MSGNav: Unleashing the Power of Multi-modal 3D Scene Graph for Zero-Shot Embodied Navigation

Supplementary Material

1. Implement Details

This section outlines the implementation details of our approach, including computational resource, hyper-parameters, and the vision foundation models (VFMs) used for constructing M3DSG.

1.1. Computational Resource

Our approach primarily employs a Vision-Language Model (VLM) for inference. Consequently, when utilizing API calls to closed-source models, it places only modest demands on local GPU computational resources. All experiments were conducted using NVIDIA A800-SXM4-40GB graphics cards, with each card requiring approximately 8 GB to 10 GB of VRAM to perform VFMs inference. This configuration ensures stable performance while maintaining cost-effectiveness in resource usage.

Meanwhile, inference with open-source large models such as GPT-OSS [1] and LLAVA [2] is also feasible. However, this configuration demands additional GPU memory that scales in proportion to the parameter size of open-source large models.

1.2. Hyper-parameters

The details of the hyperparameters of `MSGNav` are shown in Table 1. Additionally, we maintain consistency with 3D-Mem and Conceptgraph for other parameters in the 3D scene graph construction.

Hyperparameters	Value
Radii set: $\mathbf{R}$	[0.5, 0.75] (m)
Camera height: h,	1.5 (m)
Success threshold: d	0.25 (m)
Samples number: K	20
Adjacency distance: θ	3.5 (m)
Obstruction distance: τ	0.05 (m)
Top-k related objects: $\lvert\mathbf{O}^{rel}\rvert$	20
Hfov of camera	120°
Image resolution	1280x1280
Image prompt resolution	512x512

Table 1. The details of hyperparameters of `MSGNav`.

[1]GPT-OSS: https://github.com/openai/gpt-oss
[2]LLAVA: https://github.com/haotian-liu/LLaVA

Figure 1. A visualization example of perception results during constructing the M3DSG.

1.3. VFMs for Constructing M3DSG

As outlined in the main paper, the 3D scene reconstruction pipeline primarily incorporates the following Visual Feature Modules (VFMs). First, YOLO-World [2] is applied for object detection and room recognition. Next, SAM [4] is utilized to generate semantic masks. Finally, CLIP [5] is employed to extract visual feature representations of the objects.

An example of visualized perception results is shown as Fig. 1, illustrating the outputs obtained after processing with YOLO-World and SAM for open-vocabulary object detection and semantic segmentation.

And the detailed information of VFMs is shown in Table 2.

VFM	Version	Size
Yolo-World [2]	YOLOv8x-world	141.11 MB
SAM [4]	SAM-Large	1.16 GB
CLIP [5]	ViT-H-14-quickgelu/dfn5b	12.20 GB

Table 2. The detailed information of VFMs for constructing M3DSG.

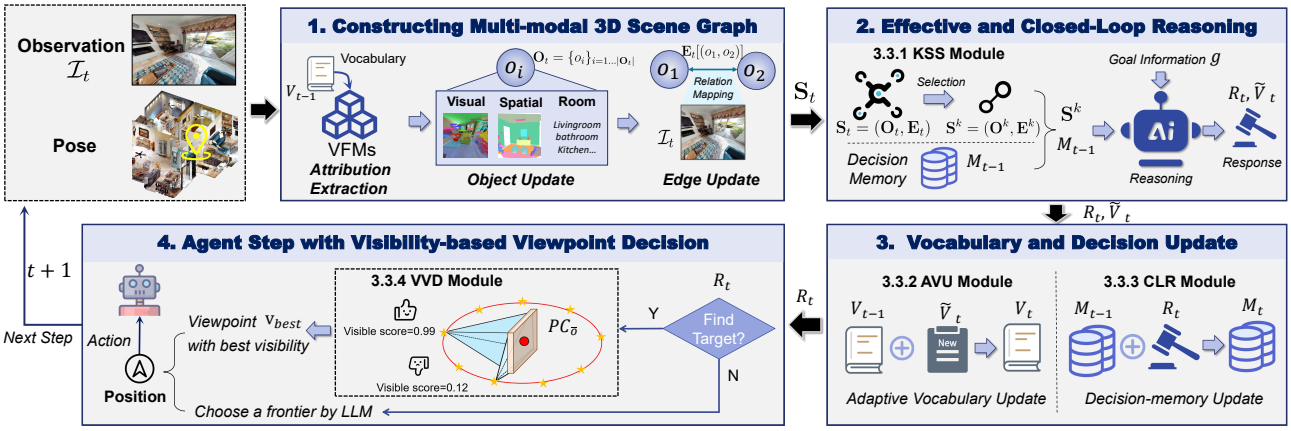

Figure 2. **Detailed end-to-end processing flow of the proposed framework.** The system executes a continuous sense-reason-act loop across four key stages: (1) multi-modal 3D scene graph construction via Vision Foundation Models (VFMs), (2) closed-loop reasoning based on the extracted sub-graph $\mathbf{S}^k$ and historical memory, (3) adaptive updates to the semantic vocabulary and decision memory, and (4) visibility-based viewpoint decision or frontier exploration. Please refer to the corresponding text for a detailed description of each mathematical symbol and data flow.

Method	Training-free	SR ↑	SPL ↑
BCRL [6]	×	20.2	8.2
DAgger [6]	×	18.1	9.4
DAgRL [6]	×	41.3	21.2
RL [7]	×	39.2	18.7
VLFM [9]	✓	35.2	18.6
Uni-NaVid [11]	×	41.3	21.2
DAgRL+OD [10]	×	35.8	21.2
MTU3D [12]	×	**55.0**	23.6
MSGNav (Ours)	✓	48.3	**27.0**

Table 3. Comparison of navigation performance on the **Val Seen** split of HM3D-OVON. This experiment evaluates the model's optimization effectiveness and navigation efficiency within the training distribution.

1.4. Illustration of End-to-end Framework

In addition to the framework diagram in the main paper, we also provide a more intuitive end-to-end framework diagram that systematically illustrates the data flow and processing logic between modules, as shown in Fig. 2.

The system operates in a continuous sense-reason-act loop at each timestep t, which can be detailed in four key stages: **(1) Constructing Multi-modal 3D Scene Graph:** Given the current observation $\mathcal{I}_t$ and agent pose, Vision Foundation Models (VFMs) extract visual, spatial, and room-level attributes to incrementally update the object nodes and relational edges of the scene graph $\mathbf{S}_t$. **(2) Effective and Closed-Loop Reasoning:** A Key Sub-graph Selection (KSS) module filters $\mathbf{S}_t$ into a task-relevant sub-graph $\mathbf{S}^k$. An LLM agent then reasons over $\mathbf{S}^k$, historical decision memory M_{t-1}, and goal information $\mathcal{G}$ to generate an action response R_t and proposed vocabulary updates $\tilde{V}_t$. **(3) Vocabulary and Decision Update:** The Adaptive Vocabulary Update (AVU) module integrates $\tilde{V}_t$ to form the current vocabulary V_t, while the Closed-Loop Reasoning (CLR) module records R_t to update the decision memory M_t. **(4) Agent Step with Visibility-based Viewpoint Decision:** If the target is identified, the VVD module selects the optimal viewpoint $\mathbf{v}_{best}$ that maximizes the visibility score of the target point cloud $PC_{\bar{o}}$; otherwise, the LLM selects a new frontier for exploration in step $t+1$.

2. More Experimental Results

2.1. Additional Analysis on HM3D-OVON

To further investigate the foundational perception and optimization stability of the proposed MSGNav, we conduct supplementary experiments on the HM3D-OVON dataset [10]. For this analysis, we specifically report the results on the "*Val Seen*" split to evaluate the model's performance when navigating within familiar environment distributions, which serves as a benchmark for the model's upper-bound efficiency and basic semantic mapping capabilities.

As presented in Table 3, we compare MSGNav with several state-of-the-art training-based and training-free methods on the seen environments. While the training-based method MTU3D [12] achieves a higher Success Rate (SR) of 55.0% by leveraging extensive environment-specific learning, MSGNav significantly outperforms all methods in terms of Success weighted by Path Length (SPL), reaching **27.0%**.

This result is particularly noteworthy as MSGNav is a

VLM	**Overall** (2669)		**Object Category** (991)		**Language** (856)		**Image** (822)	
(MSGNav)	Success Rate	SPL	Success Rate	SPL	Success Rate	SPL	Success Rate	SPL
GPT-4o [3]	51.97	29.56	53.38	29.98	**47.43**	25.08	54.99	**33.71**
Qwen-VL-Max [1]	**52.79**	**30.79**	**56.10**	**31.53**	45.56	**27.18**	**56.33**	33.67

Table 4. Experiments of our MSGNav method using different VLMs on the "*Val Unseen*" split of GOAT-Bench. The number following each category represents the sample size.

training-free approach. The superior SPL indicates that even in seen scenarios, our method generates more efficient and purposeful navigation paths compared to models that might overfit to specific trajectories. These supplementary results confirm that MSGNav maintains robust basic navigation logic and high path efficiency, which provides a solid foundation for the zero-shot generalization capabilities demonstrated in our primary experiments (Table 1 in the main paper).

2.2. Quantitative experiments on Goat-Bench

We present the detailed experimental results in Table 2 of the main paper, showing the detailed performance of MSGNav across different categories with various VLM. As shown in the table 4, MSGNav demonstrates strong adaptability across different VLMs instead of relying on the specific model. MSGNav achieves comparable results on both state-of-the-art VLMs, Qwen-VL-Max and GPT-4o. It is worth noting that the results on Qwen-VL-Max are slightly better than those on GPT-4o. This is primarily due to the costly token fees of GPT-4o, our prompts were primarily implemented based on Qwen-VL-Max, and they may be better suited for Qwen-VL-Max. However, the excellent results achieved on GPT-4o without any prompt adjustments demonstrate that the system does not overly rely on prompt engineering.

3. Analysis of the Exploration

3.1. Additional Details of VVD Module

We provide an example illustrating how the VVD module selects optimal viewpoints. As shown in Fig. 3, candidate viewpoints located closer to the ground-truth (GT) viewpoints receive higher visibility scores. In this example, the highest-scoring viewpoint (Viewpoint 1, score 0.97) is positioned directly in front of the target, offering an unobstructed field of view that clearly captures the target object.

By contrast, the medium-scoring viewpoint (Viewpoint 2, score 0.83) is positioned on the side of the target, thereby capturing only partial visual information. Notably, Viewpoint 3 (score 0.05) is severely occluded by environmental structures, which may be less discernible purely from a simplified 2D bird's-eye view (BEV) representation. In such cases, the VVD module correctly identifies the occlusion by computing 3D line-of-sight, consistently assigning low visibility scores to these obstructed viewpoints.

To further clarify the mechanics of the VVD module, we expand upon the operations detailed in Algorithm 2 of the main paper. The algorithm systematically evaluates line-of-sight occlusion in 3D space to identify the optimal navigation endpoint.

1. **Candidate Generation (Lines 1-4):** The algorithm first computes the centroid $\mathbf{c}_{\bar{o}}$ of the localized target's point cloud $\mathcal{PC}_{\bar{o}}$. It then generates a set of K candidate viewpoints in a concentric circle (or multiple concentric circles defined by radii set $\mathbf{R}$) around this centroid. Crucially, it filters this set to retain only viewpoints $\mathbf{V}_c$ that reside in traversable free space, ensuring the agent can physically reach them.
2. **Ray Construction (Line 7):** For a given candidate viewpoint $\mathbf{v}_i$ and a specific point $\mathbf{p}$ belonging to the target object, the algorithm mathematically defines a ray segment $\mathcal{Q}(\mathbf{v}_i, \mathbf{p})$. This segment connects the viewpoint and the target point. To account for the physical volume of the agent and potential sensor noise, the parameter τ introduces a safety margin at both ends of the ray.
3. **Occlusion Evaluation (Line 8):** The core visibility condition $\mathcal{E}(\mathbf{v}_i, \mathbf{p})$ evaluates whether the defined ray $\mathcal{Q}$ is obstructed by the broader scene point cloud $\mathcal{PC}$. A target point $\mathbf{p}$ is considered visible if and only if the minimum distance between any point $\mathbf{q}$ on the ray and any point $\mathbf{s}$ in the environment point cloud is strictly greater than or equal to the obstruction margin τ.
4. **Visibility Scoring (Lines 9-13):** The algorithm computes an aggregate visibility score $S_{\mathbf{v}_i}$ for the candidate viewpoint. This score represents the ratio of target points $\mathbf{p} \in \mathcal{PC}_{\bar{o}}$ that satisfy the visibility condition $\mathcal{E}$. The viewpoint achieving the highest ratio is selected as the optimal navigation target $\mathbf{v}_{best}$, guaranteeing the agent finishes its trajectory with a robust, unoccluded view of the target.

3.2. Visualization about VVD Module

We provide a real-world example illustrating how the VVD module selects optimal viewpoints. As shown in Fig. 4, candidate viewpoints located closer to the ground-truth (GT) viewpoints, marked by the purple triangle, receive higher visibility scores. In this example, the highest-scoring

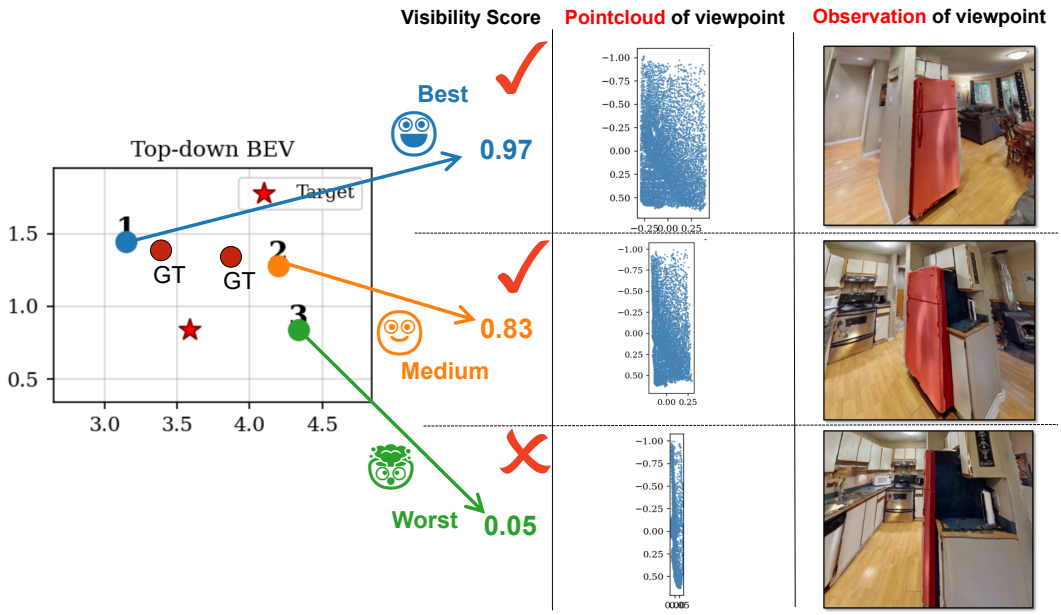

Figure 3. **Qualitative Visualization of the Visibility-based Viewpoint Decision (VVD) Module.** The central plot shows a top-down Bird's-Eye View (BEV) of the agent's spatial environment. Red stars indicate potential targets, while red circles mark the Ground Truth (GT) target viewpoints. The VVD module evaluates candidate viewpoints (numbered 1, 2, and 3), calculating a visibility score based on ray-casting to the target point cloud. Viewpoint 1 achieves the 'Best' visibility score (0.97), and its corresponding first-person observation provides a clear, unoccluded view of the red refrigerator. Viewpoint 2 receives a 'Medium' score (0.83), capturing a partial, side-angle view. Viewpoint 3 receives the 'Worst' score (0.05), as its line of sight is heavily occluded by other environmental structures.

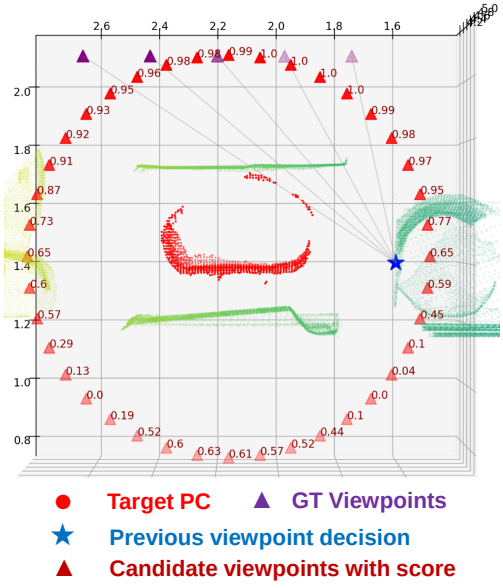

Figure 4. An example of viewpoint decision by the VVD module. This shows the BEV view of the point cloud in the scene and the corresponding viewpoint information.

viewpoint is almost coincident with the GT viewpoint and is positioned directly in front of the target (top), offering an unobstructed field of view.

By contrast, the previous viewpoint selection was farther from the GT viewpoint and positioned on the side of the target, thereby capturing only partial visual information. Notably, no GT viewpoint exists at the rear (lower side) of the target due to occlusion caused by point cloud data (green dots) at the same height within the scene, which may be less discernible in the bird's-eye view (BEV) representation. In such cases, the VVD module consistently assigns low visibility scores (0.1–0.6) to these occluded viewpoints.

3.3. Visualization about Exploration

As shown in Fig. 5, we present the natural language-guided target navigation by visualizing the agent's process of exploring and localizing the target described as "refrigerator in the kitchen. It is located next to the kitchen cabinet and the worktop" to intuitively demonstrate its closed-loop reasoning.

The agent starts at the predefined Start Position and begins to explore adjacent areas. At this stage, the target refrigerator is still unseen (indicated by a red dot). The agent continuously moves along the Frontier (the boundary between explored and unexplored areas, marked by purple arrows) to uncover more regions. Its movement trajectory is recorded as the Exploration path (light gray line), and its Current Position (blue dot) is updated dynamically as it nav-

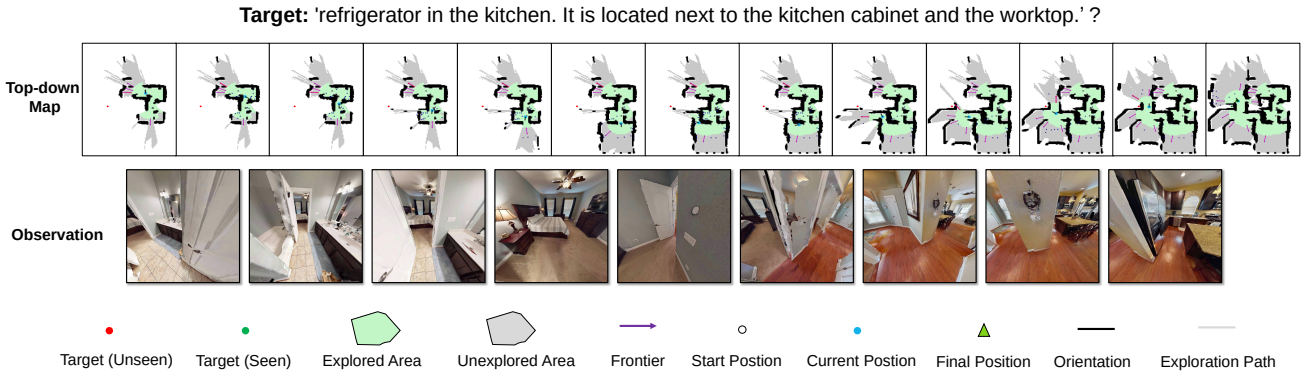

Figure 5. This visualization illustrates the process of an agent exploring and navigating to a target described by natural language—"refrigerator in the kitchen. It is located next to the kitchen cabinet and the worktop"—in an indoor environment. It is composed of three parts from top to bottom: the Top-down Map sequence, the Observation sequence (first-person visual inputs), and the legend, which respectively display the global exploration state, the agent's local visual perception, and the meaning of each visual element.

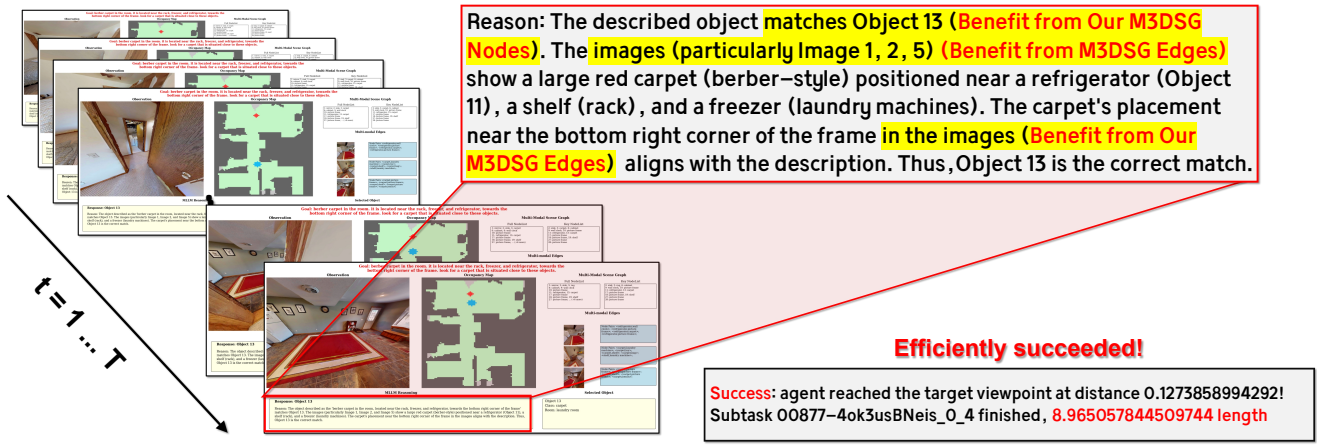

Figure 6. **Visualization of the VLM reasoning process based on M3DSG.** The agent successfully locates the target "berber carpet" by leveraging M3DSG Nodes to match the object candidates (Object 13) and Multi-modal Edges (images) to verify complex spatial relationships, such as the carpet's placement near the refrigerator (Object 11) and its position in the bottom right corner of the frame. This accurate grounding leads to an efficiently successful navigation task.

igates through different rooms. As the agent ventures into the kitchen area, the Observation sequence captures the visual input of the refrigerator adjacent to the kitchen cabinet and worktop. Simultaneously, in the Top-down Map, the target marker changes from a red dot (Target (unseen)) to a green dot (Target (seen)), and the agent reaches the Final Position (green triangle), completing this hard navigation task.

This visualization comprehensively demonstrates how the agent uses the multi-modal scene graph to sequentially reason and localize the target object in an unknown environment. With the help of our multi-modal scene graph, agents can gradually explore and locate targets even when they are extremely distant. This further validates the effectiveness of our multi-modal scene graph for reasoning in zero-shot embodied navigation tasks.

Furthermore, to explicitly illustrate the critical role of our proposed M3DSG in the granular decision-making process, we visualize the Vision-Language Model (VLM) reasoning process in Fig. 6. In this episode, the agent is tasked with finding a target based on a complex spatial description: "herber carpet is the room. It is located near the rack, freezer, and refrigerator towards the bottom right corner of the frame". As the exploration proceeds ($t = 1 \dots T$), the M3DSG incrementally updates. When matching the target, the VLM relies on the M3DSG Nodes to identify the candidate, explicitly noting that the described object matches Object 13. More importantly, it heavily depends on the Multi-modal Edges (stored images) to verify fine-grained spatial relations. The reasoning process highlights that specific im-

ages (particularly Image 1, 2, and 5) show a large red carpet positioned near a refrigerator (Object 11), a shelf, and a freezer. Additionally, the visual edges confirm the carpet's placement near the bottom right corner of the frame, perfectly aligning with the language description. By combining node attributes and visual edge evidence, the agent confidently concludes that Object 13 is the correct match. Ultimately, this accurate multimodal reasoning allows the agent to efficiently succeed in the task, reaching the target viewpoint at a remarkably close distance of 0.127m.

3.4. Analysis of Reasoning based on M3DSG

We present an example illustrating the use of a Vision-Language Model (VLM) for multimodal scene graph reasoning. As shown in Fig. 7, when given the query "plant that is located near the window glass and radiator", the conventional textual description in the scene graph failed to include information pertinent to the target. In contrast, the VLM, utilizing the supplementary visual information embedded in the scene graph, successfully identified the target matching the query. The model then enriched the scene graph through vocabulary supplementation and updates. As a result, the task was completed with a Success weighted by Path Length (SPL) of 89.79, requiring only about 3 m of exploration length. This demonstrates that retaining images within the multimodal scene graph substantially improves exploration efficiency.

1. Question?

Could you find the object exactly described as the 'plant that is located near the window glass and radiator.'?

2. Input Multi-modal Scene Graph

Objects Attribution:

radiator: 8, (10.99, -2.15, 0.36), kitchen room
chair: 10, (11.30, -1.26, 0.87), kitchen room
bed: 11, (10.85, -0.64, 0.64), kitchen room
chair: 12, (11.33, -0.10, 0.49), kitchen room
chair: 18, (10.74, -1.28, 0.87), kitchen room

Relationship Attribution:

8, 8, [Image 0, Image 1]
8, 10, [Image 0]
8, 11, [Image 0, Image 1]
8, 12, [Image 0]
8, 18, [Image 1]

Image List:

Image 0 [iVBORw0KGg...]
Image 1 [iVBORw0KGg...]

3. Response and Reason from VLM

Response: [image 0, plant]

Reason: [The image shows a plant located near the window and radiator, matching the description.]

4. Task Complete

- Success: agent reached the target viewpoint at distance 0.1468908041715622!
- Subtask 00890-6s7QHgap2fW_1_0 finished, 3.2773782577578823 length
- SPL by distance: 89.79
- Filtered frames/Total frames: 2/26

Figure 7. Example illustrating the use of a Vision-Language Model (VLM) for multimodal scene graph reasoning.

System Prompt:

You are an agent in an indoor scene who can observe the environment and explore to find a target object. You must choose an Image or an Object as the answer, in order to find the specified target object.

To efficiently solve the problem, you should identify key objects that are most helpful for guiding exploration toward the target.
Please follow these strict instructions:
1. Read and understand the full 3D scene graph. Each object includes its id, class, room, and nearby objects (i.e., its neighbors in the graph).
2. Rank objects by how helpful they are for locating the target, based on: Semantic relevance to the target; Co-occurrence with the target in typical environments; Presence in the same room as the target.
3. Choose only the most informative and strategically diverse objects for exploration. To maximize coverage: Avoid choosing objects that are directly connected (i.e., neighbors) in the scene graph.Here is is the format for input 3D scene graph:
Object ID: Class, Located room, nearby objects ID
Question: {*Example Question*}.
{*Example Scene Graph*}
Answer:
1
5

Content Prompt:

Following is the concrete content of the task and you should retrieve helpful key objects in order.
Question: {*Task Question*}
Following is the 3D scene graph based on the above input format
1: spa bench, laundry room, [1, 2]
2: wall cabinet, laundry room, [1, 2]
{. . .}
Do not print any object that are not included in the 3D scene graph or include any additional information other than the ID in your response:
Answer:

Figure 8. Prompt 1 for "top-k object nodes selection in KSS module". The placeholders {...} will be replaced by the corresponding information. The gray-highlighted text represents the information of compressed M3DSG.

4. VLM Reasoning and Prompt

Our **MGSNav** method leverages Vision-Language Model (VLM) reasoning in three scenarios during the navigation process:

- **Top-k object nodes selection in KSS** (As shown in Fig. 8) — Within the Key Scene Selection (KSS) module, the VLM is applied to infer the top-k target-related nodes from the compressed scene graph.
- **Exploration inference** — The KSS-compressed key

scene subgraph is processed by the VLM to identify either the target node or the exploration frontier. This reasoning is performed in two sequential steps:

1. **Find target**. If the target is present in the 3D scene graph, input the scene graph information to identify the target ID. (As shown in Fig. 9)
2. **Explore frontier**. If absent, input the frontier image to determine the exploration frontier image ID. (As shown in Fig. 10)

This decomposition facilitates more accurate VLM reasoning.

- **Task completion verification** — Once the agent considers the task completed, the VLM utilizes observation images from the past steps to perform a final inference, thereby validating task credibility. (As shown in Fig. 11)

To better understand the process of VLM reasoning, we present the prompts used for the three VLM reasoning scenarios described above.

References

[1] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. ARXIV, 2023. 3

[2] Tianheng Cheng, Lin Song, Yixiao Ge, Wenyu Liu, Xinggang Wang, and Ying Shan. Yolo-world: Real-time open-vocabulary object detection. In CVPR, pages 16901–16911, 2024. 1

[3] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. ARXIV, 2024. 3

[4] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In ICCV, pages 4015–4026, 2023. 1

[5] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In ICML, pages 8748–8763. PmLR, 2021. 1

[6] Ram Ramrakhya, Dhruv Batra, Erik Wijmans, and Abhishek Das. Pirlnav: Pretraining with imitation and rl finetuning for objectnav. In CVPR, pages 17896–17906, 2023. 2

[7] 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, 2019. 2

[8] Yuncong Yang, Han Yang, Jiachen Zhou, Peihao Chen, Hongxin Zhang, Yilun Du, and Chuang Gan. 3d-mem: 3d scene memory for embodied exploration and reasoning. In CVPR, pages 17294–17303, 2025. 9

[9] Naoki Yokoyama, Sehoon Ha, Dhruv Batra, Jiuguang Wang, and Bernadette Bucher. vlfm: Vision-language frontier maps for zero-shot semantic navigation. In ICRA, pages 42–48. IEEE, 2024. 2

[10] Naoki Yokoyama, Ram Ramrakhya, Abhishek Das, Dhruv Batra, and Sehoon Ha. Hm3d-ovon: A dataset and benchmark for open-vocabulary object goal navigation. In IROS, pages 5543–5550. IEEE, 2024. 2

[11] Jiazhao Zhang, Kunyu Wang, Shaoan Wang, Minghan Li, Haoran Liu, Songlin Wei, Zhongyuan Wang, Zhizheng Zhang, and He Wang. Uni-navid: A video-based vision-language-action model for unifying embodied navigation tasks. ARXIV, 2024. 2

[12] Ziyu Zhu, Xilin Wang, Yixuan Li, Zhuofan Zhang, Xiaojian Ma, Yixin Chen, Baoxiong Jia, Wei Liang, Qian Yu, Zhidong Deng, et al. Move to understand a 3d scene: Bridging visual grounding and exploration for efficient and versatile embodied navigation. ARXIV, 2025. 2

System Prompt:

```
You are an agent in an indoor scene who can observe the environment and explore
to find a target object. You must choose an Image, a Frontier (for further
exploration), or directly select an Object as the answer, in order to find the
specified target object.
Scene Graph Definitions:
    Objects Attribution: Each object's name, ID, position, and location.
    Relationship Attribution: Relationships between objects, often with a
reference image.
    Image List: Supplementary images from the nodes of scenegraph.
    Frontier: Unexplored regions that may provide new information.
    History Decision: All previous choices you made (avoid repeating them).
Instructions:
    Step 1: Examine the Objects Attribution section. If the target object is
explicitly listed (by name or ID), and fits the question, select it immediately
as the answer.
    Step 2: If the target object is not explicitly in Objects Attribution, check
the Relationship Attribution section. Use relationships and referenced images in
the Image List to help you identify the target.
    Step 3: If you still cannot identify the object, select to further explore
and gather more information.
    Step 4: Provide your answer in one of the following formats:
        'Object i': If the object is found in Objects Attribution.
        'Image i, j': If the object is likely to exist in image i, and j is the
required object category. Ensure you include the category name.
        'Continue Exploration': If the object is not found and further
exploration is necessary.
Additional Notes:
1. Try not to select any object, image that is in the "History Decision" list
unless you are very confident.
2. The detected class name and located room in the Objects Attribution may be
inaccurate, use the images to verify.
3. Only provide the required answer(with optional brief reasoning in a new line).
```

Content Prompt:

```
Here is the Question you need to solve: {Question}'?
Objects Attribution:
    ventilation hood: 16, (1.99, 1.55, 1.63), kitchen room
    wall cabinet: 19, (3.17, 1.97, 0.46), kitchen room
    {...}
Relationship Attribution:
    16, 7, [Image 0]
    16, 16, [Image 0, Image 1]
    {...}
Image List:
Please note that the class name and Located room may not be accurate due to the
limitation of the detection model. So you still need to utilize the images to
make the decision.
    Image 0 [iVBORw0KGg...]
    Image 1 [iVBORw0KGg...]
    {...}
The followings are all the previous Decisions that you made: (now step is 10/50).
Choosing those incorrect objects or images again is prohibited:
   step 0 : Choosing a Frontier to explore.
   step 1 : Choosing a Frontier to explore.
    {...}
Answer:
You can explain the reason for your choice, but put it in a new line after the
choice.
```

Figure 9. Prompt 2 for "*find target in exploration inference*". The placeholders {...} will be replaced by the corresponding information. The gray-highlighted text represents the information of the full M3DSG.

System Prompt:

```
You are an agent tasked with finding a target object in an indoor
scene. Your mission is to choose the most promising frontier for
further exploration to locate the specified target object.
```

Content Prompt:

```
Here is the Question you need to solve: {Question}
The Frontiers that you can explore:
    Frontier 0 [iVBORw0KGg...]
    Frontier 1 [iVBORw0KGg...]
    {...}
You can explain the reason for your choice, but put it in a new
line after the choice.
The example for the answer:
Frontier 0
I chose Frontier 0 for exploration, which show direction to
kitchens where refrigerators are more likely to appear
```

Figure 10. Prompt 3 for "*explore frontier in exploration inference*". The placeholders {...} will be replaced by the corresponding information. The gray-highlighted text represents the information of frontier images as in 3D-Mem [8].

System Prompt:

```
Task: You are an agent in an indoor scene that is able to observe the
surroundings and explore the environment. You are tasked with indoor
navigation, and you are required to choose a Image or a Frontier to
explore, or directly select an Object, finally find the target object
required in the question.

Definitions:
Now that you have arrived near the previously selected answer, please
observe your surroundings and confirm whether you have really reached
the object required by the Question (close enough, less than 0.25m).
```

Content Prompt:

```
Here is the Question you need to solve: {Question}'?
The following are surrounding observations about the egocentric view
of the agent:
    Surrounding observation 1 [iVBORw0KGg...]
    Surrounding observation 2 [iVBORw0KGg...]
    {...}
Answer: (Yes or No)
You can explain the reason for your choice, but put it in a new line
after the choice.
```

Figure 11. Prompt 4 for "*task completion verification*". The placeholders {...} will be replaced by the corresponding information. The gray-highlighted text represents the information of observation images from the past steps.