RoboEXP: Action-Conditioned Scene Graph via Interactive Exploration for Robotic Manipulation

Hanxiao Jiang[1] Binghao Huang[1] Ruihai Wu[3] Zhuoran Li[4]
Shubham Garg[2] Hooshang Nayyeri[2] Shenlong Wang[1] Yunzhu Li[1]
[1]University of Illinois Urbana-Champaign [2]Amazon [3]Peking University [4]National University of Singapore

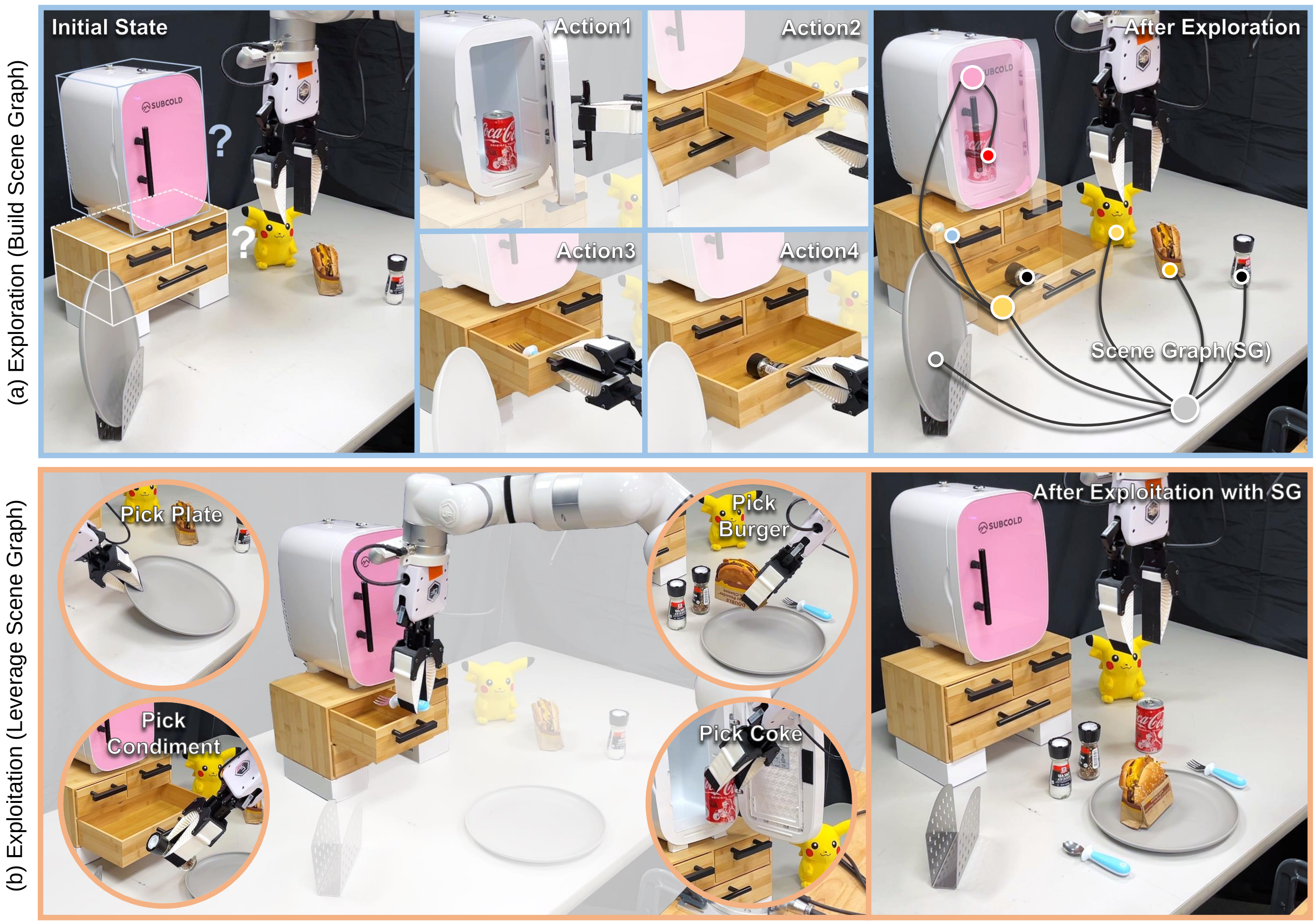


Fig. 1: **Interactive Exploration to Construct an Action-Conditioned Scene Graph (ACSG) for Robotic Manipulation.** (a) **Exploration:** The robot autonomously explores by interacting with the environment to generate a comprehensive ACSG. This graph is used to catalog the locations and relationships of items. (b) **Exploitation:** Utilizing the constructed scene graph, the robot completes downstream tasks by efficiently organizing the necessary items according to the desired spatial and relational constraints.

Abstract*—Robots need to explore their surroundings to adapt to and tackle tasks in unknown environments. Prior work has proposed building scene graphs of the environment but typically assumes that the environment is static, omitting regions that require active interactions. This severely limits their ability to handle more complex tasks in household and office environments: before setting up a table, robots must explore drawers and cabinets to locate all utensils and condiments. In this work, we introduce the novel task of interactive scene exploration, wherein robots autonomously explore environments and produce an action-conditioned scene graph (ACSG) that captures the structure of the underlying environment. The ACSG accounts for both low-level information, such as geometry and semantics, and high-level information, such as the action-conditioned relationships between different entities in the scene. To this end, we present the Robotic Exploration (RoboEXP) system, which incorporates the Large Multimodal Model (LMM) and an explicit memory design to enhance our system's capabilities. The robot reasons about what and how to explore an object, accumulating new information through the interaction process and incrementally constructing the ACSG. We apply our system across various real-world settings in a zero-shot manner, demonstrating its effectiveness in exploring and modeling environments it has never seen before. Leveraging the constructed ACSG, we illustrate the effectiveness and efficiency of our RoboEXP system in facilitating a wide range of real-world manipulation tasks involving rigid, articulated objects, nested objects like Matryoshka dolls, and deformable objects like cloth. Project Page: https://jianghanxiao.github.io/RoboEXP/

I. Introduction

Imagine a future household robot designed to prepare breakfast. This robot must efficiently perform various tasks such as conducting inventory checks in cabinets, fetching food from the fridge, gathering utensils from drawers, and spotting leftovers under food covers. Key to its success is the ability to interact with and explore the environment, especially to find items that aren't immediately visible. Equipping it with such capabilities is crucial for the robot to effectively complete its everyday tasks.

Robot exploration and active perception have long been challenging areas in robotics [1–16]. Various techniques have been proposed, including information-theoretic approaches, curiosity-driven exploration, frontier-based methods, and imitation learning [17–22, 1, 23, 14, 13, 24, 15, 25]. Nevertheless, previous research has primarily focused on exploring static environments by merely changing viewpoints in a navigation setting or has been limited to interactions with a small set of object categories, such as drawers, or a closed set of simple actions like pushing [26].

In this work, we investigate the interactive scene exploration task, where the goal is to efficiently identify all objects, including those that are directly observable and those that can only be discovered through interaction between the robot and the environment (see Fig. 1). Towards this goal, we present a novel scene representation called action-conditioned 3D scene graph (ACSG). Unlike conventional 3D scene graphs that focus on encoding static relations, ACSG encodes both spatial relationships and logical associations indicative of action effects (e.g., opening a fridge will reveal an apple inside). We then show that interactive scene exploration can be formulated as a problem of action-conditioned 3D scene graph construction and traversal.

Tackling interactive scene exploration poses challenges: how can we reason about which objects need to be explored, choose the right action to interact with them, and maintain knowledge about our exploration findings? With these challenges in mind, we propose a novel, real-world robotic exploration framework, the RoboEXP system. At the core of our system is a large foundational model-powered instantiation of action-conditioned 3D scene graph. Specifically, our framework consists of four modules: perception, memory, decision-making, and action, as shown in Fig. 3. To address the challenge of perceiving what is present in the scene, our **perception module** utilizes Grounding-DINO ([27]), Segment Anything in High Quality (SAM-HQ) [28, 29], and CLIP [30] to detect objects or parts and extract their language-embedded semantic features. Our **decision-making module** employs the rich commonsense knowledge contained in large multimodal models, such as GPT-4V [31, 32], to assist in selecting which objects to explore and what actions to take, and in validating their plausibility. Once the decision-making module has chosen a skill, our **action module** is then activated to follow the plans formulated by the prior modules. During the entire physical interaction process, our **memory model**—which maintains the action-conditioned scene graph—will be continuously updated to preserve the scene's knowledge for future exploration and exploitation. Despite its strong capacity, our hardware system is simple—it requires only a single RGB-D wrist camera as sensor input and uses a single robot arm for actions.

RoboEXP can handle diverse exploration tasks in a zero-shot manner, constructing complex action-conditioned 3D scene graph in various scenarios, including those involving obstructing objects and requiring multi-step reasoning (Fig. 2). We evaluate our system across various settings, spanning simple, single-object scenarios to complex environments, demonstrating its adaptability and robustness. The system also effectively manages different human interventions. Moreover, we show that our reconstructed action-conditioned 3D scene graph demonstrates strong capacity in performing multiple complex downstream tasks. Action-conditioned 3D scene graph advances LLM/LMM-guided robotic manipulation and decision-making research [33, 34], extending their operation domain from environments with known or observable objects to complicated environments with unknown or unobserved ones. To our knowledge, this is the first of its kind.

Our contributions are as follows: i) we propose action-conditioned 3D scene graph and introduce the interactive scene exploration task to address the challenging interaction aspect of exploration; ii) we develop the RoboEXP system, capable of exploring complicated environments with unseen objects in a wide range of settings; iii) through extensive experiments, we demonstrate our system's ability to construct complex and complete action-conditioned 3D scene graph, demonstrating significant potential for various manipulation tasks. Our experiments involve rigid and articulated objects, nested objects like Matryoshka dolls, and deformable objects like cloth, showcasing the system's generalization ability across objects, scene configurations, and downstream tasks.

II. Related Works

Scene graphs [35, 36] represent objects and their relations [37–39] in a scene via a graph structure. Previous studies generate scene graphs from images [40, 36] or 3D scenes [41] with hierarchical and semantic information, and further with the assistance of large language models (LLMs) [42]. They leverage scene graphs for image captioning [43, 44], image retrieval and generation [35, 45], visual-language tasks [37, 46], navigation [47, 48] and task planning [49, 50]. While previous works model scene graphs in static 2D or 3D scenes, we generate action-conditioned scene graphs that integrate actions as core elements, depicting interactive relationships between objects and actions. This action-centric approach opens avenues for physical exploration and diverse downstream robotics tasks.

Neuro-symbolic representations integrates neural networks' perceptual abilities with the symbolic reasoning for robots in complex and dynamic environments. Prior works explored understanding scenes and describing robotic skills in symbolic texts to interpret demonstrations [51, 52], ground abstract actions for robotic primitives [53] and generate action plans [54–57]. Our proposed framework also constructs symbolic

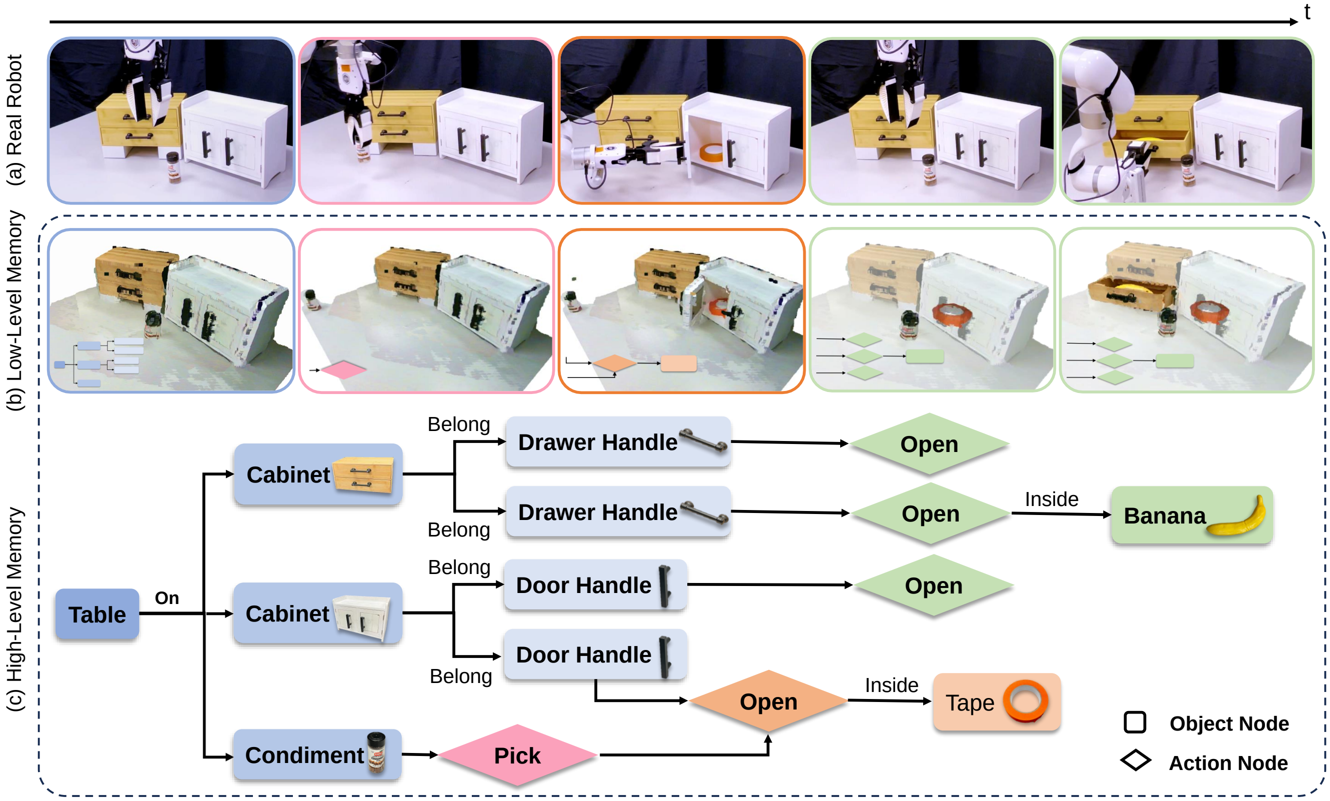


Fig. 2: **Action-Conditioned 3D Scene Graph from Interactive Scene Exploration.** To illustrate the construction process of our ACSG in the interactive scene exploration, we depict a scenario wherein a robot arm explores a tabletop scene containing two cabinets and a condiment obstructing the left door. (a) The robot arm actively interacts with the scene, completing the interactive scene exploration process. (b) We showcase the corresponding low-level memory in our ACSG, which represents the geometry and semantic information of the scene. The small graph within each visualization represents a segment of the final scene graph. (c) We present the high-level memory of our action-conditioned scene graph. The graph reveals that picking up the condiment serves as a precondition for opening the door, and opening the bottom drawer allows the observation of the concealed tape and banana.

representations of the environment, but in the form of action-conditioned scene graphs for robotic manipulation.

Robotic exploration aims to autonomously navigate, interact with, and gather information from environments it has never encountered before. It is applicable in search and rescue [1, 2, 58–64], planetary exploration [3, 4, 65, 66], object goal navigation [5, 6, 67–84], and mobile manipulation [7, 8, 85–88]. The primary guiding principle behind robotic exploration is to reduce the uncertainty of the environment [89, 17, 58, 18, 19, 90], making uncertainty quantification key for robotic exploration tasks. Curiosity-driven exploration has recently emerged as a promising approach, showing effective results in various contexts [15, 20, 21, 91]. Most past works have focused on exploration in the context of mobility [92, 1, 2, 58–62, 5, 6, 67–81, 7, 8, 85–88], with the primary goal of modeling and understanding the static environment to complete specific tasks. Recently, exploration has also been studied in the context of manipulation [23, 93, 94, 16, 95], aiming to better understand the scene by changing the state of the environment. Our work introduces a new active exploration strategy for manipulation, uniquely defining a novel scene graph-guided objective to guide the exploration process.

Active perception aims to select specific actions for an agent to improve its ability to perceive and understand the environment [9, 10]. Unlike passive perception, actions offer more flexibility, such as control over better viewpoints [11–13], sensor configurations [96, 14], or adjustments to environmental configurations [97]. It can also reveal certain scene properties that cannot be perceived in a passive manner, such as dynamic parameters [15, 24] or articulation [98, 16, 99]. Previous studies have explored active perception in 3D reconstruction [100, 101], object recognition [102–104], camera localization [105], and robotic manipulation [106, 107]. Our work falls into the category of actively exploring the environment to reveal what's inside or underneath objects. Differing from most previous active perception efforts, which are driven by handcrafted rules [108], information gain [22, 109], or reinforcement learning [15, 25], our approach to active perception is guided by grounding the rich commonsense knowledge encoded in a large language model into an explicit scene graph representation.

Language models for robotics. Large language models (LLMs) [110–112] and large multimodality models (LMMs) [31, 32] are bringing overwhelming influence into the robotics field, for their strong capacity in common-sense knowledge and long-horizon reasoning. Previous studies have harnessed the common-sense knowledge of such large models to generate action candidates [113] and action sequences for task planning [114, 112, 115, 84], and generate code for robotic control and manipulation [116, 33, 117]. More recently, VILA [34] utilized GPT-4V [31, 32] for vision-language planning. In our RoboEXP system, we leverage GPT-4V for decision-making in two crucial roles. First, as the *action proposer*, it ensures both effectiveness and efficiency in proposing appropriate strategies to expand potential nodes in our action-conditioned 3D scene graph. Second, as the *action verifier*, it ensures the plausibility and smoothness of actions and operations in our system. Moreover, instead of memorizing everything using large models in a brute force way, our system employs explicit memory to enhance the decision-making process.

III. Problem Statement

We unfold this section with an introduction of action-conditioned 3D scene graph, a novel scene representation illustrating interactive object relationships (Sec. III-A). We then formulate interactive scene exploration as an action-conditioned 3D scene graph construction and traversal problem (Sec. III-B).

A. Action-Conditioned 3D Scene Graph

An action-conditioned 3D scene graph (ACSG) is an actionable, spatial-topological representation that models objects and their interactive and spatial relations in a scene. Formally, ACSG is a directed acyclic graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ where each node represents either an object (e.g., a *door*) or an action (e.g., *open*), and edges $\mathbf{E}$ represent their interaction relations. The object node $\mathbf{o}_i = (\mathbf{s}_i, \mathbf{p}_i) \in \mathbf{V}$ encodes the semantics and geometry of each object (e.g., the semantic embedding of a fridge $\mathbf{s}_i$, and its shape in the form of a point cloud $\mathbf{p}_i$), whereas the action node $\mathbf{a}_k = (a_k, \mathbf{T}_k) \in \mathbf{V}$ encodes high-level action type a_k and low-level primitives $\mathbf{T}_k$ to perform the actions. Between the nodes are edges encoding their relations, which we categorize into four types: 1) between objects $\mathbf{e}_{\mathbf{o}\to\mathbf{o}}$ (e.g., the *door handle belongs* to the *fridge*), 2) from objects to actions $\mathbf{e}_{\mathbf{o}\to\mathbf{a}}$ (e.g., *toy* can be *picked up*), 3) from action to objects $\mathbf{e}_{\mathbf{a}\to\mathbf{o}}$ (e.g., a *banana* can be reached if we *open* the cabinet), or 4) from one action to another $\mathbf{e}_{\mathbf{a}\to\mathbf{a}}$ (e.g., the cabinet can be *opened* only if we *move away* the *condiment*). Our action-conditioned 3D scene graph greatly enhances existing 3D scene graphs, as it explicitly models the action-conditioned relations between objects. Fig. 2 depicts a complete action-conditioned 3D scene graph of a tabletop scene.

One advantage of our interaction-aware scene graph lies in its simplicity for retrieving and taking actions on an object. Regardless of how complicated the scene is, given our scene graph and a target object, an agent merely needs to sequentially execute all the actions on the paths from the root to the object node in a topological order to retrieve the object. For example, in Fig. 2, to reach the tape inside a cabinet whose door is blocked by a condiment, according to the graph, one simply needs to: 1) pick up the condiment on the table that blocks the cabinet door, and 2) open the cabinet through the door handle.

B. Interactive Exploration

This subsection describes how we can construct a complete action-conditioned scene graph of a real-world scene. This is a challenging problem due to partial observability. For instance, a banana cannot be populated without *opening* the cabinet. To solve this task, we formulate the scene graph construction as an active perception and exploration problem using POMDP-inspired notations. Formally, at each time t, based on our past graph estimation $\mathbf{G}^{t-1}$, and past sensor observations $\mathbf{O}^{t-1}$, our agent takes an action $\mathbf{A}^t$, which causes the environment to transition to a new state, and the agent receives a new observation $\mathbf{O}^t$, which is used to update its current inferred graph $\mathbf{G}^t$. This update might include adding new nodes to the graph or updating the state of an existing node. We will then continue with exploration and keep updating the set of remaining unexplored nodes $\mathbf{U} \subset \mathbf{V}$ (see Algorithm 1).

Algorithm 1 Interactive Exploration

input: $\mathbf{O}^0$, $\mathbf{G}^0 = (\mathbf{V}^0, \mathbf{E}^0), \mathbf{U}^0 \leftarrow \mathbf{V}^0$
while $|\mathbf{U}^{t-1}| \neq 0$ **do**
  if choose object $\mathbf{o}_i \in \mathbf{U}^{t-1}$ **then** `% explore object`
    add spatial relations `% memory`
    obtain action $\mathbf{a}$ to explore $\mathbf{o}_i$ `% decision-making`
    if action $\mathbf{a} \notin \mathbf{V}^{t-1}$ **then**
      $\mathbf{V}^t, \mathbf{U}^t = \mathbf{V}^{t-1} \cup \{\mathbf{a}\}, \mathbf{U}^{t-1} \cup \{\mathbf{a}\}$ `% add node`
      $\mathbf{E}^t = \mathbf{E}^{t-1} \cup \{\mathbf{e}_{\mathbf{o}_i\to\mathbf{a}}\}$ `% add edge`
      $\mathbf{U}^t = \mathbf{U}^t \setminus \mathbf{o}_i$ `% mark as explored`
    end if
  else choose action $\mathbf{a}_k \in \mathbf{U}^{t-1}$
    if no obstruction **then** `% decision-making`
      take action $\mathbf{a}_k$ `% action`
      obtain new observation $\mathbf{O}^t$ `% perception`
      if found new objects $\mathcal{O} \not\subset \mathbf{V}^{t-1}$ **then**
        $\mathbf{V}^t, \mathbf{U}^t = \mathbf{V}^t \cup \{\mathcal{O}\}, \mathbf{U}^{t-1} \cup \{\mathcal{O}\}$ `% add nodes`
        $\mathbf{E}^t = \mathbf{E}^t \cup \{\mathbf{e}_{\mathbf{a}_k\to\mathcal{O}}\}$ `% add edges`
        $\mathbf{U}^t = \mathbf{U}^t \setminus \mathbf{a}_k$ `% mark as explored`
      end if
    else
      add action preconditions `% memory`
    end if
  end if
end while
output: $\mathbf{G}^t$ `% final scene graph`

The goal of the exploration is simple: discover and explore all the nodes of the scene graph in as little time as possible. Towards this, we formulate a reward function with three terms:

$$\mathbf{R}^t = \mathbf{R}^t_{\text{graph}} + \mathbf{R}^t_{\text{explore}} + \mathbf{R}^t_{\text{time}}$$

where $\mathbf{R}^t_{\text{graph}} = |\mathbf{V}^t| - |\mathbf{V}^{t-1}|$ is the graph construction term, which promotes our agent to discover as many nodes as possible to the graph, $\mathbf{R}^t_{\text{explore}} = \max(0, |\mathbf{U}^{t-1}| - |\mathbf{U}^t|)$ gives positive reward to actions that reduce unexplored node set, which prioritize the agent to explore previously unexplored nodes, and immediate reward $\mathbf{R}^t_{\text{time}} = -\lambda, 0 < \lambda < 1$ is a negative time reward that optimizes the time efficiency and allows the exploration to terminate when there is no more node to explore.

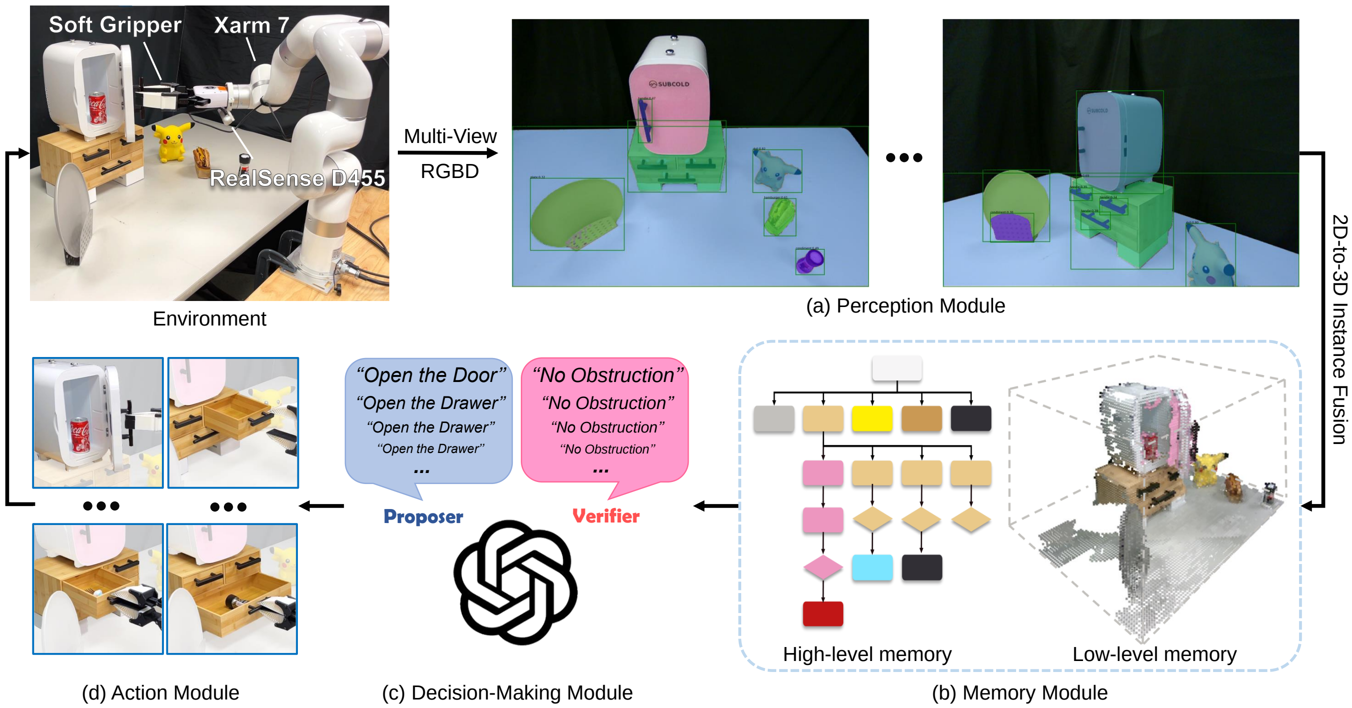


Fig. 3: **Overview of Our RoboEXP System.** We present a comprehensive overview of our RoboEXP system, comprised of four modules. (a) Our **perception module** takes RGBD images as input and produces the corresponding 2D bounding boxes, masks, object labels, and associated semantic features as output. (b) The **memory module** seamlessly integrates 2D information into the 3D space, achieving more consistent 3D instance segmentation. Additionally, it constructs the high-level graph of our ACSG through the merging of instances. (c) Our **decision-making module** serves dual roles as a proposer and verifier. The proposer suggests various actions, such as opening doors and drawers, while the verifier assesses the feasibility of each action, considering factors like obstruction. (d) The **action module** executes the proposed actions, enabling the robot arm to interact effectively with the environment.

Intuitively, to maximize this reward at each discrete timestamp, we should prioritize exploring the unexplored nodes in the current scene graph that are likely to lead to the discovery of new nodes (e.g., opening a cabinet that has not been opened, or lifting a piece of clothing that might cover a small object). The key challenge lies in how we can perceive the objects in the scene, infer possible actions and their relations from the sensory data, and take actions with the current scene graph. In the next section, we will comprehensively describe our system implementation to achieve this goal.

IV. Method

In this section, we outline the structure of our RoboEXP system, including perception, memory, decision-making, and action modules, in Sec. IV-A. We then discuss our system's design for the interactive scene exploration task in Sec. IV-B, focusing on its application in closed-loop exploration processes that may require multi-step or recursive reasoning and handle potential interventions.

A. RoboEXP System

To tackle the task outlined in Section Sec. III, we present our RoboEXP system, designed to autonomously explore unknown environments by observing and interacting with them. The system comprises four key components: perception, memory, decision-making, and action modules (see Fig. 3). Raw RGBD images are captured through the wrist camera in different viewpoints and processed by the perception modules to extract scene semantics, including object labels, 2D bounding boxes, segmentations, and semantic features. The obtained semantic information is then transmitted to the memory module, where the 2D data is merged into the 3D representation. Such 3D information serves as a valuable guide for the decision module, aiding in the selection of appropriate actions to further interact or observe the environment and unveil hidden objects. The action module is activated to execute the planned action, generating new observations for the perception modules. This closed-loop system ensures the thoroughness of our task in interactive scene exploration.

Perception Module. Given multiple RGBD observations from different viewpoints, the objective of the perception module (Fig. 3a) is to detect and segment objects while extracting their semantic features. To enhance generality, we opt for the open-vocabulary detector GroundingDINO [27] and the Segment Anything in High Quality (SAM-HQ) [28], an advanced version of SAM [29]. For the extraction of semantic features used in subsequent instance merging within the memory module, we employ CLIP [30]. To obtain per-instance CLIP features, we implement a strategy similar to the one proposed by Jatavallabhula et al. [118]. Specifically, we extend the local-global image feature merging approach by incorporating additional label text features to augment the semantic CLIP feature for each instance. Furthermore, we exclusively focus on instance-level features, disregarding pixel-level features, thereby accelerating the entire semantic feature extraction process.

Memory Module. The memory module (Fig. 3b) is designed to construct our ACSG of the environment by assimilating observations over time. For the low-level memory, to ensure stable instance merging from 2D to 3D, we employ a similar instance merging strategy as presented in Lu et al. [119], consolidating observations from diverse RGBD sources across various viewpoints and time steps. In contrast to the original algorithm, which considers only 3D IoU and semantic feature similarity we additionally incorporate label similarity and instance confidence. To enhance algorithm efficiency, we represent low-level memory using a voxel-based representation, which allows for more efficient computation and memory updates. Meanwhile, given the crowded nature of objects in our tabletop setting, we have implemented voxel-based filtering designs to obtain a cleaner and more complete representation of the objects for storage in our memory.

The memory module handles merging across different viewpoints and time steps. To merge across different viewpoints, we project 2D information (RGBD, semantic features, mask, bounding box) to 3D and leverage the instance merging strategy mentioned earlier to attain consistent 3D information. Addressing memory updates across time steps presents a challenge due to dynamic changes in the environment. For instance, a closed door in the previous time step may be opened by our robot in the current time step. To accurately reflect such changes, our algorithm evaluates whether elements within our memory have become outdated, primarily through depth tests based on the most recent observations. This process ensures that the memory accurately represents the environment's current state, effectively managing scenarios where objects may change positions or states across different time steps.

For the high-level graph of our ACSG, the memory module analyzes the relationships between objects and the logical associations between actions and objects. Depending on changes in low-level memory and relationships, the memory module is tasked with updating the graph. This involves adding, deleting, or modifying nodes and edges within our graph.

Decision-Making Module. The primary goal of the decision module (Fig. 3c) is to identify the appropriate object and corresponding skill to enhance the effectiveness and efficiency of interactive scene exploration. In the context of our task, distinct objects may necessitate distinct exploration strategies. While humans can easily discern the most suitable skill to apply (e.g., picking up the top Matryoshka doll to inspect its contents), achieving such decisions through heuristic-based methods is challenging. The utilization of a Large Multi-Modal Model (LMM), such as GPT-4V [31, 32], shows instrumental in addressing this difficulty, as it captures commonsense knowledge that facilitates decision-making.

The LMM brings commonsense knowledge to our decision-making process and serves in two pivotal roles. Firstly, it functions as an action proposer. Given the current digital environment from the memory module, GPT-4V is tasked with selecting the appropriate skill for unexplored objects in our system. For instance, when presented with a visual prompt of an object within a green bounding box from various viewpoints, GPT-4V can discern the suitable "pick up" skill for the Matryoshka doll in the environment. For unexplored objects, our ACSG includes the attribute of whether each object node is explored or unexplored. GPT-4V, in its role as the proposer, also functions to assess whether the object holds value for further exploration. If not, the corresponding node is marked as explored, indicating that no further actions are needed.

Secondly, the LMM also serves as the action verifier. For the proposer role, it analyzes the object-centric attributes and doesn't consider surrounding information when choosing the proper skill. For example, if the proposed action involves opening a door, the proposer alone may struggle with cases where obstructions exist in front of the door (e.g., a condiment bottle). To address this, we use another LMM program to verify the feasibility of the action and identify any objects in the scene that may impede the action based on information from our ACSG.

In summary, the decision module, with its dual roles, effectively guides our system to choose efficient actions that minimize uncertainty in the environment and successfully locate all relevant objects.

Action Module. In the action module (Fig. 3d), our primary focus is on autonomously constructing the ACSG through effective and efficient interaction with the environment. We employ heuristic-based action primitives within our action module, leveraging the geometry cues in our ACSG. These primitives encompass seven categories: "open the door", "open the drawer", "close the door", "close the drawer", "pick object to idle space", "pick back object", "move wrist camera to position". Strategic utilization of these skills plays a pivotal role in accomplishing intricate tasks seamlessly within our system (more details in the Appendix).

B. *Other Design in Interactive Exploration*

One desiderata for robot exploration is the ability to handle scenarios that necessitate multi-step or recursive reasoning. An example of this is the Matryoshka doll case (Fig. 6b), which cannot be addressed using previous one-step LLM-based code generation approaches [34, 33]. In contrast, our modular design allows agents to dynamically plan and adapt in a closed-loop manner, enabling continuous LLM-based exploration based on environmental feedback.

To manage multi-step reasoning, our system incorporates an action stack as a simple but effective "planning" module. Guided by decisions from the decision module, the stack structure adeptly organizes the order of actions. For instance, upon picking up the top Matryoshka doll, if the perception and memory modules identify another smaller Matryoshka doll in the environment, the decision module determines to pick it up. Our action stack dynamically adds this pickup action to the top of the stack, prioritizing the new action over picking back the previous, larger Matryoshka doll. This stack structure facilitates multi-step reasoning and constructs the system's logic in a deep and coherent structure.

Moreover, for the interactive scene exploration task, maintaining scene consistency is crucial in practice (e.g., the agent should close the fridge after exploring it). We employ a greedy strategy returning objects to their original states. This approach keeps the environment close to its pre-exploration state, making RoboEXP more practical for real-world applications.

Fig. 4: **All Testing Objects.** We present various objects utilized in our work, encompassing different types of cabinets, fruits, dolls, condiments, beverages, food items, tapes, tableware, and fabric.

V. EXPERIMENTS

In this section, we assess the performance of our system across a variety of tabletop scenarios in the interactive scene exploration setting. Our primary objective is to address two key questions through experiments: 1) How does our system effectively and efficiently deal with diverse exploration scenarios and successfully construct comprehensive ACSG? 2) What is the utility of our ACSG in facilitating downstream tasks?

A. Robot and Environment Setups

All our experiments are conducted in a real-world setting. In these scenarios, we mount one RealSense-D455 camera on the wrist of the robot arm to collect RGBD observations, with the execution of actions performed by the UFACTORY xArm 7. The end effector for our robot arm is the soft gripper (see Fig. 3). Our experimental setup encompasses a diverse range of objects, as illustrated in Fig. 4.

B. Interactive Exploration and Scene Graph Building

To assess our system's efficacy across various exploration scenarios, we compared it with a strong baseline by augmenting GPT-4V with ground truth actions. We designed five types of experiments, each with 10 different settings varying in object number, type, and layout (see the Appendix for more details). Our quantitative analysis reveals that our RoboEXP system consistently surpasses the baseline across various tasks. Furthermore, we validate the performance of our system in constructing ACSG through qualitative demonstrations.

Baseline. We employ the pure GPT-4V as our baseline model along with the chain-of-thoughts (CoT) to enhance its capabilities, as outlined in a method similar to that proposed by Hu et al. [34]. This baseline operates in a closed-loop fashion, receiving three RGB observations from different viewpoints during each iteration. At each turn, it generates the current scene graph, encompassing hidden objects, and suggests the next action to be taken. Upon determining that all tasks are completed, the model outputs "Done" (refer to the complete prompts in the Appendix). To ensure the baseline is robust, we utilize manual actions as ground truth references for the proposed actions. For instance, if the baseline suggests opening a specific drawer, we manually perform the action and prompt the model with the new observation to generate another action. In contrast, in the exploration experiments described below, all actions from our system are automatically executed by our action module on the physical robot.

Evaluation. To thoroughly assess the efficacy of our system compared to the baseline, we have designed five key metrics to measure its performance. It is crucial to note that the output of our task, represented by ACSG, aligns precisely with the format of ACSG for our system. Conversely, for the baseline, we manually construct ACSG based on its actions and the new observations it uncovers. Due to the unstructured nature of the raw scene graph from the baseline, we carefully refine it according to the observable objects, providing an upper-bound baseline for comparison during evaluation.

To assess the effectiveness and efficiency of ACSG, we engage human evaluators in the tasks to construct the ground truth version of ACSG. The five main metrics employed for evaluation are as follows:

1) **Success:** This metric evaluates the success percentage across 10 variants for each task. We define success for each experiment as 1 when the final outputted ACSG exactly matches the GT version, and 0 otherwise.

2) **Object Recovery:** This metric quantifies the percentage of hidden objects successfully identified.

3) **State Recovery:** A binary value indicates whether the final state resembles the original state before exploration. This includes considerations for partial states and object positions (e.g., in the top drawer of a cabinet or on the table).

4) **Unexplored Space:** Evaluating the percentage of successfully explored need-to-explore space to reduce the robot's uncertainty about the scene. The identification of the need-to-explore space relies on human annotation.

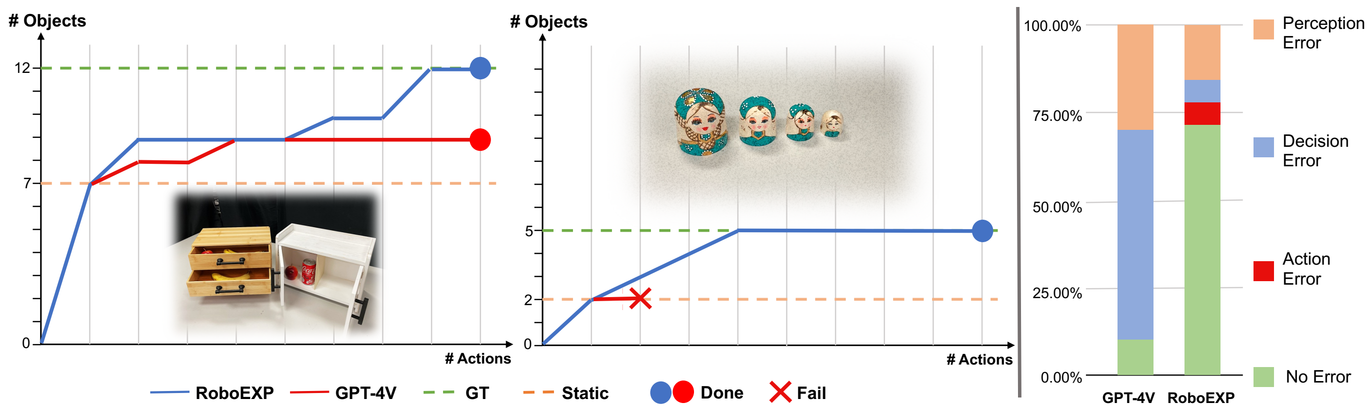
5) **Graph Edit Distance (GED):** GED measures the disparity between the outputted graph and the GT graph. We adopt a simplified version of GED with six operations—three for nodes (add, delete, edit) and three for edges (add, delete, edit), with each operation incurring a cost of 1.

These metrics provide a comprehensive evaluation of the method's performance. Additionally, we visualize the number of objects and actions during the exploration process to show the exploration strategies employed by different methods.

Comparison. The quantitative findings presented in Tab. I underscore the superior performance of our system compared to the baseline method. Our approach showcases a notable enhancement across all metrics, outperforming the baseline by a considerable margin. The collective assessment of success rate, object recovery, and unexplored space metrics unequivocally validates the efficacy of our system in exploring unfamiliar scenes through interactive processes. It is essential to highlight that in the case of object recovery, the baseline method may occasionally choose to randomly open certain drawers or doors

TABLE I: **Quantitative Results on Different Tasks.** We compare the performance of both the GPT-4V baseline and our system across various tasks. We assess the outcomes using five distinct metrics to illustrate diverse facets of the interactive exploration process. Our system consistently outperforms the baseline across all tasks and metrics.

Task (10 variance for each)	Drawer-Only		Door-Only		Drawer-Door		Recursive		Occlusion	
Metric	GPT-4V	Ours	GPT-4V	Ours	GPT-4V	Ours	GPT-4V	Ours	GPT-4V	Ours
Success % ↑	20±13.3	**90**±10.0	30±15.2	**90**±10.0	10±10.0	**70**±15.3	0±0.0	**70**±15.3	0±0.0	**50**±16.7
Object Recovery % ↑	83±11.0	**97**±3.3	50±16.7	**100**±0.0	62±10.7	**91**±4.7	20±13.3	**80**±11.7	17±11.4	**67**±14.9
State Recovery % ↑	60±16.3	**100**±0.0	80±13.3	**100**±0.0	70±15.3	**100**±0.0	70±15.3	**100**±0.0	10±10.0	**70**±15.3
Unexplored Space % ↓	15±7.6	**0**±0.0	40±14.5	**0**±0.0	25±6.5	**0**±0.0	63±15.3	**15**±8.9	85±7.6	**30**±15.3
Graph Edit Dist. ↓	2.8±1.04	**0.2**±0.20	4.4±1.42	**0.1**±0.10	5.6±1.46	**0.5**±0.27	8.8±2.06	**2.1**±1.49	7.3±0.97	**2.5**±1.15



(a) Action-object Graph (b) Error Breakdown

Fig. 5: **Visualization of Quantitative Results.** (a) The action-object graph captures the change in the number of discovered objects relative to the number of actions taken. Our RoboEXP efficiently discovers all objects. Sometimes, the object count doesn't increase during actions due to the absence of objects in storage after opening. Additionally, some actions are employed to restore the scene state (e.g., closing the door after exploration). (b) The error breakdown of all our quantitative experiments includes 5 task settings with 10 variations each. We categorize errors into perception, decision, action, and no-error cases. For the GPT-4V baseline, manual assistance in action execution eliminates failure cases, serving as an upper bound for baseline performance. Even in this scenario, our RoboEXP largely outperforms the baseline.

to unveil objects. This randomness contributes to a seemingly higher object recovery rate for the baseline, which may not necessarily correlate with its overall success. The unexplored space metric shows that our system is much more stable in exploring all need-to-explore spaces.

Moreover, both the success rate and graph edit distance underscore the close alignment of our system with human actions, highlighting the efficiency of our approach across diverse scenarios. The state recovery metric assesses whether the final state post-exploration resembles the initial state. Our system consistently shows effective state recovery; however, the baseline may trick this metric by opting not to take any action, resulting in an artificially high score in this aspect.

Fig. 5a provides additional insights, illustrating that as the number of actions increases, so does the number of objects. Specifically, we present the ground truth object number alongside the directly-observable object number that can be represented by the traditional 3D scene graph. These results underscore our system's ability to achieve robust and efficient exploration throughout the exploration process. Our system excels in efficiently discovering all concealed objects, whereas the baseline fails either due to a lack of early-stage actions or an inability to explore all need-to-explore spaces even upon completion. The analysis of errors (Fig. 5b) in both our system and the baseline reveals the specific failure cases encountered by the baselines. In contrast, our system demonstrates enhanced robustness in both perception and decision-making.

Fig. 6 further illustrates various exploration scenarios along with their corresponding ACSG. These scenarios encompass ACSG with varying width or depth, highlighting our system's adaptive capability across diverse objects such as rigid, articulated objects, nested objects, and deformable objects. In addition, the scenario in Fig. 2 shows that our system is able to deal with the scenario with obstruction.

C. Utility of our ACSG

The scenarios depicted in Fig. 1 exemplify the efficacy of our generated output (ACSG) in manipulation tasks. Consider the table-rearranging scenario: without our ACSG, the robot struggles to swiftly prepare the table due to the lack of precise prior knowledge about the location of objects (e.g., the fork stored in the top-left drawer of the wooden cabinet). Beyond comprehensive layout guidance, our ACSG also addresses a crucial question regarding task feasibility for the robot. For instance, if there is no spoon in the scene, the robot recognizes its inability to perform the task and asks for human help.

In addition to enhancing downstream manipulation tasks, our ACSG possesses the capability to autonomously adapt to environmental changes. In the human intervention setting, our system seamlessly explores newly added components, such as a cabinet, ensuring continuous adaptability. Check our Appendix and supplemental video for more details.

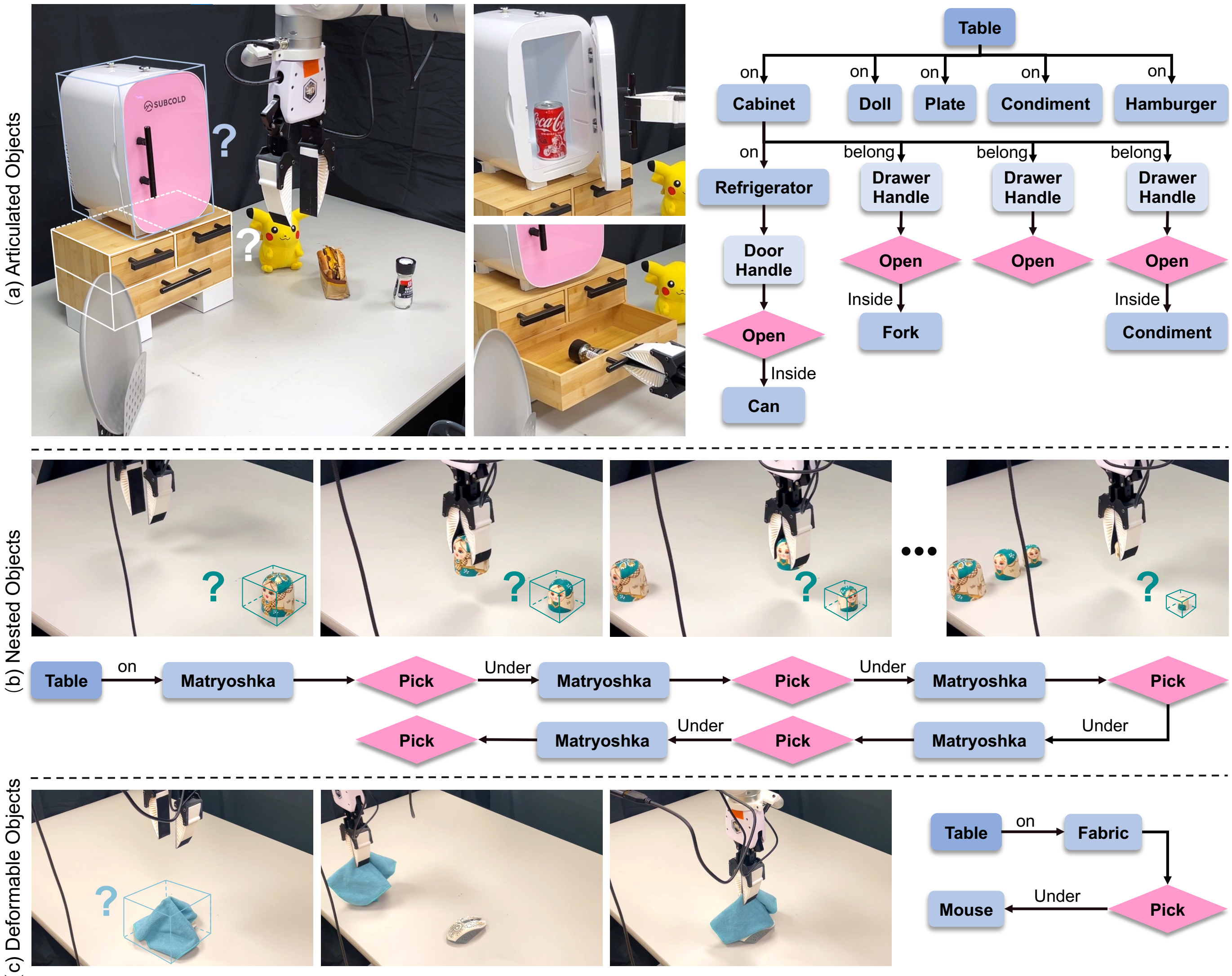


Fig. 6: **Qualitative Results on Different Scenarios.** We visualize the interactive exploration process and the corresponding constructed ACSG. (a) This scenario involves a tabletop environment with two articulated objects, accompanied by additional items either on the table or concealed in storage space. The constructed scene graph demonstrates the success of our system in identifying all objects within the environment through a series of physical interactions. (b) This scenario includes nested objects, five Matryoshka dolls, with only the top one being directly observable. Our system autonomously decides to explore the contents through a recursive reasoning process, showcasing its ability to construct deep ACSG. (c) This scenario involves a fabric covering a mouse, showcasing exploration scenarios that involve a deformable object. Our system interacts with the fabric and successfully uncovers what lies beneath it.

D. Remaining Challenges

Although our system has proven effective, there is room for improvement. The breakdown of the failure rate in Fig. 5.b suggests that failures primarily arise from detection and segmentation errors within the perception module. To address this issue, we envision two future directions: 1) enhancing the capabilities of visual foundation models for open-world semantic understanding, and 2) utilizing temporal cues and semantic fusion techniques to improve perception robustness through continuous observations.

Furthermore, our system would benefit from enhanced LMM capacities and the integration of sophisticated skill modules, including learning-based or model-based path planning. Such improvements would improve both the decision-making and action modules, thereby further reducing failure cases.

VI. Conclusion

We introduced RoboEXP, a foundation-model-driven robotic exploration framework capable of effectively identifying all objects in a complex scene, both directly observable and those revealed through interaction. Central to our system is action-conditioned 3D scene graph, an advanced 3D scene graph that goes beyond traditional models by explicitly modeling interactive relations between objects. Experiments have shown RoboEXP's superior performance in interactive scene exploration across various challenging scenarios, significantly outperforming a strong GPT4V-based baseline. Notably, the reconstructed action-conditioned 3D scene graph is crucial for guiding complex downstream manipulation tasks, like preparing breakfast in a mock-kitchen environment with fridges, cabinets, and drawer sets. Our system and its action-conditioned scene graph lay the groundwork for practical robotic deployment in complex settings, especially in environments like households and offices, facilitating their everyday use.

Acknowledgements

Yunzhu Li is partly supported by the Amazon AICE Award. Shenlong Wang is supported by the Amazon AICE Award, IBM IIDAI Grant, Nvidia Hardware Grants, the Insper-Illinois Innovative Grant, the NCSA Faculty Fellow, NSF Award #2331878 and #2312102, and gifts from Intel and Meta. This work does not relate to the positions of Shubham Garg and Hooshang Nayyeri at Amazon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the sponsors.

References

[1] Farzad Niroui, Kaicheng Zhang, Zendai Kashino, and Goldie Nejat. Deep reinforcement learning robot for search and rescue applications: Exploration in unknown cluttered environments. *RA-L*, 2019. 2, 3

[2] Yugang Liu and Goldie Nejat. Robotic urban search and rescue: A survey from the control perspective. *Journal of Intelligent & Robotic Systems*, 2013. 3

[3] Philip Arm, Gabriel Waibel, Jan Preisig, Turcan Tuna, Ruyi Zhou, Valentin Bickel, Gabriela Ligeza, Takahiro Miki, Florian Kehl, Hendrik Kolvenbach, et al. Scientific exploration of challenging planetary analog environments with a team of legged robots. *Science robotics*, 2023. 3

[4] Martin J Schuster, Marcus G Müller, Sebastian G Brunner, Hannah Lehner, Peter Lehner, Ryo Sakagami, Andreas Dömel, Lukas Meyer, Bernhard Vodermayer, Riccardo Giubilato, et al. The arches space-analogue demonstration mission: Towards heterogeneous teams of autonomous robots for collaborative scientific sampling in planetary exploration. *RA-L*, 2020. 3

[5] KAI-QING Zhou, Kai Zheng, Connor Pryor, Yilin Shen, Hongxia Jin, L. Getoor, and X. Wang. Esc: Exploration with soft commonsense constraints for zero-shot object navigation. *ICML*, 2023. 3

[6] Ram Ramrakhya, Eric Undersander, Dhruv Batra, and Abhishek Das. Habitat-web: Learning embodied object-search strategies from human demonstrations at scale. In *CVPR*, 2022. 3

[7] Dipendra Misra, Andrew Bennett, Valts Blukis, Eyvind Niklasson, Max Shatkhin, and Yoav Artzi. Mapping instructions to actions in 3d environments with visual goal prediction. *arXiv preprint arXiv:1809.00786*, 2018. 3

[8] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *CVPR*, 2020. 3

[9] R. Bajcsy. Active perception. *Proceedings of the IEEE*, 1988. 3

[10] Active perception vs. passive perception. In *Proc. of IEEE Workshop on Computer Vision*, 1985. 3

[11] Andreas Bircher, Mina Kamel, Kostas Alexis, Helen Oleynikova, and Roland Siegwart. Receding horizon" next-best-view" planner for 3d exploration. In *ICRA*, 2016. 3

[12] Ana Batinovic, Antun Ivanovic, Tamara Petrovic, and Stjepan Bogdan. A shadowcasting-based next-best-view planner for autonomous 3d exploration. *RA-L*, 2022.

[13] Menaka Naazare, Francisco Garcia Rosas, and Dirk Schulz. Online next-best-view planner for 3d-exploration and inspection with a mobile manipulator robot. *RA-L*, 2022. 2, 3

[14] Peihao Chen, Dongyu Ji, Kunyang Lin, Weiwen Hu, Wenbing Huang, Thomas Li, Mingkui Tan, and Chuang Gan. Learning active camera for multi-object navigation. *NeurIPS*, 2022. 2, 3

[15] Tushar Nagarajan and Kristen Grauman. Learning affordance landscapes for interaction exploration in 3d environments. In *NeurIPS*, 2020. 2, 3

[16] Neil Nie, Samir Yitzhak Gadre, Kiana Ehsani, and Shuran Song. Structure from action: Learning interactions for articulated object 3d structure discovery. *arXiv preprint arXiv:2207.08997*, 2022. 2, 3

[17] Benjamin Charrow, Gregory Kahn, Sachin Patil, Sikang Liu, Ken Goldberg, Pieter Abbeel, Nathan Michael, and Vijay Kumar. Information-theoretic planning with trajectory optimization for dense 3d mapping. In *RSS*, 2015. 2, 3

[18] Georgios Georgakis, Bernadette Bucher, Anton Arapin, Karl Schmeckpeper, Nikolai Matni, and Kostas Daniilidis. Uncertainty-driven planner for exploration and navigation. In *ICRA*, 2022. 3

[19] Christos Papachristos, Shehryar Khattak, and Kostas Alexis. Uncertainty-aware receding horizon exploration and mapping using aerial robots. In *ICRA*, 2017. 3

[20] Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A. Efros. Large-scale study of curiosity-driven learning. In *ICLR*, 2019. 3

[21] Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *ICML*, 2017. 3

[22] Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Embodied question answering. In *CVPR*, 2018. 2, 3

[23] Pulkit Agrawal, Ashvin V Nair, Pieter Abbeel, Jitendra Malik, and Sergey Levine. Learning to poke by poking: Experiential learning of intuitive physics. *NeurIPS*, 2016. 2, 3

[24] Yian Wang, Ruihai Wu, Kaichun Mo, Jiaqi Ke, Qingnan Fan, Leonidas Guibas, and Hao Dong. AdaAfford: Learn-

ing to adapt manipulation affordance for 3d articulated objects via few-shot interactions. In *ECCV*, 2022. 2, 3

[25] Steven D Whitehead and Dana H Ballard. Active perception and reinforcement learning. In *Machine Learning Proceedings 1990*. 1990. 2, 3

[26] Fei Xia, William B Shen, Chengshu Li, Priya Kasimbeg, Micael Edmond Tchapmi, Alexander Toshev, Roberto Martín-Martín, and Silvio Savarese. Interactive gibson benchmark: A benchmark for interactive navigation in cluttered environments. *RA-L*, 2020. 2

[27] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023. 2, 5

[28] Lei Ke, Mingqiao Ye, Martin Danelljan, Yifan Liu, Yu-Wing Tai, Chi-Keung Tang, and Fisher Yu. Segment anything in high quality. *arXiv preprint arXiv: 2306.01567*, 2023. 2, 5

[29] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023. 2, 5

[30] Alec Radford, Jong Wook Kim, Chris Hallacy, A. Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *ICML*, 2021. 2, 5

[31] OpenAI. Gpt-4v(ision) system card. *https://cdn.openai.com/papers/GPTV System Card.pdf*, 2023. 2, 3, 4, 6

[32] Zhengyuan Yang, Linjie Li, Kevin Lin, Jianfeng Wang, Chung-Ching Lin, Zicheng Liu, and Lijuan Wang. The dawn of lmms: Preliminary explorations with gpt-4v(ision). *arXiv preprint arXiv: 2309.17421*, 2023. 2, 3, 4, 6

[33] Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023. 2, 4, 6

[34] Yingdong Hu, Fanqi Lin, Tong Zhang, Li Yi, and Yang Gao. Look before you leap: Unveiling the power of gpt-4v in robotic vision-language planning. *arXiv preprint arXiv: 2311.17842*, 2023. 2, 4, 6, 7

[35] Justin Johnson, Ranjay Krishna, Michael Stark, Li-Jia Li, David Shamma, Michael Bernstein, and Li Fei-Fei. Image retrieval using scene graphs. In *CVPR*, 2015. 2

[36] Danfei Xu, Yuke Zhu, Christopher B Choy, and Li Fei-Fei. Scene graph generation by iterative message passing. In *CVPR*, 2017. 2

[37] Cewu Lu, Ranjay Krishna, Michael Bernstein, and Li Fei-Fei. Visual relationship detection with language priors. In *ECCV*, 2016. 2

[38] Ruihai Wu, Kehan Xu, Chenchen Liu, Nan Zhuang, and Yadong Mu. Localize, assemble, and predicate: Contextual object proposal embedding for visual relation detection. In *AAAI*, 2020.

[39] Hanwang Zhang, Zawlin Kyaw, Shih-Fu Chang, and Tat-Seng Chua. Visual translation embedding network for visual relation detection. In *CVPR*, 2017. 2

[40] Jianwei Yang, Jiasen Lu, Stefan Lee, Dhruv Batra, and Devi Parikh. Graph r-cnn for scene graph generation. In *ECCV*, 2018. 2

[41] Iro Armeni, Zhi-Yang He, JunYoung Gwak, Amir R Zamir, Martin Fischer, Jitendra Malik, and Silvio Savarese. 3d scene graph: A structure for unified semantics, 3d space, and camera. In *ICCV*, 2019. 2

[42] Jared Strader, Nathan Hughes, William Chen, Alberto Speranzon, and Luca Carlone. Indoor and outdoor 3d scene graph generation via language-enabled spatial ontologies. *arXiv preprint arXiv:2312.11713*, 2023. 2

[43] Xu Yang, Kaihua Tang, Hanwang Zhang, and Jianfei Cai. Auto-encoding scene graphs for image captioning. In *CVPR*, 2019. 2

[44] Jonathan Krause, Justin Johnson, Ranjay Krishna, and Li Fei-Fei. A hierarchical approach for generating descriptive image paragraphs. In *CVPR*, 2017. 2

[45] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image generation from scene graphs. In *CVPR*, 2018. 2

[46] Marcel Hildebrandt, Hang Li, Rajat Koner, Volker Tresp, and Stephan Günnemann. Scene graph reasoning for visual question answering. *arXiv preprint arXiv:2007.01072*, 2020. 2

[47] Zachary Ravichandran, Lisa Peng, Nathan Hughes, J Daniel Griffith, and Luca Carlone. Hierarchical representations and explicit memory: Learning effective navigation policies on 3d scene graphs using graph neural networks. In *ICRA*, 2022. 2

[48] Zachary Seymour, Niluthpol Chowdhury Mithun, Han-Pang Chiu, Supun Samarasekera, and Rakesh Kumar. Graphmapper: Efficient visual navigation by scene graph generation. In *ICPR*, 2022. 2

[49] Krishan Rana, Jesse Haviland, Sourav Garg, Jad Abou-Chakra, Ian Reid, and Niko Suenderhauf. Sayplan: Grounding large language models using 3d scene graphs for scalable task planning. *arXiv preprint arXiv:2307.06135*, 2023. 2

[50] Christopher Agia, Krishna Murthy Jatavallabhula, Mohamed Khodeir, Ondrej Miksik, Vibhav Vineet, Mustafa Mukadam, Liam Paull, and Florian Shkurti. Taskography: Evaluating robot task planning over large 3d scene graphs. In *CoRL*, 2022. 2

[51] Jiayuan Mao, Tomás Lozano-Pérez, Joshua B Tenenbaum, and Leslie Pack Kaelbling. Learning reusable manipulation strategies. In *CoRL*, 2023. 2

[52] Kexin Yi, Chuang Gan, Yunzhu Li, Pushmeet Kohli, Jiajun Wu, Antonio Torralba, and Joshua B Tenenbaum. Clevrer: Collision events for video representation and reasoning. *arXiv preprint arXiv:1910.01442*, 2019. 2

[53] Renhao Wang, Jiayuan Mao, Joy Hsu, Hang Zhao, Jiajun

Wu, and Yang Gao. Programmatically grounded, compositionally generalizable robotic manipulation. *ICLR*, 2023. 2

[54] Zhutian Yang, Jiayuan Mao, Yilun Du, Jiajun Wu, Joshua B. Tenenbaum, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. Compositional Diffusion-Based Continuous Constraint Solvers. In *CoRL*, 2023. 2

[55] Weiyu Liu, Jiayuan Mao, Joy Hsu, Tucker Hermans, Animesh Garg, and Jiajun Wu. Composable part-based manipulation. In *CoRL*, 2023.

[56] Zhenfang Chen, Kexin Yi, Yunzhu Li, Mingyu Ding, Antonio Torralba, Joshua B Tenenbaum, and Chuang Gan. Comphy: Compositional physical reasoning of objects and events from videos. In *ICLR*, 2022.

[57] Jiayuan Mao, Tomas Lozano-Perez, Joshua B. Tenenbaum, and Leslie Pack Kaelbling. PDSketch: Integrated Domain Programming, Learning, and Planning. In *NeurIPS*, 2022. 2

[58] Farzad Niroui, Ben Sprenger, and Goldie Nejat. Robot exploration in unknown cluttered environments when dealing with uncertainty. In *IRIS*, 2017. 3

[59] Barzin Doroodgar, Yugang Liu, and Goldie Nejat. A learning-based semi-autonomous controller for robotic exploration of unknown disaster scenes while searching for victims. *IEEE Transactions on Cybernetics*, 2014.

[60] Nicola Basilico and Francesco Amigoni. Exploration strategies based on multi-criteria decision making for searching environments in rescue operations. *Autonomous Robots*, 2011.

[61] Yongguo Mei, Yung-Hsiang Lu, CS George Lee, and Y Charlie Hu. Energy-efficient mobile robot exploration. In *ICRA*, 2006.

[62] Stefan Oßwald, Maren Bennewitz, Wolfram Burgard, and Cyrill Stachniss. Speeding-up robot exploration by exploiting background information. *RA-L*, 2016. 3

[63] Matej Petrlik, Pavel Petracek, Vit Kratky, Tomas Musil, Yurii Stasinchuk, Matous Vrba, Tomas Baca, Daniel Hert, Martin Pecka, Tomas Svoboda, et al. Uavs beneath the surface: Cooperative autonomy for subterranean search and rescue in darpa subt. *arXiv preprint arXiv:2206.08185*, 2022.

[64] Marco Tranzatto, Takahiro Miki, Mihir Dharmadhikari, Lukas Bernreiter, Mihir Kulkarni, Frank Mascarich, Olov Andersson, Shehryar Khattak, Marco Hutter, Roland Siegwart, et al. Cerberus in the darpa subterranean challenge. *Science Robotics*, 2022. 3

[65] Florian Cordes, Ingo Ahrns, Sebastian Bartsch, Timo Birnschein, Alexander Dettmann, Stéphane Estable, Stefan Haase, Jens Hilljegerdes, David Koebel, Steffen Planthaber, et al. Lunares: Lunar crater exploration with heterogeneous multi robot systems. *Intelligent Service Robotics*, 2011. 3

[66] Takahiro Sasaki, Kyohei Otsu, Rohan Thakker, Sofie Haesaert, and Ali-akbar Agha-mohammadi. Where to map? iterative rover-copter path planning for mars exploration. *RA-L*, 2020. 3

[67] Albert J Zhai and Shenlong Wang. Peanut: predicting and navigating to unseen targets. In *ICCV*, 2023. 3

[68] Wei Yang, Xiaolong Wang, Ali Farhadi, Abhinav Gupta, and Roozbeh Mottaghi. Visual semantic navigation using scene priors. *arXiv preprint arXiv:1810.06543*, 2018.

[69] Oleksandr Maksymets, Vincent Cartillier, Aaron Gokaslan, Erik Wijmans, Wojciech Galuba, Stefan Lee, and Dhruv Batra. Thda: Treasure hunt data augmentation for semantic navigation. In *ICCV*, 2021.

[70] Apoorv Khandelwal, Luca Weihs, Roozbeh Mottaghi, and Aniruddha Kembhavi. Simple but effective: Clip embeddings for embodied ai. In *CVPR*, 2022.

[71] Karmesh Yadav, Ram Ramrakhya, Arjun Majumdar, Vincent-Pierre Berges, Sachit Kuhar, Dhruv Batra, Alexei Baevski, and Oleksandr Maksymets. Offline visual representation learning for embodied navigation. *arXiv preprint arXiv:2204.13226*, 2022.

[72] Heming Du, Xin Yu, and Liang Zheng. Learning object relation graph and tentative policy for visual navigation. In *ECCV*, 2020.

[73] Joel Ye, Dhruv Batra, Abhishek Das, and Erik Wijmans. Auxiliary tasks and exploration enable objectgoal navigation. In *ICCV*, 2021.

[74] Matthew Chang, Arjun Gupta, and Saurabh Gupta. Semantic visual navigation by watching youtube videos. *NeurIPS*, 2020.

[75] Matt Deitke, Eli VanderBilt, Alvaro Herrasti, Luca Weihs, Kiana Ehsani, Jordi Salvador, Winson Han, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. Procthor: Large-scale embodied ai using procedural generation. *NeurIPS*, 2022.

[76] Abhishek Kadian, Joanne Truong, Aaron Gokaslan, Alexander Clegg, Erik Wijmans, Stefan Lee, Manolis Savva, Sonia Chernova, and Dhruv Batra. Sim2real predictivity: Does evaluation in simulation predict real-world performance? *RA-L*, 2020.

[77] Erik Wijmans, Abhishek Kadian, Ari Morcos, Stefan Lee, Irfan Essa, Devi Parikh, Manolis Savva, and Dhruv Batra. Dd-ppo: Learning near-perfect pointgoal navigators from 2.5 billion frames. *arXiv preprint arXiv:1911.00357*, 2019.

[78] Georgios Georgakis, Bernadette Bucher, Karl Schmeckpeper, Siddharth Singh, and Kostas Daniilidis. Learning to map for active semantic goal navigation. *arXiv preprint arXiv:2106.15648*, 2021.

[79] Santhosh K Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alex Clegg, John Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, et al. Habitat-matterport 3d dataset (hm3d): 1000 large-scale 3d environments for embodied ai. *arXiv preprint arXiv:2109.08238*, 2021.

[80] Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural slam. *arXiv preprint arXiv:2004.05155*, 2020.

[81] Haokuan Luo, Albert Yue, Zhang-Wei Hong, and Pulkit

Agrawal. Stubborn: A strong baseline for indoor object navigation. In *IROS*, 2022. 3

[82] Santhosh Kumar Ramakrishnan, Devendra Singh Chaplot, Ziad Al-Halah, Jitendra Malik, and Kristen Grauman. Poni: Potential functions for objectgoal navigation with interaction-free learning. In *CVPR*, 2022.

[83] Naoki Yokoyama, Sehoon Ha, Dhruv Batra, Jiuguang Wang, and Bernadette Bucher. Vlfm: Vision-language frontier maps for zero-shot semantic navigation. *arXiv preprint arXiv:2312.03275*, 2023.

[84] Yinpei Dai, Run Peng, Sikai Li, and Joyce Chai. Think, act, and ask: Open-world interactive personalized robot navigation. *arXiv preprint arXiv:2310.07968*, 2023. 3

[85] Luca Weihs, Matt Deitke, Aniruddha Kembhavi, and Roozbeh Mottaghi. Visual room rearrangement. In *CVPR*, 2021. 3

[86] Dhruv Batra, Angel X Chang, Sonia Chernova, Andrew J Davison, Jia Deng, Vladlen Koltun, Sergey Levine, Jitendra Malik, Igor Mordatch, Roozbeh Mottaghi, et al. Rearrangement: A challenge for embodied ai. *arXiv preprint arXiv:2011.01975*, 2020.

[87] Fei Xia, Chengshu Li, Roberto Martín-Martín, Or Litany, Alexander Toshev, and Silvio Savarese. Relmogen: Leveraging motion generation in reinforcement learning for mobile manipulation. *arXiv preprint arXiv:2008.07792*, 2020.

[88] Kiana Ehsani, Winson Han, Alvaro Herrasti, Eli VanderBilt, Luca Weihs, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. Manipulathor: A framework for visual object manipulation. In *CVPR*, 2021. 3

[89] Cyrill Stachniss, Giorgio Grisetti, and Wolfram Burgard. Information gain-based exploration using rao-blackwellized particle filters. In *RSS*, 2005. 3

[90] Fanfei Chen, John D Martin, Yewei Huang, Jinkun Wang, and Brendan Englot. Autonomous exploration under uncertainty via deep reinforcement learning on graphs. In *IROS*, 2020. 3

[91] Simone Parisi, Victoria Dean, Deepak Pathak, and Abhinav Gupta. Interesting object, curious agent: Learning task-agnostic exploration. In *NeurIPS*, 2021. 3

[92] C Cao, H Zhu, Z Ren, H Choset, and J Zhang. Representation granularity enables time-efficient autonomous exploration in large, complex worlds. *Science Robotics*, 2023. 3

[93] Lerrel Pinto and Abhinav Gupta. Learning to push by grasping: Using multiple tasks for effective learning. In *ICRA*, 2017. 3

[94] Tim Schneider, Boris Belousov, Georgia Chalvatzaki, Diego Romeres, Devesh K Jha, and Jan Peters. Active exploration for robotic manipulation. In *IROS*, 2022. 3

[95] Cheng-Chun Hsu, Zhenyu Jiang, and Yuke Zhu. Ditto in the house: Building articulation models of indoor scenes through interactive perception. *arXiv preprint arXiv:2302.01295*, 2023. 3

[96] Shengyong Chen, Youfu F Li, Wanliang Wang, and Jianwei Zhang. *Active sensor planning for multiview vision tasks*. 2008. 3

[97] Mahsa Ghasemi, Erdem Bulgur, and Ufuk Topcu. Task-oriented active perception and planning in environments with partially known semantics. In *ICML*, 2020. 3

[98] Roberto Martín-Martín and Oliver Brock. Building kinematic and dynamic models of articulated objects with multi-modal interactive perception. In *2017 AAAI Spring Symposium Series*, 2017. 3

[99] Zhenyu Jiang, Cheng-Chun Hsu, and Yuke Zhu. Ditto: Building digital twins of articulated objects from interaction. In *CVPR*, 2022. 3

[100] Christopher Collander, William J Beksi, and Manfred Huber. Learning the next best view for 3d point clouds via topological features. In *ICRA*, 2021. 3

[101] Daryl Peralta, Joel Casimiro, Aldrin Michael Nilles, Justine Aletta Aguilar, Rowel Atienza, and Rhandley Cajote. Next-best view policy for 3d reconstruction. In *ECCV Workshops*. Springer, 2020. 3

[102] Zhirong Wu, Shuran Song, Aditya Khosla, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets for 2.5 d object recognition and next-best-view prediction. *arXiv preprint arXiv:1406.5670*, 2014. 3

[103] Yiheng Han, Irvin Haozhe Zhan, Wang Zhao, and Yong-Jin Liu. A double branch next-best-view network and novel robot system for active object reconstruction. In *ICRA*, 2022.

[104] Björn Browatzki, Vadim Tikhanoff, Giorgio Metta, Heinrich H Bülthoff, and Christian Wallraven. Active in-hand object recognition on a humanoid robot. *IEEE Transactions on Robotics*, 2014. 3

[105] Qihang Fang, Yingda Yin, Qingnan Fan, Fei Xia, Siyan Dong, Sheng Wang, Jue Wang, Leonidas Guibas, and Baoquan Chen. Towards accurate active camera localization. In *ECCV*, 2022. 3

[106] Jun Lv, Yunhai Feng, Cheng Zhang, Shuang Zhao, Lin Shao, and Cewu Lu. Sam-rl: Sensing-aware model-based reinforcement learning via differentiable physics-based simulation and rendering. *RSS*, 2023. 3

[107] Youssef Zaky, Gaurav Paruthi, Bryan Tripp, and James Bergstra. Active perception and representation for robotic manipulation. *arXiv preprint arXiv:2003.06734*, 2020. 3

[108] Quoc V Le, Ashutosh Saxena, and Andrew Y Ng. Active perception: Interactive manipulation for improving object detection. *Standford University Journal*, 2008. 3

[109] Snehal Jauhri, Sophie Lueth, and Georgia Chalvatzaki. Active-perceptive motion generation for mobile manipulation. *arXiv preprint arXiv:2310.00433*, 2023. 3

[110] John Schulman, Barret Zoph, Christina Kim, Jacob Hilton, Jacob Menick, Jiayi Weng, Juan Felipe Ceron Uribe, Liam Fedus, Luke Metz, Michael Pokorny, et al. Chatgpt: Optimizing language models for dialogue. *OpenAI blog*, 2022. 3

[111] R OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[112] Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey

Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. Palm-e: An embodied multimodal language model. In *arXiv preprint arXiv:2303.03378*, 2023. 3

[113] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022. 3

[114] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*, 2022. 3

[115] Jianing Yang, Xuweiyi Chen, Shengyi Qian, Nikhil Madaan, Madhavan Iyengar, David F Fouhey, and Joyce Chai. Llm-grounder: Open-vocabulary 3d visual grounding with large language model as an agent. *arXiv preprint arXiv:2309.12311*, 2023. 3

[116] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *ICRA*, 2023. 4

[117] William Shen, Ge Yang, Alan Yu, Jansen Wong, Leslie Pack Kaelbling, and Phillip Isola. Distilled feature fields enable few-shot language-guided manipulation. In *CoRL*, 2023. 4

[118] Krishna Murthy Jatavallabhula, Alihusein Kuwajerwala, Qiao Gu, Mohd Omama, Tao Chen, Shuang Li, Ganesh Iyer, Soroush Saryazdi, Nikhil Keetha, Ayush Tewari, et al. Conceptfusion: Open-set multimodal 3d mapping. *arXiv preprint arXiv:2302.07241*, 2023. 5

[119] Shiyang Lu, Haonan Chang, Eric Pu Jing, Abdeslam Boularias, and Kostas Bekris. Ovir-3d: Open-vocabulary 3d instance retrieval without training on 3d data. In *CoRL*, 2023. 5

[120] Qiao Gu, Alihusein Kuwajerwala, Sacha Morin, Krishna Murthy Jatavallabhula, Bipasha Sen, Aditya Agarwal, Corban Rivera, William Paul, Kirsty Ellis, Rama Chellappa, Chuang Gan, Celso Miguel de Melo, Joshua B. Tenenbaum, Antonio Torralba, Florian Shkurti, and Liam Paull. Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning. *arXiv preprint arXiv: 2309.16650*, 2023. 15

Appendix

I. Additional Details of RoboEXP system

A. Interactive Exploration

Due to space constraints, we did not include a comprehensive explanation of the algorithm proposed in the problem statement, but include more details here for clarity. We formulate the interactive scene exploration task into an active perception and exploration problem to construct the action-conditioned 3D scene graph (ACSG).

The algorithm shown in the main paper simply mentions "add spatial relations" and "add action preconditions" as part of the function of the memory module, but without detailed explanation. In the algorithm, we have demonstrated how to construct the edges from objects to actions $\mathbf{e}_{\mathbf{o}\to\mathbf{a}}$ and from actions to objects $\mathbf{e}_{\mathbf{o}\to\mathbf{a}}$; however, there is a lack of description for the other two types of edges.

Add Spatial Relations. The logic involves analyzing the spatial relationships among objects using spatial heuristics and incorporating the resulting spatial relation edges between objects $\mathbf{e}_{\mathbf{o}\to\mathbf{o}}$ (see Algorithm 2).

Algorithm 2 Add Spatial Relations

input: $\mathbf{G}^{t-1} = (\mathbf{V}^{t-1}, \mathbf{E}^{t-1})$
$\mathbf{E}^t = \mathbf{E}^{t-1}$
for $\mathbf{o} \in \mathbf{V}^{t-1}$ **do** `% check relations`
  if relation from $\mathbf{o}$ to $\mathbf{o}_i$ **then** `% memory`
    $\mathbf{E}^t = \mathbf{E}^t \cup \{\mathbf{e}_{\mathbf{o}\to\mathbf{o}_i}\}$ `% add edge`
  end if
  if relation from $\mathbf{o}_i$ to $\mathbf{o}$ **then**
    $\mathbf{E}^t = \mathbf{E}^t \cup \{\mathbf{e}_{\mathbf{o}_i\to\mathbf{o}}\}$ `% add edge`
  end if
end for
output: $\mathbf{G}^t$ `% new scene graph`

Add Action Preconditions. The approach is to assess the feasibility of implementing the actions. We utilize the decision-making module to verify whether there are any prerequisite actions that need to be completed beforehand, and then adjust the plan accordingly (see Algorithm 3).

Algorithm 3 Add Action Preconditions

input: $\mathbf{G}^{t-1} = (\mathbf{V}^{t-1}, \mathbf{E}^{t-1}), \mathbf{U}^{t-1}$
if object $\mathbf{o}$ obstruct **then** `% decision-making`
  choose action $\mathbf{a}$
  $\mathbf{V}^t = \mathbf{V}^{t-1} \cup \{\mathbf{a}\}$, $\mathbf{U}^{t-1} \cup \{\mathbf{a}\}$ `% add node`
  $\mathbf{E}^t = \mathbf{E}^{t-1} \cup \{\mathbf{e}_{\mathbf{o}\to\mathbf{a}}\}$ `% add edge`
  $\mathbf{E}^t = \mathbf{E}^{t-1} \cup \{\mathbf{e}_{\mathbf{a}\to\mathbf{a}_k}\}$ `% add edge`
end if
output: $\mathbf{G}^t, \mathbf{U}^t$ `% new scene graph & plan`

B. Usage of ACSG

The ACSG constructed during the exploration stage shows beneficial for scenarios that require a comprehensive understanding of scene content and structure, such as household environments like kitchens and living rooms, office environments, etc. We list several examples illustrating the potential usage of the scene graph in various tasks.

Judging Object Existence. A direct application of our ACSG is to determine the presence or absence of specific objects in the current environment. For instance, during the exploitation stage of the scenario to set the dining table, if the spoon is missing, the robot can further seek human assistance.

Object Retrieval. One notable advantage of our ACSG is its ability to capture all actions and their preconditions. Utilizing this information, retrieving any object becomes straightforward by following the graph structure and executing actions in topological order along the paths from the root to the target object node. For example, in the obstruction scenario, the ACSG can provide the sequence of actions required to fetch the tape: 1) removing the condiment blocking the cabinet door, 2) opening the cabinet via the door handle, and 3) retrieving the tape. Such insights are crucial for tasks like cooking.

Advanced Usage. The high-level representation of the environment provided by our ACSG serves as a simplified yet effective model. Similar to the approach proposed by Gu et al. [120], integrating the scene graph with Large Language Models (LLM) or Large Multi-modal Models (LMM) offers enhanced capabilities, including natural language interaction. This enables the robot to respond to human preferences expressed in natural language (e.g., fetching a coke when the person is thirsty) or through visual cues (e.g., fetching a mug when the table is dirty).

C. Decision-Making Module

As illustrated in the main paper, the decision-making module fulfills two crucial functions within our system. The first function serves as an action proposer (Fig. 7a), proposing the appropriate skill for the query object node. The subsequent role functions as the action verifier (Fig. 7b), tasked with confirming the feasibility of implementing the action and determining the action preconditions. The complete prompts for both roles are detailed in Fig. 7.

D. Action Module

The action module focuses on providing useful action primitives to aid in constructing our ACSG. We have designed seven action primitives: "open the `[door]`", "open the `[drawer]`", "close the `[door]`", "close the `[drawer]`", "pick `[object]` to idle space", "pick back `[object]`", "move wrist camera to `[position]`". To fully support autonomous actions, we employ a heuristic-based algorithm leveraging geometric cues.

For the door and drawer relevant primitives, engagement with handles is required. In our implementation, we exploit the handle's position and geometry to discern its motion type (prismatic or revolute) and motion parameters (motion axis and motion origin). Executing this action involves utilizing the detected handle and its geometry to adeptly open doors or drawers. Upon identifying the specific handle to be operated, our system retrieves the point cloud converted from our voxel-based representation corresponding to that handle from

(a) Prompts of Proposer

System: You are an assistant tasked with aiding in the construction of a complete scene graph for a tabletop environment. The objective is to identify all objects hidden from the current observation in the tabletop setting. Your role involves selecting appropriate actions or opting not to take any action based on commonsense knowledge in response to queries with current observations. Your responses will guide a robot in efficiently exploring the environment. Approach each step thoughtfully, and analyze the fundamental problem deeply, considering the potential vagueness or inaccuracy in the queries. Adhere to the provided formats in your instructions.

User: Analyze and provide your final answer for each new query object/part category, considering the given surrounding objects and observations in the tabletop scene from different viewpoints. The query object/part will be enclosed in a green bounding box, though it may not always be fully accurate. Format your responses as follows: "[Analysis]: <your reasoning process>; \n\n [Final Answer]: <skill>". Be comprehensive and avoid repeating my question. Choose from three skills: 1. Open the doors or drawers. 2. Pick up / Open the top object. 3. No action. The primary goal is to select an action that has the potential to reveal hidden objects. The secondary goal is to act efficiently, performing only necessary actions to uncover hidden objects. For example, if an object contains doors or drawers and can potentially store something inside, opt for the first skill "Open the doors or drawers". If an object has no bottom side and can potentially cover something beneath it, choose the second skill " Pick up / Open the top object"; otherwise, select the third skill "No action" to ensure efficiency.

Assistant: Got it. I will output the reasoning process step-by-step, explain why I choose the skill but not others and follow the output format.

User: [Query Object] + [Query Images]

Assistant: [Reply from GPT-4V]

(b) Prompts of Verifier

System: You are an assistant tasked with evaluating the feasibility of actions within a tabletop environment. Your role is to select suitable objects that could obstruct open actions based on queries and current observations. Provide guidance for a robot's planning process. Approach each step thoughtfully, analyzing the underlying problem thoroughly while considering potential vagueness or inaccuracy in the queries. Follow the provided formats in your instructions.

User: Provide an analysis and your final answer each time I present a new query object/part category, the list of surrounding objects you need to consider and observations of the corresponding in the tabletop scene from different viewpoints. The query object/part is enclosed in a green bounding box, which may not always be fully accurate. Present your reasoning process and final answer in the format "[Analysis]: <your reasoning process>; \n\n [Final Answer]: <list of objects>". Be comprehensive and avoid repeating my question. Use the given list of surrounding objects, maintaining the provided names. Only consider the surrounding objects in the given list. The objective is to identify all objects that could potentially block open actions. If an object obstructs the door or drawer from opening, include it in the final list of objects. Analyze the action movement and identify the blocking objects.

Assistant: Got it. I will output the reasoning process step-by-step, explain why I choose the object but not others and follow the output format.

User: [Query Object] + [Query Images]

Assistant: [Reply from GPT-4V]

Fig. 7: **Prompts of the Decision-Making module.** We present the full prompts for the two pivotal roles of our decision-making module, **proposer** in (a), **verifier** in (b). The prompts are used for all our experiments without modification and extra examples.

System: You are an assistant tasked with aiding in the construction of a complete scene graph for a tabletop environment. The objective is to identify all objects hidden from the current observation in the tabletop setting. Your role involves selecting appropriate actions or opting not to take any action based on commonsense knowledge in response to queries with current observations. Your responses will guide a robot in efficiently exploring the environment. Approach each step thoughtfully, and analyze the fundamental problem deeply, considering the potential vagueness or inaccuracy in the queries. Adhere to the provided formats in your instructions.

User: Analyze and provide the current scene graph and your final answer for the next action given the latest observations in the tabletop scene from different viewpoints. Each time you need to pick an action to do or choose "Done" to terminate. The action you can choose should be composed of (<object/part>, <skill>). Be specific on which object or part you refer to. The skills you can choose: [1. Open the door. 2. Close the door. 3. Open the drawer. 4. Close the drawer. 5. Pick up the object to idle space. 6. Pick back the object from the idle space]. Each time after you choose an action, you will receive the new observations after the action. Format your responses as follows: "[Analysis]: <your reasoning process>; \n\n [Scene Graph]: <current scene graph> \n\n [Final Answer]: <skill>". Be comprehensive and avoid repeating my question. The primary goal is to select an action that has the potential to reveal hidden objects. The secondary goal is to act efficiently, performing only necessary actions to uncover hidden objects. The third goal is to make the object go back to the initial state after exploration. For the output scene graph, you need to output all the objects in the scene, including those found during the exploration process.

Assistant: Got it. I will output the reasoning process step-by-step, explain why I choose the skill but not others and follow the output format.

User: [Query Images]

Assistant: [Reply from GPT-4V]

User: [Query Images]

Assistant: [Reply from GPT-4V]

...

Fig. 8: **Prompts of the GPT-4V baseline.** To ensure fairness in comparison to this baseline, we choose to use similar prompts, employing the chain-of-thoughts technique to enhance its performance.

our memory module. Subsequently, we employ Principal Component Analysis (PCA) to determine the principal direction of the handle, aiding in aligning the gripper for optimal engagement. Additionally, understanding the opening direction is pivotal for effectively handling doors or drawers. To ascertain this, we analyze neighboring points and deduce the most common normal as the opening direction. The combined information of the handle direction and the opening direction provides sufficient guidance for our robot arm to grasp the handle and open the prismatic part. However, in the case of a revolute joint, the motion becomes more intricate. Therefore, we further utilize the motion parameters inferred from the geometry to simulate the evolving opening direction based on the revolute joint's opening process. This well-designed heuristic empowers our system to reliably open drawers or doors in our tabletop setting.

For the pickup-related primitives, we simplify the pickup logic to exclusively consider a top-down direction. Consequently, our focus narrows down to acquiring essential information such as the object's height and xy location. We

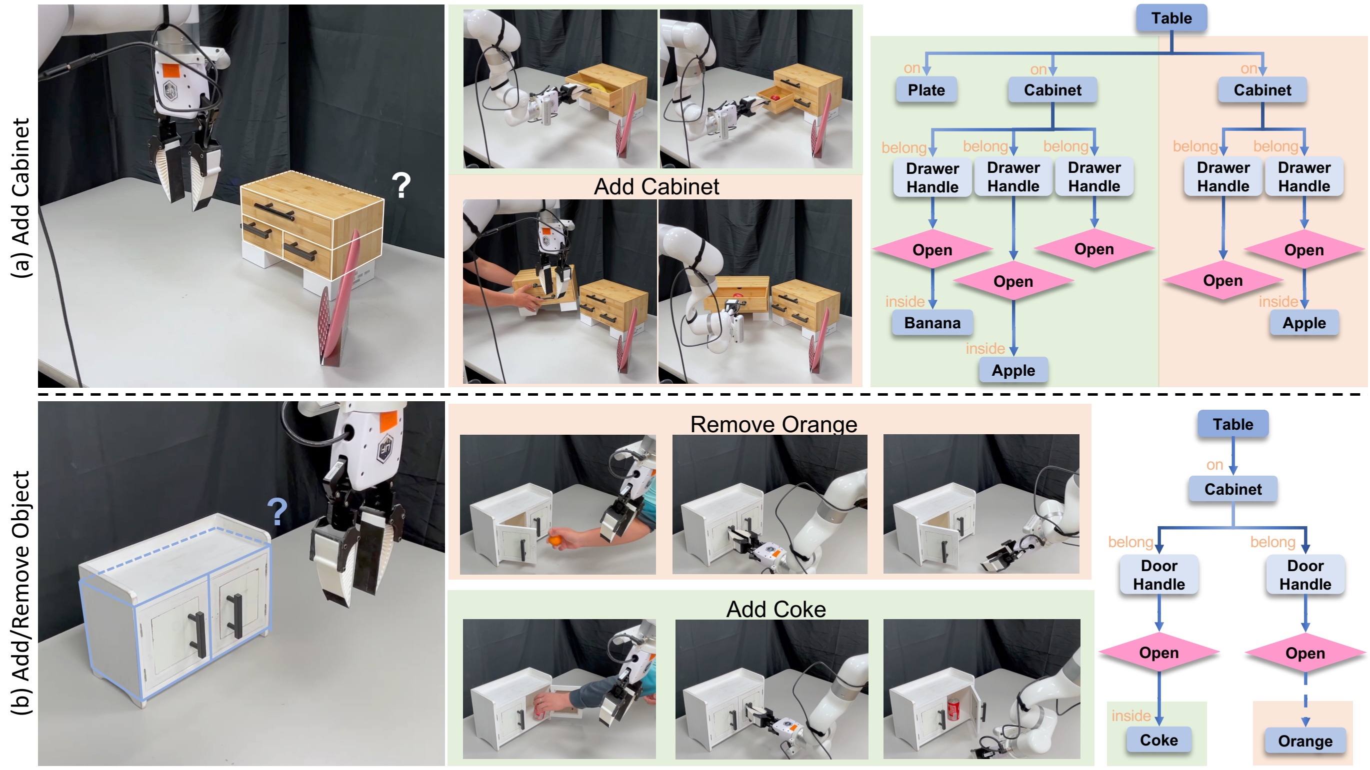


Fig. 9: **Qualitative Results on Different Intervention Scenarios.** (a) This scenario involves adding a cabinet to the tabletop setting, and our system can auto-detect the new cabinet and explore the objects inside. (b) This scenario includes removing and adding objects from and into the cabinet. Our system can monitor hand interactions and re-explore the corresponding doors.

achieve this by extracting the object's point cloud from its associated voxel-based representation. Subsequently, we pinpoint the highest points within the cloud, calculating their mean to determine the optimal pickup point. This calculated point serves as a precise reference for our gripping mechanism, facilitating the successful grasping of objects in the specified direction.

Regarding viewpoint change, the primitive is parameterized with the expected pose. For example, after opening the door/drawer, to see inside, we develop the heuristic to choose the proper viewpoint from the open direction as the parameter for the primitive, allowing for the implementation of the action primitive.

II. Additional Details of Experiments

A. GPT-4V Baseline

We have developed the pure GPT-4V baseline, incorporating ground truth action primitives (where humans manually perform the actions), establishing it as a robust benchmark for comparison with our approach. The full prompt of the GPT-4V baseline is illustrated in Fig. 8.

B. Experiment Settings

To assess the effectiveness of our system, we devised five types of experiments, each encompassing 10 distinct settings. These settings vary in terms of object number, type, and layout, as illustrated in Fig. 10.

C. Human Intervention

Our RoboEXP system possesses the capability to autonomously adapt to changes in the environment. We employ two types of human interventions to demonstrate these points.

The first type of intervention (Fig. 9a) involves adding new cabinets to the scene. In this scenario, we add a cabinet to the explored area, allowing our system to automatically explore the newly added cabinets and update the ACSG.

The second type of intervention (Fig. 9b) involves adding new objects to or removing existing ones from the cabinets in the current scene. Our system can monitor human interactions and discern which objects require re-exploration. Subsequently, it autonomously updates the ACSG based on re-exploration.

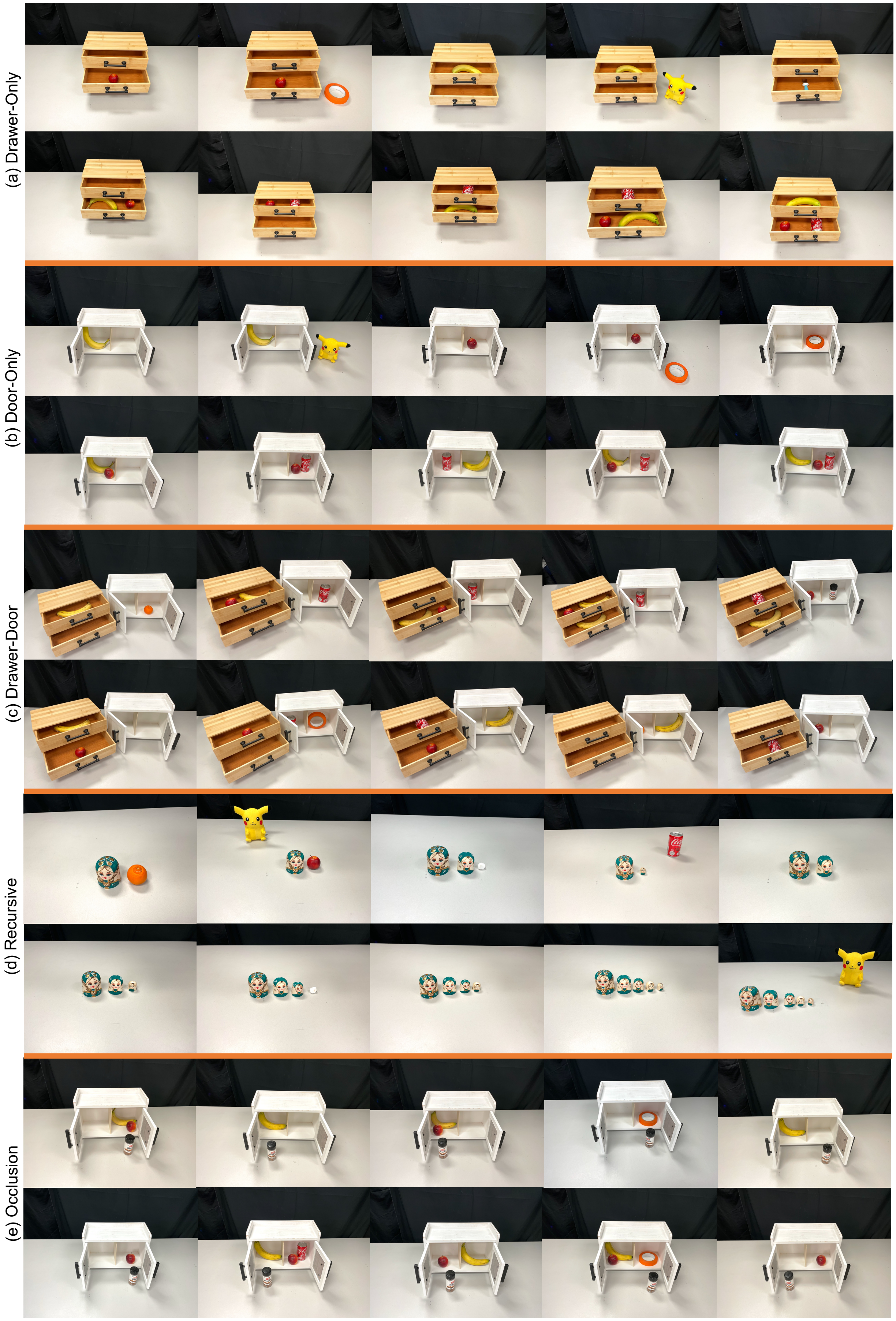


Fig. 10: **Experiment Settings.** Varied object numbers, types, and layouts in our experimental settings of the quantitative results.