VLFM: Vision-Language Frontier Maps for Zero-Shot Semantic Navigation

Naoki Yokoyama[1,2], Sehoon Ha[2], Dhruv Batra[2], Jiuguang Wang[1], Bernadette Bucher[1]

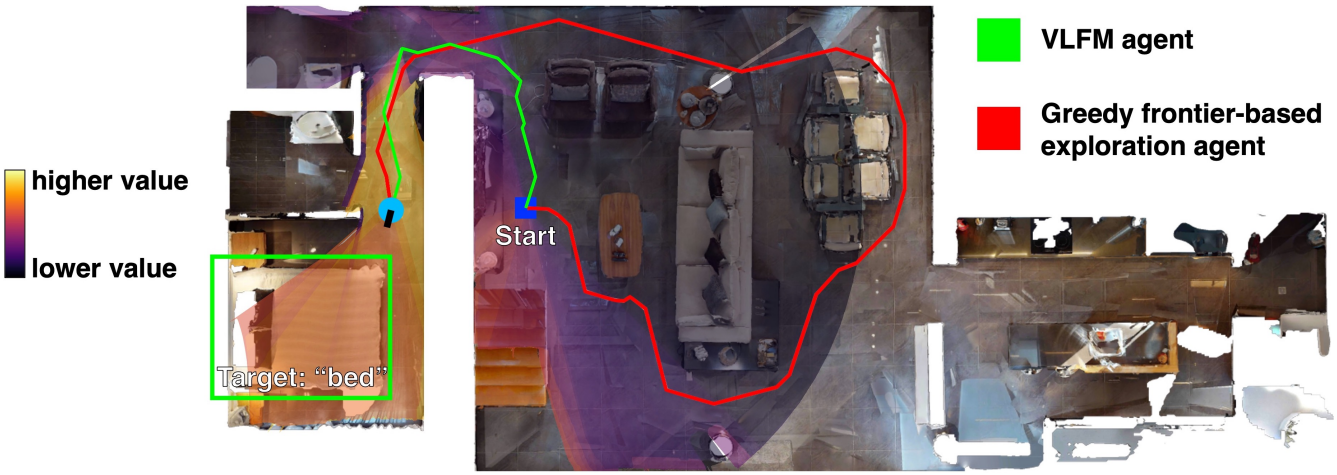

Fig. 1: VLFM achieves state-of-the-art semantic Object Goal Navigation performance in unfamiliar environments, without task-specific training, pre-built maps, or prior knowledge of the surroundings. It utilizes a vision-language model to explore the environment, capitalizing on visual semantic cues that are likely to guide the agent towards the goal to explore the environment more efficiently than a greedy frontier-based exploration agent.

***Abstract*— Understanding how humans leverage semantic knowledge to navigate unfamiliar environments and decide where to explore next is pivotal for developing robots capable of human-like search behaviors. We introduce a zero-shot navigation approach, Vision-Language Frontier Maps (VLFM), which is inspired by human reasoning and designed to navigate towards unseen semantic objects in novel environments. VLFM builds occupancy maps from depth observations to identify frontiers, and leverages RGB observations and a pre-trained vision-language model to generate a *language-grounded value map*. VLFM then uses this map to identify the most promising frontier to explore for finding an instance of a given target object category. We evaluate VLFM in photo-realistic environments from the Gibson, Habitat-Matterport 3D (HM3D), and Matterport 3D (MP3D) datasets within the Habitat simulator. Remarkably, VLFM achieves state-of-the-art results on all three datasets as measured by success weighted by path length (SPL) for the Object Goal Navigation task. Furthermore, we show that VLFM's zero-shot nature enables it to be readily deployed on real-world robots such as the Boston Dynamics Spot mobile manipulation platform. We deploy VLFM on Spot and demonstrate its capability to efficiently navigate to target objects within an office building in the real world, without any prior knowledge of the environment. The accomplishments of VLFM underscore the promising potential of vision-language models in advancing the field of semantic navigation. Videos of real world deployment can be viewed at naoki.io/vlfm.**

I. Introduction

How do humans navigate in novel environments? The process of human navigation in unfamiliar environments is complex, often relying on a combination of explicit maps and internal knowledge. This internal knowledge is typically an accumulation of semantic knowledge, which can be used to infer the layout of the space, including the locations of specific objects and geometric configurations. For instance, we know that toilets and showers are usually found together in bathrooms, often located near bedrooms. Natural language can further enhance this prior semantic knowledge, depending on the context.

In the development of robots capable of human-like navigation, learned foundation models that mimic this human reasoning process can be invaluable. Many methods, known as *zero-shot* methods, utilize these models to facilitate semantic navigation without any task-specific training or fine-tuning. Zero-shot methods are convenient because they can be easily adapted or repurposed for future robotic systems performing complex tasks, and they provide intermediate representations that improve interpretability. The remarkable performance of large language models (LLMs) and vision-language models (VLMs) has facilitated task-independent solutions for the semantic inference of out-of-view scene information [1]–[3].

In this work, we propose Vision-Language Frontier Maps (VLFM), a zero-shot approach for target-driven semantic navigation to an unseen object in a novel environment. VLFM builds occupancy maps from depth observations to identify frontiers of the explored map region. To find semantic target objects, VLFM prompts a pre-trained VLM to select which of these frontiers is most likely to lead to the semantic target. In contrast to prior language-based zero-shot semantic navigation methods [2]–[4], our method does not rely on object detectors and language models (*e.g.*, ChatGPT, BERT) to evaluate frontiers using text-only

semantic reasoning. Instead, VLFM uses a vision-language model to directly extract semantic values from RGB images in the form of a cosine similarity score with a text prompt involving the target object. VLFM uses these scores to generate a *language-grounded value map* that is used to identify the most promising frontier to explore. This spatially-grounded joint vision-language-based semantic reasoning increases computational inference speed and overall semantic navigation performance.

We demonstrate VLFM in photorealistic environments within the Habitat [5] simulator, where we achieve state-of-the-art results on the Object Goal Navigation (ObjectNav) task, even when compared to methods trained directly on the task. Specifically, we achieve absolute increases in success rates weighted by path length over prior state-of-the-art approaches of 12% on Gibson [6], 5% on Matterport 3D (MP3D) [7] and 3% on Habitat-Matterport 3D (HM3D) [8] datasets. We also demonstrate our approach in the real world on a Boston Dynamics Spot mobile manipulation platform by navigating efficiently to unseen semantic targets across a novel office building floor, without access to a pre-built map.

II. Related Works

ObjectNav. Object Goal Navigation (ObjectNav) involves executing semantic target-driven navigation in a novel environment, where performance is primarily measured by the efficiency of the robot's path to an instance of a given target object category. This is based on the premise that effective use of semantic priors should enable a robot to locate objects more efficiently [10]. Learning approaches to train robots with semantic navigation abilities have typically leveraged reinforcement learning [11]–[14], learning from demonstration [15], or prediction of semantic top-down maps [3], [16]–[20] on which waypoint planners can be used.

However, these task-specific trained approaches only work with the closed-set of object categories that they were trained on, and are often trained exclusively on simulated data, which can impede deployment of these policies onto real-world platforms. In contrast, our work proposes a zero-shot method that can take in an open-set of object categories, uses models that were trained on large amounts of real-world data, and demonstrates successful semantic navigation in the real world.

Zero-shot ObjectNav. Recent works in zero-shot methods for ObjectNav involve adapting the frontier-based exploration method proposed by [21]. Frontier-based exploration involves visiting the boundaries between explored and unexplored areas on a map that is iteratively built by the agent as it explores. Many methods for choosing the next frontier to explore have been proposed, such as classical methods that select frontiers based on the expected amount of information the agent would gain [22], [23]. CLIP on Wheels (CoW) [1] adopts a straightforward approach in which the robot explores the closest frontier until the target object is detected using either CLIP [24] features or an open-vocabulary object detector. LGX [4] and ESC [2] use a large language model (LLM) that processes object detections presented in the form of text to identify which frontiers would most likely harbor an instance of the target object. Instead of an LLM, SemUtil [3] uses BERT [25] to embed the class labels of objects detected near frontiers, and then compare them to the text embedding of the target object to select the frontier to explore next. However, these methods introduce a bottleneck in which visual cues from the environment must be converted into text by an object detector before they can be used to semantically evaluate frontiers. Additionally, reliance on an LLM requires a large amount of compute that may require a remote server the robot must connect to. In contrast, VLFM uses a vision-language model that can be easily loaded onto a consumer laptop to generate semantic value scores directly from RGB observations and text prompts, without generating any text from visual observations.

III. Problem Formulation

We address the task of ObjectNav [10], where a robot is tasked with searching for an instance of a target object category (*e.g.*, 'bed') in a previously unseen environment. This semantic navigation task encourages the robot to understand and navigate the environment based on high-level semantic concepts, such as the type of room it's in or the types of objects it sees, rather than relying solely on geometric cues. The robot only has access to an egocentric RGB-D camera and an odometry sensor that provides its current forward and horizontal distance and heading relative to its starting pose. The action space consists of the following: MOVE_FORWARD ($0.25m$), TURN_LEFT ($30°$), TURN_RIGHT ($30°$), LOOK_UP ($30°$), LOOK_DOWN ($30°$), and STOP. An episode is defined as successfully completed if STOP is called within 1 m of any instance of the target object in 500 or fewer steps.

IV. Vision-Language Frontier Maps

As depicted in Fig. 2, our approach is divided into three phases: initialization, exploration, and goal navigation. In the *initialization* phase, the robot rotates in place for a complete turn to set up its frontier and value maps, which are crucial for the subsequent exploration phase. During *exploration*, the robot persistently updates the frontier and value maps to create frontier waypoints and select the most valuable one for locating the specified target object category and navigating to it. Once it detects a target object instance, it transitions to the goal navigation phase. In the *goal navigation* phase, the robot simply navigates to the nearest point on the detected target object and triggers STOP once it is within sufficient proximity.

A. Frontier waypoint generation

We utilize depth and odometry observations to build a top-down 2D map of obstacles that the robot has encountered. The explored area within this map is updated based on the robot's location, its current heading, and any obstacles that obstruct parts of its current view from being explored. To identify obstacle locations, we transform the current depth image into a point cloud, filter out any points that are either too short or too tall to be considered an obstacle, transform

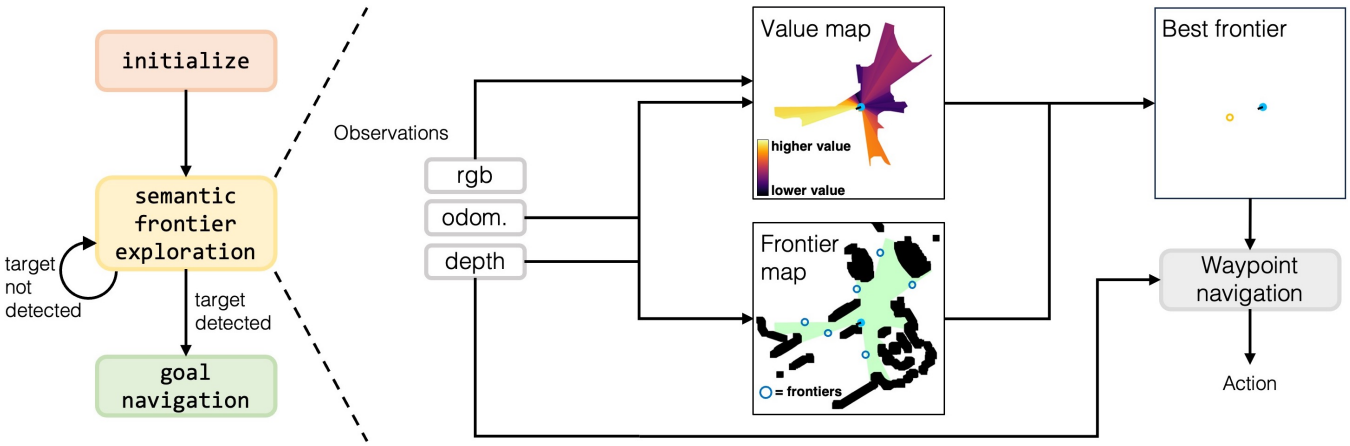

Fig. 2: VLFM constructs an occupancy map of the scene identifying frontiers of explored space as well as a value map of the likelihood of each region to lead toward the out-of-view target object. In a navigation episode, the robot first spins in a circle to initialize these maps and then begins executing frontier-based exploration by selecting waypoints from the current map frontier using the value map. Navigation to each waypoint is executed with a PointNav policy trained with Variable Experience Rollout (VER) [9]. The policy is also used to navigate to the target object once it is detected ('goal navigation').

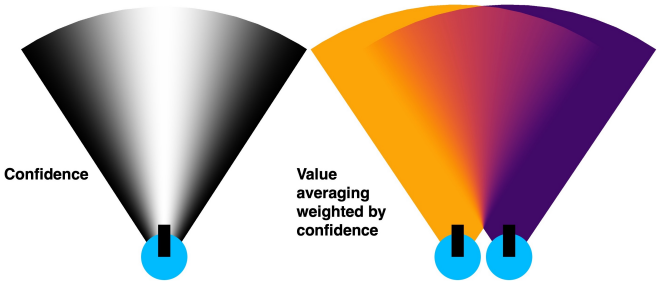

Fig. 3: *Left:* Visualization of how the confidence score of a pixel within the robot's FOV is determined based on its location relative to the optical axis. *Right:* The confidence scores are used when the robot's current FOV overlaps with the previously seen area; the new semantic values within this region become an average of the previous and current values, weighted by their confidence scores.

the points to the global frame, and then project them onto a 2D grid. We then identify each boundary separating the explored and unexplored areas, identifying its midpoint as a potential frontier waypoint. As the robot explores the area, the quantity and locations of frontiers will vary until the entire environment has been explored and no more frontiers remain. If the robot has not detected a target object at this point, it will simply trigger STOP to end the episode (unsuccessfully).

B. Value map generation

At the core of our approach is a value map, a 2D grid similar to the frontier map. This map assigns a value to each pixel within the explored area, quantifying its semantic relevance in locating the target object. The value map is used to evaluate each frontier, and the frontier with the highest value is chosen as the next one to explore. Similar to the frontier map, the value map uses depth and odometry observations to build a top-down map iteratively. However, the value map differs in that it has two channels representing semantic value scores and confidence scores.

Similar to how humans derive semantic cues directly from visual observations (*e.g.*, lighting, room type, room size, navigability to other rooms), rather than attempting to first represent what is currently visible to the robot with text (*e.g.*, using detected object bounding boxes like [2]–[4]), we use a pre-trained BLIP-2 [26] vision-language model to compute a cosine similarity score directly from the robot's current RGB observation and a text prompt containing the target object. BLIP-2 adapts CLIP [24] to achieve state-of-the-art results for image-to-text retrieval, which relies on accurately measuring how well a text prompt is represented by a given image. When used for image-to-text retrieval, BLIP-2 outputs a cosine score given an input RGB image and text prompt, where higher values indicate higher accuracy. We use a text prompt to measure how valuable the area represented by the current RGB image is for finding the target object ("`Seems like there is a <target_object> ahead.`"). These scores are then projected onto their own channel of the top-down value map.

The confidence channel aims to determine how a pixel's value in the semantic value channel should be updated if it has a value assigned from a previous time step and is within the robot's field-of-view (FOV) at the current time step. It does not affect a pixel's semantic value score if that pixel was not seen until the current time step. The confidence score of a pixel within the robot's FOV depends on its location relative to the optical axis. Pixels along the optical axis have a full confidence of 1, while those at the left and right edges have a confidence of 0. Specifically, we set the confidence of a pixel as $cos^2(\theta/(\theta_{fov}/2) * \pi/2)$, where θ is the angle between the pixel and the optical axis, and θ_{fov} is the horizontal FOV of the robot's camera.

When the robot moves to a new position where its FOV overlaps with a previously seen region, the semantic value and confidence scores for each pixel in that region are both updated with new scores. Each of these pixels' new semantic value score, $v_{i,j}^{new}$, is computed by averaging its current and previous value scores, weighted by $c_{i,j}^{curr}$ and $c_{i,j}^{prev}$, its current and previous confidence scores: $v_{i,j}^{new} =$

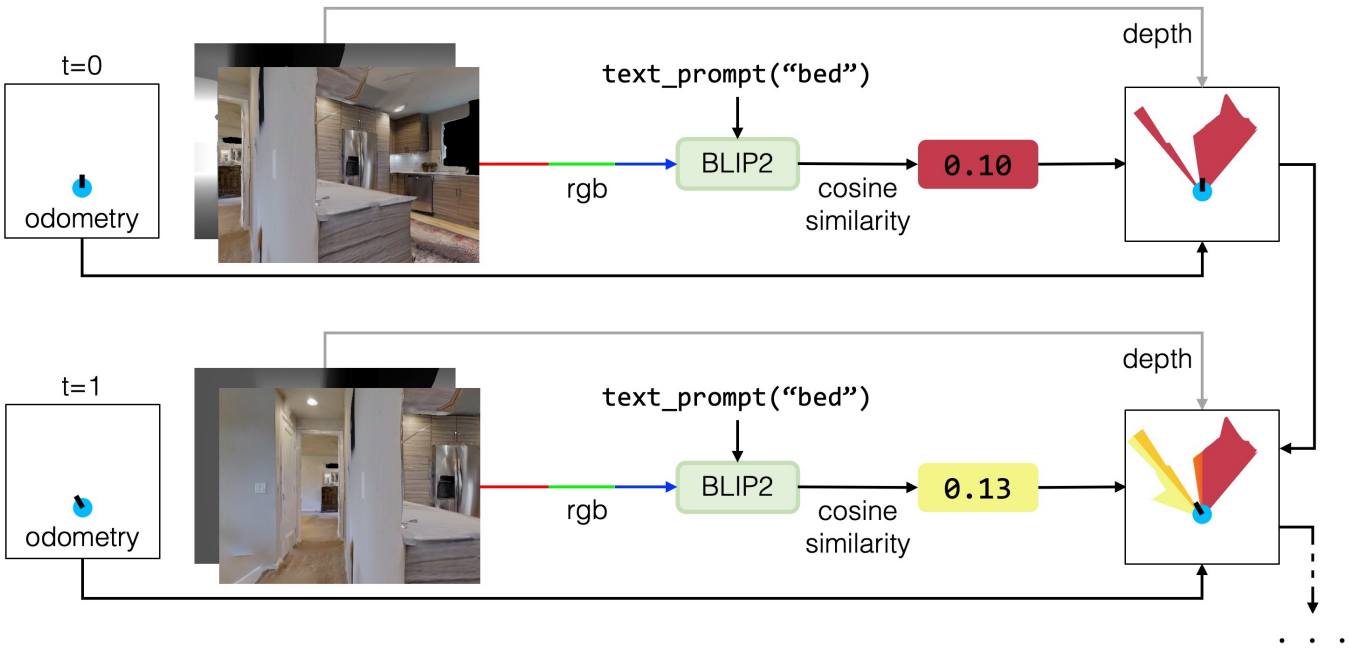

Fig. 4: VLFM iteratively constructs value maps for target-driven navigation by using BLIP-2 to compute the cosine similarity between a text prompt incorporating the target object and an RGB image taken from the robot's current pose. These semantic value scores are projected onto a top-down 2D pixel grid in the shape of the camera's FOV and exclude regions occluded by obstacles captured in the depth image.

$(c_{i,j}^{curr} v_{i,j}^{curr} + c_{i,j}^{prev} v_{i,j}^{prev})/(c_{i,j}^{curr} + c_{i,j}^{prev})$. Its new confidence score is also updated using a weighted average that is biased towards the higher confidence value: $c_{i,j}^{new} = ((c_{i,j}^{curr})^2 + (c_{i,j}^{prev})^2)/(c_{i,j}^{curr} + c_{i,j}^{prev})$. The confidence channel and its role in updating previously seen values is visualized in Fig. 3.

The map updating procedure is summarized as follows: 1) a cone-shaped mask depicting the camera FOV in a top-down manner at the camera's current pose is created, where pixels closer to the optical axis of the camera have a higher confidence score; 2) using the depth image, the mask is updated to exclude areas of the FOV obstructed by obstacles; 3) using BLIP-2, cosine similarity scores are computed between the current RGB image and the text prompt to update the semantic value channel within the masked portion of the value map; 4) previous semantic value and confidence scores within the masked portion of the value map are updated using weighted averaging using the confidence scores. The full process is depicted in Fig. 4.

C. Object detection

To determine whether or not a target object instance is currently visible to the robot, we use pre-trained object detectors that infer bounding boxes with semantic labels. Specifically, we utilize YOLOv7 [27] for target objects that fall within the COCO [28] classes, and Grounding-DINO [29] for all other categories. Grounding-DINO is an open-vocabulary object detector capable of detecting arbitrary objects based on language inputs. We do this because we find that YOLOv7 is better at detecting the objects within the COCO categories. If a target object instance is detected, we use Mobile-SAM [30] to extract its contour using the RGB image and the detected bounding box. The contour is then used with the depth image to determine the point on the object that is closest to the robot's current position, which is then used as the goal waypoint to navigate to. Once the robot's distance to this point falls below the success radius, STOP is called.

D. Waypoint navigation

After initialization, the robot is always provided with either a frontier waypoint or a target object waypoint to navigate towards, depending on whether a target object has been detected yet. To determine the action at each step for reaching the current waypoint, we employ a Point Goal Navigation (PointNav) [31] policy. To determine the action at each step for reaching the current waypoint, we use Variable Experience Rollout (VER) [9], a distributed deep reinforcement learning algorithm, to train a PointNav [31] policy. PointNav is a task that challenges the robot to navigate to a designated waypoint (2D coordinate) solely relying on visual observations and odometry. We trained our PointNav policy using scenes from the training split of the HM3D dataset [8], using the same hyperparameters as those described in [9], with 4 GPUs (64 workers each), and train the policy for 2.5 billion steps (around 7 days). Unlike ObjectNav, PointNav does not necessitate a semantic understanding of the environment for efficient and successful completion and can be accomplished using only geometric understandings. Our PointNav policy exclusively uses the egocentric depth image and the robot's relative distance and heading towards the desired goal point as input (see Fig. 2). It does not use RGB images.

It is entirely feasible to replace this PointNav policy with any alternative method capable of guiding the robot to a visually observed waypoint, be it a frontier or a detected

target object. For example, we do not use a PointNav policy for our real-world demonstrations. Our preference for the PointNav policy stems from its speed and ease-of-use; it alleviates several concerns associated with traditional mapping-based approaches, particularly when the waypoint resides outside the navigable area (*e.g.*, when the waypoint is on a target object, which itself may also be on a different obstacle), as navigability of the goal does not affect the policy or its observations.

V. Experimental Setup

Datasets. We evaluate our approach using the Habitat [5] simulator on the validation splits of three different datasets of 3D scans of real-world environments; Gibson [6], HM3D [8], and MP3D [7]. We use the ObjectNav validation split for Gibson developed in SemExp [16] which contains 1000 episodes across 5 scenes. HM3D's validation split contains 2000 episodes across 20 scenes and 6 object categories. MP3D's validation split contains 2195 episodes across 11 scenes and 21 object categories.

Metrics. For all approaches, we report success rate (SR) and Success weighted by inverse Path Length (SPL) [31]. SPL scores the efficiency of an agent's path by comparing it to the length of the shortest path from the start position to the closest instance of the target object category. It is zero if the agent did not succeed; otherwise, it is the shortest path length divided by the agent's path length (larger is better).

Baselines. We evaluate VLFM by comparing it to several state-of-the-art (SOTA) techniques for zero-shot object navigation: CLIP on Wheels (CoW) [1], ESC [2], SemUtil [3], and ZSON [32]. ZSON is an open vocabulary method that uses CLIP to transfer a method for a different navigation task (ImageNav) zero-shot to the ObjectNav task. It is trained on ImageNav in which images are used as goals instead of object names; at test-time, text-embeddings of object category names are used as goals instead. CoW explores the closest frontier until the target object is detected using either CLIP features or an open-vocabulary object detector, and then navigates directly to the detected target. Similarly to our approach, ESC and SemUtil both perform semantic frontier-based exploration, but frontiers are evaluated using nearby object detections that are converted to text and evaluated using a text-only model (*e.g.*, an LLM, BERT) that also considers the target object category.

In addition to the above zero-shot SOTA methods, we also include a comparison with supervised methods: PONI [19], PIRLNav [15], RegQLearn [14], and SemExp [16]. SemExp and PONI both build maps during navigation and train task-specific policies to perform semantic inference to predict likely target locations to explore. PIRLNav is an end-to-end policy trained with behavioral cloning on 77*k* human demonstrations and fine-tuned with online deep reinforcement learning. RegQLearn is an end-to-end policy trained only with deep reinforcement learning.

VI. Results

In this section, we aim to address the following questions:

TABLE I: Zero-shot Object Navigation results on Gibson [6], HM3D [8], and MP3D [7] benchmarks. Our method outperforms previous zero-shot methods and performs competitively against methods directly trained on the Object Navigation task.

Approach	Semantic Nav Training	Gibson		HM3D		MP3D	
		SPL↑	SR↑	SPL↑	SR↑	SPL↑	SR↑
PONI [19]	ObjectNav	41.0	73.6	-	-	12.1	31.8
PIRLNav [15]	ObjectNav	-	-	27.1	**64.1**	-	-
RegQLearn [14]	ObjectNav	31.3	63.7	-	-	-	-
SemExp [16]	ObjectNav	33.9	65.7	-	-	-	-
ZSON [32]	ImageNav	-	-	12.6	25.5	4.8	15.3
CoW [1]	None	-	-	-	-	3.7	7.4
ESC [2]	None	-	-	22.3	39.2	14.2	28.7
SemUtil [3]	None	40.5	69.3	-	-	-	-
VLFM (Ours)	None	**52.2**	**84.0**	**30.4**	52.5	**17.5**	**36.4**

1) How well can VLFM perform ObjectNav in various datasets in comparison to other trained or zero-shot methods?
2) How do different methods of fusing current and previously seen values affect the performance of VLFM?
3) Can VLFM be deployed successfully in the real world?

A. Benchmark results

The performance of VLFM in comparison to other methods on the Gibson, HM3D, and MP3D datasets is summarized in Table I. Unfilled cells indicate that the cited work did not evaluate their approach on the corresponding dataset (SemExp only evaluated on a subset of MP3D episodes using COCO classes). VLFM significantly outperforms all zero-shot methods across all benchmarks, with an increase of +11.7% SPL and +14.7% success in Gibson compared to SemUtil; +8.1% SPL and +13.3% success in HM3D compared to ESC; and +3.3% SPL and +7.7% success in MP3D compared to ESC.

In the Gibson and MP3D datasets, VLFM even surpasses methods that were trained directly within those datasets for ObjectNav, achieving +19.2% SPL and +19.0% success in Gibson compared to SemExp, and +5.4% SPL and +4.6% success in MP3D compared to PONI. This establishes new state-of-the-art metrics for these datasets.

In the HM3D dataset, VLFM only falls short of PIRLNav in terms of success (-11.6% success, but +3.3% SPL), which was trained on 77*k* human demonstrations collected within the train split of the HM3D dataset, while our method is entirely zero-shot.

Our superior performance within the Gibson dataset can be attributed to the lack of episodes that require the robot to navigate stairs in order to reach a target object, a scenario present in both HM3D and MP3D. The only non-zero-shot approach to outperform VLFM in success rate, PIRLNav, can traverse up or down stairs to find objects. However, VLFM currently only supports single-floor episodes due to the lack of a z coordinate in the odometry observation provided by the simulator, which complicates the resetting of top-down frontier and value maps upon changing floors. Consequently,

TABLE II: Comparisons of performance with different value update methods used with VLFM.

Value update method	Gibson		HM3D		MP3D	
	SPL↑	SR↑	SPL↑	SR↑	SPL↑	SR↑
Replacement	48.0	76.1	26.5	44.5	16.6	31.7
Unweighted avg.	50.9	83.0	30.0	51.8	17.1	35.0
Weighted avg.	**52.2**	**84.0**	**30.4**	**52.5**	**17.5**	**36.4**

we fail 14.6% and 9.6% of the HM3D and MP3D episodes (respectively) because they require traversing stairs to encounter the target object.

We also attribute the higher performance on the Gibson and HM3D datasets compared to the MP3D dataset to the quality of the 3D scans. The MP3D dataset has significantly lower visual fidelity [8] than the HM3D dataset, while the scenes from the Gibson dataset were manually repaired and verified to be free of holes and artifacts [6].

B. Ablations

When an area that was already seen previously is encountered again, its representation in the value map used by VLFM is updated using a combination of its previous and current values. We explore different methods of combining these values and their impact on the performance of VLFM in Table II. In the *Replacement* method, the previous value is disregarded and the new value simply overwrites it. The *Unweighted avg.* method calculates the new value as the average of the previous and current values. Our full approach, *Weighted avg.*, uses confidence scores to weight the average between the previous and new values, taking into account the parts of the FOV the area occupied when it was previously observed and at the current timestep. Our findings indicate that the *Weighted avg.* method consistently enhances performance compared to the other two methods across all three datasets.

C. Real-world deployment

We deploy VLFM on a Spot robot from Boston Dynamics (BD) to demonstrate VLFM successfully navigating to objects in the real world. For waypoint navigation, we utilize the BD API instead of a PointNav policy, which can guide the robot towards a specified waypoint, provided the path is relatively free of obstacles. The camera within the robot's gripper is used to detect objects and feed inputs to BLIP-2. However, due to its limited range in depth sensing, we employ the ZoeDepth depth estimation model [33] to generate a depth image and approximate a waypoint for the detected target object. We rely on the depth cameras on the body of Spot to detect obstacles for frontier generation. All models used, including BLIP-2, GroundingDINO, MobileSAM, and ZoeDepth, were loaded and executed in real-time on a laptop equipped with an RTX 4090 MaxQ Mobile GPU with 16 GB of VRAM. Videos can be viewed at naoki.io/vlfm.

VII. Conclusion

This paper presents VLFM, a zero-shot framework for ObjectNav in novel environments. Our key innovation is spatially grounding joint vision-language-based semantic reasoning with pre-trained models in a new approach to frontier waypoint selection in order to perform target-driven navigation in novel environments. VLFM uses a semantic prompt-based method to infer which frontier to navigate to using a pre-trained vision-language model, detects visible target objects with a pre-trained object detector, and navigates to these frontiers and target objects using a pre-trained policy. This modular nature of our method allows different components to be swapped in as improved models become available. Experiments in simulated 3D home environments demonstrate that VLFM achieves state-of-the-art zero-shot navigation performance on the Object Goal Navigation benchmark. Demonstrations in an office building on a Spot Arm platform prove the viability of our method in real-world scenarios.

VLFM has a number of limitations that could be addressed by future work. First, we assume target objects will be easily visible in the scene from the default height of the robot camera. Future work could investigate policies to increase interaction with the environment during the search process, including actively directing the robot camera to look in promising locations and using manipulation to execute search actions such as looking inside closed drawers. Furthermore, we build a map of semantic information and occupancy information with our frontier and value map, but the value map, which contains semantic information, provides only task-specific semantic information. So, we cannot leverage this map in sequentially executed semantic navigation tasks to different objects or in executing other navigation tasks requiring targets specified by language, such as vision-language navigation. Future work could explore alternate prompt formulations, value map designs, and methods of tracking semantic information during task execution to enable long-horizon planning and multitask execution.

Overall, our results highlight the promise of leveraging foundation models like BLIP-2 in a zero-shot manner within robotic systems to provide spatially grounded joint vision-language-based semantic reasoning without task-specific training.

References

[1] S. Y. Gadre, M. Wortsman, G. Ilharco, L. Schmidt, and S. Song, "CoWs on Pasture: Baselines and Benchmarks for Language-Driven Zero-Shot Object Navigation," *CVPR*, 2023.

[2] K. Zhou, K. Zheng, C. Pryor, Y. Shen, H. Jin, L. Getoor, and X. E. Wang, "ESC: Exploration with Soft Commonsense Constraints for Zero-shot Object Navigation," *arXiv preprint arXiv:2301.13166*, 2023.

[3] J. Chen, G. Li, S. Kumar, B. Ghanem, and F. Yu, "How To Not Train Your Dragon: Training-free Embodied Object Navigation with Semantic Frontiers," in *RSS*, 2023.

[4] V. S. Dorbala, J. F. Mullen Jr, and D. Manocha, "Can an Embodied Agent Find Your "Cat-shaped Mug"? LLM-Based Zero-Shot Object Navigation," *arXiv preprint arXiv:2303.03480*, 2023.

[5] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, D. Parikh, and D. Batra, "Habitat: A Platform for Embodied AI Research," in *ICCV*, 2019.

[6] F. Xia, A. R. Zamir, Z.-Y. He, A. Sax, J. Malik, and S. Savarese, "Gibson Env: Real-World Perception for Embodied Agents," in *CVPR*, 2018.

[7] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niessner, M. Savva, S. Song, A. Zeng, and Y. Zhang, "Matterport3D: Learning from RGB-D Data in Indoor Environments," *International Conference on 3D Vision (3DV)*, 2017.

[8] S. K. Ramakrishnan, A. Gokaslan, E. Wijmans, O. Maksymets, A. Clegg, J. M. Turner, E. Undersander, W. Galuba, A. Westbury, A. X. Chang, M. Savva, Y. Zhao, and D. Batra, "Habitat-Matterport 3D Dataset (HM3D): 1000 Large-scale 3D Environments for Embodied AI," in *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021. [Online]. Available: https://arxiv.org/abs/2109.08238

[9] E. Wijmans, I. Essa, and D. Batra, "VER: Scaling On-Policy RL Leads to the Emergence of Navigation in Embodied Rearrangement," in *NeurIPS*, vol. 35, 2022, pp. 7727–7740.

[10] D. Batra, A. Gokaslan, A. Kembhavi, O. Maksymets, R. Mottaghi, M. Savva, A. Toshev, and E. Wijmans, "ObjectNav Revisited: On Evaluation of Embodied Agents Navigating to Objects," *arXiv preprint arXiv:2006.13171*, 2020.

[11] J. Ye, D. Batra, A. Das, and E. Wijmans, "Auxiliary Tasks and Exploration Enable ObjectNav," *arXiv preprint arXiv:2104.04112*, 2021.

[12] M. Chang, A. Gupta, and S. Gupta, "Semantic Visual Navigation by Watching YouTube Videos," in *NeurIPS*, 2020.

[13] M. Deitke, E. VanderBilt, A. Herrasti, L. Weihs, K. Ehsani, J. Salvador, W. Han, E. Kolve, A. Kembhavi, and R. Mottaghi, "ProcTHOR: Large-Scale Embodied AI Using Procedural Generation," *NeurIPS*, vol. 35, pp. 5982–5994, 2022.

[14] N. Gireesh, D. A. S. Kiran, S. Banerjee, M. Sridharan, B. Bhowmick, and M. Krishna, "Object Goal Navigation using Data Regularized Q-Learning," in *IEEE CASE*, 2022.

[15] R. Ramrakhya, D. Batra, E. Wijmans, and A. Das, "PIRLNav: Pretraining with Imitation and RL Finetuning for ObjectNav," in *CVPR*, 2023.

[16] D. S. Chaplot, D. Gandhi, A. Gupta, and R. Salakhutdinov, "Object Goal Navigation using Goal-Oriented Semantic Exploration," in *NeurIPS*, 2020.

[17] S. Zhang, X. Song, Y. Bai, W. Li, Y. Chu, and S. Jiang, "Hierarchical Object-to-Zone Graph for Object Navigation," *ICCV*, 2021.

[18] H. Luo, A. Yue, Z.-W. Hong, and P. Agrawal, "Stubborn: A Strong Baseline for Indoor Object Navigation," in *IROS*, 2022, pp. 3287–3293.

[19] S. K. Ramakrishnan, D. S. Chaplot, Z. Al-Halah, J. Malik, and K. Grauman, "PONI: Potential Functions for ObjectGoal Navigation with Interaction-free Learning," in *CVPR*, 2022.

[20] J. Zhang, L. Dai, F. Meng, Q. Fan, X. Chen, K. Xu, and H. Wang, "3D-Aware Object Goal Navigation via Simultaneous Exploration and Identification," in *CVPR*, 2023.

[21] B. Yamauchi, "A Frontier-Based Approach for Autonomous Exploration," in *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97.'Towards New Computational Principles for Robotics and Automation'*. IEEE, 1997, pp. 146–151.

[22] A. Mobarhani, S. Nazari, A. H. Tamjidi, and H. D. Taghirad, "Histogram Based Frontier Exploration," in *IROS*, 2011, pp. 1128–1133.

[23] Y. Li, A. Debnath, G. J. Stein, and J. Kosecka, "Learning-Augmented Model-Based Planning for Visual Exploration," in *IROS*, 2023.

[24] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning Transferable Visual Models from Natural Language Supervision," in *ICML*, 2021, pp. 8748–8763.

[25] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 4171–4186.

[26] J. Li, D. Li, S. Savarese, and S. Hoi, "BLIP-2: Bootstrapping Language-Image Pre-Training with Frozen Image Encoders and Large Language Models," *arXiv preprint arXiv:2301.12597*, 2023.

[27] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "YOLOv7: Trainable Bag-of-Freebies Sets New State-Of-The-Art for Real-Time Object Detectors," in *CVPR*, 2023.

[28] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common Objects in Context," in *ECCV*, 2014, pp. 740–755.

[29] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, C. Li, J. Yang, H. Su, J. Zhu, *et al.*, "Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection," *arXiv preprint arXiv:2303.05499*, 2023.

[30] C. Zhang, D. Han, Y. Qiao, J. U. Kim, S.-H. Bae, S. Lee, and C. S. Hong, "Faster Segment Anything: Towards Lightweight SAM for Mobile Applications," *arXiv preprint arXiv:2306.14289*, 2023.

[31] P. Anderson, A. X. Chang, D. S. Chaplot, A. Dosovitskiy, S. Gupta, V. Koltun, J. Kosecka, J. Malik, R. Mottaghi, M. Savva, and A. R. Zamir, "On Evaluation of Embodied Navigation Agents," *arXiv preprint arXiv:1807.06757*, 2018.

[32] A. Majumdar, G. Aggarwal, B. Devnani, J. Hoffman, and D. Batra, "ZSON: Zero-Shot Object-Goal Navigation using Multimodal Goal Embeddings," in *NeurIPS*, 2022.

[33] S. F. Bhat, R. Birkl, D. Wofk, P. Wonka, and M. Müller, "ZoeDepth: Zero-shot Transfer by Combining Relative and Metric Depth," *arXiv preprint arXiv:2302.12288*, 2023.