Sec. 3.3) utilizes the agent's memory of the goal environment and compares it with the current environment to identify all scene changes. Then to recover the goal configuration, the scene change matching module (Sec. 3.4) is proposed to correlate these changes across different areas within the scene and infer the rearrangement goals.

3.1. Visual Rearrangement Task

According to the commonly accepted norms in the community[2], the rearrangement task is defined in a general form, where an agent is initialized in a starting state s^0 and required to transform the environment from s^0 to the goal state $s^* \in S^*$ with the possible actions $a \in A$. The environment state space is denoted as the Cartesian product of the pose spaces of all rigid parts: $S = (R^3 \times SO3) \times (R^3 \times SO3) \ldots \times (R^3 \times SO3)$, where R^3 and $SO3$ represent the 3D locations and rotations space. Follow the Partially Observable Markov Decision Processes (POMDP), the agent typically has no access to any state space and must operate purely based on the sensory observations $o \in O$ and the given goal specification $g = \phi(s^0, S^*)$. Based on different goal specification forms (GeometricGoal, ImageGoal, LanguageGoal, ExperienceGoal, et al.), the general rearrangement task has various levels of difficulty.

We consider an instance of rearrangement task proposed by Weihs et al.[45], which adopts the ExperienceGoal as the goal specification g and is defined as a two-stage task, including the **walkthrough** and **unshuffle** stages. During the walkthrough stage, the agent is immersed in a room of goal state s^* and allowed to explore autonomously. Sequentially, the walkthrough environment is shuffled and some objects' states are changed, denoted as the unshuffle stage, where the agent officially starts the rearrangement task and reorganizes the shuffled scene configuration back.

3.2. Visual Exploration

Under the two-stage rearrangement task, the initial exploration of the target environment is critical for the subsequent stages, since the agent is expected to acquire more object information in the fewest number of steps. Previous works adopt coverage-based exploration [38] or a search policy based on the expert distribution of objects [41]. However, there are usually many small-sized objects distributed across the scene, which can be easily overlooked or obscured by large entities when observed from a distance. Therefore, we propose a closer-aligned-retrace exploration policy, aiming to observe more objects at a closer distance to improve the observation accuracy and completeness.

The core idea of the proposed policy is to build an observation distance map $m_o \in \mathbb{R}^{H \times W}$, where each grid denotes the minimum distance at which the current coordinate point is observed by the agent. Through the optimization of m_o, the agent can be guided to observe objects closer.

In exploration, at each timestep, the agent obtains visual observation RGBD and updates its own pose. Following previous work [8, 28, 41], we also build a 2D obstacle map $m_t \in \mathbb{R}^{H \times W}$ with the proposed observation distance map m_o. At the beginning of the exploration, due to the limited range of movement, the observation distance map m_o predominantly consists of high distance values. Therefore, the visual exploration policy π can be represented by optimizing a function f of m_o and a distance thresh ϵ_d.

$$\pi(a) = f(m_o, \epsilon_d)$$

The goal of optimization is to minimize the observation distance map (i.e., $\min(m_o) \leq \epsilon_d$)). We employ an analytical approach to obtain the solution. Specifically, based on the current observation distance map m_o, we select a waypoint as the next exploration goal and apply the route planning Dijkstra algorithm [13] to generate a path on the obstacle map m_t. As to the waypoint selection, we prioritize selecting those with higher distance values on the distance map, aiming to observe objects closer. To better compare the shuffled and goal state of the scene for rearrangement goals inference, in the unshuffle stage, the agent tries its best to replicate the trajectory of the walkthrough stage.

3.3. Scene Change Detection

Detecting changes within the scene is a critical capability for an agent to perform rearrangement tasks. We maintain a diff-cloud to represent the scene changes. As shown in Fig. 2 (b), the diff-cloud consists of two parts. The red and blue points respectively represent the moved and protruding

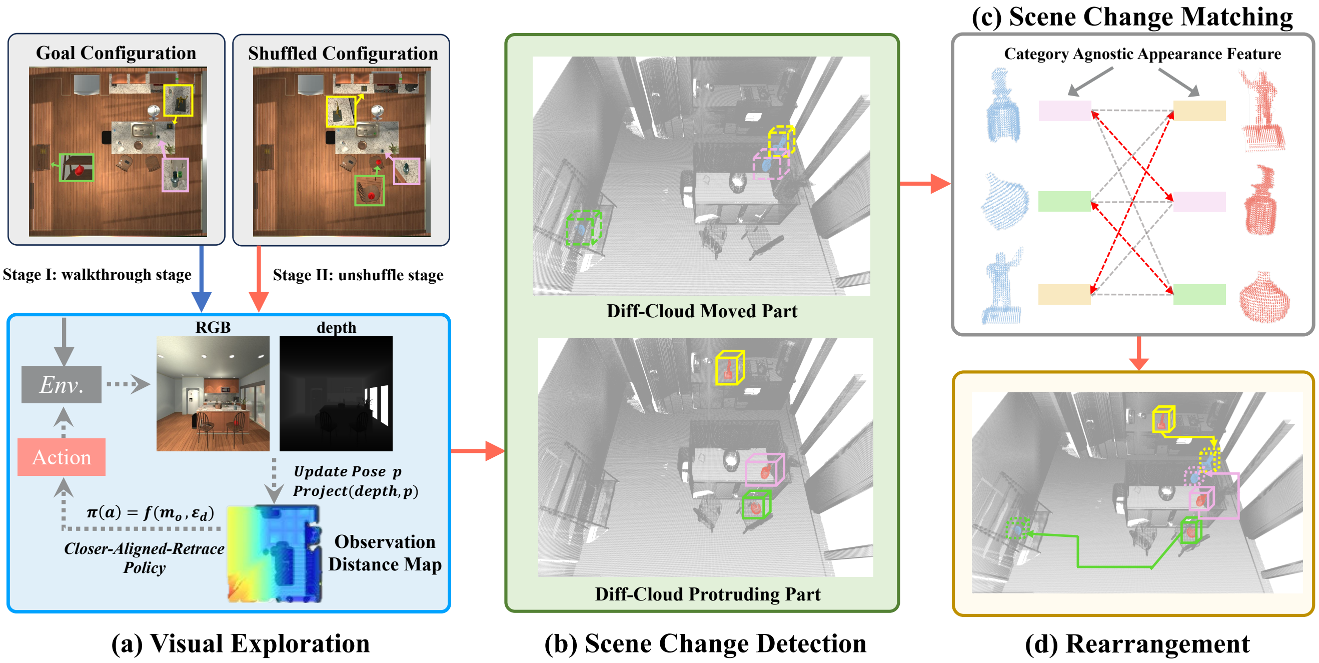
(c) Scene Change Matching
Goal Configuration
Shuffled Configuration
Stage I: walkthrough stage
Stage II: unshuffle stage
RGB
depth
Env.
Action
Update Pose p
Project(depth, p)
π(a) = f(m_o, ε_d)
Closer-Aligned-Retrace Policy
Observation Distance Map
Diff-Cloud Moved Part
Diff-Cloud Protruding Part
Category Agnostic Appearance Feature
(a) Visual Exploration
(b) Scene Change Detection
(d) Rearrangement

Figure 2. **Pipline of our CAVR model.** (a) The gradient color transitioning from blue to red in the observation distance map represents the distances ranging from 0 m to 5 m. We adopt a closer-aligned-retrace exploration policy to observe more details by optimizing a function of the distance map. (b) Scene change detection is performed by comparing the point clouds corresponding to the goal configuration and the shuffled configuration of the scene, recording the moved part (blue points) and the protruding part (red points) to construct the diff-cloud. (c) We extract the entity-layer information from the two parts of the diff-cloud and match these entities depending on the similarity of category agnostic appearance feature. (d) After the matching process, we obtain a series of rearrangement goals with their goal states (indicated by the dashed bounding boxes) and shuffled states (indicated by the solid bounding boxes) .

point clouds in the shuffled configuration, compared to the goal configuration. Next, we explain how to construct the diff-cloud using visual inputs from the two stages.

During the walkthrough stage, at each pose p^w of the agent, we employ the depth information D_{p^w} to generate an egocentric point cloud $c^{ego}_{p^w}$. Each point in $c^{ego}_{p^w}$ is associated with a pixel in depth D_{p^w}. Then we convert $c^{ego}_{p^w}$ from the agent's coordinate system to global coordinate system, resulting in a geocentric point cloud $c^{geo}_{p^w}$. For the observed RGB image I_{p^w}, we adopt the pre-trained resnet18 model [20] provided by the official PyTorch to extract a visual feature map f_{p^w}.

During the unshuffle stage, we use the same method to generate the geocentric point cloud $c^{geo}_{p^u}$ and the feature map f_{p^u} for each pose p^u. If p^u aligns with a previous pose p^w in the walkthrough stage, we compare the two corresponding point clouds, $c^{geo}_{p^w}$ and $c^{geo}_{p^u}$. A considerable shift between two point coordinates associated with the same pixel indicates the changes have occurred in this location. Specifically, the increase in distance from the agent suggests removal of some objects, while the decrease signifies objects addition. Based on the distance variations, these points are allocated to the moved part and the protruding part of the diff-cloud, respectively. Moreover, for the area implying scene changes, we extract the corresponding visual feature from the feature map and assign it to each point in that region. Each point in the diff-cloud is represented as $\{x, y, z, v\}$, where x, y, z is the 3D coordinate in the global coordinate system and v is the visual feature.

3.4. Scene Change Matching

After exploration of the walkthrough stage and the unshuffle stage, we acquire the comprehensive diff-cloud that encompasses changes in all areas of the scene. Note that in the visual rearrangement task settings, objects cannot disappear into thin air, they are simply moved from one place to another. Therefore, to recover the scene configuration, we need to match changes across various locations in the scene. Since the diff-cloud contains only some points in space, we first extract the entity-layer information from it and then perform matching operations on the entity-level.

We apply the density-based clustering algorithm DBSCAN [15] separately to the two parts of the diff-cloud, resulting in two sets of entities, a moved entity set

$\Omega^m = \{\omega_1^m, \omega_2^m, \dots, \omega_k^m\}$ and an protruding entity set $\Omega^p = \{\omega_1^p, \omega_2^p, \dots, \omega_l^p\}$. Each entity in these two sets is a collection of some points in the diff-cloud: $\omega = \{(x_1, y_1, z_1, v_1), (x_2, y_2, z_2, v_2), \dots, (x_n, y_n, z_n, v_n)\}$.

As shown in Fig. 2 (c), the scene change matching process can be regarded as the weighted bipartite graph matching between Ω^m and Ω^p. We construct a bipartite graph $G = (\Omega^m \cup \Omega^p, E)$, where $\Omega^m \cup \Omega^p$ is the node set and E represents the all fully connected edge set. Every edge $e \in E$ has one end node in Ω^m and the other end node in Ω^p . The function ϕ assigns a positive weight value to each edge. A matching M is a subset of E such that each node in $\Omega^m \cup \Omega^p$ appears in at most one edge in M. Our goal is to find the maximum matching:

$$M^* = \underset{M}{argmax} \sum_{e \in M} \phi(e),$$

where $e = e(\omega_i^m, \omega_j^p)$ represents the edge matching node ω_i^m and ω_j^p.

The role of the weight function $\phi(e)$ is to determine the possibility that ω_i^m and ω_j^p belong to the same instance. Based on this, we design the weight function to calculate the similarity in appearance of these two nodes. The appearance of each entity ω is considered from two aspects: geometric feature geo and visual feature vis. For vis, we use the average of the visual features of all the points in this entity. In terms of geo, we train a geometric feature extractor, which builds upon PointNet++[32] and embeds the raw point cloud data. The weight function ϕ is specifically defined as

$$\phi(e(\omega_i^m, \omega_j^p)) = Cosim(geo_i^m, geo_j^p) + Cosim(vis_i^m, vis_j^p),$$

where $Cosim$ refers to the cosine similarity.

Then the Kuhn-Munkres algorithm [24] is adopted to solve this maximum matching problem. Once entities are matched, we acquire entity pairs $\{(\omega_1^m, \omega_{j_1}^p), (\omega_2^m, \omega_{j_2}^p), \dots, (\omega_t^m, \omega_{j_t}^p)\}$ as rearrangement goals. Each entity pair $(\omega_i^m, \omega_{j_i}^p)$ represents the two different states of the same instance, where ω_i^m denotes the goal state and $\omega_{j_i}^p$ denotes the current state of the instance. Subsequently, for the inferred rearrangement goals, we transport them to their goal states in succession, during which, we leverage the 2D obstacle map and Dijkstra algorithm [13] to conduct obstacle avoidance and navigation path planning.

4. Experiments

Rearrangement task remains a practical challenge for embodied agents that assist humans in real life, whose goal is to bring a given physical environment into the goal state with a goal specification [2].each pixel is classified into a category of change[36].

Table 1. Comparison on RoomR dataset

Method	Suc (%)↑	FS (%)↑	E↓	Mis↓
TIDEE	11.7	28.9	0.715	0.734
MaSS	4.7	16.5	1.016	1.018
Our	**14.2**	**33.1**	**0.714**	**0.707**

“Suc”: Success; “FS”: Fixed Strict; “E”: Energy Remain; “Mis”: Misplaced.

Table 2. Comparison on our MrMiR dataset

Method	Suc (%)↑	FS (%)↑	E↓	Mis↓
TIDEE	1.0	14.1	0.917	0.924
MaSS	0.6	10.5	1.019	1.026
Our	**5.0**	**28.7**	**0.7327**	**0.7134**

“Suc”: Success; “FS”: Fixed Strict; “E”: Energy Remain; “Mis”: Misplaced.

4.1. Experiment Setup

Dataset We evaluate our method on the AI2THOR Rearrangement Challenge based on the RoomR dataset[45], which consists of 80 rooms and 4000 tasks for training, and 20 rooms with 1000 tasks each for both validation and test. Each task in RoomR involves 1 to 5 objects with state changes, characterized by object locations or openness.

In RoomR[45] dataset, the spatial range of object changes is limited due to the confined area with single-room scenes and the target objects to be rearranged are mainly category-wise, i.e., most categories only have one instance. To cater for the prevalent characteristics of indoor environments in reality, we build a more practical and challenging dataset MrMiR for the two-stage rearrangement task on the ProcTHOR simulator[12], where the change in the state of an object can involve a broader spatial range, even extending across different rooms. Besides, there exists multiple instances within the same category that have different appearance. The simulator ProcTHOR[12] respectively provides 10,000 training, 1000 valid and 1000 test apartments. For our task need, we totally select 6000 apartments in the simulator, splitting 5000 apartments for training, 500 apartments for validation, and 500 apartments for test. Each apartment contains multiple instances within the same category that have different appearance. For each apartment, we randomly generate one rearrangement task. Therefore, our MrMiR dataset totally contains 6000 rearrangement tasks, the same as RoomR. Fig. 3 illustrates the comparison of scene area distribution between our MrMiR dataset and RoomR dataset. It can be seen that our dataset encompasses a diverse range of scene area, while RoomR mainly focusing on small rooms under $100\,\mathrm{m}^2$.

To train the geometric feature extractor based on PointNet++[32], which embeds point cloud, we generate a dataset using AI2THOR[23]. We collect 77K sample pairs, of which 70K are used for training and 7K for testing. Each sample pair is composed of two point clouds, which may either represent the same instance (the positive pair) or different instances (the negative pair). The distribution of the positive and negative pairs is balanced, with a 1:1 ratio. Within each room of AI2THOR, we generate positive sample pairs by applying different transformation operations to the point cloud of the same object. We also perform transformation operations on the point clouds of two different objects to generate negative sample pairs. For the transformation operations, we consider random rotation, adding random noise, and randomly deleting 20% of the original point cloud data.

Metrics To evaluate an agent's performance, we consider several metrics as follows: **(1) Success**. The success metric is a binary indicator of each task, which is strictly defined as whether the whole objects' states have been restored to their goal states. **(2) Fixed Strict**. This metric records the proportion of successfully fixed objects per task. If there are any newly misplaced objects at the end of a task, this metric will be set as 0. **(3) Misplaced**. This metric is denoted as the number of misplaced objects after the unshuffle stage divided by the number of misplaced objects at the start of the unsuffle stage. **(4) Energy Remaining (E)**. The above metrics are quite strict, which is not possible to measure the distance to task completion. The energy is used to represent the difference between two possible states of an object, which can be functioned as $D : S \times S \Rightarrow [0, 1]$. The larger the energy value, the greater the difference between the two states, whereas if the two states are approximately equal, the energy value is 0. Therefore, this metric can be computed as the sum of all objects' energy after the unshuffle stage, divided by the sum of all objects' energy at the beginning of the unshuffle stage.

Implementation details The distance threshold ϵ_d is set to $1.5\,\mathrm{m}$, which is determined through hyper-parameter tuning, as detailed in Sec. 4.4. To ensure a fair comparison, we limit the maximum step number for both the exploration and rearrangement stages. In RoomR dataset[45], the exploration step limit is set to 300 and the navigation step limit for each object's rearrangement is set to 50. In our MrMiR dataset, we categorize the apartments by area into five levels: $<10\,\mathrm{m}^2$, $10-60\,\mathrm{m}^2$, $60-150\,\mathrm{m}^2$, $150-300\,\mathrm{m}^2$, $>300\,\mathrm{m}^2$. Correspondingly, the the exploration step limits are set to 50, 200, 300, 500 and 800 and the navigation step limits for each object's rearrangement are set to 50, 80, 100, 200, 300. When we train the geometric feature extractor, we use Adam as our optimizer and the hyper-parameters $(lr, \beta_1, \beta_2, \epsilon)$ are set to $(0.001, 0.9, 0.999, 1\mathrm{e}-8)$. The parameters and models are tuned only on the RoomR dataset and are directly tested on the MrMiR dataset.

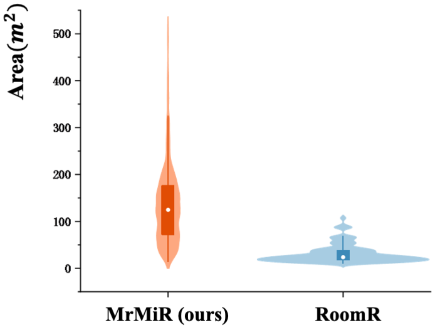

Figure 3. **Comparison of scene area distribution between MrMiR and RoomR[45] datasets.**

4.2. Comparisons with Related Works

We report the quantitative comparisons on the RoomR dataset in Table 1 and the MrMiR dataset in Table 2 with the two state-of-the-art modular methods MaSS[41] and TIDEE[38].

MaSS [41] employs a Gaussian mixture model to train a semantic search strategy, aiming to guide the agent towards regions where the likelihood of object occurrence is higher. During the exploration process, the 3D voxel semantic map is constructed, which is then used to match and identify objects that need to be rearranged.

TIDEE [38] employs a coverage-based exploration policy to extract the spatial relationships between objects. After the exploration of two stages, the relationship changes are used to identify the rearrangement goals.

Given that the original work of TIDEE is based on category-level (i.e., only records the category information of objects, and for multiple instances under the same category, only chooses one as the target), it cannot be directly applicable to our MrMiR dataset. To be fair, we make modifications to TIDEE by extracting all spatial relationships between instances when testing on the MrMiR dataset.

As shown in Table 1, our proposed method CAVR outperforms the related works in all metrics. Specifically, it improves the success rate by 2.5% and the proportion of successfully fixed objects by 5.38%. Beyond the primary improvements, the decrease in energy and misplaced metrics suggests that our CAVR method could rearrange the environment closer to the goal configuration, even without fully completing the task. As shown in Table 2, the disparity between the related works and our CAVR method has further increased, fully demonstrating the superiority of our method in dealing with more complex and challenging environment.

Table 3. Ablation Study

Visual Exploration	Scene Change Matching	Success (%) ↑	FixedStrict (%) ↑	E ↓	Misplaced ↓
coverage	✓	13.1	31.0	0.722	0.717
MaSS's	✓	8.7	25.8	0.763	0.754
✓	uniform	11.3	24.6	0.818	0.807
✓	visual	14.0	32.6	0.724	0.720
✓	geometric	14.2	32.3	0.723	0.717
✓	✓	**14.2**	**33.1**	**0.714**	**0.707**

"✓" represents utilizing our proposed corresponding modules (closer-aligned-retrace exploration policy and scene change matching based on similarity of appearance including visual feature and geometric feature introduced in Sec. 3); "E": Energy Remaining.

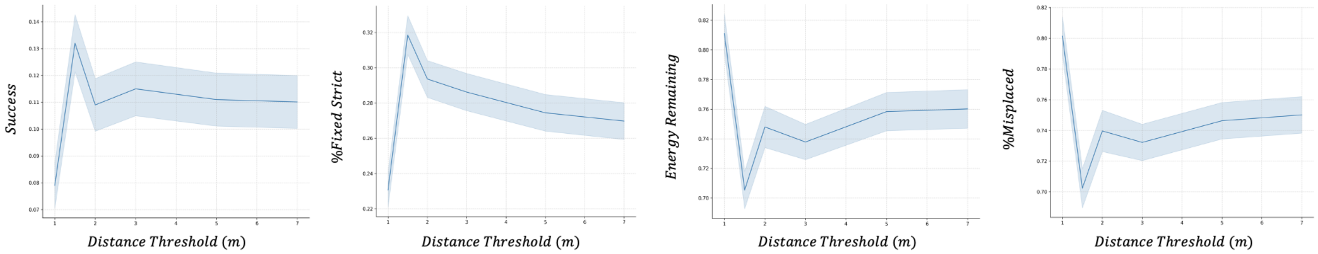

Figure 4. **Rearrangement performance relative to distance threshold ϵ_d.** The blue lines represent the average metrics across the tasks of validation set of RoomR[45], with the shaded area representing the 68% confidence interval. Higher values of $Success$ and $\%FixedStrict$ indicate superior performance, whereas lower $EnergyRemaining$ and $\%Misplaced$ indicate better results.

4.3. Ablation Study

Considering the complexity of visual rearrangement task, we conduct ablation studies on RoomR dataset [45] to further investigate the importance of different modules within the overall task. In the ablation studies, we keep the diff-cloud as the representation of scene changes.

Ablation on the visual exploration module We replace our closer-aligned-retrace exploration policy with: **a) Coverage-based exploration policy** This strategy randomly selects target points from unexplored areas, which are used in TIDEE [38]. **b) MaSS's semantic policy** This ablation directly adopts the semantic policy proposed in [41], which trains a network to search the object distribution.

Ablation on the scene change matching module The process of scene change matching can be abstracted as a maximum weight matching problem in bipartite graph. We substitute the weights of edges with: **a) Uniform weights** This ablation set all edge weights to the same value regardless of the objects' appearance, which leads to a random matching. **b) Similarity of visual feature** This ablation only utilize the similarity of visual feature as the weight. **c) Similarity of geometric feature** This ablation only use the similarity of geometric feature as the weight.

The experimental results are presented in Table 3. In the ablation study on the visual exploration module, the model with MaSS's exploration policy perform worst due to the substantial variation in objects distribution within rooms, making it challenging to model them effectively with a uniform network. The model with coverage-based policy also underperform as it is likely to overlook minor changes when the observation distance is considerable. In the ablation study on the scene change matching module, removing any part of the appearance feature clearly decreases the performance in all metrics, which illustrates the noticeable impact of our extracted appearance feature on the visual rearrangement task.

4.4. Hyper-parameter Tuning

We conduct experiments on the validation set of RoomR[45] to determine the distance threshold in the optimization criteria for our closer-aligned-retrace exploration policy. A very small threshold value means visiting nearly every grid space on the map, while a large threshold value ignores the underlying concern of non-ambiguous scene change detection. The exploration happens in the unshuffle stage as well and our exploration policy leads the agent to try its best to replicate the previous trajectory. Therefore the threshold value determines the trade-off between optimality in terms of the agent traversal for exploration and a

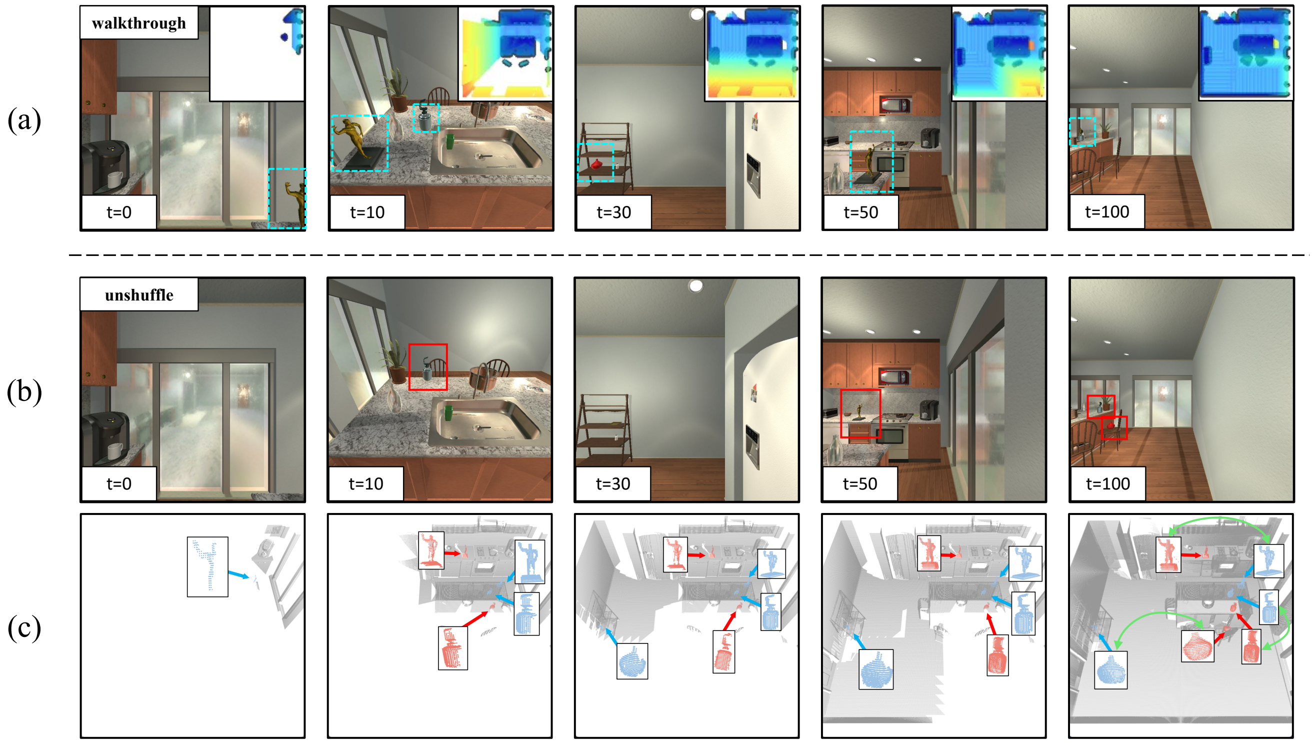

Figure 5. **Visualization of optimization process of observation distance map and construction of diff-cloud** (a) In the walkthrough stage, objects begin in the positions indicated by the dashed blue bounding boxes. Observation distance map is positioned at the top right corner of each image. The color transitioning from blue to red represents the distances ranging from $0\,\mathrm{m}$ to $5\,\mathrm{m}$. (b) In the unshuffle stage, objects are moved to the locations indicated by the solid red box. (c) The diff-cloud is gradually built up, including the moved part (blue points) and the protruding part (red points).

non-ambiguous scene change detection.

As shown in the Fig. 4, we set observation distance thresholds from $1\,\mathrm{m}$ to $7\,\mathrm{m}$ and compute the average metrics of 1000 tasks. Optimal performance on the validation set is achieved with a distance threshold at $1.5\,\mathrm{m}$, which is the threshold consistently applied in the other experiments throughout this paper. In this experiment, error bars are calculated based on a 68% confidence interval.

4.5. Visualization

We visualize and analyze the optimization of the observation distance map during the walkthrough stage and the construction of the diff-cloud in the unshuffle stage, as shown in Fig. 5. As the exploration progresses, the distance map increasingly exhibit hues of blue, which indicates that our exploration policy enables the agent to observe the scene details up close. In the unshuffle stage, as the diff-cloud is gradually built up, we develop a distinct understanding of the changes occurring throughout the scene. After matching these changes according to the similarity of their appearance, we can carry out the rearrangement execution procedurally.

5. Conclusion

We propose a category agnostic model for visual rearrangement task in this paper. Our method is composed of a closer-aligned-retrace exploration policy, a scene change detection module based on point cloud and a scene change matching module utilizing the similarity of appearance feature, each specifically designed to recover the scene configuration regardless of any category labels. To validate the proposed method, we conduct experiments on the RoomR dataset and a more practical dataset MrMiR collected by us, where multiple instances distribute across multiple rooms. Experimental results on these two datasets demonstrate that our method is able to perform the visual rearrangement task effectively without any category information.

Acknowledgements: This work was supported by the National Natural Science Foundation of China under Grant 62125207, 62272443, 62032022 and U23B2012, in part by Beijing Natural Science Foundation under Grant JQ22012, Z190020.

References

[1] Pablo F Alcantarilla, Simon Stent, German Ros, Roberto Arroyo, and Riccardo Gherardi. Street-view change detection with deconvolutional networks. *Autonomous Robots*, 42: 1301–1322, 2018. 3

[2] Dhruv Batra, Angel X. Chang, Sonia Chernova, Andrew J. Davison, Jia Deng, Vladlen Koltun, Sergey Levine, Jitendra Malik, Igor Mordatch, Roozbeh Mottaghi, Manolis Savva, and Hao Su. Rearrangement: A challenge for embodied ai, 2020. 1, 2, 3, 5

[3] Edward Beeching, Jilles Dibangoye, Olivier Simonin, and Christian Wolf. Learning to plan with uncertain topological maps. In *European Conference on Computer Vision*, pages 473–490. Springer, 2020. 2

[4] Shuhui Bu, Qing Li, Pengcheng Han, Pengyu Leng, and Ke Li. Mask-cdnet: A mask based pixel change detection network. *Neurocomputing*, 378:166–178, 2020. 3

[5] Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A. Efros. Large-scale study of curiosity-driven learning. In *ICLR*, 2019. 2

[6] Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural slam. In *International Conference on Learning Representations (ICLR)*, 2020. 2

[7] Devendra Singh Chaplot, Helen Jiang, Saurabh Gupta, and Abhinav Gupta. Semantic curiosity for active visual learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16*, pages 309–326. Springer, 2020. 2

[8] Devendra Singh Chaplot, Murtaza Dalal, Saurabh Gupta, Jitendra Malik, and Russ R Salakhutdinov. Seal: Self-supervised embodied active learning using exploration and 3d consistency. *Advances in neural information processing systems*, 34:13086–13098, 2021. 3

[9] Chao-Peng Chen, Jun-Wei Hsieh, Ping-Yang Chen, Yi-Kuan Hsieh, and Bor-Shiun Wang. Saras-net: scale and relation aware siamese network for change detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 14187–14195, 2023. 2, 3

[10] Shuo Chen, Kailun Yang, and Rainer Stiefelhagen. Dr-tanet: Dynamic receptive temporal attention network for street scene change detection. In *2021 IEEE Intelligent Vehicles Symposium (IV)*, pages 502–509. IEEE, 2021. 3

[11] Tao Chen, Saurabh Gupta, and Abhinav Gupta. Learning exploration policies for navigation. In *International Conference on Learning Representations*, 2019. 2

[12] Matt Deitke, Eli VanderBilt, Alvaro Herrasti, Luca Weihs, Jordi Salvador, Kiana Ehsani, Winson Han, Eric Kolve, Ali Farhadi, Aniruddha Kembhavi, et al. Procthor: Large-scale embodied ai using procedural generation. *arXiv preprint arXiv:2206.06994*, 2022. 2, 5

[13] Edsger W Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959. 3, 5

[14] H. Durrant-Whyte and T. Bailey. Simultaneous localization and mapping: part i. *IEEE Robotics & Automation Magazine*, 13(2):99–110, 2006. 2

[15] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, pages 226–231, 1996. 4

[16] Kuan Fang, Alexander Toshev, Li Fei-Fei, and Silvio Savarese. Scene memory transformer for embodied agents in long-horizon tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[17] Kuan Fang, Alexander Toshev, Li Fei-Fei, and Silvio Savarese. Scene memory transformer for embodied agents in long-horizon tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 538–547, 2019. 2

[18] Samir Yitzhak Gadre, Kiana Ehsani, Shuran Song, and Roozbeh Mottaghi. Continuous scene representations for embodied ai. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14849–14859, 2022. 1, 2

[19] Enqiang Guo, Xinsha Fu, Jiawei Zhu, Min Deng, Yu Liu, Qing Zhu, and Haifeng Li. Learning to measure change: Fully convolutional siamese metric networks for scene change detection. *arXiv preprint arXiv:1810.09111*, 2018. 3

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 4

[21] Dinesh Jayaraman and Kristen Grauman. Learning to look around: Intelligently exploring unseen environments for unknown tasks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[22] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. *arXiv preprint arXiv:2304.02643*, 2023. 2

[23] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, et al. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017. 6

[24] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955. 5

[25] Weijie Li, Xinhang Song, Yubing Bai, Sixian Zhang, and Shuqiang Jiang. ION: instance-level object navigation. In *MM '21: ACM Multimedia Conference, Virtual Event, China, October 20 - 24, 2021*, pages 4343–4352. ACM, 2021. 2

[26] Haotong Lin, Qianqian Wang, Ruojin Cai, Sida Peng, Hadar Averbuch-Elor, Xiaowei Zhou, and Noah Snavely. Neural scene chronology. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20752–20761, 2023. 2, 3

[27] Kevin Matzen and Noah Snavely. Scene chronology. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VII 13*, pages 615–630. Springer, 2014. 2, 3

[28] So Yeon Min, Devendra Singh Chaplot, Pradeep Ravikumar, Yonatan Bisk, and Ruslan Salakhutdinov. Film: Following instructions in language with modular methods. *arXiv preprint arXiv:2110.07342*, 2021. 3

[29] Medhini Narasimhan, Erik Wijmans, Xinlei Chen, Trevor Darrell, Dhruv Batra, Devi Parikh, and Amanpreet Singh. Seeing the un-scene: Learning amodal semantic maps for room navigation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16*, pages 513–529. Springer, 2020. 2

[30] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pages 2778–2787. PMLR, 2017. 2

[31] Deepak Pathak, Dhiraj Gandhi, and Abhinav Gupta. Self-supervised exploration via disagreement. In *International conference on machine learning*, pages 5062–5071. PMLR, 2019. 2

[32] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017. 5, 6

[33] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 2

[34] Santhosh K Ramakrishnan, Ziad Al-Halah, and Kristen Grauman. Occupancy anticipation for efficient exploration and navigation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pages 400–418. Springer, 2020. 2

[35] Santhosh K Ramakrishnan, Dinesh Jayaraman, and Kristen Grauman. An exploration of embodied visual exploration. *International Journal of Computer Vision*, 129:1616–1649, 2021. 2

[36] Vijaya Raghavan T Ramkumar, Elahe Arani, and Bahram Zonooz. Differencing based self-supervised pretraining for scene change detection. In *Conference on Lifelong Learning Agents*, pages 952–965. PMLR, 2022. 2, 3, 5

[37] Ken Sakurada, Mikiya Shibuya, and Weimin Wang. Weakly supervised silhouette-based semantic scene change detection. In *2020 IEEE International conference on robotics and automation (ICRA)*, pages 6861–6867. IEEE, 2020. 2, 3

[38] Gabriel Sarch, Zhaoyuan Fang, Adam W Harley, Paul Schydlo, Michael J Tarr, Saurabh Gupta, and Katerina Fragkiadaki. Tidee: Tidying up novel rooms using visuo-semantic commonsense priors. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIX*, pages 480–496. Springer, 2022. 1, 2, 3, 6, 7

[39] Nikolay Savinov, Anton Raichuk, Raphaël Marinier, Damien Vincent, Marc Pollefeys, Timothy Lillicrap, and Sylvain Gelly. Episodic curiosity through reachability. In *International Conference on Learning Representations (ICLR)*, 2019. 2

[40] Grant Schindler and Frank Dellaert. Probabilistic temporal inference on reconstructed 3d scenes. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1410–1417. IEEE, 2010. 2, 3

[41] Brandon Trabucco, Gunnar Sigurdsson, Robinson Piramuthu, Gaurav S Sukhatme, and Ruslan Salakhutdinov. A simple approach for visual rearrangement: 3d mapping and semantic search. *arXiv preprint arXiv:2206.13396*, 2022. 1, 2, 3, 6, 7

[42] Guo-Hua Wang, Bin-Bin Gao, and Chengjie Wang. How to reduce change detection to semantic segmentation. *Pattern Recognition*, 138:109384, 2023. 2, 3

[43] Xiaohan Wang, Yuehu Liu, Xinhang Song, Beibei Wang, and Shuqiang Jiang. Generating explanations for embodied action decision from visual observation. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 2838–2846, 2023. 2

[44] Xiaohan Wang, Yuehu Liu, Xinhang Song, Beibei Wang, and Shuqiang Jiang. Camp: Causal multi-policy planning for interactive navigation in multi-room scenes. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[45] Luca Weihs, Matt Deitke, Aniruddha Kembhavi, and Roozbeh Mottaghi. Visual room rearrangement. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5922–5931, 2021. 1, 2, 3, 5, 6, 7

[46] Haitao Zeng, Xinhang Song, and Shuqiang Jiang. Multi-object navigation using potential target position policy function. *IEEE Transactions on Image Processing*, 2023. 2

[47] Sixian Zhang, Weijie Li, Xinhang Song, Yubing Bai, and Shuqiang Jiang. Generative meta-adversarial network for unseen object navigation. In *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XXXIX*, pages 301–320.

[48] Sixian Zhang, Xinhang Song, Yubing Bai, Weijie Li, Yakui Chu, and Shuqiang Jiang. Hierarchical object-to-zone graph for object navigation. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 15110–15120. IEEE, 2021.

[49] Sixian Zhang, Xinhang Song, Weijie Li, Yubing Bai, Xinyao Yu, and Shuqiang Jiang. Layout-based causal inference for object navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10792–10802, 2023. 2