WoMAP: World Models For Embodied Open-Vocabulary Object Localization

Tenny Yin[1*] **Zhiting Mei**[1] **Tao Sun**[1,2]
Lihan Zha[1] **Jeremy Bao**[1†] **Miyu Yamane**[1†]
Emily Zhou[1†] **Ola Shorinwa**[1*] **Anirudha Majumdar**[1]
[1]Princeton University [2]McGill University
robot-womap.github.io

Abstract: Language-instructed active object localization is a critical challenge for robots, requiring efficient exploration of partially observable environments. However, state-of-the-art approaches either struggle to generalize beyond demonstration datasets (e.g., imitation learning methods) or fail to generate physically grounded actions (e.g., VLMs). To address these limitations, we introduce **WoMAP** (**Wo**rld **M**odels for **A**ctive **P**erception): a recipe for training open-vocabulary object localization policies that: (i) uses a Gaussian Splatting-based real-to-sim-to-real pipeline for scalable data generation without the need for expert demonstrations, (ii) distills dense rewards signals from open-vocabulary object detectors, and (iii) leverages a latent world model for dynamics and rewards prediction to ground high-level action proposals at inference time. Rigorous simulation and hardware experiments demonstrate WoMAP's superior performance in a broad range of zero-shot object localization tasks, with more than 9× and 2× higher success rates compared to VLM and diffusion policy baselines, respectively. Further, we show that WoMAP achieves strong generalization and sim-to-real transfer on a TidyBot.

Keywords: Active Perception, World Models, Object Localization

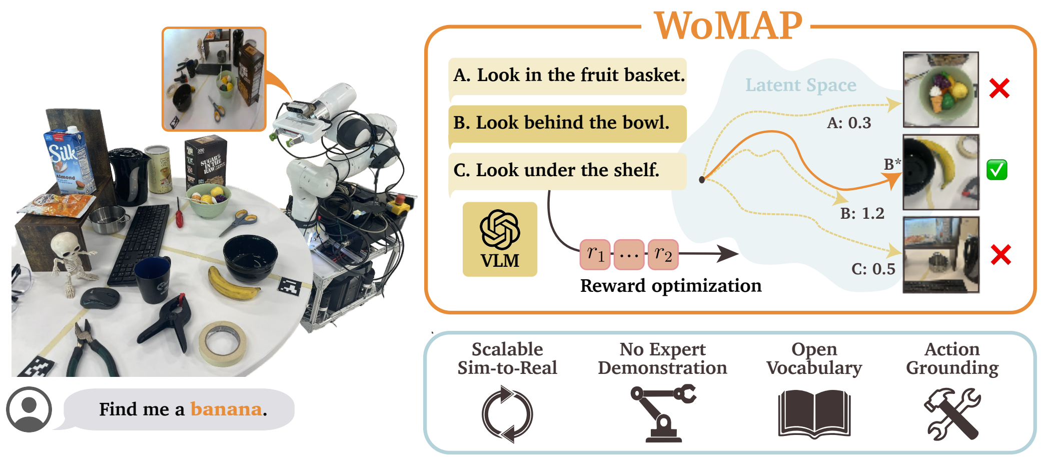

Figure 1: **WoMAP** uses a world model to ground high-level action proposals and maximize predicted rewards. In this example, given three high-level VLM proposals, WoMAP selects "look behind the bowl" as the optimal choice after evaluating outcomes of each action roll-out in latent space.

1 Introduction

Perceptual activity in biological agents is inherently active and exploratory [1, 2]. As an example, consider the task of *open-vocabulary object localization*, where an agent needs to approach a target

*Equal contribution. †Authors contributed equally.

object specified by natural language in a previously unseen environment. In such settings, humans will actively seek information guided by prior expectations to search efficiently. For example, when looking for keys, we preferentially inspect locations where they are most likely to be found, e.g., near the door or on the couch.

However, reproducing intelligent search behavior for robots remains challenging, as it requires semantic understanding of open-vocabulary object descriptions and commonsense reasoning from partial observations in unfamiliar environments. While vision-language models (VLMs) provide useful heuristics for exploration [3–6], effectively grounding these high-level action proposals to physical execution is a non-trivial problem. This grounding can be achieved via imitation learning methods [7, 8], which require large-scale expert demonstrations and can struggle to generalize beyond demonstration datasets. Alternatively, reinforcement learning (RL) [9–11] offers another route to grounding, but is challenging to employ without an accurate simulation environment.

To address these challenges, we present **WoMAP** (**Wo**rld **M**odels for **A**ctive **P**erception): a novel recipe for efficient active object localization that can be trained without expert demonstrations or online interactions with the environment (Figure 1). In order to achieve this, we propose an approach that learns a *latent world model* [12] using three key ingredients (Figure 2, left). First, we introduce a scalable data generation pipeline based on Gaussian Splatting [13] that allows us to generate photorealistic data with broad coverage using real-world videos. Second, we propose a training framework that is *reconstruction-free*; instead of using image reconstruction as a supervisory training signal (which can lead to poor generalization, training stability, and sample efficiency [14]), we construct dense rewards from the confidence outputs of open-vocabulary object detectors and *distill* these into the latent space of the world model. Finally, we present an inference-time planning scheme that optimizes high-level action proposals from VLMs using the trained world model.

Taken together, we contribute a novel approach to open-vocabulary object localization that can be trained in a data-efficient manner, generalize to novel scenes and object descriptions, and exploit commonsense reasoning abilities of VLMs. We demonstrate our approach on a suite of simulated and real-world object localization tasks with significant improvements ($2\times - 9\times$ higher success rates) over baselines that only utilize imitation learning or VLMs.

2 Related Work

Active Object Localization. Broadly, active object localization has been explored with both end-to-end approaches [15, 16], such as imitation learning (IL) [7, 8] and reinforcement learning (RL) [9–11], and modular approaches incorporating foundation models [3–5]. End-to-end methods map visual observations directly to actions but typically require large amounts of expert demonstrations or interactions to learn effective exploration behaviors [7, 15] and generalize poorly to new environments or tasks [17]. In contrast, WoMAP does not require on-policy demonstrations or interactions with the environment, and leverages a latent world model to plan sequences of actions at inference time in order to achieve strong generalization across tasks and sim-to-real gaps.

Modular approaches incorporate foundation models such as pre-trained object detectors or VLMs [4, 5, 18] to reason over observations and plan exploration. However, they rely heavily on the accuracy of each system component and require engineered spatial scene representations to ground executions in the real world. Instead, WoMAP directly optimizes VLM outputs within a learned environment model, offering a light-weight solution for grounded actions with minimal reliance on external representations. Finally, in terms of task setup, most prior work focuses on simulated indoor navigation, utilizing rich contextual cues (e.g., sofas are more likely to be in the living room) and restricting action spaces for tractability. In contrast, our framework makes no such assumptions and tackles more general settings requiring full 6-DoF camera control to locate objects in cluttered scenes.

World Models for Robotics. World models have become increasingly prominent in robotics, providing predictive foresight in learning action-conditioned dynamics for long-horizon planning [14, 19–21]. To capture environment dynamics that generalize to test-time distributions, existing

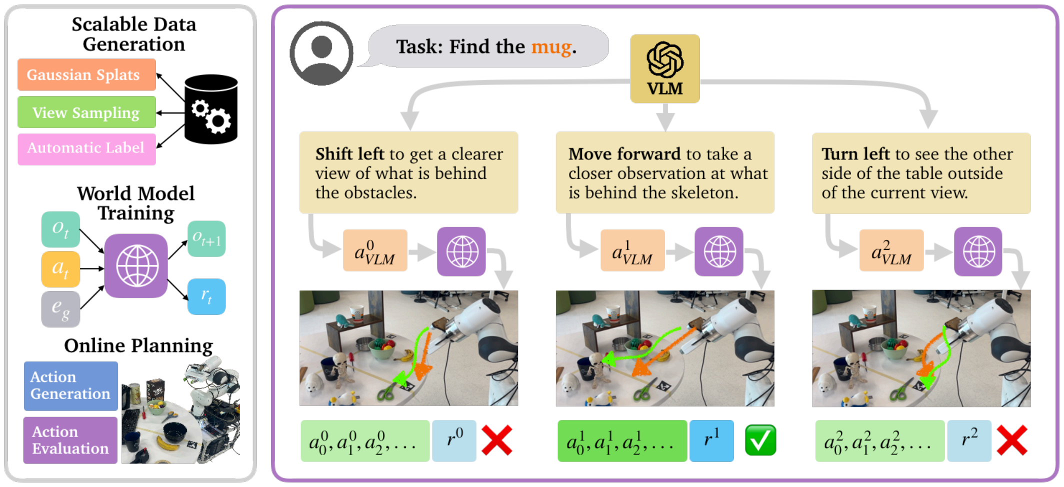
Scalable Data Generation
Gaussian Splats
View Sampling
Automatic Label
World Model Training
o_t
a_t
e_g
o_{t+1}
r_t
Online Planning
Action Generation
Action Evaluation
Task: Find the mug.
VLM
Shift left to get a clearer view of what is behind the obstacles.
Move forward to take a closer observation at what is behind the skeleton.
Turn left to see the other side of the table outside of the current view.
a^0_{VLM}
a^1_{VLM}
a^2_{VLM}
$a^0_0, a^0_1, a^0_2, \ldots$
r^0
$a^1_0, a^1_1, a^1_2, \ldots$
r^1
$a^2_0, a^2_1, a^2_2, \ldots$
r^2

Figure 2: **Left: Core components of WoMAP.** Scalable data generation with Gaussian Splats (Section 3.2), world modeling with object detection reward supervision (Section 3.3), and latent space action planning (Section 3.4). **Right: The action optimization process.** Given the task and current observation, a VLM generates high-level proposals, which we transform into coarse actions (green arrows); each action is further optimized within WoMAP's reward gradient field (red arrows), and the action sequence with the highest predicted reward is executed.

works typically rely on large quantities of uncurated data [14, 20], expert demonstrations [21, 22], or on-policy interactions [23], which are labor-intensive and costly to collect in real-world settings. In contrast, WoMAP learns a policy-agnostic environment model from synthetically generated offline data via Gaussian Splatting [13]. Many existing works also employ an image reconstruction loss to provide dense learning signals [20, 24], which often results in greater model complexity and unstable training. Without image reconstruction, WoMAP instead leverages dense reward signals distilled from pretrained models to encode rich visual, spatial, and semantic information in a pretrained latent representation, improving both scalability and robustness for downstream tasks.

3 Method

3.1 Problem Formulation

We consider an open-vocabulary object localization task with a robot equipped with an onboard RGB camera with a six degree-of-freedom (6-DoF) action space, operating in an environment $E \in \mathcal{E}$. We model the problem as a partially observable Markov decision process (POMDP), where given the current state of the environment and the robot, the camera returns a partial observation o_t under sensing uncertainty, occlusions, and limited field of view. At each time step, the robot selects a continuous action $a_t \in \mathbb{R}^6$, corresponding to camera translation and rotation in 3D space, to obtain a new observation. Given a language description l of the target object $\mathcal{T}_g$, the robot seeks to efficiently obtain an *optimal* view (i.e., a view that maximizes the object's visibility for localization) over a planning horizon T: $\max_{a_{0:T}} \mathcal{R}(o_T, \mathcal{T}_g)$, where $\mathcal{R} \in [0, 1]$ is the object localization reward describing how well the target object $\mathcal{T}_g$ is identified in o_T.

Our proposed framework, WoMAP, uses a world model to capture latent space dynamics and reward predictions that can generalize to any $E \in \mathcal{E}$. However, as discussed in Section 1, learning a world model that operates across such diverse task settings is non-trivial, requiring training data with sufficient coverage, strong supervisory reward signals, and the ability to incorporate high-level commonsense reasoning during planning. In the following sections, we describe the three core components of WoMAP as illustrated in Figure 2, addressing each of these fundamental challenges.

3.2 Scalable Data Generation

Unlike imitation learning methods, world models do not require expert trajectories for training, which are generally expensive to collect. However, they require sufficient data coverage to effectively capture the dynamics of the environment, which is challenging to scale as the number of training environments increases. Gathering diverse observation data from the real-world also poses significant challenges, which necessitates strategic data collection to maximize sample efficiency.

In WoMAP, we introduce a scalable real-to-sim-to-real data generation pipeline that utilizes only a few real-world videos to efficiently generate diverse training data. Our pipeline leverages Gaussian Splatting [13] to generate a photorealistic simulation environment from video input and can render arbitrary views at any camera pose. Shown in Figure 3, we automatically annotate the location and physical dimension of each target in the training scene by distilling language semantics from CLIP [27] into semantic Gaussian Splats [25]. Within the trained Gaussian Splat, we collect a training dataset $\mathcal{D}$ consisting of M observation-reward-pose tuples, i.e., $\mathcal{D} = \{(o_i, r_i, P_i),\ \forall i \in [M]\}$, where $P_i \in \mathbb{R}^6$ denotes the spatial camera pose, o_i is the associated rendered image, and r_i is the reward associated with o_i. At training time, given two random samples, we compute the action a_{ij} required to transition from P_i to P_j, since training the world model does not require sequentially-ordered data.

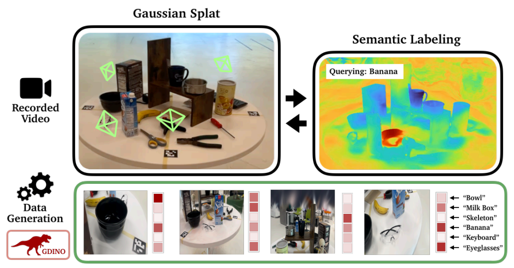

Figure 3: **Data Generation with Gaussian Splats.** We train Gaussian Splats for each scene and obtain ground truth object locations through semantic labeling [25] for informative view sampling. Each observation is labeled with GroundingDINO [26] to get confidence scores for all training targets.

Further, to improve sample efficiency, we design our data distribution to concentrate on trajectories starting from random initial positions and leading towards sampled target objects, with added linear and angular perturbations for data augmentation. These trajectories can be generated with any planning algorithm and require no human demonstration. We provide more implementation details in Appendix A.1.3. In Section 4.4, we demonstrate that despite only training on synthetically rendered images in Gaussian Splats, WoMAP still achieves strong zero-shot sim-to-real performance.

3.3 World Models for Active Perception

Given data generated from Sec. 3.2, we outline key design choices that enable the world model to accurately learn dynamics and rewards. A central innovation of our pipeline is the use of dense reward distillation from open-vocabulary object detectors, which allows for data-efficient training without relying on image reconstruction objectives.

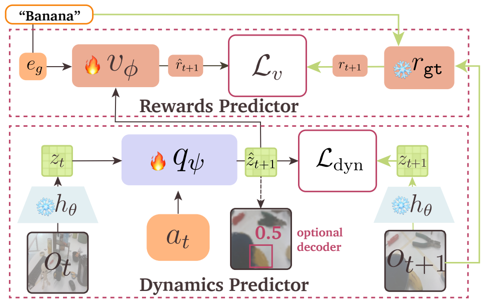

Figure 4: **World Model Architecture** for simultaneous dynamics and rewards prediction.

3.3.1 World Model Architecture

As shown in Figure 4, the world model consists of three standard core components [12, 28]:

$$\begin{aligned}
&\text{Observation Encoder:} \quad && z_t = h_\theta(o_t), \\
&\text{Dynamics Predictor:} \quad && z_{t+1} \sim q_\psi(z_{t+1} \mid z_t, a_t), \\
&\text{Rewards Predictor:} \quad && r_t \sim v_\phi(r_t \mid z_t, e_g),
\end{aligned} \tag{1}$$

where $z_t \in \mathbb{R}^d$ denotes the latent state, e_g represents the language embedding computed from a description l of the target object $\mathcal{T}_g$, $r_t \in \mathbb{R}$ denotes the associated reward with z_t when querying for $\mathcal{T}_g$, and θ, ψ, ϕ denote network parameters for each component of the world model.

The observation encoder h_θ maps a high-dimensional camera observation o_t to a compact latent space z_t. Inspired by [14], we leverage a pre-trained vision encoder model DINOv2 [29] to directly compute flattened patch embeddings of the given image as z_t to retain rich visual and spatial features resulting from large-scale pre-training. The dynamics predictor q_ψ models the transition distribution $p(z_{t+1} \mid z_t, a_t)$ using a standard ViT architecture [30]. With variational inference, we parametrize q_ψ as a Gaussian distribution $q_\psi(z_{t+1} \mid z_t, a_t) \sim \mathcal{N}(\mu_t, \sigma_t^2)$ and minimize the Kullback-Leibler (KL) divergence between the true state transition, $p(z_{t+1} \mid z_t, a_t)$ and q_ψ, with the loss function: $\mathcal{L}_{\text{dyn}} = \mathrm{KL}\big(p(z_{t+1} \mid z_t, a_t) \| q_\psi(z_{t+1} \mid z_t, a_t)\big)$. During training, we supervise q_ψ recurrently on a sequence of H observation-action pairs $\{(o_i, a_i)\}_{i=1}^{H}$ to enforce dynamics consistency. The rewards predictor v_ϕ estimates the reward for each latent state, conditioned on the language embedding of the task $\mathcal{T}_g$. Additional implementation details can be found in Appendix A.2.

3.3.2 Reward Distillation

Despite providing dense reward signals, image reconstruction objectives often lead to training instability [31, 32], which we further demonstrate in Appendix A.2.1. To tackle this challenge, WoMAP introduces a novel reward distillation procedure that generates dense rewards signals without reconstruction objectives for data-efficient training. As shown in Figure 3, during data generation, WoMAP computes a per-frame reward for each object in the observation using the detection confidence provided by a pretrained object detector, e.g., GroundingDINO [26], scaled by the associated detection bounding-box size. This annotation procedure yields a rich, task-relevant training signal for each object in the scene, and scales efficiently with environment complexity by enabling parallel processing of detections. At training time, we distill the ground-truth reward signal $r_{\texttt{gt}}(o_{t+1}, e_g) \in [0, 1]$ into the rewards predictor v_ϕ conditioned on the language embedding e of each relevant object. This distillation pipeline enables an effective planning framework using world models, which we discuss in the following subsection.

3.4 Planning with WoMAP

While world models can directly plan actions via sampling or gradient-based optimization, such methods are often inefficient in continuous action spaces without informed guidance. Particularly, in complex problem spaces, gradient-based methods struggle to localize target objects within a finite optimization budget and frequently converge to suboptimal solutions. WoMAP addresses this limitation by leveraging VLMs' commonsense reasoning to generate informed action proposals, which are then refined via model predictive control using a world model for spatial grounding.

For a given task instruction, we prompt a VLM using chain-of-thought prompting [33] to provide high-level guidance for promising locations for the robot to explore. In preliminary experiments, we observed that VLMs struggle with spatial understanding when prompted for numerical relative actions given an input image. Consequently, we query the VLM using a multiple-choice prompt with the options given by a fixed set of textual description of the possible actions, e.g., "turn left," or "move forward." We provide additional implementation details in Appendix A.3. Subsequently, WoMAP optimizes a set of candidate actions provided by the VLM to maximize the expected rewards:

$$\begin{aligned} \max_{a_{t:t+T}} \quad & \sum_{\tau=1}^{T} \big(\mathbb{E}_{v_\phi}[r_{t+\tau} \mid z_{t+\tau}, e_g)] + \gamma \left\| a_{t+\tau-1} - a_{t+\tau-2} \right\|_1\big), \\ \text{subject to} \quad & z_{t+\tau} \sim q_\psi(z_{t+\tau} \mid z_{t+\tau-1}, a_{t+\tau-1}) \;\; \forall \tau \in [T], \end{aligned} \tag{2}$$

at each timestep t with latent state z_t, target-object language embedding e_g, previous control action a_{t-1}, MPC planning horizon T, and weight $\gamma \in \mathbb{R}_+$. While the first objective term seeks to maximize the expected rewards, the second objective term incentivizes smoothness of the robot's trajectories.

Figure 2 illustrates the trajectory planning process on the right panel. The VLM provides three action proposals. WoMAP optimizes each of the three action proposals, estimates their rewards, and then executes the optimized action sequence with the highest reward.

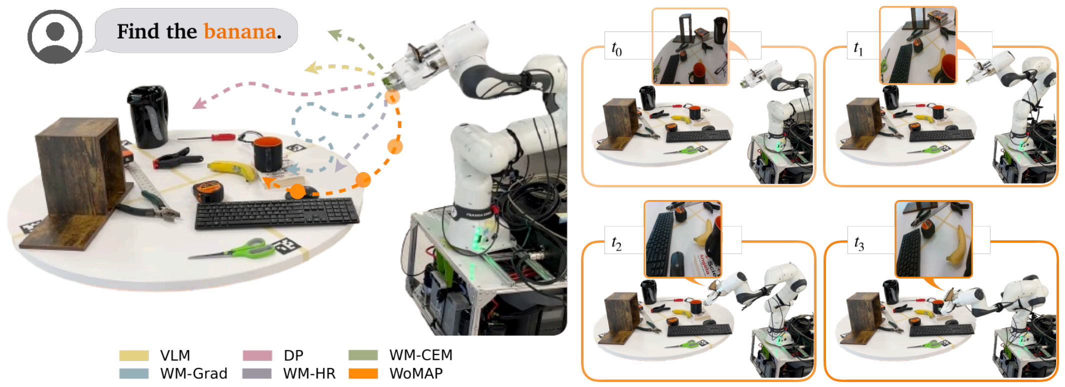

Figure 5: **Visualization of the TidyBot's trajectories for all planners.** When asked to find an object, e.g., a banana occluded by a mug, WoMAP finds the target object (banana) more efficiently than the other planners. As illustrated, the WM-Grad computes inefficient, circuitous paths, while the DP does not look behind occlusions. See Section 4.3 and the paper's video for more details. Further, we show images from the scene and wrist cameras at different timesteps when planning with WoMAP (right).

4 Experiments

We evaluate WoMAP on open-vocabulary active object localization tasks both in simulation and on a TidyBot [34] to answer the following questions: (1) How does WoMAP perform across environments and tasks of varying difficulty (defined by scene complexity and initial conditions)? (2) Can WoMAP transfer effectively from simulation to the real world when trained only on photorealistic data? (3) Can WoMAP achieve zero-shot generalization to visual (unseen lighting, backgrounds) and semantic (unseen instructions, target objects) conditions? Further, in Appendix A.2.1, we ablate training with image reconstruction objectives, freezing vs. finetuning pretrained image encoders, and training image encoders from scratch.

4.1 Environments and Tasks

We consider four PyBullet (PB) simulation environments [35] and three real-world environments both within Gaussian Splats (GS) and on a TidyBot, focusing on practical scene configurations, e.g., with *kitchen*, *office*, and *random* environment themes. Within each environment, we randomly vary object configurations and target identities to create challenging localization tasks with occlusions and distractors. For a comprehensive evaluation of all methods, we vary the task difficulty along two axes: (i) *scene difficulty*, determined by the number, diversity, and layout of objects and (ii) *initial-pose difficulty*, representing the task difficulty due to occlusions, viewability, and distance to the target object which depends on the initial pose of the robot. In Appendix B.1, we provide a detailed discussion of the evaluation setup, as well as illustrations of the tasks and environments.

4.2 Baselines, Ablations, and Evaluation Metrics

For baselines, we first benchmark WoMAP against a VLM-based planner using GPT-4o [36], similar to [6, 37], prompted with the same template as WoMAP (Appendix A.3). We also evaluate a multi-task diffusion policy (DP), trained on expert demonstrations across multiple target objects for the same set of training environments. Additionally, we compare against world model-only planners: (i) *WM-CEM* [14], using the gradient-free, cross-entropy method for action proposals; (ii) *WM-Grad*, using gradient-based action optimization; and (iii) *WM-HR*, a heuristic-based, non-VLM planner using a fixed set of atomic actions (the same set used to prompt the VLM), refined via gradient descent without VLM input. We use success rate and efficiency (given by the success rate weighted by the path length [38]) as evaluation metrics, where success is defined by a threshold on the detection

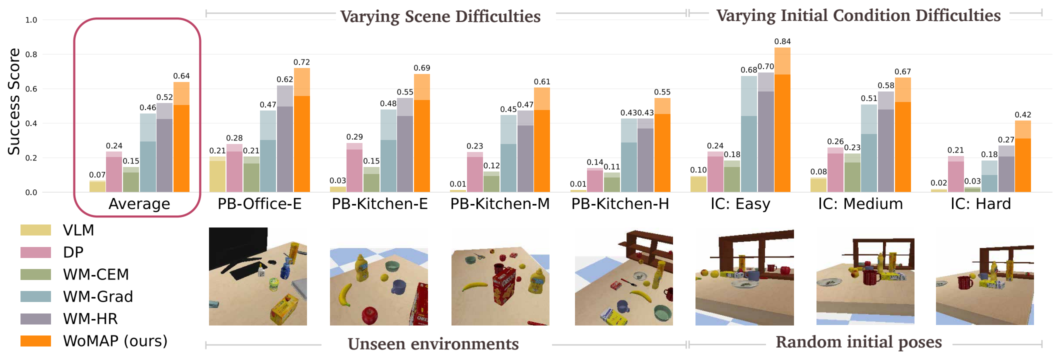

Figure 6: **PyBullet evaluation tasks and results.** Success rates (translucent bars) and efficiency scores (solid bars) in active object localization across PyBullet scenes (presented in the order of increasing difficulty) and initial-pose conditions: easy (E), medium (M), and hard (H). WoMAP outperforms all baseline methods in all scenes and initial-pose conditions.

confidence of the target object and the proportion of the associated bounding-box in the robot's camera observation. See Appendix B.3 for additional implementation details.

4.3 Evaluation across Varying Task Difficulty

We evaluate each method on novel (unseen) scenes in PyBullet and Gaussian Splat with varying scene and initial-pose difficulty, across a total of 150 tasks per environment. Figures 6 and 7 illustrate that on average WoMAP outperforms the VLM and diffusion policy (DP) baselines by over $9\times$ and $2\times$, respectively. While the VLM planner fails to account for physical grounding and the DP policy struggles to generalize beyond the training distribution, WoMAP generates grounded actions across diverse scenes. Moreover, we observe a progressive increase in the performance of the world-model-based planners with more informed search methods, in the order of (i) sampling-based WM-CEM, which is sample-inefficient even with $4\times$ as many action proposals, (ii) gradient-based WM-Grad, which relies on myopic local gradients and generates inefficient, circuitous actions, evidenced by its much lower efficiency compared to success scores, (iii) heuristics-based WM-HR, which does not leverage intelligent guidance from the VLM, limiting its performance in challenging problems, and ultimately (iv) WoMAP, which evaluates and optimizes the VLM action proposals with the world model. Figure 5 visualizes example trajectories from all planners on a TidyBot tasked with finding a banana hidden behind a mug, illustrating these results. Notably, the VLM fails to approach the target closely, while DP does not produce useful exploration behavior. Next, we discuss the performance of the planners with respect to the difficulty of the scene and initial pose and direct readers to Appendix B for ablations on the correlation between data quantity/scene diversity and performance.

Varying Scene Difficulty. As expected, the performance of all methods decreases with increasing scene difficulty, with a relative drop in success rates over 50% for VLM and DP baselines from the easiest scene to the hardest. In comparison, WoMAP's performance only drops by 23.6% in PyBullet and 40.2% in Gaussian Splat. Notably, all world-model-based planners exhibit lower performance drops, suggesting the importance of planning with physical priors provided by world models.

Varying Initial Conditions. WoMAP achieves the second-smallest relative performance drop in the PyBullet environments (after DP) and the smallest performance drop in the Gaussian Splat environments, even with its highest absolute scores. By leveraging the high-level reasoning capabilities of the VLM for action proposals, WoMAP effectively mitigates hallucinations in world models that arise in difficult-to-predict scenarios, e.g., occlusions.

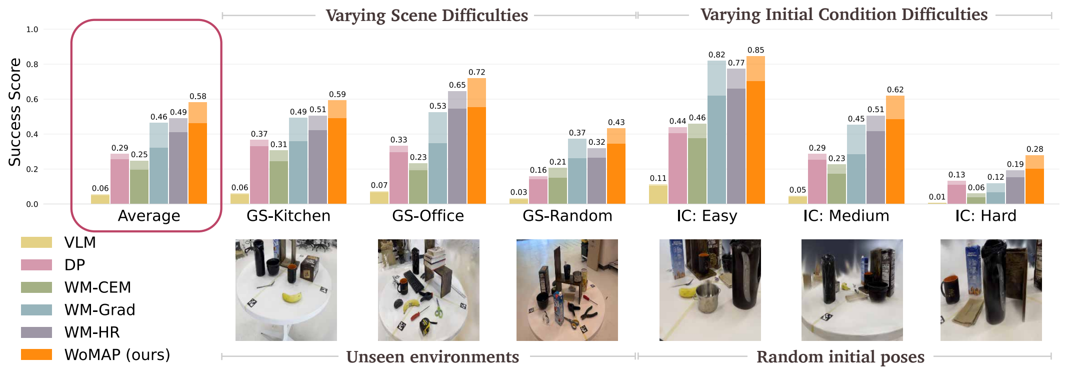

Figure 7: **Gaussian Splat evaluation tasks and results.** Success rates (translucent bars) and efficiency scores (solid bars) in active object localization across Gaussian Splat scenes and initial-pose conditions: easy (E), medium (M), and hard (H). As in the PyBullet scenes, WoMAP outperforms all baseline methods via effective action grounding and optimization.

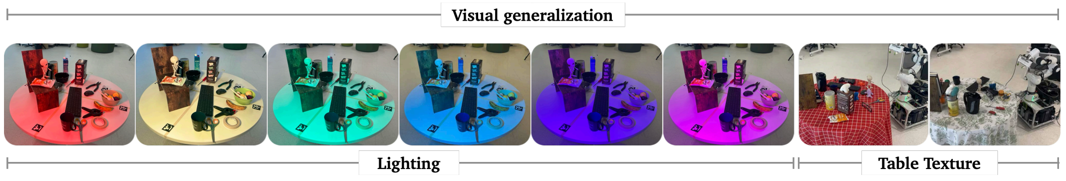

Figure 8: **Visual generalization setup:** lighting and background conditions.

4.4 Sim-to-Real Transfer with Gaussian Splats

We evaluate WoMAP's sim-to-real transfer ability on 20 hardware trials for each of the 3 corresponding real-world tasks using the TidyBot. For each trial, we randomize both the scene configuration and target object to include a diverse range of initial conditions and task difficulties. As shown in Table 1, despite being trained entirely in the Gaussian Splat simulation, WoMAP transfers effectively to the real world, achieving the same success rate in *GS-Office* and a higher success rate in *GS-Random* compared to the sim success rates, with only a moderate performance drop of 23% in *GS-Kitchen*. In contrast, the VLM baseline typically predicts unrealistic actions that violate joint limits, resulting in a substantial performance drop of 62% or more. This finding highlights WoMAP's strong generalization capabilities in producing reliable reward predictions under domain shifts, enabling effective sim-to-real transfer.

Table 1: Success rates (%) for zero-shot sim-to-real transfer for VLM and WoMAP.

Model	**GS-Kitchen**	**GS-Office**	**GS-Random**
VLM (sim)	6	13	3
VLM (real)	0	5	0
WoMAP (sim)	71	65	32
WoMAP (real)	55	65	63

4.5 Generalization to Novel Task Conditions

We examine the visual and semantic generalization capabilities of WoMAP trained only on nominal conditions in *GS-Random* on 10 scenes with 30 trajectories each. We evaluate WoMAP in out-of-distribution lighting and background conditions, illustrated in Figure 8. In Table 2, we show that WoMAP achieves strong zero-shot generalization with a success rate and efficiency score of 50% and 47% compared to 63% and 60% in nominal conditions, respectively (a decrease of less than 22%), even under extreme lighting conditions, with a further performance drop with out-of-distribution backgrounds. In general, these findings show that WoMAP learns robust latent-space features for generalization to out-of-distribution test-time conditions.

Table 2: Visual generalization results for various background and lighting conditions.

Axis	**Success Rate %**	**Efficiency %**
Nominal	63	60
Lighting	50	47
Backgrounds	30	28

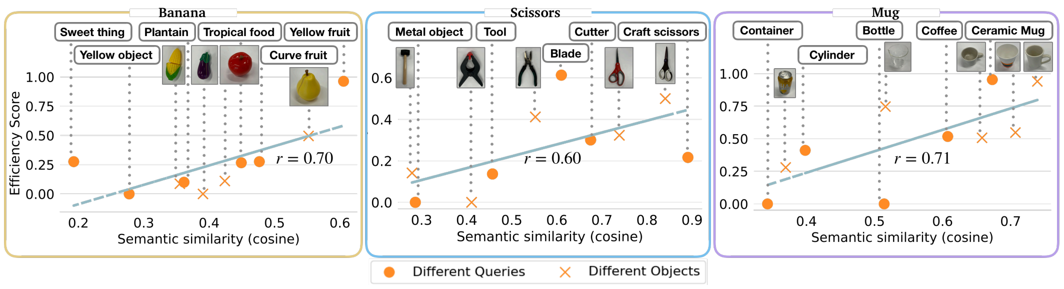

Figure 9: **Generalization plots for unseen queries and objects in the same category:** (left) *banana*, (center) *scissors*, (right) *mug*. We see a positive correlation in semantic similarity (cosine distance) of the objects/queries with the most similar object present in our training objects, and the efficiency score suggesting the model's performance.

In addition, we evaluate semantic generalization of WoMAP to unseen target objects and task instructions across two axes: (i) to unseen language instructions to find target objects that were seen during training and (ii) to unseen target objects with unseen language instructions. As illustrated in Figure 9, we consider three representative object categories: "banana," "scissors," and "mug," where WoMAP is only trained on a single instance for each. We find that WoMAP achieves strong semantic generalization, with an expected decrease in performance with decreasing *semantic similarity* as measured by the the cosine similarity metric. For example, we ask WoMAP to find the following unseen objects: "pear," "pliers," and "beaker." Even though WoMAP has not seen these objects during training, WoMAP is able to find each of these objects at test time. We discuss these results further in Appendix B.

5 Conclusion

We present WoMAP, a recipe for open-vocabulary active object localization that uses a scalable data generation pipeline to train a latent world model without expert demonstrations or online interaction data. WoMAP distills dense reward signals into the world model with a reconstruction-free training architecture for strong generalization from a few training samples. At inference time, WoMAP utilizes the world model for dynamics and rewards prediction to ground high-level action proposals from VLMs, demonstrating more efficient object localization and strong generalization to novel tasks.

6 Limitations and Future Work

Interactive Active Object Localization. Although we limit our problem to non-interactive object localization problems in this work, interaction between the robot and its environment is crucial to efficient exploration in many problem setting. Consequently, active object localization with interactive feedback from the environment is a promising direction for future research to enable more expressive and manipulation-intensive tasks.

Uncertainty Quantification in Rewards Distillation. WoMAP distills the confidence of pretrained object detectors into a world model as a rewards signal. However, learned object detectors sometimes produce uncalibrated confidence estimates, which could corrupt the training data, negatively impacting its effectiveness in grounding action proposals. Future work will explore calibration methods for pretrained object detectors to ensure data fidelity during training.

Hallucination Detection and Uncertainty Quantification in World Models. WoMAP's action optimization fails when the world model hallucinates the dynamics/rewards, usually in areas where the world model is not confident. Uncertainty quantification of world models remains critical to identifying when to trust predictions from world models, which has been relatively unexplored.

In future work, we will derive calibrated uncertainty quantification methods to enable uncertainty-aware planning to ensure effective action grounding and optimization. In addition, we will explore incorporating calibrated uncertainty estimates from the VLM on the action proposals into WoMAP to enable risk-sensitive planning.

Acknowledgments

The authors were partially supported by the NSF CAREER Award #2044149, the Office of Naval Research (N00014-23-1-2148), a Sloan Fellowship, and the Princeton SEAS Innovation Fund.

References

[1] J. J. Gibson. *The Ecological Approach to Visual Perception*. Houghton Mifflin, Boston, 1979.

[2] R. Bajcsy. Active perception. *Proceedings of the IEEE*, 76(8):966–1005, 1988. doi:10.1109/5.5968.

[3] M. Chang, T. Gervet, M. Khanna, S. Yenamandra, D. Shah, S. Y. Min, K. Shah, C. Paxton, S. Gupta, D. Batra, R. Mottaghi, J. Malik, and D. S. Chaplot. Goat: Go to any thing, 2023. URL https://arxiv.org/abs/2311.06430.

[4] D. S. Chaplot, D. P. Gandhi, A. Gupta, and R. R. Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. *Advances in Neural Information Processing Systems*, 33: 4247–4258, 2020.

[5] C. Wen, Y. Huang, H. Huang, Y. Huang, S. Yuan, Y. Hao, H. Lin, Y.-S. Liu, and Y. Fang. Zero-shot object navigation with vision-language models reasoning. In *International Conference on Pattern Recognition*, pages 389–404. Springer, 2025.

[6] A. Z. Ren, J. Clark, A. Dixit, M. Itkina, A. Majumdar, and D. Sadigh. Explore until confident: Efficient exploration for embodied question answering. *arXiv preprint arXiv:2403.15941*, 2024.

[7] R. Ramrakhya, D. Batra, E. Wijmans, and A. Das. Pirlnav: Pretraining with imitation and rl finetuning for objectnav. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17896–17906, 2023.

[8] N. Yokoyama, R. Ramrakhya, A. Das, D. Batra, and S. Ha. Hm3d-ovon: A dataset and benchmark for open-vocabulary object goal navigation. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5543–5550. IEEE, 2024.

[9] J. Ye, D. Batra, A. Das, and E. Wijmans. Auxiliary tasks and exploration enable objectgoal navigation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 16117–16126, 2021.

[10] L. Fan, M. Liang, Y. Li, G. Hua, and Y. Wu. Evidential active recognition: Intelligent and prudent open-world embodied perception, 2023. URL https://arxiv.org/abs/2311.13793.

[11] S. K. Ramakrishnan, D. Jayaraman, and K. Grauman. Emergence of exploratory look-around behaviors through active observation completion. *Science Robotics*, 4(30), May 2019. ISSN 2470-9476. doi:10.1126/scirobotics.aaw6326. URL http://dx.doi.org/10.1126/scirobotics.aaw6326.

[12] D. Ha and J. Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

[13] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023.

[14] G. Zhou, H. Pan, Y. LeCun, and L. Pinto. Dino-wm: World models on pre-trained visual features enable zero-shot planning. *arXiv preprint arXiv:2411.04983*, 2024.

[15] R. Ramrakhya, E. Undersander, D. Batra, and A. Das. Habitat-web: Learning embodied object-search strategies from human demonstrations at scale. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5173–5183, 2022.

[16] T. Gervet, S. Chintala, D. Batra, J. Malik, and D. S. Chaplot. Navigating to objects in the real world. *Science Robotics*, 8(79):eadf6991, 2023.

[17] 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. In *International Conference on Machine Learning*, pages 42829–42842. PMLR, 2023.

[18] H. Jiang, B. Huang, R. Wu, Z. Li, S. Garg, H. Nayyeri, S. Wang, and Y. Li. Roboexp: Action-conditioned scene graph via interactive exploration for robotic manipulation. *arXiv preprint arXiv:2402.15487*, 2024.

[19] R. Mendonca, S. Bahl, and D. Pathak. Alan: Autonomously exploring robotic agents in the real world. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3044–3050. IEEE, 2023.

[20] P. Wu, A. Escontrela, D. Hafner, P. Abbeel, and K. Goldberg. Daydreamer: World models for physical robot learning. In *Conference on robot learning*, pages 2226–2240. PMLR, 2023.

[21] K. Nakamura, L. Peters, and A. Bajcsy. Generalizing safety beyond collision-avoidance via latent-space reachability analysis. *arXiv preprint arXiv:2502.00935*, 2025.

[22] A. Bar, G. Zhou, D. Tran, T. Darrell, and Y. LeCun. Navigation world models. *arXiv preprint arXiv:2412.03572*, 2024.

[23] N. Hansen, H. Su, and X. Wang. Td-mpc2: Scalable, robust world models for continuous control. *arXiv preprint arXiv:2310.16828*, 2023.

[24] H. Qi, H. Yin, Y. Du, and H. Yang. Strengthening generative robot policies through predictive world modeling. *arXiv preprint arXiv:2502.00622*, 2025.

[25] O. Shorinwa, J. Sun, M. Schwager, and A. Majumdar. Siren: Semantic, initialization-free registration of multi-robot gaussian splatting maps. *arXiv preprint arXiv:2502.06519*, 2025.

[26] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, Q. Jiang, C. Li, J. Yang, H. Su, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *European Conference on Computer Vision*, pages 38–55. Springer, 2024.

[27] 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 *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[28] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019.

[29] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[30] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[31] N. Hansen, X. Wang, and H. Su. Temporal difference learning for model predictive control. *arXiv preprint arXiv:2203.04955*, 2022.

[32] M. Burchi and R. Timofte. Mudreamer: Learning predictive world models without reconstruction. *arXiv preprint arXiv:2405.15083*, 2024.

[33] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[34] J. Wu, W. Chong, R. Holmberg, A. Prasad, Y. Gao, O. Khatib, S. Song, S. Rusinkiewicz, and J. Bohg. Tidybot++: An open-source holonomic mobile manipulator for robot learning. *arXiv preprint arXiv:2412.10447*, 2024.

[35] E. Coumans and Y. Bai. Pybullet, a Python module for physics simulation for games, robotics and machine learning. `http://pybullet.org`, 2016–2022.

[36] A. Hurst, A. Lerer, A. P. Goucher, A. Perelman, A. Ramesh, A. Clark, A. Ostrow, A. Welihinda, A. Hayes, A. Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

[37] D. Goetting, H. G. Singh, and A. Loquercio. End-to-end navigation with vision language models: Transforming spatial reasoning into question-answering. *arXiv preprint arXiv:2411.05755*, 2024.

[38] P. Anderson, A. Chang, D. S. Chaplot, A. Dosovitskiy, S. Gupta, V. Koltun, J. Kosecka, J. Malik, R. Mottaghi, M. Savva, et al. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*, 2018.

[39] O. Shorinwa, J. Tucker, A. Smith, A. Swann, T. Chen, R. Firoozi, M. Kennedy III, and M. Schwager. Splat-mover: Multi-stage, open-vocabulary robotic manipulation via editable gaussian splatting. *arXiv preprint arXiv:2405.04378*, 2024.

[40] M. N. Qureshi, S. Garg, F. Yandun, D. Held, G. Kantor, and A. Silwal. Splatsim: Zero-shot sim2real transfer of rgb manipulation policies using gaussian splatting. *arXiv preprint arXiv:2409.10161*, 2024.

[41] S. Zhou, H. Chang, S. Jiang, Z. Fan, Z. Zhu, D. Xu, P. Chari, S. You, Z. Wang, and A. Kadambi. Feature 3dgs: Supercharging 3d gaussian splatting to enable distilled feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21676–21685, 2024.

[42] O. Shorinwa, J. Sun, and M. Schwager. Fast-splat: Fast, ambiguity-free semantics transfer in gaussian splatting. *arXiv preprint arXiv:2411.13753*, 2024.

[43] M. Tancik, E. Weber, E. Ng, R. Li, B. Yi, T. Wang, A. Kristoffersen, J. Austin, K. Salahi, A. Ahuja, et al. Nerfstudio: A modular framework for neural radiance field development. In *ACM SIGGRAPH 2023 conference proceedings*, pages 1–12, 2023.

[44] S. M. LaValle and J. J. Kuffner. Rapidly-exploring random trees: Progress and prospects: Steven m. lavalle, iowa state university, a james j. kuffner, jr., university of tokyo, tokyo, japan. *Algorithmic and computational robotics*, pages 303–307, 2001.

[45] D. Shah, M. Equi, B. Osinski, F. Xia, B. Ichter, and S. Levine. Navigation with large language models: Semantic guesswork as a heuristic for planning, 2023. URL `https://arxiv.org/abs/2310.10103`.

[46] A. Sridhar, D. Shah, C. Glossop, and S. Levine. Nomad: Goal masked diffusion policies for navigation and exploration. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 63–70. IEEE, 2024.

[47] R. J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280, 1989.

[48] N. Reimers and I. Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019. URL https://arxiv.org/abs/1908.10084.

Appendix A Implementation Details

A.1 Details on Data Generation

A.1.1 Preliminaries: Gaussian Splatting for Photorealistic Novel View Rendering

We provide a brief introduction to Gaussian Splatting and its applications to robotics. Gaussian Splatting [13] is a volumetric scene representation that uses explicit ellipsoidal primitives to represent non-empty space in any given environment. Trained entirely from RGB poses and associated camera poses, Gaussian Spatting enables real-time photorealistic novel-view synthesis without any structural priors, unlike many existing scene reconstruction methods. Given its high-fidelity reconstruction, novel-view synthesis, and amenability to open-vocabulary semantics, Gaussian Splatting has been widely applied in robotics, e.g., robot manipulation [39, 40]. Training a latent world model requires abundant data coverage of the environment, which is challenging to collect in the real world. In this work, we leverage Gaussian Splatting for scalable data generation from only a few real-world videos. Specifically, we employ semantic Gaussian Splatting [41, 42] for automatic labeling of target objects, distilling language semantics from CLIP [27] into the Gaussian Splat. In the following subsections, we briefly describe the procedure used for generating, aligning, and rendering views for multiple scenes.

Collecting Videos and Training the Gaussian Splat. Trained Gaussian Splats do not share a common reference frame, in general. Hence, we align the individual Gaussian Splats using four Aruco markers in fixed positions. However, we note that other approaches such as semantics-based alignment can also be used. In each scene, we record a one-minute video as input to the Gaussian Splat. We compute the camera poses for each video using structure-from-motion and subsequently train the semantic 3D Gaussian Splat [25] for 30,000 iterations on an Nvidia L40 GPU using Nerfstudio [43].

Scene Alignment and Annotation. The reconstructed scene representation could have an arbitrary reference frame. However, a common reference frame is necessary for data consistency when generating data from multiple scenes. Consequently, we first perform alignment of the scene coordinates by matching Aruco tags detected in the reconstruction with its measured ground-truth position and orientation. Finally, given coordinates of the detected points in the world frame, we solve the Perspective-n-Point (PnP) problem and the point-registration problem using RANSAC to compute the camera-to-world and the Gaussian Splat-to-world transforms, respectively. We query the semantic field of the Gaussian Splat to annotate the position and dimension of each target object in the scene, visualized in Figure 10, with different target objects.

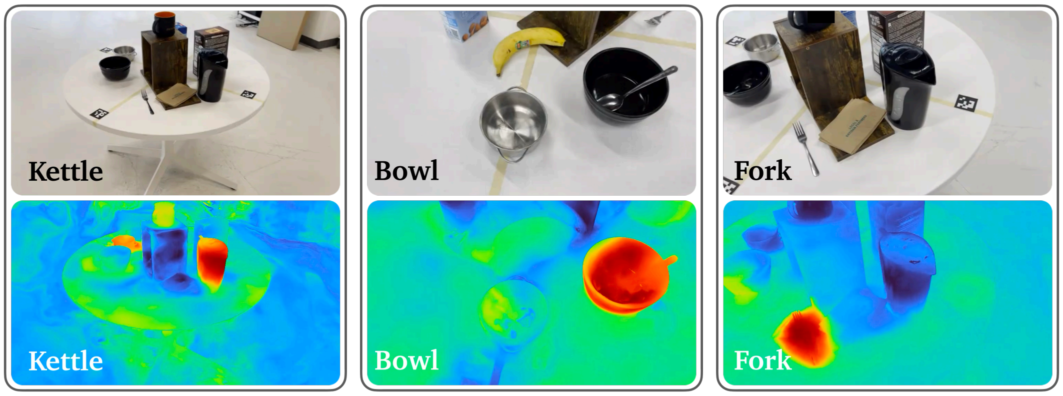

Figure 10: Querying the semantic field of the Gaussian Splat.

A.1.2 Ablation: Generating Training Data without Novel Views

We examine the need for data scalability beyond real-world data, ablating WoMAP limited to real-world video frames compared to the data generated from the Gaussian Splats. First, we extract a set of image frames uniformly across each real-world video and use these images in training the Gaussian Splat. Next, we train two world models with: (i) only the video frames of each scene (*Video-Only*) and (ii) on the rendered images from the Gaussian Splat (*GSplat-Data*). The training data of the Video-Only and GSplat-Data world models consist of about 2100 and 9000 images, respectively. We evaluate the trained model using the gradient-based planner WM-Grad and summarize our results in Table 3. Across all scenes and initial conditions, the GSplat-Data model outperforms the Video-Only model by significant margins, ranging between 50% and 200%. The relatively poor performance of the Video-Only model can be explained by the lack of sufficient data coverage in the real-world videos, underscoring the importance of our scalable data pipeline. Our data generation pipeline provides not only additional training images, but also *diverse* viewpoints, which makes the GSplat-Data model more robust to initial conditions compared to the Video-Only model (sweeping from *easy* to *hard*).

Table 3: Success scores of the world model trained using only real-world video frames (Video-Only model) compared to the (GSplat-Data model). With more diverse data, the GSplat-Data model outperforms Video-Only model across scenes and initial conditions.

Training Data	**GS-Kitchen**			**GS-Office**			**GS-Random**		
	init-easy	init-medium	init-hard	init-easy	init-medium	init-hard	init-easy	init-medium	init-hard
Video-Only	0.06	0.04	0.02	0.24	0.06	0.02	0.04	0	0
GSplat-Data	0.84	0.50	0.14	0.86	0.50	0.22	0.76	0.36	0

A.1.3 Details on Trajectory Data Generation

To get sufficient coverage of diverse viewpoints in the the PyBullet environment and Gaussian Splat scenes, we generate synthetic collision-free trajectories that start from randomized initial positions and navigate towards various target locations. Specifically, we leverage the RRT* planner [44] to compute feasible, diverse paths between the start and goal. We concatenate these camera poses and add random linear and angular perturbations to further increase data diversity to cover a more realistic range of viewpoints encountered at deployment time. Figure 11 provides a visualization of trajectories generated in our training dataset. We collect observations at relatively low frequencies where the delta distance between consecutive observations is around 1-5cm. The largest model that we train contains about 10,000 observation-camera pose pairs for 500 trajectories (or 20 observations per trajectory), which is considerably smaller in scale than the training data used in many other imitation learning or reinforcement learning-based visual navigation policies [45, 46]. Moreover, we show in Appendix B.4.1 that WoMAP's performance remains competitive in much smaller training configurations.

A.2 World Model Implementation

Here, we summarize the implementation details of the world model—composed of the observation encoder, dynamics predictor, and rewards predictor—and provide the hyperparameters used in training the world model. For interpretability, we train a decoder to map latent states to the image space without backpropagating the gradients through the other components of the world model. In addition, to better visualize the objects the world model is focusing on, we train the rewards models to predict bounding-boxes along with the scalar rewards, which we overlay in the decoded RGB images.

Observation Encoder. We encode raw image observations into the latent space using the pre-trained DINOv2 model *dinov2_vits14* [29]. We use the norm of the patch tokens of image o_t as its feature embedding $h_\theta(o_t) \in \mathbb{R}^{384}$. In addition, we freeze the weights of the DINOv2 model during training and do not apply any image data augmentation, e.g., color jittering and random cropping, since the pre-trained DINOv2 model already utilizes data augmentation for training. Moreover, random perturbation of the image, such as random rotation or cropping, may compromise the fidelity of the ground-truth image-action pairs. In Appendix A.2.1, we ablate the observation encoder.

Dynamics Predictor. To condition the ViT-based dynamics predictor on both the latent state z_t and action a_t, we map z_t and a_t to a 384-dimensional embedding space using an affine transformation and

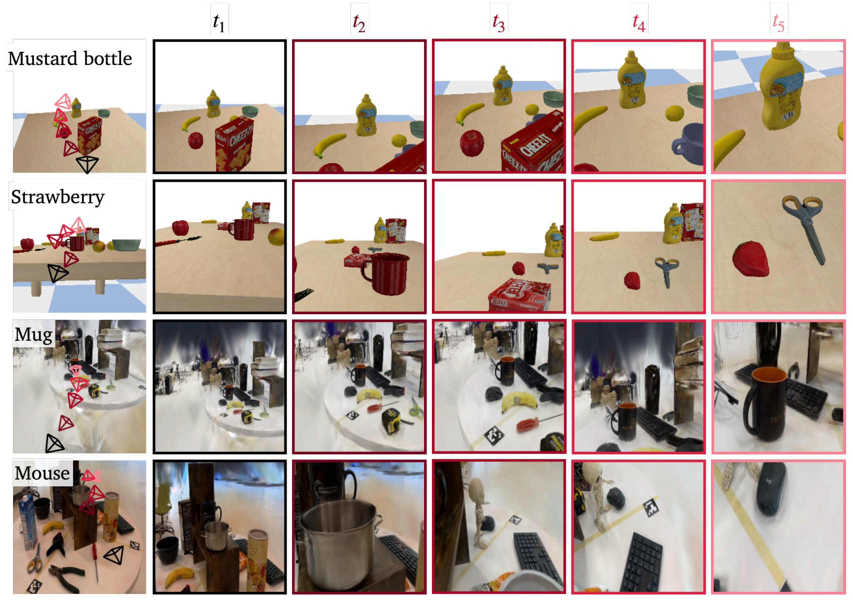

Figure 11: Visualization of training trajectories generated in PyBullet and Gaussian Splat.

concatenate the resulting embeddings. In preliminary experiments, we found that a longer historical context for dynamics prediction did not provide any significant improvement in prediction accuracy. As a result, we provide only the last latent state to the dynamics predictor. To improve the multi-step prediction accuracy, we supervise the dynamics predictor over a sequence of observation-action pairs, recursively passing in the previous prediction into the model. Consequently, we do not optimize the dynamics predictor using teacher forcing [47]. Although teacher forcing facilitates faster training through parallelism, teacher forcing contributes to significant accumulation of dynamics errors over multi-step predictions.

Rewards Predictor. Like the dynamics predictor, we condition the rewards predictor on z_t and the language embedding e_g, each mapped to $\mathbb{R}^{384}$. We concatenate the embeddings and apply full cross-attention to predict the scalar reward for that latent state and target object. We train the rewards predictor using the binary cross-entropy loss function given by: $\ell_t = -[r_{\text{gt},t} \log(r_t) + (1 - r_{\text{gt},t}) \log(1 - r_t))]$, for datapoint $(r_{\text{gt},t}, r_t)$ with ground-truth reward $r_{\text{gt},t}$ and predicted reward r_t.

Training Setup and Hyperparameters. We train the world model on a single Nvidia L40 GPU with 48GB of GPU VRAM using a batch size of 25 for between 8 to 10 hours, depending on the task environment. In Table 4, we present the hyperparameters used in training the world model. We warmup training with a learning rate (LR) of $1e^{-3}$ before training for the full number of epochs (100) with an LR of $5e^{-4}$ for stable training. We observed that the training loss diverged for learning rates greater than $1e^{-3}$. Further, in Table 5, we report the number of trainable and non-trainable parameters in each component of the world model. The dynamics q_ψ and rewards v_ϕ predictors have about 7.6 and 4.1 million trainable parameters, respectively. Meanwhile, we do not fine-tune the observation encoder h_θ.

A.2.1 Ablations

We ablate different components of the world model, examining the effects of training an encoder from scratch, finetuning a pre-trained encoder, and using an image reconstruction loss. We report our findings in Figure 12, 13, and 14, where *ViT-R* and *ViT-NR* denote ViT trained from scratch with image reconstruction and without image reconstruction, respectively, *DINO-Frozen* denote a frozen DINOv2 model, and *DINO-R-FT* and *DINO-NR-FT* denote a finetuned DINOv2 model trained with

Table 4: WoMAP's Hyperparameters.

Name	Value
Image Size	224
Lang. Embed Dim	384
Pred. Embed Dim	384
Start LR	1e-3
LR	5e-4
Warmup Epochs	2
Weight Decay	4e-2
Final Weight Decay	0.4
Batch Size	25
Total Epoch	100
Planning Horizon	4

Table 5: WoMAP's Number of Parameters (in millions).

Name	# trainable	# non-trainable
h_θ	0	22
q_ψ	7.6	0.1
v_ϕ	4.1	0.1

and without image reconstruction, respectively. In all applicable plots, we represent the success rate of each method by the solid-color bars and the efficiency scores by the translucent bars. We discuss these results in the following subsections.

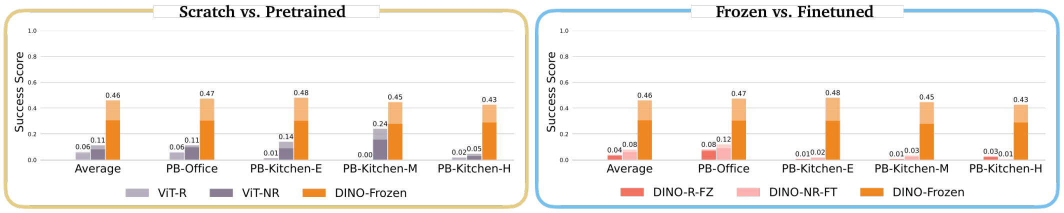

Figure 12: **World Model Architecture Ablations in the PyBullet Scenes.** We explore training the observation encoder from scratch, finetuning, and training the observation encoder with image reconstruction, where *ViT-R* denotes a ViT trained from scratch **with** image reconstruction, *ViT-NR* denote a ViT trained from scratch **without** image reconstruction, respectively, *DINO-Frozen* denotes a **frozen** DINOv2 encoder, *DINO-R-FT* denotes a finetuned DINOv2 model trained **with** image reconstruction, and *DINO-NR-FT* denotes a **finetuned** DINOv2 encoder trained **without** image reconstruction. WoMAP uses DINO-Frozen.

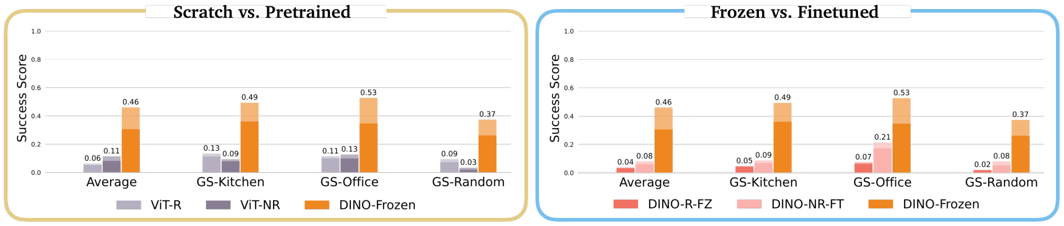

Figure 13: **World Model Architecture Ablations in the Gaussian Splatting Scenes.** We ablate training the observation encoder from scratch, finetuning, and training the observation encoder with image reconstruction, where *ViT-R* denotes a ViT trained from scratch **with** image reconstruction, *ViT-NR* denote a ViT trained from scratch **without** image reconstruction, respectively, *DINO-Frozen* denotes a **frozen** DINOv2 encoder, *DINO-R-FT* denotes a finetuned DINOv2 model trained **with** image reconstruction, and *DINO-NR-FT* denotes a **finetuned** DINOv2 encoder trained **without** image reconstruction. WoMAP uses DINO-Frozen.

Training the Observation Encoder from Scratch. We compare a ViT-based observation encoder trained from scratch (ViT-NR) to a frozen pre-trained DINOv2 model (DINO-NR-FZ), without an image reconstruction objective. From Figure 12 and 13, DINO-NR-FZ achieves higher success rates and efficiency scores across the PyBullet and Gaussian Splat scenes. Although both models have access to the same data when training the world model, the results suggest that DINO-NR-FZ model benefits from large-scale pre-training, which provides a robust latent state for dynamics and rewards prediction even without any finetuning. Further, we observed that the ViT was more unstable to train, given the total number of trainable parameters. In general, the ViT-NR and ViT-R models may require more training data to learn more useful visual features compared to the frozen DINOv2 models.

Finetuning the Pre-trained Observation Encoder. We explore finetuning the DINOv2 encoder, comparing its performance to that of the frozen model. We find that finetuning the DINOv2 encoder leads to training instability that adversely impacts the performance of the world model. In fact, in many of our experiments, the training loss failed to decrease or raised Not-a-Number (NaN) errors. In Figure 14, we show the training loss for the dynamics predictor across the four PyBullet environment, highlighting the increase in the training loss at the initial stages of the training procedure in the fine-tuned DINOv2 model. This training instability may be attributed to the more complicated loss landscape with many local minima during finetuning. In contrast, the training loss for the frozen DINOv2 models decreases relatively monotonically. These training dynamics are reflected in the success rates and efficiency scores achieved by both models. The finetuned models DINO-NR-FT and DINO-R-FT have notably lower scores on the performance metrics compared to frozen DINO encoder *DINO-Frozen*.

Training with a Reconstruction Objective. Here, we investigate training the world model with an image reconstruction objective. From Figures 12 and 13, we find that training with an image reconstruction objective generally leads to a degradation in the performance of the world model, e.g., in the ViT-R and DINO-R-FT models. Although image reconstruction objectives can provide dense rewards supervision, training instability often eliminates this potential advantage, underscoring the challenge with image reconstruction-based training. In some cases, the training instability can lead to significant drops in performance, e.g., in the ViT-R model when trained in the PB-Kitchen-M scene. In summary, the results from the ablations indicate that the frozen pretrained DINOv2 provides generalizable latent features that enable accurate dynamics and rewards prediction and effective action grounding.

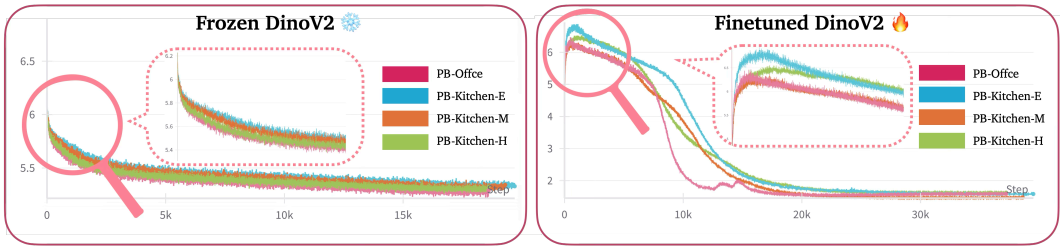

Figure 14: **Frozen vs. Finetuned DINOv2 Encoder.** Finetuning the DINOv2 encoder generally leads to training instability, negatively impacting performance.

A.3 Details on Planning Integration

A.3.1 Planning with VLMs

Using the world model as a local planner poses many challenges given the noisy reward landscape, particularly in the low training data regime. However, world models can serve as good evaluators/optimizers of imperfect action proposals generated from another policy. For example, generating good 6D action proposals with VLMs is difficult due to the limited spatial understanding ability of VLMs, a challenge that can be addressed using a world model. We experimented with the following prompting strategies using GPT-4o [36]:

Direct 6D action output. We first provide rough dimensions of the scene and ask the VLM to directly output 6D actions (x, y, z, roll, pitch, yaw). However, due to inconsistencies in coordinate system conventions across its pretraining data (e.g., variations in the orientation of the positive x and y axes), the vision-language model often fails to consistently interpret spatial directions and rotations correctly.

Direction output. To minimize confusion on the coordinate axes definition, we experimented with expressing the translation and rotation actions with natural language descriptions, such as move

```
This is what you currently see. Please carefully analyze the current observation
and think about what is the best action to take given what you see.
If you think the current observation is a good enough view of {self.target}
and you can't get any closer, please say 'DONE'.
Otherwise, please select the top {self.k} choices from the action options
below that you think would help you achieve the goal given the current observation.
The options are:
    (A) Move directly forward for 15 cm -- this lets you approach the objects in view
    (B) Move directly to the left for 15 cm -- this expands your left view by a bit
    (C) Move directly to the right for 15 cm -- this expands your right views by a bit
    (D) Look to the left by 45 degrees -- this lets you look around
    (E) Look to the right for 45 degrees -- this lets you look around
    (F) Move forward-left for 15 cm -- this lets you approach objects on the
    left side of your view
    (G) Move forward-right for 15 cm -- this lets you approach objects on the
    right side of your view
    (H) Move forward-left for 15 cm and then look right by 45 degrees -- this lets
    you look behind an object
    (I) Move forward-right for 15 cm and then look left by 45 degrees -- this lets
    you look behind an object
All these actions should be executed relative to the current position.
Please think carefully step by step and reason about why your choice could help you
get the desired observation.
Please also review your movement history to see where you've already explored.
Output the results in a structured JSON format as follows:
{
    "descriptions": <what you observe in the the current scene and
    whether there are hints of the {self.target}.>
    "actions": [
        {"rank": 1, "choice": <choice1>, "confidence": <confidence_score1>,
        "explanation": <explanation1>},
        ...
        {"rank": {self.k}, "choice": <choice{self.k}>, "confidence":
        <confidence_score{self.k}>, "explanation": <explanation{self.k}>},
    ]
}
<choice1>, ... <choice{self.k}> should be a single letter representing one of the
9 choices above.
Ensure the confidence scores are in descending order.
Do not include any extra explanation outside of the JSON structure.
```

Figure 15: Prompt provided to the VLM

forward/backward, tilt up/down, etc. However, we have found that these directions are not fully descriptive of exploration behavior. For example, if the VLM suggests looking behind an obstacle that is directly in front, its action outputs are often axis-aligned, which is not expressive enough to encode this "look around" behavior.

Action Primitives. Our final version frames the prompt as a multiple-choice question, providing the VLM with a list of action primitives. Empirically, we find that using the coarse action primitives as shown in Fig. 15, the VLM is capable of matching its high-level suggestions with the correct action outputs consistently. Despite covering only a limited set of actions with coarse magnitudes, we demonstrate that WoMAP can still successfully optimize these action proposals into grounded actions, as shown in Figure 2.

Appendix B Experiment Details

B.1 Task Design Details

We provide more details on the design choices for the tasks we examine WoMAP on.

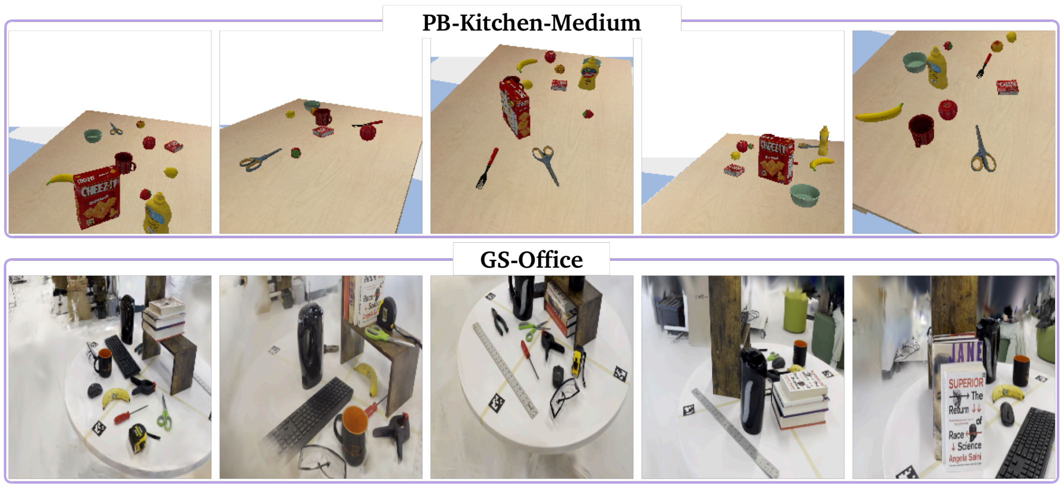

Figure 16: Scene variations for a single environment. We create multiple scenes within each environment by varying the configuration and degree of occlusions of the objects in the environment. We show a few scenes in the PB-Kitchen-Medium and GS-Office environments.

B.1.1 Designing task environments

As discussed in Section 4.1, we evaluate WoMAP on four simulation environments and three real environments. Figures 6 and 7 show example scenes for each environment, and Figure 16 shows the different variations for a single environment across different scenes.

Our key motivation to selecting the suite of environments and objects lies on two axes: practicality and difficulty. We design each task based on a particular theme resembling a cluttered living area where active object localization is a a core challenge requiring spatial reasoning, viewpoint planning, and the ability to handle occlusions. For each environment, we pick representative objects along the axis of difficulty and try to cover a diverse set of objects. For *easy* scenes, we pick distinct objects, which are typically small in size, whereas for difficult scenes, we place large shelves and visually similar objects (e.g., apple and peach), creating occlusions and distractions. We provide a comprehensive list of environments and target objects in each environment in Table 6.

Table 6: Target objects used for each environments.

PB-Kitchen-Easy	**PB-Kitchen-Medium**	**PB-Kitchen-Hard**	**PB-Office**	**GS-Kitchen**	**GS-Office**	**GS-Random**
banana	banana	banana	gum	banana	banana	banana
apple	apple	apple	lipton tea	mug	mug	mug
green bowl	green bowl	green bowl	small marker	bowl	scissors	bowl
lemon	lemon	lemon	glue	fork	books	fork
mustard	mustard	mustard	book	pot	keyboard	keyboard
cracker box	cracker box	sugar box	stapler	kettle	mouse	mouse
blue cup	mug	mug	mug		screwdriver	screwdriver
	scissors	scissors	scissors		eyeglass	eyeglasses
	peach	peach	remote			pot
	fork	fork	cleanser			scissors
	strawberry	strawberry	potato chip			milk box
	jello box	jello box				skeleton
		plate				
		potato chip				

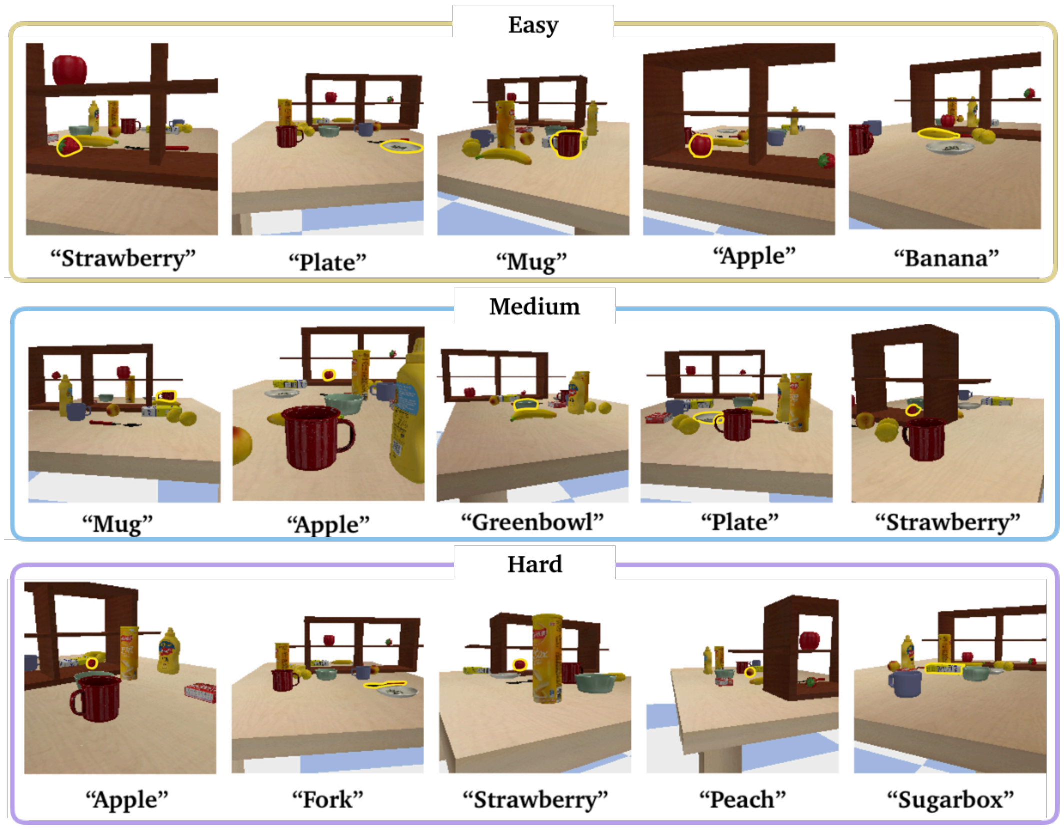

Figure 17: **PB-Kitchen-Hard initialization difficulty.** Initial observations and target query for different initial-pose difficulty levels in the PB-Kitchen-Hard scene.

B.1.2 Evaluating Task Difficulty

To systematically evaluate our framework under various settings, we define task difficulty across two dimensions: *scene difficulty*, which is determined by the diversity of objects present and the level of compactness, and *initial-pose difficulty*, which is determined by a heuristics function that is computed from three factors: initial detection confidence, ground truth distance to target, and level of occlusion of the bounding box. The thresholds for these factors are environment-dependent to account for scene dimensions, detection qualities, etc. Though the difficulty metrics are not strictly quantifiable, from empirical results we observe consistent trends in degrading performance as task difficulty increases, and show promising trend on WoMAP's robustness to exhibiting less performance degradation compared to the baselines. Figure 17 and 18 show a more comprehensive visualization of different initial condition levels in two PyBullet and Gaussian Splat environments to offer readers a better intuition.

B.2 Hardware Experiment Setup

We evaluate our policy trained entirely in the Gaussian Splat directly on a TidyBot platform with a Panda Franka arm. With a mobile base and a 6-DOF arm equipped with an onboard Intel RealSense camera, the TidyBot is well suited for active object localization tasks since it has a larger effective workspace, compared to tabletop, fixed-base Franka robots that are constrained to mostly top-down views. We interface with the TidyBot through an onboard NUC, which publishes images to a desktop for inference using the world model or VLM.

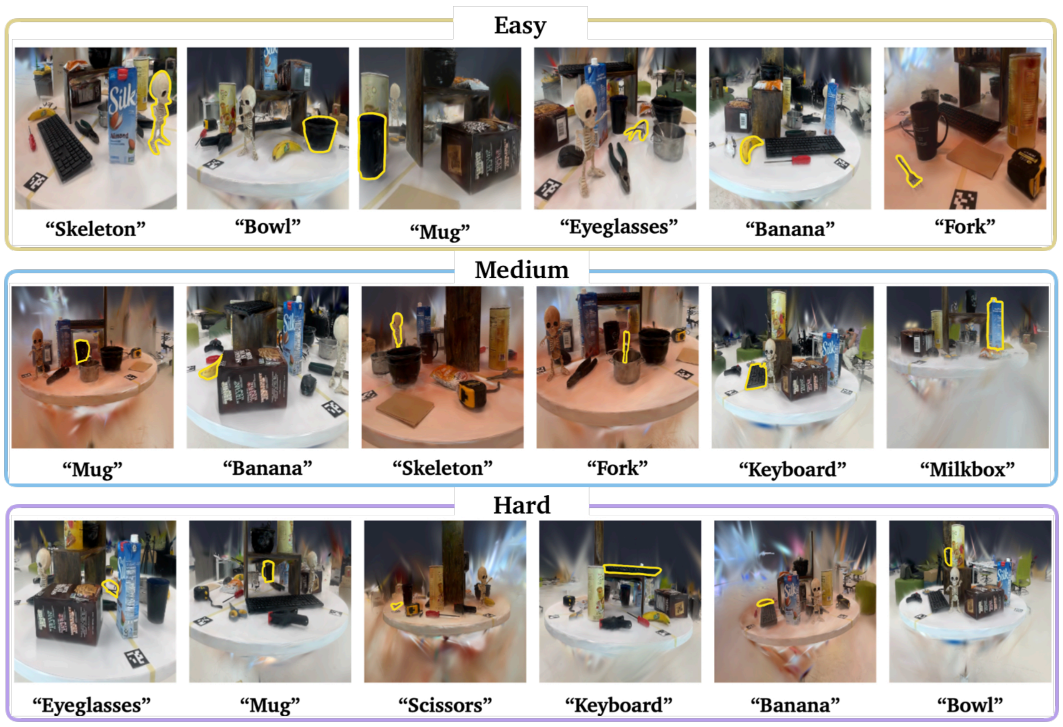

Figure 18: **GS-Random initialization difficulty.** Initial observations and target query for different initial-pose difficulty levels in the GS-Random scene.

B.3 Evaluation Setup

B.3.1 Choice of Baselines and Ablations

In the following sections, we discuss in more detail how we setup the evaluation framework, and motivation for the choice of baselines and ablations that we choose to include. Selecting the proper baseline is particularly challenging in our case. For one, many policies are not open-vocabulary and require more constrained problem/action spaces. Secondly, the setup of our task (large environment variations and observations from only an onboard camera) also makes finetuning state-of-the-art manipulation policies difficult, since they are not designed for the task. In our experiments, we compare WoMAP to a VLM-based planner with GPT-4o [36] as the VLM. We compute the gradients during planning using automatic differentiation in the WM-Random, WM-HR, and WoMAP.

Diffusion Policy. One might expect the diffusion policy to perform better in our experiments. Upon further investigation, we observed that a *single-task* diffusion policy (trained to localize a single object in an environment) performs well; however, the *multi-task* diffusion policy (DP), which is more relevant to the active object localization problem, failed to perform well. We found that the DP generally moved forward in the direction the robot was initialized at, without exhibiting any intelligent exploration behavior, e.g., looking behind objects and in occluded areas. We visualize some of the DP trajectories in Figure 19 in the PB-Kitchen-Easy and PB-Kitchen-Hard scenes. The arrows in the figure indicate the direction of travel. From Figure 19, we see that the DP does not seem to make intentional decisions to move towards the target objects, annotated in the figure.

B.3.2 Choice of Metrics

We make two key observations when designing the evaluation metrics. First, in order to make sure that the final observation is of good quality, our *success score* is defined to be 1 if both (i) the detection confidence labeled by GroundingDINO is above a certain threshold, and (ii) the labeled bounding box proportion is above a certain threshold. Since GroundingDINO's detection confidence and bounding

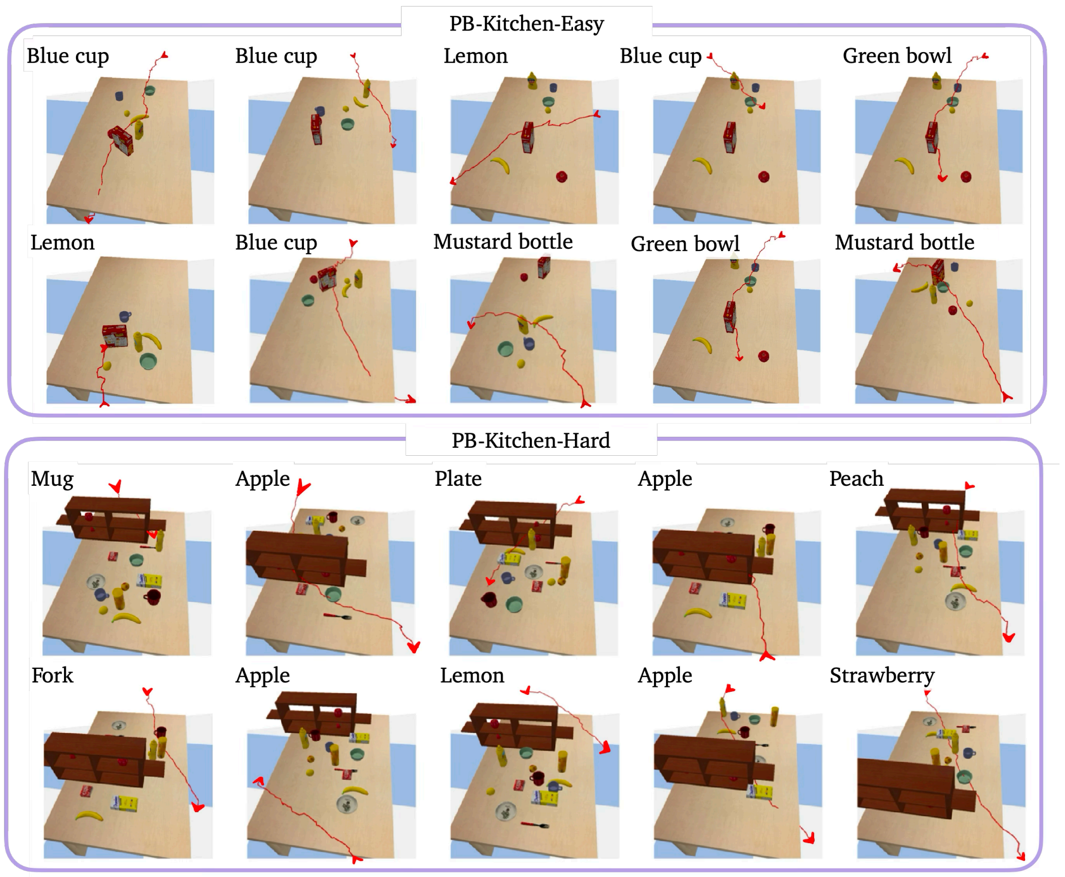

Figure 19: **Diffusion policy trajectory visualizations.** Top two rows: PB-Kitchen-Easy with easy initial conditions, Bottom two rows: PB-Kitchen-Hard with hard initial conditions. The DP generally moved forward in the direction the robot was initialized at, without showing any intelligent exploration behavior, e.g., looking behind objects.

box size on different objects can behave differently depending on the object identity, shape, or size, we choose object-specific scaling parameters from the best view.

Secondly, the quality of task completion should also be dependent on the amount of distance traveled to reach the target as some policies are more efficient than others. To this end, we define the *efficiency score* as efficiency $= r * \exp(-d/d^\star)$, where r denotes the success rate, d denotes the distance traveled, and $d^\star$ the optimal distance to the object. In practice, we use an estimate of $d^\star$.

B.4 Additional Experiment Results

B.4.1 Ablation Studies on Training Data

Though we performed all of our experiments with a fixed training setup (50 scenes-500 trajectories for the PyBullet environments and 10 scenes-300 trajectories for the Gaussian Splat/Real environments), we conducted additional ablation studies to investigate the influence of the size and diversity of the training data on model performance. We perform this ablation study in the PB-Kitchen-Easy scene, since we can easily vary the size and diversity of the training data in PyBullet.

In Figure 20, we observe that the performance of WoMAP and WM-Grad increases as the total number of trajectories in the training dataset increases. We show the performance of WM-Grad alongside WoMAP to indicate the base performance of the world model without VLM action proposals. As expected, WoMAP's performance is strongly correlated with the size of the training data, which

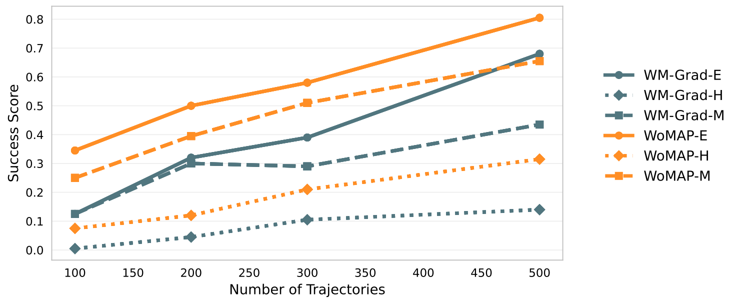

Figure 20: Average success score for different number of trajectories sampled from 50 scenes in the PB-Kitchen-Easy task. We observe a positive correlation between the number of training trajectories and the success rate of WM-Grad and WoMAP across different initial conditions: Easy (E), Medium (M), and Hard (H). With only 200 training trajectories, WoMAP outperforms the VLM and DP baselines (see Figure 6).

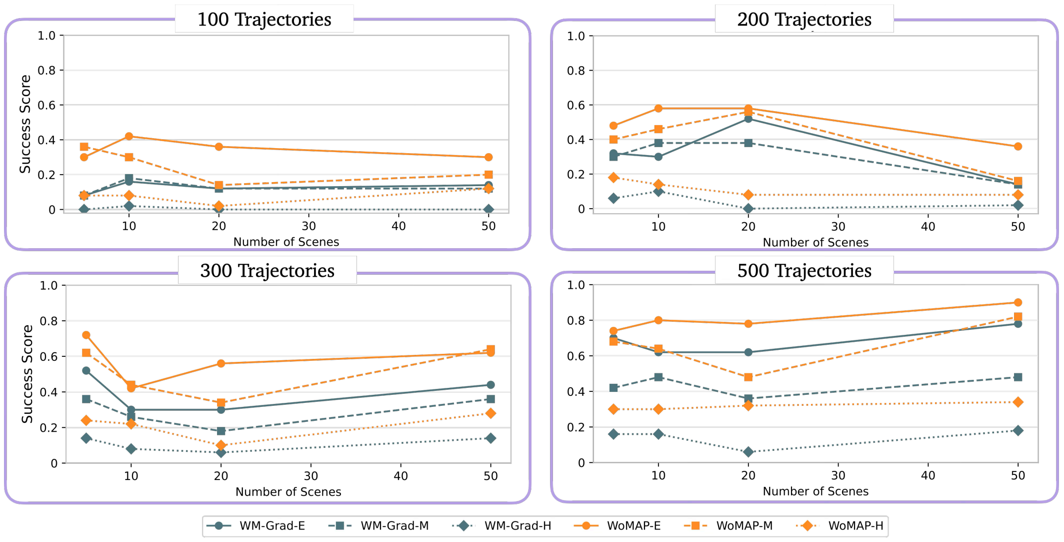

Figure 21: Average success score for total number of trajectories $\in \{100, 200, 300, 400\}$ sampled from different numbers of scenes in the PB-Kitchen-Easy task. When trained with very few trajectories (e.g., 100 or 200 trajectories), the model performance decreases as the number of training scenes increases, due to the tradeoff between learning scene-specific features versus generalizable features. However, we observe a positive correlation between the number of training scenes and the success rate when training with more trajectories, e.g., 500 trajectories.

influences the dynamics and rewards prediction accuracy of the world model. Notably, even with only 200 training trajectories, WoMAP outperforms the VLM and DP baselines by about 1033% and 17%, respectively, (see Figure 6), showing its remarkable data efficiency.

Figure 21 shows the performance of WoMAP as we vary the scene diversity while keeping the total number of trajectories fixed. Overall, we do not observe a strong correlation between the number of training scenes and the success score. However, when training with fewer trajectories (e.g., 100 or 200 trajectories), the performance of the models decreases with the number of scenes. This finding may be attributed to the tradeoff between learning more specialized (scene-specific) features versus more generalizable features across multiple scenes, with a tight budget on the number of training trajectories. In contrast, when training with 500 trajectories, we find that the success rate improves as the number of scene increases, resulting in much higher success rates.

B.4.2 Full Experiment Results

We provide the full results for the experiments used to compute the values in Figure 6 and 7 in Table 7. Each task under a given environment and initialization difficulty is performed under 50 independent trials with randomized scene configuration and initial position.

B.4.3 Semantic Generalization Experiment Results

As discussed in Section 4.5, we evaluate WoMAP's semantic zero-shot generalization to novel tasks. We quantify the semantic similarity between different tasks using the cosine similarity between language embeddings computed from the task instructions by Sentence-Bert [48]. For seen target objects, we vary the task instructions to capture the diversity of possible user descriptions, e.g., asking WoMAP to find a "sweet thing" or a "yellow fruit" with the goal of locating a banana. We utilize partial success scores to better evaluate the degradation in performance with more semantically dissimilar target object queries, where the success scores is linearly scaled based on the bounding-box proportion and the detection confidence. From Figure 9, we find that WoMAP achieves strong generalization with a correlation coefficient between 0.6 and 0.71 across the three object categories. Likewise in Figure 9, we observe that WoMAP generalizes well to unseen target objects. For example, WoMAP localizes a pair of "pliers" and a "hammer," which WoMAP was not trained on. We attribute the strong generalization performance of WoMAP to the generalizable semantic features captured in WoMAP's latent space.

B.4.4 Additional Discussion

Here, we provide additional discussion for the main results in Section 4.

World Model-Heuristics (WM-HR) Baseline: We use the set of candidate actions provided to the VLM as the heuristic actions in the WM-HR baseline. This clever set of actions enabled the WM-HR to perform similarly to WoMAP, since the world model performed well at evaluating these candidate actions and selecting the most promising one. However, the WM-HR baseline lacks any high-level intelligence, posing a limitation in practical situations, since a brute-force approach would not scale, in general.

Sim-to-Real Performance for VLM: We observe that the performance of the VLM drops significantly when moving from simulation to the real-world. This drop in performance is in part due to the limitations in directly executing the VLM's action proposals on the physical robot. We find that the robot reaches its range limits often when executing the VLM action proposals, which effectively ends the experiments, leading to an increase in the failure of the VLM. Further, the VLM occasionally struggled to fully approach the object in the real-world and often stopped a good distance away from the object.

B.4.5 Extension: Non-tabletop Scenes

Despite using tabletop scenes for straightforward, standardized benchmarking in this paper, we also illustrate that our framework works for more general environments. We provide a video in the supplementary material, demonstrating WoMAP on a larger scale, particularly in a living room.

Table 7: Full Experiment Results.

IC: Easy	PB-Office		PB-Kitchen-Easy		PB-Kitchen-Medium		PB-Kitchen-Hard	
Planner	Success	Efficiency	Success	Efficiency	Success	Efficiency	Success	Efficiency
VLM	0.30	0.28	0.06	0.05	0.00	0.00	0.02	0.02
DP	0.28	0.25	0.28	0.25	0.22	0.19	0.16	0.15
WM-CEM	0.30	0.24	0.18	0.13	0.10	0.08	0.16	0.13
WM-Grad	0.60	0.41	0.78	0.50	0.66	0.41	0.66	0.46
WM-HR	0.74	0.63	0.74	0.61	0.68	0.56	0.62	0.54
WoMAP	0.88	0.73	0.90	0.72	0.78	0.60	0.80	0.68

IC: Medium	PB-Office		PB-Kitchen-Easy		PB-Kitchen-Medium		PB-Kitchen-Hard	
Planner	Success	Efficiency	Success	Efficiency	Success	Efficiency	Success	Efficiency
VLM	0.24	0.22	0.04	0.04	0.04	0.04	0.02	0.02
DP	0.42	0.36	0.26	0.23	0.28	0.25	0.08	0.07
WM-CEM	0.26	0.22	0.24	0.17	0.24	0.18	0.16	0.11
WM-Grad	0.58	0.39	0.48	0.30	0.52	0.34	0.46	0.32
WM-HR	0.74	0.61	0.66	0.52	0.50	0.41	0.44	0.38
WoMAP	0.70	0.55	0.82	0.63	0.64	0.52	0.50	0.41

IC: Hard	PB-Office		PB-Kitchen-Easy		PB-Kitchen-Medium		PB-Kitchen-Hard	
Planner	Success	Efficiency	Success	Efficiency	Success	Efficiency	Success	Efficiency
VLM	0.08	0.05	0.00	0.00	0.00	0.00	0.00	0.00
DP	0.14	0.10	0.32	0.28	0.20	0.18	0.18	0.16
WM-CEM	0.06	0.04	0.02	0.01	0.02	0.02	0.02	0.02
WM-Grad	0.24	0.11	0.18	0.11	0.16	0.09	0.16	0.09
WM-HR	0.38	0.26	0.24	0.19	0.24	0.19	0.22	0.19
WoMAP	0.58	0.39	0.34	0.25	0.40	0.32	0.34	0.28

IC: Easy	GS-Kitchen		GS-Office		GS-Random	
Planner	Success	Efficiency	Success	Efficiency	Success	Efficiency
VLM	0.12	0.11	0.14	0.13	0.08	0.07
DP	0.50	0.47	0.54	0.49	0.28	0.26
WM-CEM	0.58	0.48	0.40	0.35	0.40	0.30
WM-Grad	0.84	0.67	0.86	0.63	0.76	0.56
WM-HR	0.78	0.66	0.96	0.83	0.58	0.49
WoMAP	0.86	0.74	0.96	0.77	0.72	0.59

IC: Medium	GS-Kitchen		GS-Office		GS-Random	
Planner	Success	Efficiency	Success	Efficiency	Success	Efficiency
VLM	0.06	0.06	0.06	0.05	0.02	0.01
DP	0.42	0.36	0.32	0.29	0.12	0.11
WM-CEM	0.26	0.20	0.22	0.18	0.20	0.13
WM-Grad	0.50	0.32	0.50	0.30	0.36	0.23
WM-HR	0.56	0.45	0.66	0.56	0.30	0.23
WoMAP	0.68	0.55	0.72	0.55	0.46	0.36

IC: Hard	GS-Kitchen		GS-Office		GS-Random	
Planner	Success	Efficiency	Success	Efficiency	Success	Efficiency
VLM	0.00	0.00	0.02	0.02	0.00	0.00
DP	0.18	0.16	0.14	0.11	0.08	0.06
WM-CEM	0.08	0.06	0.08	0.05	0.02	0.01
WM-Grad	0.14	0.09	0.22	0.11	0.00	0.00
WM-HR	0.18	0.15	0.32	0.25	0.08	0.07
WoMAP	0.24	0.18	0.48	0.34	0.12	0.08