2025 IEEE International Conference on Robotics and Automation (ICRA)
May 19-23, 2025. Atlanta, USA

Interaction-Driven Updates: 3D Scene Graph Maintenance During Robot Task Execution

Qingfeng Li[1†], Xinlei Zhang[1†], Chen Chen[2*], Haochen Zhao[1], Jianwei Niu[1]

***Abstract*— Robots powered by large language model (LLM) demonstrate significant research and application potential by effectively interpreting scene information to respond to human commands. However, when robots rely on static scene information during task execution, they face difficulties in adapting to changes in the environment, posing a major challenge for dynamic scene perception. To address the above issues, we propose an innovative interaction-driven approach to enhance robots' ability to perceive dynamic scene information. This approach consists of two contributions, the observation point selection module and the dynamic scene maintenance module. Specifically, first, the robot uses the 3D scene graph (3DSG) containing assets and objects to perceive static scene information through the LLM planner. Next, the best observation point for each asset is obtained through the observation point selection module. Then, with the help of the best observation point, the dynamic scene maintenance module interacts with the asset-related objects to dynamically update all the object node information related to the asset node. This approach enables robots to maintain dynamic scene information, enhancing their adaptability in unpredictable environments and improving task reliability. We evaluated our method using the iTHOR and RoboTHOR datasets within the AI2-THOR simulator and in real-world scenarios. Experimental results demonstrate that our method effectively and accurately maintains robots' perception of dynamic scene information.**

I. INTRODUCTION

Robots equipped with LLM harness their sophisticated language comprehension abilities to plan tasks as instructed by humans [1], [2], [3], [4], [5], [6], [7], [8]. In real-world settings, however, these robots often depend on precise perception of dynamic scene information. The complexity and variability of human environments lead to frequent and unpredictable changes in the scene, presenting substantial challenges for robots in perceiving these dynamics. Specifically, in large, multi-room settings, it is impractical to expect robots to continuously monitor all potential changes. Taking the above issues into account, a straightforward question arise: *How does a robot perceive changes in information within a scene?* Inspired by human behavior, for example, when we go to a room and pick up a cup from a table, during this process, we often only pay attention to the current objects on the table, ignoring the specific scene information along the way. Therefore, in this paper, we propose an interaction-driven method to enable robots to perceive dynamic scene information.

[1]Xinlei Zhang, Jianwei Niu, Qingfeng Li, Haochen Zhao are with Beihang University, Beijing, China.
[2]Chen Chen is with the Hangzhou Innovation Institute of Beihang University, Hangzhou, China.
† These authors contributed equally to this work.
* Corresponding author

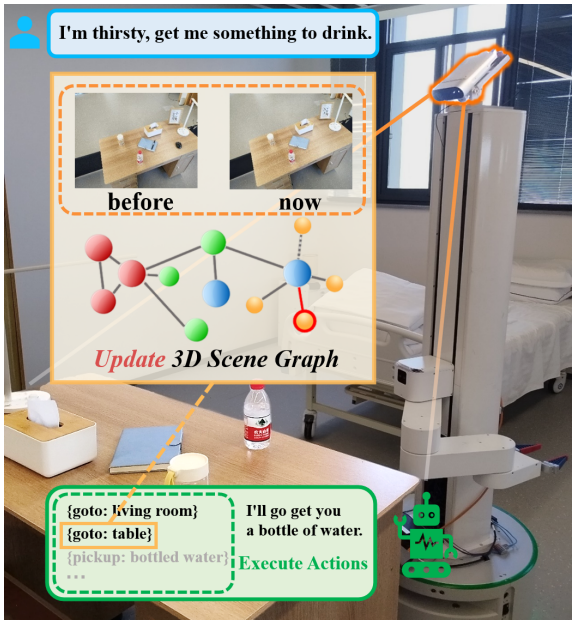

Fig. 1: The interaction-driven method proposed in this paper is capable of real-time updating of object information in the scene during the robot's interaction with the environment.

Our method consists of two modules. First, in the observation points selection module, we use a mesh of a 3DSG [9] containing assets and objects as input, generating potential observation points for each asset in the scene. Then, by analyzing the information of the scene corresponding to each potential observation point, we determine the best observation point for each asset. Second, in the dynamic scene maintenance module, when the robot approaches the observation point of an asset, it detects all objects on the asset. If changes are detected in the objects, the robot uses a visual language model (VLM) to mine the latest information about all objects on the asset, updating this information to the 3DSG, and then executing subsequent commands. If no changes are detected, the robot proceeds directly to execute subsequent commands. Through this method, we enable the robot to perceive dynamic scene information and improve the reliability of command execution.

In this work, our main contributions are:

- We propose an interaction-driven approach to scene information update, which embeds the update process into each interaction between the robot and the scene.
- We utilize LLM and VLM to update scene information, resulting in a zero-shot and open-vocabulary scene information update method.

2025 IEEE International Conference on Robotics and Automation (ICRA) | 979-8-3315-4139-2/25/$31.00 ©2025 IEEE | DOI: 10.1109/ICRA55743.2025.11128194

- Experiments in simulated and real-world scenarios show the effectiveness of our method in maintaining dynamic scene information.

II. RELATED WORKS

A. LLM-Empowered Robotic Agents

The application of LLM in the construction of robotic agents capable of performing complex tasks has garnered extensive attention from both academia and industry [29], [30], [31], [32], [37]. Previous studies have demonstrated the potential of pre-trained LLM to generate executable plans for robotic agents that can understand and execute complex instruction sequences [1], [2], [3], [4], [5], [6], [7], [8]. Nevertheless, current LLM-based mission planning approaches still face two major challenges: first, their long-term planning capabilities in vast or complex environments; and second, how to effectively respond to dynamic changes in the environment. Llm-Planner [1] addresses the scalability problem by allowing LLM to plan iteratively. SayCan [2] uses a visibility function to execute the basics of the LLM planner, but it works well only for small, static scenes and a limited object vocabulary. As a cutting-edge study, SayPlan [3] overcomes the scalability bottleneck of traditional methods in large-scale environments by utilizing static 3DSG. Although these methods based on LLM to drive robots have achieved good results, these methods lack the ability to adapt to dynamic scenes.

B. Traditional Scene Update Methods

In the field of robotics, real-time updates of map information are essential for robots to efficiently perform tasks in dynamic and complex environments. Life-long SLAM(Simultaneous Localization and Mapping) [18], [23], [24], [25], [26], [27], [28] is capable of processing changes in the environment, identifying these changes, and updating the map accordingly, assuming the initial mapping is successfully completed. Hydra [13] is a real-time spatial awareness system that builds a 3DSG from sensor data in real time. ConceptGraphs [19] utilizes SLAM technology to create a 3DSG, a powerful representation of the environment that allows the robot to understand and navigate by describing the environment in the form of a 3DSG. Additionally, it can update some scene information based on the semantic common sense abilities of LLM.

In a hierarchical structure like a 3DSG [13], [33], [34], [35], nodes represent different levels of abstraction, from basic geometric shapes to high-level semantic information, such as objects, places, rooms, and even entire buildings. Edges define the relationships between these concepts. While this representation provides rich environmental awareness, it is often considered static and difficult to update in real time once constructed, unless the entire scene is fully rescaned and reconstructed. However, the above method updates the scene information by means of re-mapping, which cannot make the robot understand the dynamic scene information in time.

III. METHOD

In order to realize the robot's ability to perceive dynamic scene information, we first introduce the 3DSG representation of an environment which we leverage throughout our approach in III-A. Our method flow diagram is shown in Fig. 2 and consists of two modules, namely the observation point selection module and the dynamic scene update module, the details of which are described in III-B and III-C.

A. Preliminaries

Here, we introduce a 3DSG representation of the environment [3]. 3DSG have recently become a practical representation for robotic worlds [11], [13], [14], [15], [16], providing a hierarchical abstraction that encompasses spatial semantics and object relations. This abstraction captures pertinent states, affordances, and predicates of the entities within the environment. Formally, a 3DSG is structured as a hierarchical multigraph $G = (V, E)$, where the vertex set V is composed of $V_1 \cup V_2 \cup \ldots \cup V_K$, with each V_k denoting the vertices at hierarchy level k. The edges originating from any vertex $v \in V_k$ can only link to vertices in V_{k-1}, V_k, or V_{k+1}, meaning they connect nodes at the same level or one level above or below.

We assume the existence of a pre-built 3DSG representation of a large-scale environment, created using established methods [13], [11], [9]. This 3DSG can be modeled as a `NetworkX` Graph object [17] and serialized into a JSON format, which can be directly interpreted by a pre-trained LLM. A single asset node within the 3DSG might look like this: {name: coffee_machine, type: asset, location: kitchen, affordances: [turn_on, turn_off, release], state: off, attributes: [red, automatic], position: [3.14, 0.32, 2.05]} with connections between nodes described as {kitchen $\leftrightarrow$ coffee_machine}. The 3DSG is structured hierarchically across four main levels: floors, rooms, assets, and objects. Each room contains assets, which are fixed entities, and objects, which are movable. Both types of nodes—assets and objects—contain details such as their state, affordances, additional attributes like color or weight, and their 3D positions.

B. Observation Point Selection Module

We utilize the original mesh in the 3DSG generated by existing technology [13], [38], as input for the observation point selection module. This process generates an observation viable domain for each asset in the scene, with the size of the domain dependent on the robot parameters. Different robots, in the same scenario, have distinct feasible domains. To prevent collisions with the environment, the robot sets a safe radius d_{safe}, which is twice the expansion distance, to generate an area covering the asset and its surroundings. Potential observation points are then evenly generated within this area. By combining two-dimensional raster information, infeasible points are screened out, retaining the set of feasible observation points. These observation points are three-dimensional and represent geometric positions, not orientations. The feasible observation points are extended

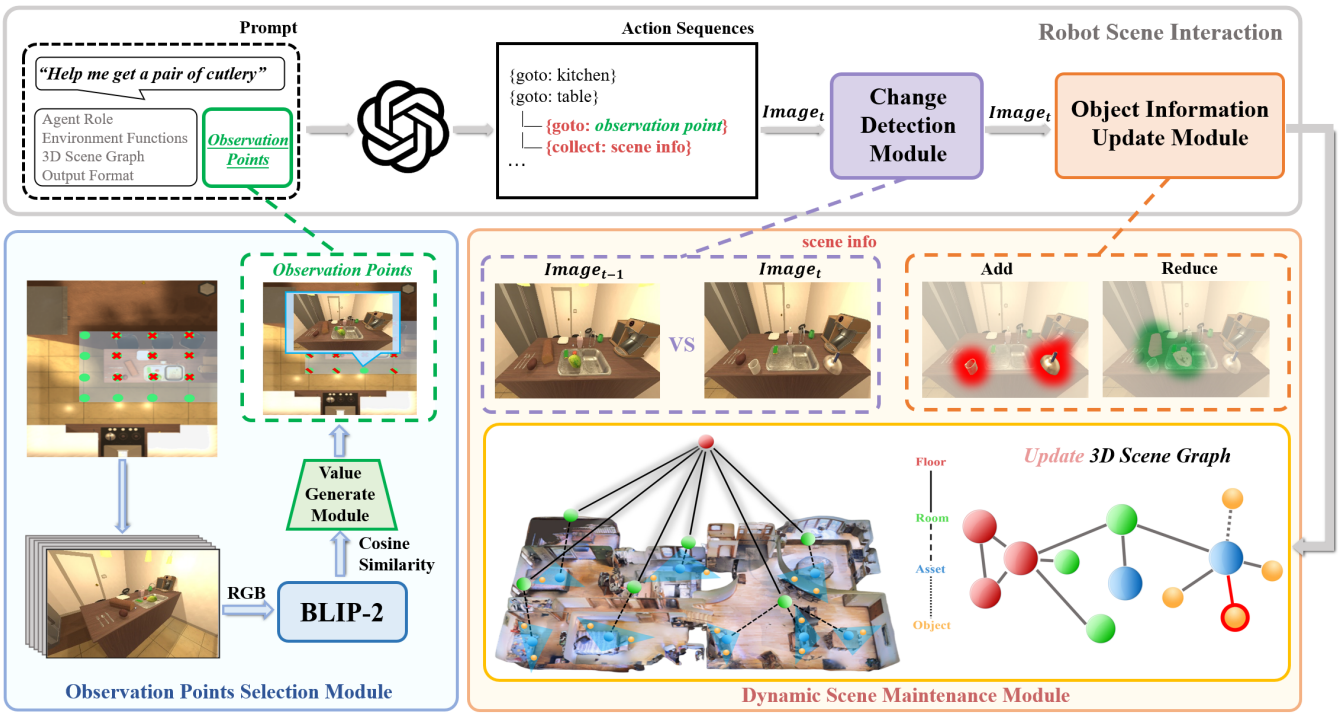

Fig. 2: Our method consists of two modules: 1) observation point selection module: Before generating specific task instructions, multiple observation points are generated around each asset. The best observation point for each asset is determined by analyzing the image information from different perspectives. 2) dynamic scene maintenance module: The robot performs specific action execution. After the robot arrives near the asset, it detects the object information on the asset before performing any operation. If an object change is detected, the image information obtained from the observation point is used to update the latest information of all objects on the asset. This information is then integrated into the 3DSG to maintain the accuracy of the scene.

from (x, y, z) to (x, y, z, a, b, c, d) through the geometric center coordinates of the asset, where the last four values represent the quaternions indicating the orientation of the robot.

Then, the height of the robot camera and the coordinates of the observation points are used to obtain the observation image information of all observation points in the original mesh (note that when obtaining the image information, adjust the tilt angle parameter of the camera so that the asset is located in the middle of the image). Since we have a JSON representation of the 3DSG [3], we can obtain all the object text information related to the corresponding asset. This text information is input into the BLIP-2 model [22] simultaneously with the RGB image to obtain the cosine similarity.

In the value generation module, for the set of viable observations for each asset, we assign a value to each observation point using cosine similarity. The specific generated value is as follows:

$$\text{value} = \frac{d_{\max} - d}{d_{\max}} \cdot \text{cosine similarity}, \quad (1)$$

where $d_{\max}$ represents the distance from the farthest point of the asset among all the observation points; d represents the distance from the asset to the observation point. For each asset, the observation point corresponding to the largest value is used as the best observation point by comparing the values of all observation points. This process is repeated for all assets in the scene to obtain the best observation point for each asset. The resulting set of observation points is then used as part of the prompt when the robot performs decision planning.

C. Dynamic Scene Maintenance Module

We are embedding dynamic scene updates into the robot's interaction with the environment. Specifically, after receiving the instruction, the robot obtains a specific and executable sequence of actions through LLM analysis, and our scene information update process is part of the action subtask. For example, the instruction *"Help me get the water bottle on the table"* will be broken down into specific action sequences: *"goto (living room), goto (table), pickup (bottle of water)..."*. By integrating the dynamic scene maintenance module into specific robot action primitives, action primitives such as *"goto (table)"* can be refined into smaller action steps, namely, *"goto (observation point)"* and *"collect (scene info)"*. After arriving at the observation point, the robot first collects the scene information, which involves collecting the RGB image related to the asset at the observation point. This also represents the scene information of the asset's changes.

In the change detection module, one can ascertain whether the scene information has altered between the current scene and the preceding 3DSG, as well as whether the scene information undergoes changes upon the execution of asset-related commands. The module stores the scene image re-

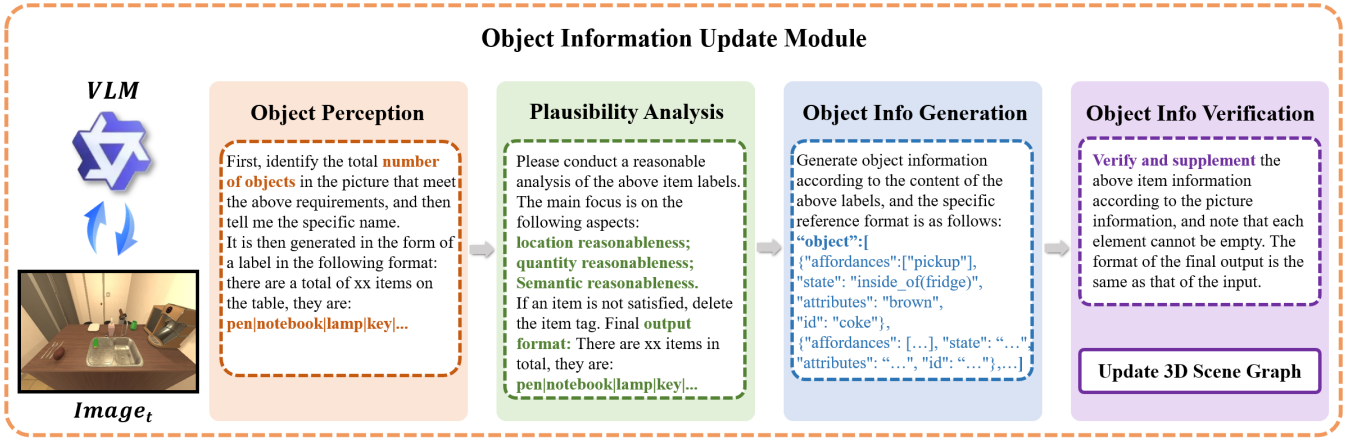

Fig. 3: The object information update module consists of four stages: object perception stage; plausibility analysis stage; object information generation stage; object information verification stage.

lated to the asset at the time of the initial or last execution of the command $image_{t-1}$, and the new scene image obtained by the robot when executing the command is $image_t$. These two images are input into the visual transformer model [36] to obtain embeddings $embedding_{t-1}$ and $embedding_t$. We utilize these two vectors to compute the cosine similarity score, thereby ascertaining whether the scene information has undergone a change. A threshold, denoted as θ, is established to ascertain whether the object information has altered. It is crucial not to set θ excessively high, as this may be influenced by dynamic changes in the scene and variations in lighting. θ setting necessitates specific adjustments based on the robot's camera parameters. If the cosine similarity score falls below θ, it signifies that the object information has changed; conversely, if it remains above θ, the object information is deemed unchanged. In the event of no change, the robot proceeds with the current action. However, if a change is detected, the robot transitions into the object information update module.

The object information update module is responsible for incorporating the most current object information pertinent to an asset into the 3DSG, accounting for both additions and deletions. This module receives two inputs: the original scene graph data and the scene image captured from the asset's observation point. To ensure that the updated scene information accurately reflects the real world, the process of updating object information is segmented into four distinct stages. During the object perception stage, the number of objects associated with the asset and their respective tags are identified. In the plausibility analysis stage, the plausibility of the detected object tags is assessed, enabling the elimination of implausible tags. The object information generation stage involves transforming the identified tags into a format that conforms to the 3DSG standard. Finally, in the object information verification stage, the accuracy of the object information is verified and, if necessary, supplemented. The specific update process is depicted in Figure 3.

Object perception stage: Perceiving the number of objects associated with an asset in an image is crucial for accurately updating object information within the scene. Images captured from the observation point include asset, related object, and background information. We utilize information about asset-related objects from the preceding scene and employ the common-sense understanding of a LLM to formulate queries that describe the state of the space. For instance, if a scene features a table with objects placed upon and adjacent to it, the robot updates object information only pertaining to the table. By leveraging the semantic understanding capabilities of the LLM, we generate queries that meet the necessary criteria, such as *"What's on the table?"* The resulting output specifies the number of objects and their labels: there are n objects on the table, and they are: pen|notebook|lamp|...|cup.

Plausibility analysis stage: Ensuring the realism of object information associated with an asset node is crucial for the accuracy of updates. While we restrict the positions of detected objects within the object perception module, the generated label information may not always comply with requirements, due to limitations in the automatic generation capabilities of the VLM and sometimes the constraints on the number of objects. To address this, we employ a LLM to construct a plausibility analysis module, which assesses positional, semantic, and quantitative plausibility, thereby filtering out labels that do not meet the required standards.

Object information generation stage: Generating object node information that conforms to the 3DSG format is crucial for updating scene information. We utilize the VLM to augment certain attribute information within the object nodes and to identify the *"affordances," "state,"* and *"attributes"* of the objects, thereby enabling the perception of edges in the 3DSG. The original 3DSG's JSON format structures object node information as follows: *"object":[{"affordances":["pickup"], "state": "inside_of(fridge)", "attributes": "brown", "id": "coke"},{"affordances": [...], "state": "...", "attributes": "... ", "id": "..."},...]* object node.

Object information verification stage: Ensuring the accuracy and validity of the generated object node information is

crucial for updating object nodes in the 3DSG. Each object node encompasses a substantial amount of data, including affordances, state, attributes, and others. In scenarios where the scene is rich in object information, the information generated by the object information generation module may not be entirely complete or accurate. Consequently, these elements require further verification and enhancement. We employ a VLM to re-detect the image, leveraging prompt design to activate its capacity to validate certain attributes. Ultimately, this process yields object update information that aligns with the actual scene.

IV. EXPERIMENTS

A. Overview

We empirically evaluated the performance of the LLM planner in both simulated and real-world scenarios. Specifically, we aimed to test the hypothesis that the LLM planner can detect changes in the information of objects within a scene by utilizing image data from a specific viewpoint. Our findings, when compared to those of a baseline LLM planner, conclusively supported this hypothesis by accurately measuring the changes in objects. Moreover, our ablation studies highlighted the critical roles of the perspective selection module and the dynamic scene maintenance module in updating scene information.

B. Experiment Setup

For the primary evaluation of our model, we conducted tests across three distinct scenarios within the iTHOR [20] and RoboTHOR [21] simulation environments, alongside three real-world settings. In the simulation, to ensure a fair assessment, we randomly selected three scenarios that shared similar asset types, thus validating the effectiveness of our approach. Conversely, in the real-world settings, we purposefully chose three assets with varied characteristics: a coffee table featuring a full viewing angle, a desk with a semi-open viewing angle, and a shelf with a specific viewing angle. These selections were made to test the generalization capabilities of our method.

Environment settings. Each scene contains a different room layout and different placements of objects. Task instantiation commences with the random placement of 5-10 objects on each asset, followed by the addition or removal of 1-5 objects for each task instance. The objects vary in size and shape. As the process of adding and removing objects occurs, occlusions between objects also appear randomly. This task is challenging because the agent needs to recognize changes in the asset-related object information and correctly update this information into the 3DSG during the execution of specific actions.

Evaluation indicators. We evaluate the success rate of updating asset-related object information through three indicators:

1) Node Recognition Accuracy (NRA): This metric measures the recognition accuracy of the number and names of objects.

$$\text{NRA} = \frac{n_{\text{det\_chn\_nodes}}}{n_{\text{chn\_nodes}}}, \tag{2}$$

where $n_{\text{chn\_nodes}}$ represents the number of all actually changing object nodes in a set of experiments, and $n_{\text{det\_chn\_nodes}}$ represents the number of changing nodes detected during the experiment.

2) Edge Recognition Accuracy (ERA): This metric measures the accuracy of the identification of the association between the object and the asset.

$$\text{ERA} = \frac{n_{\text{det\_valid\_edges}}}{n_{\text{det\_edges}}}, \tag{3}$$

where $n_{\text{det\_edges}}$ represents the number of edges detected during the experiment, and $n_{\text{det\_valid\_edges}}$ represents the number of edges that match real-world conditions among the detected edges.

3) Weighted Jaccard Similarity (WJS): This metric measures the comprehensive recognition effect of nodes and edges, specifically the similarity between the detected local scene graph and the real scene graph.

$$\text{WJS}(S_{\text{real}}, S_{\text{detected}}) = \frac{\sum_{x \in S_{\text{real}} \cap S_{\text{detected}}} w(x)}{\sum_{x \in S_{\text{real}} \cup S_{\text{detected}}} w(x)}, \tag{4}$$

where S_{real} denotes the set of real objects; S_{detected} denotes the set of detected objects; w(x) represents the weight of element x.

4) Node Hallucination Probability (NHP): This metric is the proportion of the number of unseen objects in the scene to the total number of objects recognized.

Experimental details. For each asset in each room, there are 8 sets of experiments, each with five specific instances. The eight sets of experiments included: Adding unobstructed single objects; Reduction of single objects without cover; Adding a single object with occlusion; Reduced occluded single object; Adding multiple objects without obstruction; Reducing multiple objects without obstruction; Adding multiple objects with occlusion; Reducing multiple objects with occlusion. For clarity, occlusion here refers to the viewing angle. In each specific instance, the number of objects is 5-10 before each increase or decrease. To represent as many real-world situations as possible, objects with large differences in size, shape, color, etc., are selected for each asset-related object, and the occlusion case involves at least half of the object's volume by default from the observation point.

Architecture and Baseline. We compare our approach to the most advanced LLM planning baseline methods: SayCan [2] and SayPlan [3]. We use GPT-4 as the LLM for all comparisons, and the temperature during LLM generation is set to 0 to reduce deterministic duplication in the LLM response (the default range is 0-1). For all methods, the prompt includes instructions, a scene diagram description, and two examples. Our method uses SayPlan [3] as the base model and adds scene dynamic perception and update modules to the specific action primitives. To enable SayCan [2] and SayPlan [3] to perceive changes in the scene, we add a basic perceptual update action to the action primitives that SayCan [2] and SayPlan [3] can use.

C. Main Results

As shown in TABLE I,we compare the update of object information in different scenes, the success rate of detection did not change significantly in the three real scenes and the three simulation environments, demonstrating a relatively stable update ability. At the same time, it was observed that the accuracy of node and edge detection improved when detecting the information of objects closer to the camera. Since the height of the camera was fixed at 1.6 meters during the experiments in the real scene, the objects on the shelves were closer to the camera than the objects on the table, leading to an improvement in recognition accuracy by about 5% based on the analysis of experimental data.During the experiments, some hallucination nodes appeared. According to the analysis of specific experimental examples, most of these nodes were small objects such as keys, spoons, and pens. Due to the low resolution of the camera used in the real scene (only 1280*720) and the limitations of the pedestal model, these factors contributed to the appearance of hallucinatory nodes.

TABLE I: Performance comparison of different assets in various scenes. (room_1, room_2, and room_3 represent three different layouts of the same asset collection in a real room.)

Scene	Asset	NRA	NHP	ERA	WJS
room_1	tea table	91.25	3.07	87.50	94.57
	desk	88.75	1.86	88.75	95.65
	shelf	93.75	4.82	92.50	95.22
room_2	tea table	86.25	2.47	83.75	94.41
	desk	85.00	3.66	85.00	94.79
	shelf	92.50	4.24	90.00	95.90
room_3	tea table	88.75	3.66	86.25	95.03
	desk	83.75	2.47	81.25	94.26
	shelf	90.00	4.24	88.75	95.41
floorplan320	table	88.75	3.66	85.00	95.29
floorplan29	table	86.25	4.82	86.25	94.77
	fridge	88.75	5.39	86.25	95.17
floorplan_train6_3	table	91.25	3.07	87.50	95.96
	cabinet	83.75	4.82	80.00	94.12
Average	——	**88.48**	**4.04**	**86.34**	**95.04**

As shown in TABLE II, we compare the success rate of our node and edge detection with the baseline methods and evaluate the probability of hallucination nodes occurring during the detection process and the overall recognition of objects on the asset, taking the average value for each scene. The results show that our method significantly outperforms all baseline methods in the success rate of node and edge detection, improving performance by nearly 20%. Additionally, the probability of hallucinatory nodes is also significantly reduced.In terms of object recognition, most results are consistently good with each update. Further analysis reveals that SayCan [2] and SayPlan [3] do not select a suitable observation angle, leading to lost node detection information. Furthermore, they do not conduct a rationality analysis of the generated object information, making it difficult for them to extract edge information from the image that conforms to the actual scene.

TABLE II: Comparative Experiments

Method	NRA↑	NHP↓	ERA↑	WJS↑
SayCan[2]	67.98	5.63	62.61	86.89
SayPlan[3]	69.01	5.95	63.35	87.36
Ours	**88.48**	**4.04**	**86.34**	**95.04**

D. Ablation Studies

This section outlines ablation experiments focusing on perspective selection and dynamic scene maintenance. Six scenarios (three real-world, three simulated) were used. For each asset in every scene, eight sets of experiments were conducted, each including five specific instances.

For the experiment involving the removal of the dynamic scene maintenance stage, we disabled the object information update module while maintaining interaction with the visual model, ensuring the process was completed using only image information. For the removal of the observation point selection module, we eliminated the selection process, conducting each experiment by randomly choosing a point near the asset to obtain images.

TABLE III: Ablation Study.

Method	NRA↑	NHP↓	ERA↑	WJS↑
Ours	**88.48**	**4.04**	**86.34**	**95.04**
w/o Dynamic Scene Maintenance	68.93	5.78	62.59	87.18
w/o Observation Point Selection	74.62	4.96	71.83	90.39

As shown in TABLE III, when we ablate different modules, the performance of our method degrades to varying degrees. These ablation studies demonstrate the effectiveness of the perspective selection and dynamic scene maintenance stages in updating information for dynamic scenes.

V. CONCLUSIONS

In summary, this study presents an innovative approach to enhancing the dynamic scene perception capabilities of robots powered by LLM. The proposed interaction-driven method addresses the critical limitations of traditional static scene interpretation by enabling robots to adapt to changing environments. Through the introduction of two key modules: the observation point selection module and the dynamic scene maintenance module, robots are equipped to dynamically maintain and update scene information using 3DSG and interaction-driven updates. Experimental validation on the iTHOR and RoboTHOR datasets, as well as real-world applications, demonstrates that this method significantly improves robots' ability to perceive and respond to dynamic scene changes. The results highlight the approach's effectiveness in maintaining accurate and reliable task execution, making it a valuable contribution to the field of robotics.

ACKNOWLEDGMENT

This work was supported by the National Key R&D Program of China(2023YFB4503701).

REFERENCES

[1] C. H. Song, J. Wu, C. Washington, B. M. Sadler, W.-L. Chao, and Y. Su, "Llm-planner: Few-shot grounded planning for embodied agents with large language models," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 2998–3009.

[2] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," *arXiv preprint arXiv:2204.01691*, 2022.

[3] K. Rana, J. Haviland, S. Garg, J. Abou-Chakra, I. Reid, and N. Suenderhauf, "Sayplan: Grounding large language models using 3d scene graphs for scalable robot task planning," in *7th Annual Conference on Robot Learning*, 2023.

[4] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch, "Language models as zero-shot planners: Extracting actionable knowledge for embodied agents," in *International conference on machine learning*. PMLR, 2022, pp. 9118–9147.

[5] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng, "Code as policies: Language model programs for embodied control," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 9493–9500.

[6] J. Mai, J. Chen, G. Qian, M. Elhoseiny, B. Ghanem *et al.*, "Llm as a robotic brain: Unifying egocentric memory and control," *arXiv preprint arXiv:2304.09349*, 2023.

[7] W. Huang, F. Xia, T. Xiao, H. Chan, J. Liang, P. Florence, A. Zeng, J. Tompson, I. Mordatch, Y. Chebotar *et al.*, "Inner monologue: Embodied reasoning through planning with language models," *arXiv preprint arXiv:2207.05608*, 2022.

[8] Z. Ni, X. Deng, C. Tai, X. Zhu, Q. Xie, W. Huang, X. Wu, and L. Zeng, "Grid: Scene-graph-based instruction-driven robotic task planning," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024, pp. 13765–13772.

[9] I. Armeni, Z.-Y. He, J. Gwak, A. R. Zamir, M. Fischer, J. Malik, and S. Savarese, "3d scene graph: A structure for unified semantics, 3d space, and camera," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 5664–5673.

[10] U.-H. Kim, J.-M. Park, T.-J. Song, and J.-H. Kim, "3-d scene graph: A sparse and semantic representation of physical environments for intelligent agents," *IEEE transactions on cybernetics*, vol. 50, no. 12, pp. 4921–4933, 2019.

[11] A. Rosinol, A. Violette, M. Abate, N. Hughes, Y. Chang, J. Shi, A. Gupta, and L. Carlone, "Kimera: From slam to spatial perception with 3d dynamic scene graphs," *The International Journal of Robotics Research*, vol. 40, no. 12-14, pp. 1510–1546, 2021.

[12] P. Gay, J. Stuart, and A. Del Bue, "Visual graphs from motion (vgfm): Scene understanding with object geometry reasoning," in *Computer Vision–ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part III 14*. Springer, 2019, pp. 330–346.

[13] N. Hughes, Y. Chang, and L. Carlone, "Hydra: A real-time spatial perception system for 3d scene graph construction and optimization," *arXiv preprint arXiv:2201.13360*, 2022.

[14] C. Agia, K. M. Jatavallabhula, M. Khodeir, O. Miksik, V. Vineet, M. Mukadam, L. Paull, and F. Shkurti, "Taskography: Evaluating robot task planning over large 3d scene graphs," in *Conference on Robot Learning*. PMLR, 2022, pp. 46–58.

[15] A. Kurenkov, R. Martín-Martín, J. Ichnowski, K. Goldberg, and S. Savarese, "Semantic and geometric modeling with neural message passing in 3d scene graphs for hierarchical mechanical search," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 11 227–11 233.

[16] S. Garg, N. Sünderhauf, F. Dayoub, D. Morrison, A. Cosgun, G. Carneiro, Q. Wu, T.-J. Chin, I. Reid, S. Gould *et al.*, "Semantics for robotic mapping, perception and interaction: A survey," *Foundations and Trends® in Robotics*, vol. 8, no. 1–2, pp. 1–224, 2020.

[17] A. Hagberg, P. J. Swart, and D. A. Schult, "Exploring network structure, dynamics, and function using networkx," Los Alamos National Laboratory (LANL), Los Alamos, NM (United States), Tech. Rep., 2008.

[18] M. Zhao, X. Guo, L. Song, B. Qin, X. Shi, G. H. Lee, and G. Sun, "A general framework for lifelong localization and mapping in changing environment," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 3305–3312.

[19] Q. Gu, A. Kuwajerwala, S. Morin, K. M. Jatavallabhula, B. Sen, A. Agarwal, C. Rivera, W. Paul, K. Ellis, R. Chellappa *et al.*, "Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 5021–5028.

[20] E. Kolve, R. Mottaghi, W. Han, E. VanderBilt, L. Weihs, A. Herrasti, M. Deitke, K. Ehsani, D. Gordon, Y. Zhu *et al.*, "Ai2-thor: An interactive 3d environment for visual ai," *arXiv preprint arXiv:1712.05474*, 2017.

[21] M. Deitke, W. Han, A. Herrasti, A. Kembhavi, E. Kolve, R. Mottaghi, J. Salvador, D. Schwenk, E. VanderBilt, M. Wallingford *et al.*, "Robothor: An open simulation-to-real embodied ai platform," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 3164–3174.

[22] J. Li, D. Li, S. Savarese, and S. Hoi, "Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models," in *International conference on machine learning*. PMLR, 2023, pp. 19 730–19 742.

[23] D. Meyer-Delius, J. Hess, G. Grisetti, and W. Burgard, "Temporary maps for robust localization in semi-static environments," in *2010 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2010, pp. 5750–5755.

[24] A. Walcott-Bryant, M. Kaess, H. Johannsson, and J. J. Leonard, "Dynamic pose graph slam: Long-term mapping in low dynamic environments," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 1871–1878.

[25] F. Schuster, M. Wörner, C. G. Keller, M. Haueis, and C. Curio, "Robust localization based on radar signal clustering," in *2016 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2016, pp. 839–844.

[26] D. M. Rosen, J. Mason, and J. J. Leonard, "Towards lifelong feature-based mapping in semi-static environments," in *2016 IEEE International conference on robotics and automation (ICRA)*. IEEE, 2016, pp. 1063–1070.

[27] H. Lim, S. Hwang, and H. Myung, "Erasor: Egocentric ratio of pseudo occupancy-based dynamic object removal for static 3d point cloud map building," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2272–2279, 2021.

[28] G. Kim and A. Kim, "Lt-mapper: A modular framework for lidar-based lifelong mapping," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 7995–8002.

[29] Y. Long, X. Li, W. Cai, and H. Dong, "Discuss before moving: Visual language navigation via multi-expert discussions," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 17 380–17 387.

[30] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei, "Voxposer: Composable 3d value maps for robotic manipulation with language models," *arXiv preprint arXiv:2307.05973*, 2023.

[31] D. Driess, F. Xia, M. S. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson, Q. Vuong, T. Yu *et al.*, "Palm-e: An embodied multimodal language model," *arXiv preprint arXiv:2303.03378*, 2023.

[32] S. H. Vemprala, R. Bonatti, A. Bucker, and A. Kapoor, "Chatgpt for robotics: Design principles and model abilities," *IEEE Access*, 2024.

[33] S.-C. Wu, J. Wald, K. Tateno, N. Navab, and F. Tombari, "Scenegraphfusion: Incremental 3d scene graph prediction from rgb-d sequences," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 7515–7525.

[34] J. Wald, H. Dhamo, N. Navab, and F. Tombari, "Learning 3d semantic scene graphs from 3d indoor reconstructions," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 3961–3970.

[35] S. Koch, P. Hermosilla, N. Vaskevicius, M. Colosi, and T. Ropinski, "Sgrec3d: Self-supervised 3d scene graph learning via object-level scene reconstruction," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024, pp. 3404–3414.

[36] D. Alexey, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv: 2010.11929*, 2020.

[37] Y.-F. Tang, C. Tai, F.-X. Chen, W.-T. Zhang, T. Zhang, X.-P. Liu, Y.-J. Liu, and L. Zeng, "Mobile robot oriented large-scale indoor dataset for dynamic scene understanding," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 613–620.

[38] Z. Zhu, S. Peng, V. Larsson, W. Xu, H. Bao, Z. Cui, M. R. Oswald, and M. Pollefeys, "Nice-slam: Neural implicit scalable encoding for slam," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 12 786–12 796.