

Multi-Scale Gaussian-Language Map for Zero-shot Embodied Navigation and Reasoning

Sixian Zhang^{1,2}, Yiyao Wang^{1,2}, Xinhang Song^{1,2}, Keming Zhang^{1,2}, Zijian Xu^{1,2}, Shuqiang Jiang^{2,3*}

¹State Key Laboratory of AI Safety, Institute of Computing Technology, Chinese Academy of Sciences, Beijing

²University of Chinese Academy of Sciences, Beijing, ³Institute of Computing Technology, Chinese Academy of Sciences, Beijing

{sixian.zhang, yiyao.wang, xinhang.song, keming.zhang, zijian.xu}@vipl.ict.ac.cn,
sqjiang@ict.ac.cn

1. Formula Details

In the supplementary material, we detail the merge operator $G_{\text{new}} \leftarrow G_i \oplus G_j$ used in Eq. 5 of the main paper. Each Gaussian G_i is parameterized by its mean $\mu_i \in \mathbb{R}^3$, covariance $\Sigma_i \in \mathbb{R}^{3 \times 3}$, color $\mathbf{c}_i \in \mathbb{R}^3$, opacity α_i , and a scalar weight w_i . The weight w_i is given by the confidence score predicted by Grounding DINO for the corresponding detection. Given a pair of (G_i, G_j) to be merged, we define the merged Gaussian G_{new} as

$$\begin{aligned}\mu_{\text{new}} &= \frac{w_i \mu_i + w_j \mu_j}{w_i + w_j} \\ \Sigma_{\text{new}} &= \frac{w_i \Sigma_i + w_j \Sigma_j}{w_i + w_j} + \frac{w_i w_j}{(w_i + w_j)^2} (\mu_i - \mu_j)(\mu_i - \mu_j)^\top \\ \mathbf{c}_{\text{new}} &= \frac{w_i \mathbf{c}_i + w_j \mathbf{c}_j}{w_i + w_j} \\ \alpha_{\text{new}} &= 0.8\end{aligned}\tag{1}$$

Since all Gaussians are directly fitted from the input point cloud, the points inside each voxelized cell predominantly come from the surface of a single instance, i.e., these points are non-transparent. We therefore fix the opacity to a constant value, $\alpha_{\text{new}} = 0.8$.

2. Computational Efficiency Analysis

The computational overhead of constructing the GLMap is primarily dominated by calls to pretrained large models: an MLLM (Gemma3-27B) for generating textual descriptions from RGB observations, and an LLM (Qwen3-8B) for integrating the textual descriptions stored in the description buffer. Specifically, the MLLM is queried at every timestep to interpret the current egocentric RGB frame and to produce both instance-level and region-level textual descriptions.

In contrast, the LLM is used solely to merge descriptions that refer to the same semantic unit, and we reduce the number of LLM calls through the following strategies. 1) Spatial connectivity pre-checking. Before adding a new description to the buffer, we verify whether the corresponding semantic unit (either an instance or a region) is spatially connected, i.e., whether the instance Gaussians are spatially mergeable or, for regions, whether the instances contained within them exhibit spatial overlap. This spatial check prevents irrelevant textual descriptions from entering the buffer, and thus avoids unnecessary buffer growth. 2) Buffered merging. Instead of invoking the LLM at every new description, we accumulate concatenated descriptions in a buffer and trigger an LLM-based merge only when the buffer length exceeds a 300-token threshold. This strategy ensures that LLM invocations are not excessively frequent. Statistically, these strategies result in an average of 1.4 merges per instance and 1.9 merges per region, corresponding to approximately 0.6 Qwen3-8B calls per step.

For the use of GLMap, given an LLM-based ObjectNav method built on GLMap, the LLM estimates for each instance and region the probability that the target is nearby. The prompt to the LLM is issued in a single pass because the 32K context window of Qwen3-8B can accommodate all required information. In addition, the LLM is queried only at sparse decision points rather than at every step. Consequently, LLM calls triggered by navigation decisions occur at an average rate of 0.8 per step.

Consequently, the total FLOPs of GLMap spent on large-model inference remain comparable to existing works (e.g., SG-Nav [2] and UniGOAL [3]).

3. Real-World Robot Evaluation

3.1. Experimental Setup

We conduct real-world evaluations in a fully furnished indoor environment covering approximately 120 m². The test environment includes common household furniture cat-

*Corresponding author.

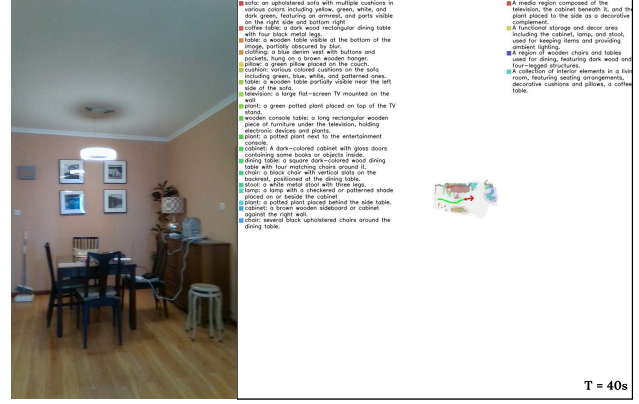
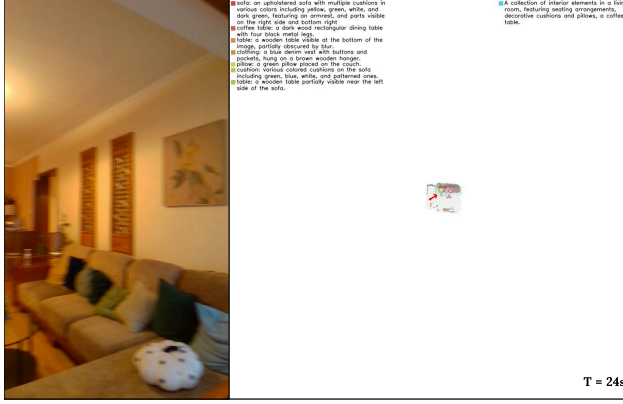


Figure 1. The Incremental Construction of GLMap in Real-World. During real-world navigation, GLMap is incrementally constructed as the agent explores the environment. Instance information is displayed in the left column of the map, while region information is presented in the right column.

egories such as beds, sofas, televisions, coffee tables, chairs, dining tables, potted plants, and toilets.

The experiments are performed using the Hello Robot Stretch 3 platform. The Stretch 3 robot features a compact wheeled base with three wheels: two independently driven wheels for differential drive and one passive omniwheel for stabilization. The base width is less than 350 mm, allowing the robot to traverse narrow doorways and constrained indoor passages. The robot is equipped with an Intel RealSense D435i depth camera as its primary perception sensor. The camera operates at a resolution of 1280×720 with a 58° horizontal field of view. For computation, we employ an external workstation with an NVIDIA RTX 4090 GPU to perform real-time inference. The workstation communicates with Stretch 3 via a local area network, enabling low-latency data transmission. The mapping and waypoint determination are computed on the server and subsequently dispatched to the robot for execution.

Table 1. **Real-World Robot Evaluation Results.** Performance comparison between our GLMap and the baseline VLFM [4] across ObjectNav and InstNav tasks.

Task	Target	VLFM [4]	GLMap
ObjectNav	bed	70.00	76.67
	TV	56.67	60.00
	dining table	83.33	83.33
	toilet	50.00	63.33
InstNav	potted plant beside the TV	30.00	56.66
	lamp on the nightstand	23.33	63.33

3.2. Evaluation Results

We evaluate GLMap in real-world environments as shown in Tab. 1. The baseline is VLFM [4]. For each target, we randomly select 10 feasible starting positions and conduct 3 trials per position. The evaluation covers 6 navigation targets, including 4 ObjectNav category-level goals (bed, TV,

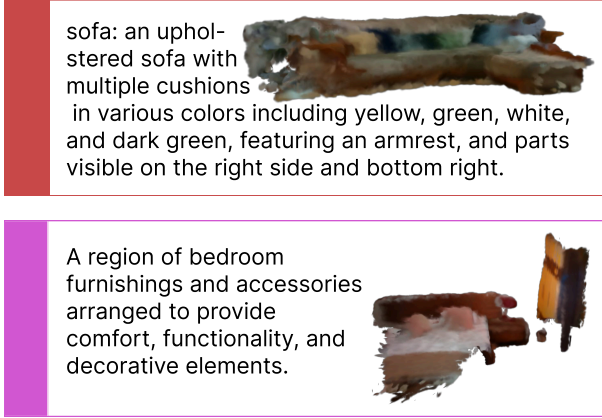


Figure 2. Real-World Gaussian Estimated Results. The figure above depicts a sofa, representing an instance along with its corresponding description; the figure below illustrates a bedroom region, accompanied by its associated description.

dining table, toilet) and 2 InstNav instance-level goals constrained by contextual semantics (lamp on the nightstand, potted plant beside the TV). A navigation episode is considered successful when the robot actively stops within 1.0 m of the target and the object is visible in the egocentric view.

In ObjectNav tasks, GLMap consistently outperforms the baseline. However, the improvements are not very large. We infer that this is because ObjectNav only requires finding any instance within a semantic category. GLMap encodes both instance-level and region-level semantics, but the benefits of instance-level information are less evident in category-based navigation. Even so, region-level semantics still provide performance gains for most ObjectNav categories. In contrast, InstNav tasks benefit substantially more from GLMap. Since these tasks require precise grounding of instance-level targets with contextual constraints (e.g., “lamp on the nightstand”), both instance-level and region-level semantics play a critical role. GLMap leverages these richer representations to achieve significantly higher success rates compared with the baseline, demonstrating the effectiveness of our GLMap for ObjectNav and InstNav in real-world.

3.3. Visual Results

In this section, we provide qualitative results to address two questions: (1) how well the Gaussian estimator in GLMap performs in real-world environments, and (2) the quality of the semantic map produced by GLMap under real-world observations.

As shown in Fig. 2, RGB-D observations from the real environment are fed into our Gaussian estimator to infer the Gaussian parameters of each instance. We then ren-

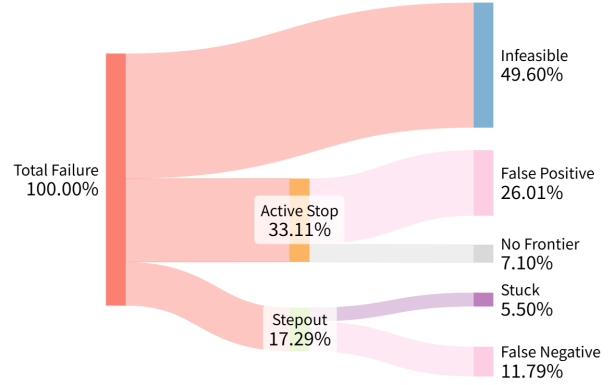


Figure 3. Failure cases of GLMap on HM3D. ‘Infeasible’ means the target does not exist on the agent’s starting floor. ‘False Positive’ means the agent navigated to an incorrect target, while ‘No Frontier’ means the target was not found after exploring all frontiers. ‘Stuck’ means the agent navigated to a fixed waypoint and exhausted its steps in a small space. ‘False Negative’ covers other cases where the agent exhausts its steps without finding the target.

der images using the estimated Gaussians for visualization. The rendered results preserve the semantic structure of the underlying instances, demonstrating that our estimator remains effective under real-world sensing conditions. We also observe that depth sensors can occasionally be inaccurate, which does introduce some disturbances to the estimation process.

Fig. 1 illustrates the incremental construction of GLMap using real RGB-D inputs, where the navigation target is a bed. The visualizations show that both instance-level and region-level semantics are accurately recorded during mapping. Guided by GLMap, the agent is able to efficiently infer the target location even the position of target is unknown. A video demonstration corresponding to Fig. 1 is also provided in the supplementary material.

4. Failure Cases

We analyze the failure cases of our method in the HM3D [1] and categorize them into three primary types: Infeasible, Active Stop, and Stepout, accounting for 49.60%, 33.11%, and 17.29% of total failures, respectively.

The first type, Infeasible, refers to scenarios where the target does not exist on the agent’s starting floor, which constitutes the largest proportion of total failures. The second type, Active Stop, occurs when the agent autonomously terminates navigation before exhausting the step limit, with the majority attributed to ‘False Positive’, where the agent incorrectly detects a non-target object as the target. The third

Please describe the objects in the image and the region they form. For objects, give the category and a short description. For regions, organize them around functional cooperation and interaction between objects. The output format should be a JSON str, as shown in the example below:

```
{
  "scene_summary": "The image shows a white flat-panel TV with a black base placed on a brown rectangular wooden side table, and a wooden bed",
  "objects": [
    {
      "instance_id": 1,
      "category": "side table",
      "description": "a brown rectangular wooden side table with a smooth surface, about 60cm in length."
    },
    {
      "instance_id": 2,
      "category": "TV",
      "description": "a white flat-panel TV with a black plastic base, screen size about 55 inches."
    },
    {
      "instance_id": 3,
      "category": "bed",
      "description": "a wooden bed with a black bedding and a white pillow."
    }
  ],
  "regions": [
    {
      "instances": [1, 2],
      "description": "A region consists of a side table and a TV, for viewing TV content and entertainment."
    }
  ]
}
```

The other rules are as follows:

1. Due to the limitations of subsequent object detectors, each object category shall appear only once.
2. Each region must consist of two or more objects. If there is no region that meets the criteria, the corresponding position shall be left as an empty list.
3. Please ignore any objects that are common in most rooms and do not reflect the specific characteristics of the scene, such as windows, doors, door frames, walls, floors, etc.
4. Do not split an integral object into parts. For example, do not split "bed" into excessively trivial phrases such as "bed frame" and "bedding"; place detailed information in the "description" field.
5. If only a litter part of something is shown in the image, please ignore it.

Figure 4. Prompt used for the MLLM.

type, Stepout, arises from two primary causes: on the one hand, path planning failures of the local policy (Stuck), and on the other hand, failure to locate the target due to missed object detections or excessively large scene scales. In particular, this category leaves substantial room for further improvement through targeted optimizations.

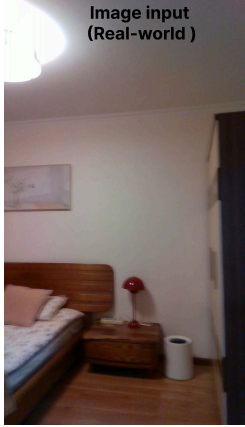
5. MLLM Prompts and Outputs

Since the construction of semantic units in GLMap relies heavily on the MLLM’s ability to interpret images, we present qualitative results to evaluate this component. We provide the prompt used for the MLLM, as shown in Fig. 4, and include one real-world example and two simulation examples as shown in Fig. 5. Each example contains the visual input given to the MLLM along with the model’s corresponding response. The results demonstrate that the MLLM is able to accurately understand both instance-level objects and region-level semantics. This level of semantic accuracy indicates that the MLLM can reliably support the construction of GLMap in practice.

References


- [1] Karmesh Yadav, Ram Ramrakhya, Santhosh Kumar Ramakrishnan, Theo Gervet, John Turner, Aaron Gokaslan, Noah Maestre, Angel Xuan Chang, Dhruv Batra, Manolis Savva, et al. Habitat-matterport 3d semantics dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4927–4936, 2023. [3](#)
- [2] Hang Yin, Xiuwei Xu, Zhenyu Wu, Jie Zhou, and Jiwen Lu. Sg-nav: Online 3d scene graph prompting for llm-based zero-shot object navigation. *Advances in Neural Information Processing Systems*, 37:5285–5307, 2024. [1](#)
- [3] Hang Yin, Xiuwei Xu, Linqing Zhao, Ziwei Wang, Jie Zhou, and Jiwen Lu. Unigoal: Towards universal zero-shot goal-oriented navigation. In *Proceedings of the Computer Vision*

Image input (Real-world)




```
{
  "scene_summary": "The image shows a bedroom with a bed, nightstand, lamp, and trash can.",
  "objects": [
    {
      "instance_id": 1,
      "category": "bed",
      "description": "a wooden bed with pillows on it."
    },
    {
      "instance_id": 2,
      "category": "nightstand",
      "description": "a small brown wooden table next to the bed."
    },
    {
      "instance_id": 3,
      "category": "lamp",
      "description": "a red desk lamp on the nightstand."
    },
    {
      "instance_id": 4,
      "category": "trash_can",
      "description": "a white cylindrical trash can near the wall."
    },
    {
      "instance_id": 5,
      "category": "picture_frame",
      "description": "a rectangular picture frame hanging on the wall."
    }
  ],
  "regions": [
    {
      "instances": [1, 2, 3],
      "description": "A group consists of the bed, nightstand, and lamp, forming a sleeping area setup."
    }
  ]
}
```

Image input (Simulation)



```
{
  "scene_summary": "The image shows a room with a bench, a coat rack, and a door.",
  "objects": [
    {
      "instance_id": 1,
      "category": "bench",
      "description": "A small wooden bench with a black cushioned seat."
    },
    {
      "instance_id": 2,
      "category": "coat rack",
      "description": "A wooden coat rack with multiple hooks, holding some garments."
    }
  ],
  "regions": [
    {
      "instances": [1, 2],
      "description": "A region consisting of a bench and a coat rack, providing seating and storage for coats or belongings."
    }
  ]
}
```

Image input (Simulation)



```
{
  "scene_summary": "The image shows a TV on a table, a chair, and a view of an outdoor garden.",
  "objects": [
    {
      "instance_id": 1,
      "category": "TV",
      "description": "A flat-screen TV mounted on a low console."
    },
    {
      "instance_id": 2,
      "category": "chair",
      "description": "A dark-colored armchair with a grey cushioned seat."
    },
    {
      "instance_id": 3,
      "category": "table",
      "description": "A low, rectangular table serving as a TV stand."
    }
  ],
  "regions": [
    {
      "instances": [1, 3],
      "description": "A region containing the TV and the table it rests on, forming a media center setup."
    },
    {
      "instances": [2, 1],
      "description": "A viewing area consisting of a chair facing the TV, suggesting a space for relaxation and entertainment."
    }
  ]
}
```

Figure 5. Images provided to the MLLM and the corresponding MLLM outputs (prompts follow the format in Fig. 4).

and Pattern Recognition Conference, pages 19057–19066, 2025. [1](#)

- [4] Naoki Yokoyama, Schoon Ha, Dhruv Batra, Jiuguang Wang, and Bernadette Bucher. Vlfm: Vision-language frontier maps for zero-shot semantic navigation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 42–48. IEEE, 2024. [2](#)