Learning Object-centric Local Navigation of Quadruped Robot from RGB Observations

A THESIS

SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL

OF THE UNIVERSITY OF MINNESOTA

BY

Fidan Mahmudova

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE

OF MASTER OF SCIENCE

KARTHIK DESINGH

August, 2025

Acknowledgements

I am deeply grateful to my advisor, Professor Karthik Desingh, for his exceptional mentorship throughout this research. His insightful guidance and genuine enthusiasm for robotics have been transformative. Professor Desingh's patience in helping us navigate complex challenges and his belief in this project kept us motivated through every obstacle. I am fortunate to have learned from such an inspiring mentor.

I would also like to thank Tzu-Hsien Lee, my research partner and collaborator. He contributed the automated data collection system using GraphNav and the action decoder used in this project. Throughout the research, we worked closely to refine the model architecture and conduct rollout experiments. His technical insight, dedication, and collaborative spirit greatly enriched the development process.

I am grateful to the RPM Lab members for their thoughtful feedback during presentations and discussions, which helped us to consistently improve our work.

I would also like to express my sincere gratitude to my Government Scholarship Program for supporting my studies through a scholarship. Their financial support made it possible for me to pursue this research and focus wholeheartedly on my academic journey.

Finally, thanks to the Minnesota Robotics Institute for providing the resources that enabled this research.

Abstract

Mobile manipulation requires seamless integration of navigation and manipulation capabilities, yet existing systems struggle with the transition from meter-level navigation to centimeter-level precision needed for manipulation tasks. This thesis tackles the "last-meter problem" by addressing this gap between navigation and manipulation systems.

Existing approaches use 3D model informed pose estimation from depth observations, confining generalization to predefined indoor objects. To remove this prerequisite and enable generalization beyond training datasets, we propose an RGB-only imitation learning framework for mobile manipulator robots such as the Boston Dynamics SPOT.

This thesis proposes an encoder-decoder architecture with cross-attention that processes visual observations from SPOT's five onboard cameras alongside natural language queries, eliminating the need for depth sensors, object models, and indoor maps. We compare three vision encoders (ResNet18, DINOv2, and Grounded SAM2) and demonstrate that only Grounded SAM2, with explicit object segmentation, successfully generalizes to new object positions (98%) and unseen environments (91%) while maintaining $\leq$30cm translational and $\leq \pm 10^{\circ}$ rotational accuracy. ResNet18 and DINOv2 fail completely, revealing they memorize spatial layouts rather than learning object-centric navigation policies. Category-level generalization remains challenging for Grounded SAM2 (48% success). This thesis work demonstrates that explicit semantic grounding is beneficial for robust object-centric local navigation in real-world environments where precise positioning enables successful manipulation.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Enabling robots to navigate effectively in real-world environments is essential for their integration into practical applications, particularly for tasks requiring physical manipulation. This thesis explores learning object-centric local navigation from RGB demonstrations, an approach that teaches robots to reach manipulation-ready poses with respect to a target object by imitating expert trajectories. Unlike traditional navigation that relies on metric coordinates, object-centric navigation grounds robot navigation decisions to its semantic understanding of the target object. The "local" aspect addresses the final meter of approach where centimeter-level precision becomes critical for achieving manipulation-ready positioning.

Recent works in semantic navigation, exemplified by Vision-Language Frontier Maps (VLFM) [1], demonstrate that robots can efficiently navigate to target objects using natural language commands, achieving state-of-the-art performance on standard benchmarks. However, like most contemporary navigation systems [2, 3, 4], VLFM defines success at meter-level accuracy, meaning episodes are considered complete when the robot stops within 1 meter of the target object.

While this represents significant progress in long-range navigation, this precision threshold creates a fundamental limitation for manipulation tasks. As demonstrated in Figure 1.1a, these approaches can successfully guide robots to target locations within the required 1 meter threshold. However, this 1 meter precision is insufficient for manipulation execution, which requires signif-

icantly higher spatial and orientational accuracy. The robot, despite navigation success, remains positioned at an inadequate distance and orientation for actual object interaction. Figure 1.1b contrasts this by showing the manipulation-ready configuration, where the robot achieves necessary positioning for effective appliance interaction. This gap between navigation success and manipulation readiness exemplifies the critical last meter problem that current navigation frameworks fail to address.

(a) Navigation Success: Robot positioned within 1 meter threshold but inadequate for manipulation

(b) Manipulation-Ready Positioning: Robot precisely positioned for kitchen appliance interaction

Figure 1.1: The Last Meter Problem: Navigation Success vs Manipulation Readiness.

We define local navigation as the final precision maneuver within 1 meter that positions a mobile manipulator at an object-relative pose suitable for physical interaction. This work focuses exclusively on achieving this positioning accuracy; we do not perform manipulation but navigate to a target pose that enables successful downstream manipulation tasks.

To meet this objective, we introduce an object-centric local navigation system for the Boston Dynamics SPOT robot. Our navigation objective requires positioning the robot directly in front of the target object (semantically appropriate orientation) to enable effective manipulation execution. Conditioned on current and goal RGB observations, and a natural-language query specifying the target object, our imitation-learning encoder–decoder model consistently reaches ≤ 30 *cm* translational and $\leq \pm 10^{\circ}$ rotational accuracy while achieving this frontal positioning requirement.

In summary, our major contributions are:

1. An encoder-decoder style imitation learning framework for learning a local-navigation policy with RGB-only expert demonstrations, and
2. A quantitative and qualitative analysis of state-of-the-art visual encoders to support the object-centric local-navigation.

The remainder of this paper is organized as follows. Chapter 2 reviews various approaches to solve the local navigation problem, highlighting their advantages and limitations to motivate our RGB-only, object-centric approach. Chapter 3 presents our IL framework and automated expert data collection pipeline. Chapter 4 discusses experiments , while Chapter 5 provides results and visualization insights for failures. Chapters 6 concludes with limitations and future work.

Chapter 2

Related Work

In this chapter we review relevant prior work, organized into four main areas: 1) Classical Visual Servoing, 2) Hybrid Visual Servoing methods, 3) Vision-Language Models, and 4) Learning-Based Navigation. We highlight the strengths and limitations of each approach to motivate our proposed system.

2.1 Classical Visual Servoing

Among the most widely explored approaches to vision-based navigation is classical visual servoing, a well-established method that encompasses two primary strategies: Image-Based Visual Servoing (IBVS) and Position-Based Visual Servoing (PBVS) [5]. Both approaches have demonstrated effectiveness in controlled laboratory settings [6, 7] and serve as a theoretical foundation for vision-based control.

However, classical visual servoing has fundamental limitations that preclude its adoption in local navigation problem. PBVS requires accurate 3D pose or scene geometry, which is an impractical constraint in real-world deployment scenarios involving diverse objects. IBVS, while avoiding explicit 3D modeling, suffers from the non-convex error surface. This property can trap the optimization process in local minima, causing the control system to converge on a suboptimal

solution that remains significantly misaligned with the target [8].

2.2 Hybrid Visual Servoing

Recent approaches have attempted to address the limitations of classical approaches by integrating learning-based components with traditional visual servoing frameworks [9, 10, 11]. Hybrid methods aim to preserve the theoretical foundations of classical approaches while enhancing robustness through data-driven components. One such method incorporates a Neural Radiance Fields (NeRF) into the visual servoing pipeline addressing limitations in 3D modeling by replacing hand-crafted depth priors with NeRF-based geometry [10]. While this expands starting viewpoints and boosts IBVS accuracy, the authors explicitly acknowledge it cannot run in real time due to slow NeRF rendering.

Another notable approach, Imagine2Servo, leverages diffusion-based image generation to iteratively produce intermediate goal images that guide classical IBVS toward task completion [12]. This method eliminates the need for predefined goal images by diffusing sub-goal images. However, each sub-goal generation requires a full diffusion pass via InstructPix2Pix, which typically takes roughly 9 seconds on an A100 GPU [13], thereby limiting its feasibility for real-time deployment.

These approaches reveal that while hybrid techniques successfully address specific classical approach limitations, they also introduce new computational constraints that render them impractical for our application.

2.3 Vision-Language Models

Vision-Language Models represent a significant paradigm shift for object-goal navigation, moving beyond geometric constraints to leverage large-scale pre-training for understanding both visual scenes and natural language instructions [14, 15].

Mem2ego [16] represents a significant advance in VLM navigation precision, achieving sub-20 cm navigation accuracy. However, this accuracy applies to any relative pose of the robot with respect to the target object, rather than the precise positioning our application requires. Moreover, the system operates entirely in simulation and relies on per-step GPT-4o reasoning over a global memory module. These design choices suggest the method would not meet real-time, on-board deployment constraints.

In contrast, VLFM [1] addresses the sim-to-real gap by demonstrating successful deployment on Boston Dynamics SPOT robots in real environments. The system constructs occupancy and language-grounded value maps through frontier exploration using BLIP-2 and other perception modules. Despite its achievement, the method still operates with a 1m success radius and needs full environmental maps throughout navigation, creating infrastructure dependencies beyond our lightweight navigation requirements.

While both approaches showcase VLM capabilities, their design targets comprehensive environment exploration. Since we focus on lightweight precise local navigation to nearby visible targets, we sought alternatives that maintain semantic understanding without the requirement of frontier exploration and continuous mapping.

2.4 Learning-based Navigation

End-to-end learning-based approaches have emerged as an alternative paradigm for local navigation, attempting to overcome classical method brittleness through data-driven policy learning.

2.4.1 Reinforcement Learning Methods

Deep reinforcement learning methods have shown promise in learning visual navigation policies without explicit geometric representations [4, 17]. Li and Košecka [18] demonstrate visual servoing through dense correspondences and adopt a strict 0.2 m pose-success threshold, directly align-

ing with our objective of centimeter-level, pose-aware local navigation. However, these methods require extensive simulation rollouts with carefully shaped reward functions that exacerbate the sim-to-real transfer challenge.

2.4.2 Imitation Learning Methods

Imitation Learning (IL) presents a compelling alternative to traditional reinforcement learning by enabling agents to learn directly from expert demonstrations, bypassing the complexity of reward function design. It has been repeatedly employed to train navigation policies [19, 20].

A particularly relevant example is the work by Meng et al. [21], which addresses the same challenge of high-precision, object-centric local navigation using multi-modal perception. Their approach employs RGB-D cameras and 2D LiDAR with a transformer-based architecture trained on 500k simulated trajectories and achieves centimeter-level accuracy. Notably, their model can reason about object approach angles and orientations, relying on reference images annotated with object masks and parametric pose descriptors. In contrast, our method seeks to teach consistent, front-facing approach behavior through demonstrations, using only RGB inputs and real-world data on a quadruped robot with details provided in Section 3.

Chapter 3

Methodology

3.1 Problem Formulation

As we address the challenge of last-meter, object-centric local navigation, our problem scope presumes successfully completed global path planning and focuses specifically on the final approach phase that positions the robot in a manipulation-ready pose relative to the target object. While doing this, system relies only on RGB perception from SPOT's five onboard cameras, with no requirement of depth, 3D models, or access to global pose. This reflects practical scenarios where additional sensor modalities may be unavailable, unreliable, or cost-prohibitive.

Assumptions. Our approach operates under key assumptions that define the problem scope. Since we focus on the final meter of navigation after global path planning has already positioned the robot near the target, we assume target objects remain within the robot's field of view throughout the local navigation phase. Additionally, as we focus on precision positioning rather than obstacle-rich path planning, we assume scenes are free of occluding obstacles along the approach path, which isolates the last-meter alignment problem from broader navigation challenges.

Goal Conditioning. We encode navigation objectives through dual specification: goal images that implicitly demonstrate desired spatial configurations, and natural language queries that explicitly identify target objects for Grounded SAM2 grounding. For our specific task, we train the model to navigate and position itself directly in front of the target object at a precise pose. Through exposure to expert demonstrations, the policy implicitly learns this precise frontal positioning by observing trajectories that consistently terminate when current observations align with goal observations, while the natural language component specifies the semantic target to approach.

3.2 Imitation Learning Framework

We define this last-meter navigation problem as behavioral cloning (BC), so that a policy π learns:

$$\pi : (O^t_{\text{curr}},\, O_{\text{goal}},\, L) \rightarrow A \tag{3.1}$$

where $O^t_{\text{curr}} = \{I^t_{\text{front}}, I^t_{\text{back}}, I^t_{\text{left}}, I^t_{\text{right}}\}$ represents the current multi-view RGB observations at time t, captured by SPOT's onboard cameras. The front view I^t_{front} is constructed by stitching SPOT's two front-facing stereo camera images. The goal observations $O_{\text{goal}} = \{I^{\text{goal}}_{\text{front}}, I^{\text{goal}}_{\text{back}}, I^{\text{goal}}_{\text{left}}, I^{\text{goal}}_{\text{right}}\}$ are the RGB views captured at the desired terminal pose, representing the target visual configuration for manipulation-ready positioning. The natural-language prompt L here specifies the target object (e.g., "green chair") and is processed by our Grounded SAM2 [22], [23] pipeline for visual grounding. Finally, A denotes the discrete action space for robot navigation commands.

Following the BC paradigm, we optimize policy parameters by minimizing the negative log-likelihood over expert demonstrations (D) :

$$\mathscr{L}(\theta) = -E_{(O^t_{\text{curr}}, O_{\text{goal}}, L, A^*)\sim D}\left[\log\, \pi_\theta\left(A^* \mid O^t_{\text{curr}}, O_{\text{goal}}, L\right)\right]$$

where A^* is the expert action. The expert demonstrations are collected using Boston Dynamics'

GraphNav system (see Section3.4).

3.3 Model

Our approach employs an end-to-end Imitation Learning (IL) framework with an encoder-attention-decoder architecture that extends traditional encoder-decoder paradigms by incorporating attention mechanisms [24] between encoding and decoding stages. This design enables goal-conditioned navigation through feature correspondence learning between current observations and target specifications. The architecture as shown in Figure 3.1 comprises: (1) vision encoder that extracts features from current and goal observations (with an optional text conditioning), (2) a bi-directional cross-attention module to establish spatial correspondences, and (3) an action decoder that maps attended features to navigation predictions.

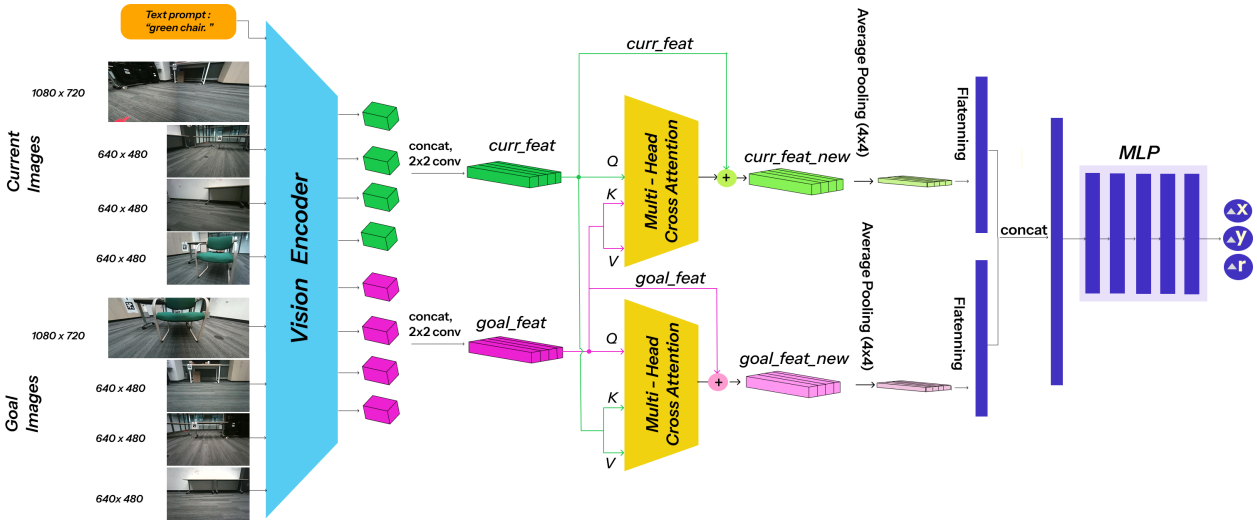


Figure 3.1: **Model Architecture :** Current and goal observations from four camera views (front, left, right, back) are independently processed through a shared vision encoder. Features are concatenated and fed to bidirectional multi-head cross-attention that learns spatial correspondences in both directions (from current-to-goal and from goal-to-current). The enhanced features from both branches undergo global average pooling, flattening, and concatenation before a 5-layer MLP decoder predicts displacement (Δx, Δy, Δr).

Panoramic Visual Representation.

We process four camera views (front, left, right, back) separately through a shared vision encoder. After encoding, we concatenate the resulting features along the spatial dimension to create a unified panoramic representation with 360° spatial coverage.

The design rationale behind this panoramic representation is to provide complete directional awareness for navigation decision-making. This mechanism allows the model to associate target objects appearing in specific spatial regions with corresponding directional responses. For example, a target object appearing in the right camera view could inform rightward navigation decisions.

3.3.1 Vision Encoders

We investigate three distinct vision encoder architectures which represent different paradigms in visual representation learning to evaluate their effectiveness for extracting meaningful features for our navigation task. ResNet18 [25] serves as our lightweight convolutional baseline, where we leverage a pre-trained ResNet18 backbone with ImageNet weights and remove the final classification layers, extracting 512-dimensional spatial feature maps. DINOv2 [26] represents our transformer-based vision encoder, utilizing the frozen DINOv2-ViT-Small model which extracts 384-dimensional patch token representations that capture rich semantic relationships learned through visual self-supervision. Finally, Grounded SAM2 [22], [23] serves as our object-centric encoder, which incorporates textual goal conditioning alongside visual processing, where the integration of language grounding allows the model to learn navigation cues that are directly relevant to the specified target object. We discuss the specific implementation of our two-stage Grounded SAM2 pipeline in greater detail next in Section 3.3.4.

3.3.2 Cross-Attention Mechanism

The cross-attention module learns spatial correspondences between current observations and goal states, enabling the model to identify which current scene elements are most relevant for navigation toward the target configuration.

Given current features and goal features $\mathbf{F}_{\text{curr}}$, $\mathbf{F}_{\text{goal}} \in R^{C\times H\times W}$, we flatten spatial dimensions to create sequences of $T = H \times W$ tokens of dimension C. We then pass these tokens into PyTorch's built-in `nn.MultiheadAttention` (MHA), which applies scaled dot-product attention:

$$\text{softmax}\left(\frac{\mathbf{QK}^\top}{\sqrt{d_k}}\right)\mathbf{V},$$

and returns attention outputs and weights.

Unidirectional cross-attention mechanism. We designate current features as *queries* and goal features as both *keys* and *values*: $\mathbf{A}_{\text{curr},\_} = \texttt{MHA}(\mathbf{F}_{\text{curr}}, \mathbf{F}_{\text{goal}}, \mathbf{F}_{\text{goal}})$ This configuration allows each current observation location to attend to relevant goal features, producing attention maps that highlight goal-relevant regions in the current scene.

Bidirectional cross-attention mechanism. To enable mutual alignment, we also run the symmetric operation: $\mathbf{A}_{\text{goal},\_} = \texttt{MHA}(\mathbf{F}_{\text{goal}}, \mathbf{F}_{\text{curr}}, \mathbf{F}_{\text{curr}})$. This allows goal features to attend back to current context, enhancing bidirectional correspondence. Attended features integrate with input features via residual connections: $\mathbf{F}^{\text{curr}}_{\text{out}} = \mathbf{F}_{\text{curr}} + \mathbf{A}_{\text{curr}}$, $\mathbf{F}^{\text{goal}}_{\text{out}} = \mathbf{F}_{\text{goal}} + \mathbf{A}_{\text{goal}}$.

We configure MHA with 8 attention heads and evaluate both unidirectional and bidirectional variants, with comparative results presented in Section 5.1 for evaluating the navigation policies.

3.3.3 Action Decoder

Following cross-attention, attended features undergo global average pooling and concatenation before action decoding as shown in architecture Figure 3.1. The decoder employs a 5-layer MLP

with 1024 hidden dimensions to map multi-view visual features to discrete movement predictions.

The decoder outputs three 3-class classification heads : x-direction, y-direction, and rotation movements.

3.3.4 Grounded SAM2 Implementation

To efficiently integrate language-driven object segmentation into our pipeline, we implement a two-stage workflow separating heavy vision computation from lightweight policy learning, avoiding repeated forward passes through expensive visual backbones during training.

Stage 1: Vision Processing We first run Grounding DINO (text threshold 0.45, box threshold 0.4) to detect objects matching textual queries. SAM2 then extracts 256-dimensional embeddings and generates segmentation masks. Element-wise multiplication between embeddings and masks produces object-specific features. During training, these features are cached to avoid redundant computation; during inference, this processing occurs in real-time.

Stage 2: Policy Learning The second stage follows the same cross-attention and decoding pipeline as other encoder variants, using the masked features as input. This two-stage approach distinguishes Grounded SAM2 from ResNet18 and DINOv2 approaches, where feature extraction and policy learning occur jointly due to their computational efficiency.

3.4 Expert Trajectory Collection

The success of our imitation learning model depends heavily on the quality of the demonstration data. Instead of using human teleoperation demonstrations, which can include suboptimal actions, we opted for automated expert trajectories. We developed a system that leverages GraphNav [27], Boston Dynamics' navigation framework for the SPOT robot, to produce consistent and high-quality demonstrations that can be reliably reproduced.

The system localizes using the SPOT's fiducial markers included in the environment. After

localization, it navigates between systematically parameterized starting poses and pre-defined goal pose. For each trajectory, the robot navigates toward a green chair (the target object) while it records synchronized visual observations using onboard vision (Figure 3.2) as well as actions. This automated approach produced 715 expert trajectories containing 7,263 image-action pairs.

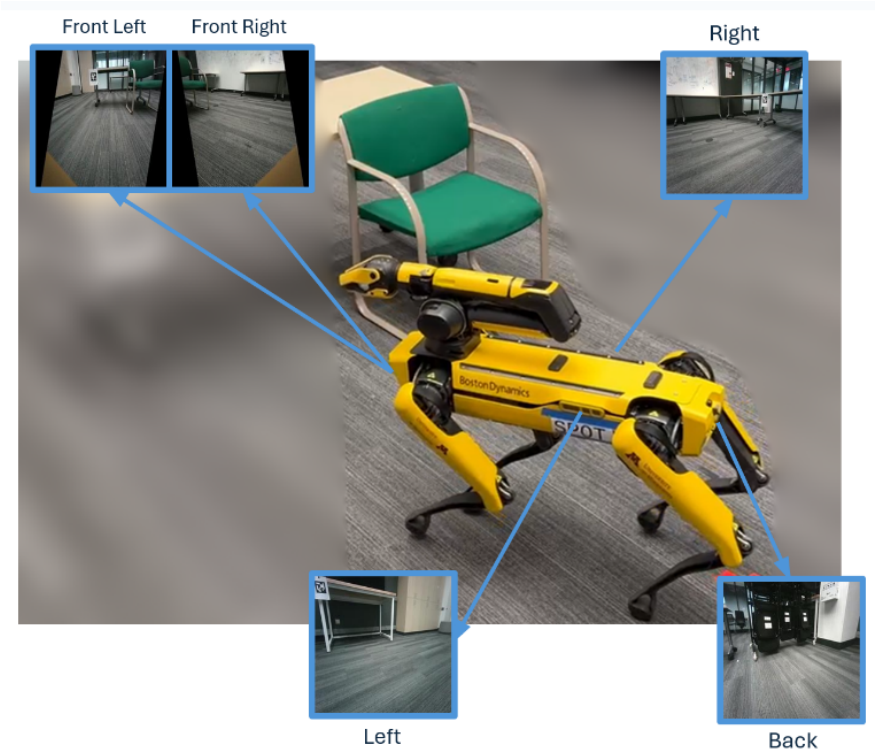


Figure 3.2: SPOT onboard vision.

3.4.1 Environmental Setup

We collected data within a mock-office environment, containing tables, green chair and chair hangers to simulate realistic navigation scenarios (see Figure 3.3). We are referring to this environment as MAP 1 throughout our evaluation.

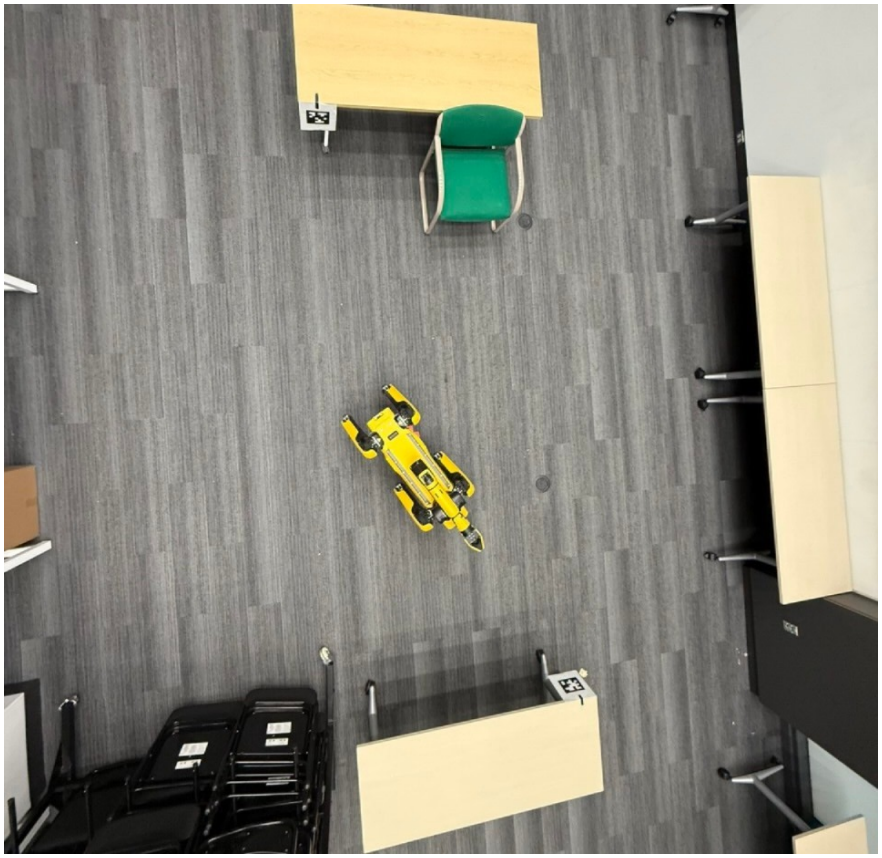


Figure 3.3: Training environment (MAP 1).

3.4.2 Starting Pose Parameterization

We systematically parameterize starting poses around the green chair to create diverse training scenarios using three key variables. **Radial Distance** (R) controls how far the robot starts from the chair, with values $R \in \{0.3, 0.45, 0.6, 0.9, 1.2\}$ meters representing close to distant starting positions. **Approach Angle** (A) determines which side of the chair the robot approaches from, ranging from -90° (left side) to 90° (right side) in 15° increments relative to the chair's front-facing direction. **Starting Orientation** (O) specifies the robot's initial heading direction, ranging from -150° to 150° in 30° increments, allowing the robot to start facing various directions regardless of its position.

Figure 3.4 visualizes the resulting training point distribution around the target object.

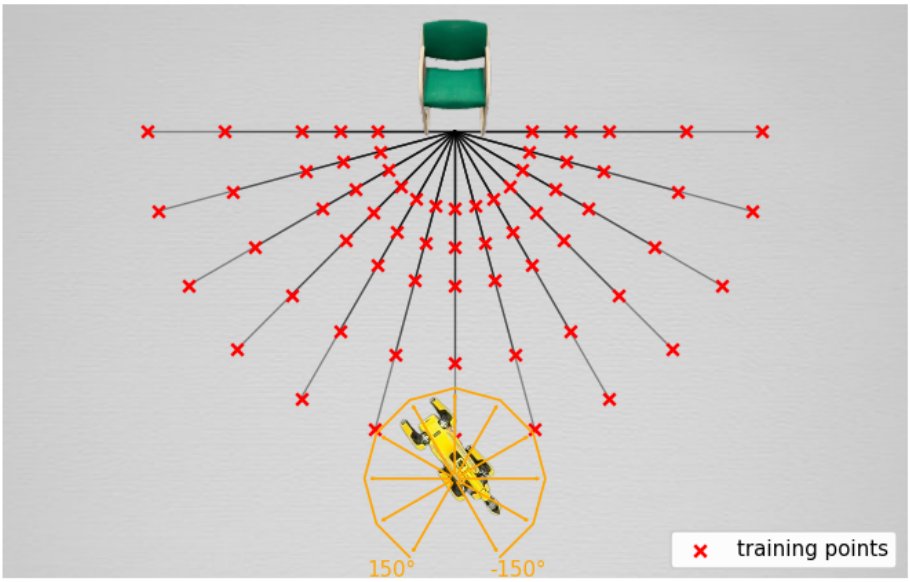


Figure 3.4: Visualization of training points. Orange lines indicate the robot's starting orientations.

3.4.3 Action Space and Termination

Our navigation system implements a discrete action space consisting of three independent control dimensions: longitudinal movement in the x-direction (Δx), lateral movement in the y-direction (Δy), and rotational movement (Δr). For each dimension, the agent can select from three discrete actions: 0 (negative movement), 1 (stop/no movement), and 2 (positive movement). The complete stop command corresponds to $[1,1,1]$, indicating no movement in any dimension.

Defining clear terminal conditions is crucial for goal-conditioned navigation. For this, we manually position the robot at the desired final pose relative to the green chair , capture a goal image that encodes the desired spatial relationship as well as record the final pose achieved. During evaluation, goal pose establishes ground-truth pose for quantitative evaluation.

3.5 Training Details

For each architecture, we train variants with both unidirectional and bidirectional cross-attention to determine which mechanism better supports navigation generalization. Models are trained ex-

clusively on MAP 1 to navigate toward a fixed green chair.

Training proceeded for 1 000 epochs with a batch size of 8. We optimised the networks with the Adam optimiser , using a fixed learning rate of 1×10^{-4} and a categorical cross-entropy loss over the three discrete action heads. All trainings were executed on a single NVIDIA RTX A6000 GPU (48 GB VRAM) running CUDA 12.4 on our lab server.

Chapter 4

Experiments

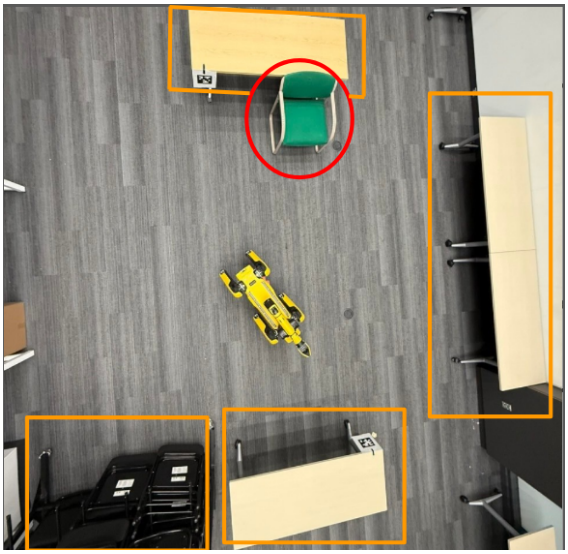
This chapter describes the experimental methodology used to evaluate our navigation approach across multiple test environments. We outline the progressive evaluation scenarios, testing points parametrization, and success metrics used to assess generalization capabilities.

4.1 Test Scenarios

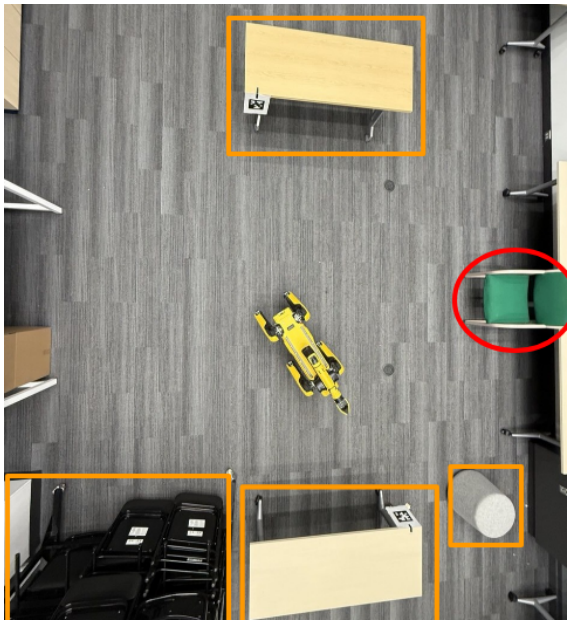
Following training exclusively on **MAP 1** for navigation toward the green chair, we test generalization capabilities of all the models through four progressively challenging scenarios.

In **MAP 2**, we evaluate novel starting configurations in the same training environment, maintaining the same target object and location as MAP 1 (Figure 4.1a). We test robustness to initialization variance by executing 98 rollouts with unseen starting positions and poses (Figure 4.2), evaluating whether models can generalize navigation policies to new approach angles and distances.

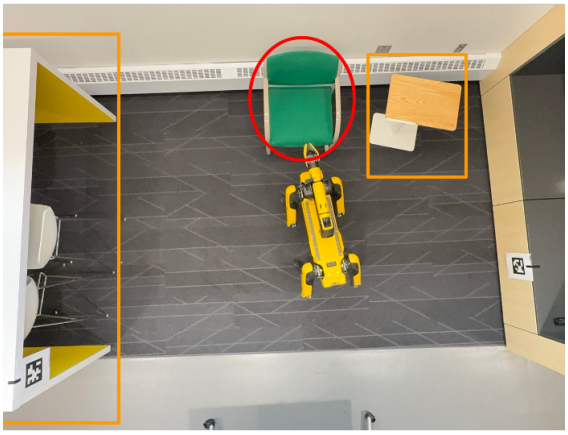
MAP 3 presents spatial relocation challenges, where we relocate the target green chair to a different position within the modified environment (Figure 4.1b), testing whether models truly understand the semantic goal (”green chair”) or memorize spatial coordinates.



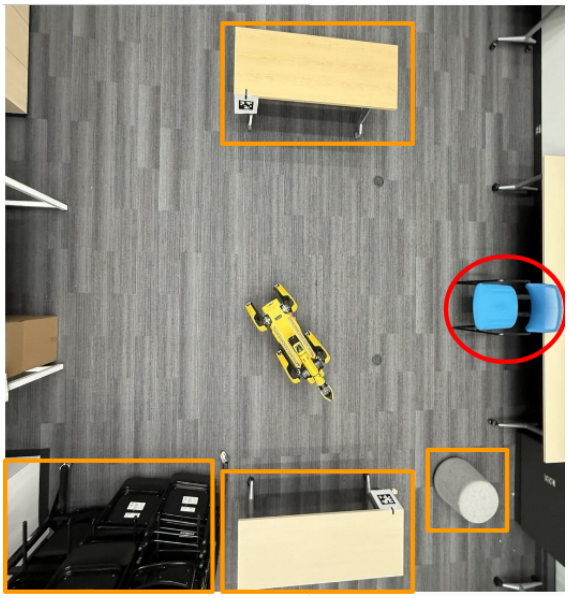
(a) MAP 2 experimental setup.



(b) MAP 3 experimental setup.



(c) MAP 4 experimental setup.



(d) MAP 5 experimental setup.

Figure 4.1: Experimental setups for different maps. Orange boxes highlight the distractors in the environment, while a red circle indicates the target object.

MAP 4 introduces environmental generalization by testing navigation to the same target object (green chair) in a completely new environment with different lighting conditions and layout (Figure 4.1c). This scenario evaluates whether our models have learned robust navigation policies that can

transfer beyond the training environment, or if they have simply memorized visual features specific to the original room configuration.

Finally, **MAP 5** provides the most challenging test of category-level generalization, requiring navigation to a different instance of the same category (blue chair) in a novel environment (Figure 4.1d). After training exclusively on "green chair," models must now navigate to a blue chair, evaluating whether learned navigation strategies can generalize across different instances within the same semantic category.

4.2 Experimental Setup

We conduct 98 rollouts per test map in same mock-office environment using a systematic parameterization approach. Radial distances are sampled at $R \in \{0.5, 1.0\}$ meters from the target, while approach angles span $A \in \{-80°, -50°, -25°, 0°, 25°, 50°, 80°\}$ to test various approach directions. Starting orientations are distributed across $O \in \{-135°, -90°, -45°, 0°, 45°, 90°, 135°\}$ to ensure comprehensive evaluation of initial heading variations. Figure 4.2 visualizes the resulting training and testing point distribution around the target object.

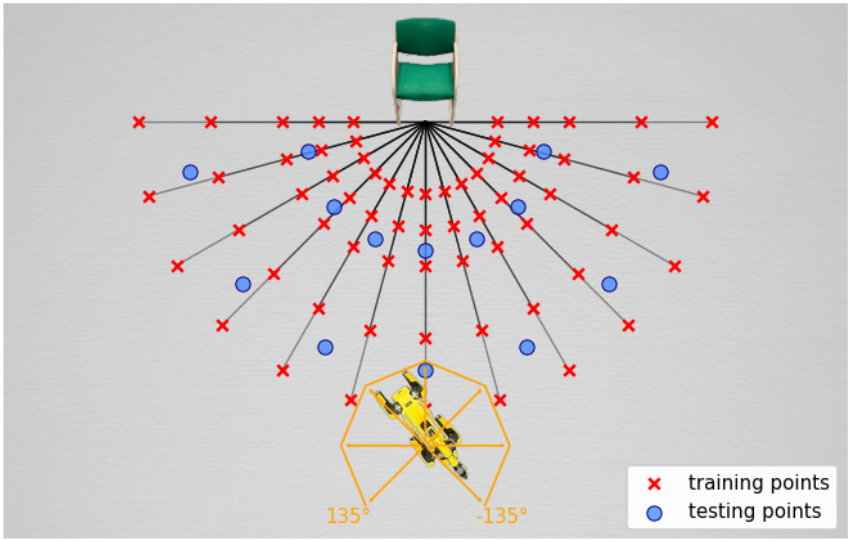


Figure 4.2: Visualization of training and testing points. Orange lines indicate the robot's starting orientations.

4.3 Success Criteria

We define successful manipulation-ready positioning through a dual-criterion evaluation framework combining geometric and semantic assessments.

For geometric evaluation, we measure position and orientation errors relative to the goal pose using two tolerance levels: strict criteria requiring $\leq 20\,\text{cm}$ translation and $\leq \pm 5^\circ$ rotation error, and relaxed criteria allowing $\leq 30\,\text{cm}$ translation and $\leq \pm 10^\circ$ rotation error.

However, since geometric criteria may indicate failure when the robot is still appropriately positioned for manipulation, we complement this with semantic evaluation. Our semantic assessment segments the target object (green chair) using Grounded SAM2 and computes the Center-of-Mass (CoM) shift between goal and achieved images as $d_{\text{CoM}} = \|\mathbf{c}_{\text{goal}} - \mathbf{c}_{\text{achieved}}\|_2$, where $\mathbf{c} = \frac{1}{N}\sum_{(x,y)\in M}(x,y)$ for mask M. We consider positioning successful when $d_{\text{CoM}} \leq 90$ pixels, a threshold determined through visual inspection of goal-achieved image pairs to identify the maximum acceptable displacement for successful manipulation.

Figure 4.3 demonstrates this evaluation approach, showing CoM results for both goal (left) and achieved (right) configurations with a shift of 83 pixels, indicating successful positioning.

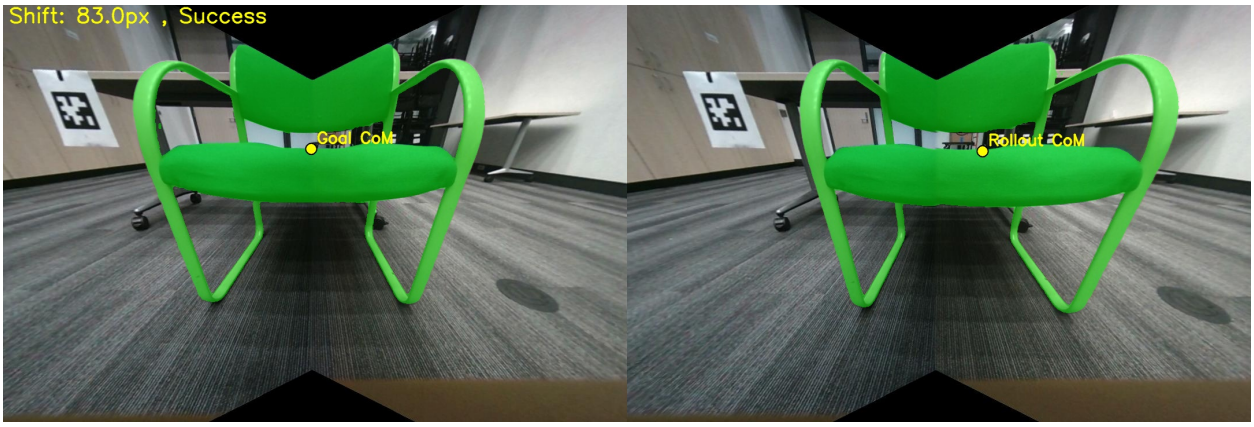


Figure 4.3: Semantic alignment evaluation. **Left:** Goal image with marked Center-of-Mass (CoM). **Right:** Achieved terminal view with marked Center-of-Mass (CoM).

Chapter 5

Results

This chapter presents the performance outcomes from our experimental evaluation across different navigation scenarios. We report quantitative results and provide analysis to understand the underlying factors driving success and failure modes.

5.1 Evaluation of Navigation Policies

We evaluate generalization capabilities of all the models through their performance on different maps.

5.1.1 Results on MAP 2 : Novel Starting Configurations

MAP 2 (Figure 4.1a) results evaluate model robustness to novel starting configurations within the same environment layout.

Table 5.1: MAP 2 rollout results

Model	Success Criteria	Uni-Attn	Bi-Attn
ResNet18-MLP	T–20 cm, R–5°	66.33%	71.43%
	T–30 cm, R–10°	100.00%	98.98%
	CoM	91.84%	90.82%
DINOv2-MLP	T–20 cm, R–5°	39.80%	89.80%
	T–30 cm, R–10°	92.86%	100.00%
	CoM	100.00%	72.45%
Grounded SAM2-MLP	T–20 cm, R–5°	89.80%	71.43%
	T–30 cm, R–10°	95.92%	93.88%
	CoM	94.9 %	84.69%

Table 5.1 reveals strong trajectory generalization across all architectures (92.86-100% success at relaxed threshold) on MAP 2. However, the choice between unidirectional and bidirectional attention proves to be architecture-dependent.

Geometric Evaluation Results. Grounded SAM2-MLP excels with unidirectional attention (89.80% vs 71.43%). This performance difference can be attributed to the fact that the Grounding DINO already provides explicit object segmentation masks that highlight the target chair in each frame and bidirectional attention introduces unnecessary complexity that disrupts this direct mask-to-action mapping.

Conversely, ResNet18-MLP and DINOv2-MLP benefit from bidirectional attention mechanisms. Unlike Grounded SAM2, these architectures must learn implicit feature correspondences between raw observations and goal images. Bidirectional attention enables this cross-modal alignment compensating for the absence of explicit target object localization.

Semantic Evaluation Results As illustrated in Table 5.1, the CoM and pose evaluation metrics show consistent alignment for Grounded SAM2 and ResNet18, where both favor the same attention

direction or show minimal differences. However, DINOv2 reveals a critical contradiction: bidirectional attention achieves higher pose success rates (89.80% vs 39.80%) but worse spatial precision (72.45% vs 100% CoM), suggesting it reaches the goal vicinity more frequently but with scattered endpoint positioning, while unidirectional attention delivers perfect centering (Figure 5.1b) when successful but fails more often at strict pose thresholds. ResNet18 maintains nearly identical tight clustering patterns for both approaches (91.84 % vs 90.82 %), suggesting its simpler features are equally effective regardless of attention complexity (Figure 5.1a).

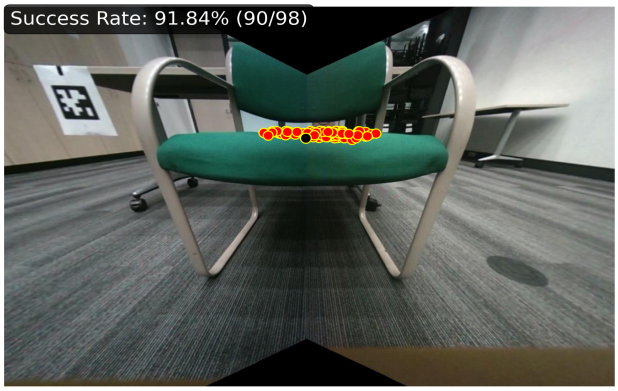
5.1.2 Results on MAP 3: Spatial Relocation

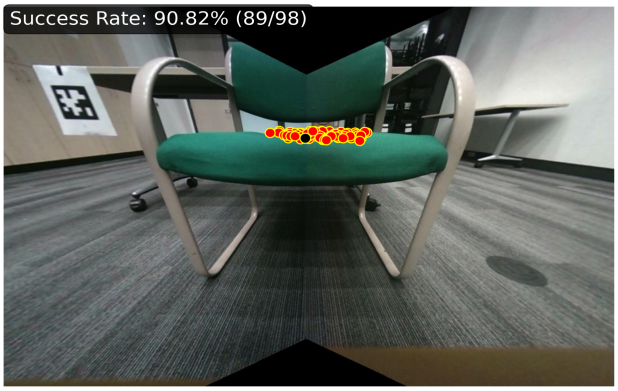
We now present MAP 3 (Figure 4.1b) results, where we relocate the target chair to test semantic understanding and generalization to new spatial locations.

Table 5.2 shows complete failure for ResNet18-MLP and DINOv2-MLP (0% success on all metrics) on MAP 3, when the target chair is moved to a new location.

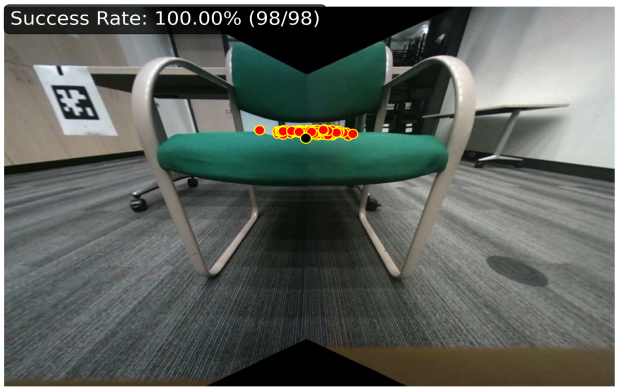
The two architectures fail in different ways. ResNet18-MLP repeatedly navigates the robot to the chair's original training location despite its absence. DINOv2-MLP, on the other hand, explores the environment without clear direction. Our visualization analysis for internal representations (Section 5.2) explains this failure mode, revealing attention for these models being concentrated on floor coordinates rather than semantic objects.

In contrast, Grounded SAM2-MLP maintains high performance (97.96% at relaxed threshold) as its object detection component identifies chairs based on the provided prompt regardless of location. The performance drop with bidirectional attention (97.96% to 59.18%) suggests that additional processing complexity interferes with the direct object-following behavior that proves to be the most effective.

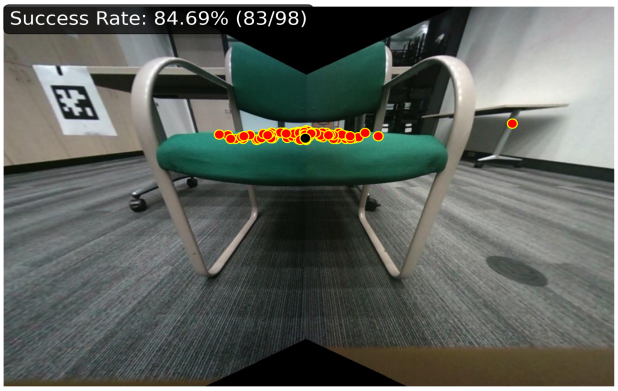




(a) ResNet18



(b) DINOv2



(c) Grounded SAM2-MLP

Figure 5.1: Center-of-Mass (CoM) points distribution for MAP 2 test cases across three models. **Red points** denote predicted CoM locations, while the **black point** marks the ground-truth CoM from the goal image. **Left:** Models with uni-directional attention. **Right:** Models with bi-directional attention.

Table 5.2: MAP 3 rollout results

Model	Success Criteria	Uni-Attn	Bi-Attn
ResNet18-MLP	T–20 cm, R–5°	0 %	0 %
	T–30 cm, R–10°	0 %	0 %
	CoM	0 %	0 %
DINOv2-MLP	T–20 cm, R–5°	0 %	0 %
	T–30 cm, R–10°	0 %	0 %
	CoM	0 %	0 %
Grounded SAM2-MLP	T–20 cm, R–5°	67.35%	32.65%
	T–30 cm, R–10°	97.96%	59.18%
	CoM	95.92%	79.59%

5.1.3 Results on MAP 4 : Unseen Environment

In MAP 4 (Figure 4.1c), we evaluate environmental generalization by testing navigation to the same target object (green chair) in a novel environment with different lighting conditions and layout.

Table 5.3 reveals complete failure for ResNet18-MLP and DINOv2-MLP (0% success across all metrics) because these models learn environment-specific visual features as discussed previously and thereby cannot transfer to new settings.

Grounded SAM2-MLP demonstrates strong environmental generalization with unidirectional attention (90.82% at relaxed threshold, 95.92% CoM), successfully transferring to an unseen environment where other models fail completely. Its main strength lies in object segmentation which isolates the target chair from background distractors and layout variations. While lighting changes affect the visual appearance of the segmented chair features themselves, causing minor performance degradation, the model maintains robust navigation capabilities. This represents effective environmental transfer, with the object-centric approach proving useful for generalization. Bidirectional attention fails dramatically (2.04%), as the additional complexity of cross-modal reasoning

in both directions interferes with navigation.

Table 5.3: MAP 4 rollout results

Model	Success Criteria	Uni-Attn	Bi-Attn
ResNet18-MLP	T–20 cm, R–5°	0 %	0 %
	T–30 cm, R–10°	0 %	0 %
	CoM	0 %	0 %
DINOv2-MLP	T–20 cm, R–5°	0 %	0 %
	T–30 cm, R–10°	0 %	0 %
	CoM	0 %	0 %
Grounded SAM2-MLP	T–20 cm, R–5°	41.84%	0 %
	T–30 cm, R–10°	90.82%	2.04%
	CoM	95.92%	25.51%

5.1.4 Results on MAP 5: Category-Level Generalization

We present MAP 5 (Figure 4.1d) results, where models must navigate to a different chair instance (blue chair) to test category-level generalization.

Table 5.4 shows complete failure for ResNet18-MLP and DINOv2-MLP (0% success), while Grounded SAM2-MLP demonstrates partial category-level understanding with unidirectional attention (47.96% at relaxed threshold, 64.29% CoM). Bidirectional attention fails dramatically (6.12%), as cross-modal complexity worsens with unfamiliar object appearances. The object segmentation property of Grounded SAM2-MLP model enables some category transfer capability, however robust generalization across different chair instances remains challenging.

Table 5.3

Table 5.4: MAP 5 rollout results

Model	Success Criteria	Uni-Attn	Bi-Attn
ResNet18-MLP	T–20 cm, R–5°	0 %	0 %
	T–30 cm, R–10°	0 %	0 %
	CoM	0 %	0 %
DINOv2-MLP	T–20 cm, R–5°	0 %	0 %
	T–30 cm, R–10°	0 %	0 %
	CoM	0 %	0 %
Grounded SAM2-MLP	T–20 cm, R–5°	22.45%	0 %
	T–30 cm, R–10°	47.96%	6.12%
	CoM	64.29%	19.39%

5.2 Internal Representations Visualization

To better understand the internal representations learned by our navigation models and gain insight into why these architectures fail catastrophically when the target object is relocated in MAP3 (Table 5.2), we employ visualization techniques tailored to each architecture. We examine both pre-attention and post-attention phases to trace spatial representations.

Pre-attention visualizations. To reveal the foundational visual features each architecture extracts before cross-modal reasoning, we employ architecture-specific visualization techniques. For visualizing ResNet18 features, we apply EigenCAM [28] to the final convolutional layer, which reveals spatial regions with highest activation variance (Figure 5.2, middle row). While for DINOv2, we implement attention rollout [29] which aggregates self-attention matrices across transformer layers to trace information flow (Figure 5.3, middle row).

Post-attention visualizations. Post-attention heatmaps visualize the enhanced visual representations after the uni-directional cross-attention mechanism, providing insight into which image

regions become prioritized after goal-conditioned processing.

5.2.1 Visualization Insights

ResNet18 Analysis. The EigenCAM visualization (Figure 5.2, middle row) shows how ResNet18 initially distributes its attention on multiple objects like chairs, tables, structural elements simultaneously. After cross-attention application (Figure 5.2, bottom row), heatmaps show intense hotspots in specific areas, suggesting the model now knows which regions are relevant to the navigation goal.

DINOv2 Analysis. In the initial state (Figure 5.3, middle row), we see crisp outlines around chair legs, table edges and AprilTags revealing that ViT learns object-level feature extraction. Following cross-attention integration (Figure 5.3, bottom row), the model transforms these object representations into navigation-focused attention maps, characterized by localized hotspots that appear strategically positioned for path planning.

Success on MAP 2. Visualizations (Figures 5.2 and 5.3) reveal that models navigate by memorizing floor coordinates near the target location. This spatial memorization strategy succeeds on MAP 2 because the memorized coordinates remain valid navigation targets from different starting positions within the similar environment layout.

Failure on MAP 3. The same coordinate-based strategy causes catastrophic failure when objects relocate on MAP 3. So, when we relocate the target, agent navigates to previously learned positions despite the absence of the target, demonstrating complete absence of object-grounded learning. This persistence in seeking empty locations reveals the models' fundamental inability to associate navigation goals to semantic entities rather than environmental positions.

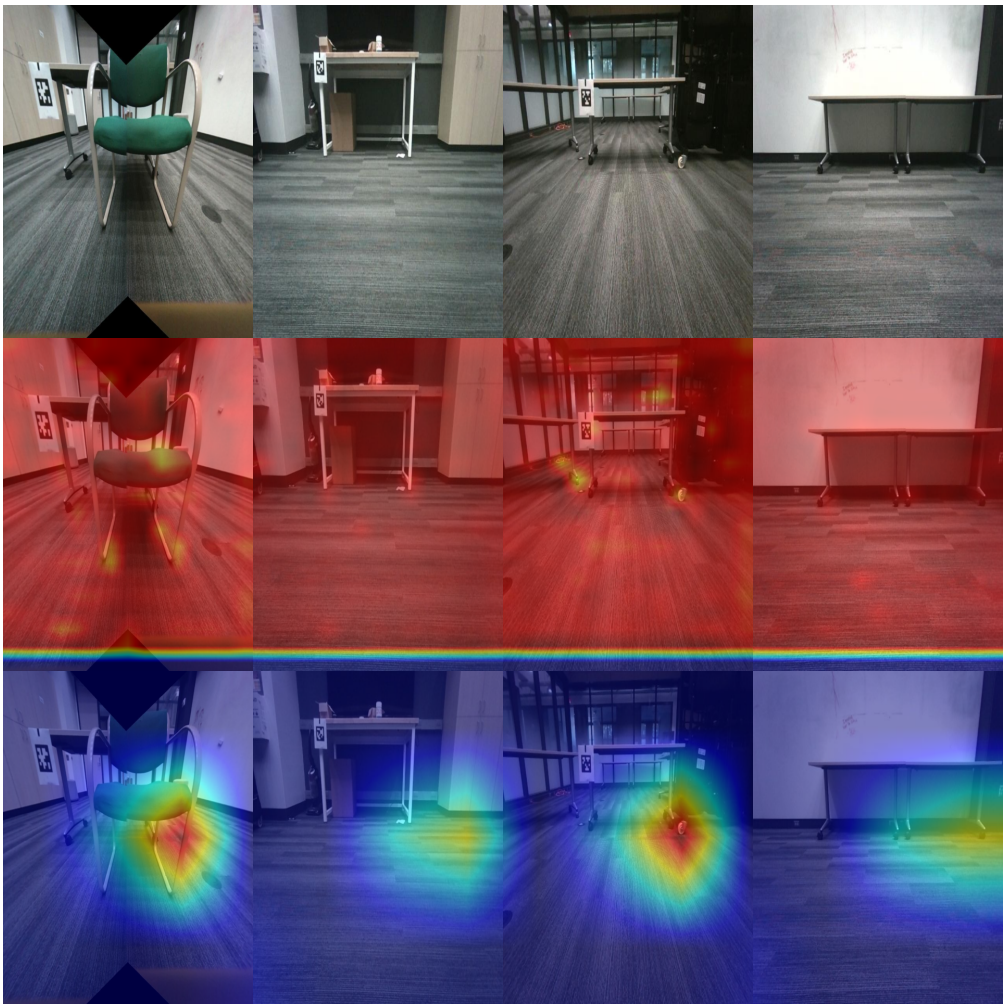


Figure 5.2: ResNet18 heatmap visualizations. **Top**: Raw RGB inputs. **Middle**: Heatmaps generated by EigenCAM (Pre-attention). **Bottom**: Heatmaps after cross-attention (Post-attention).

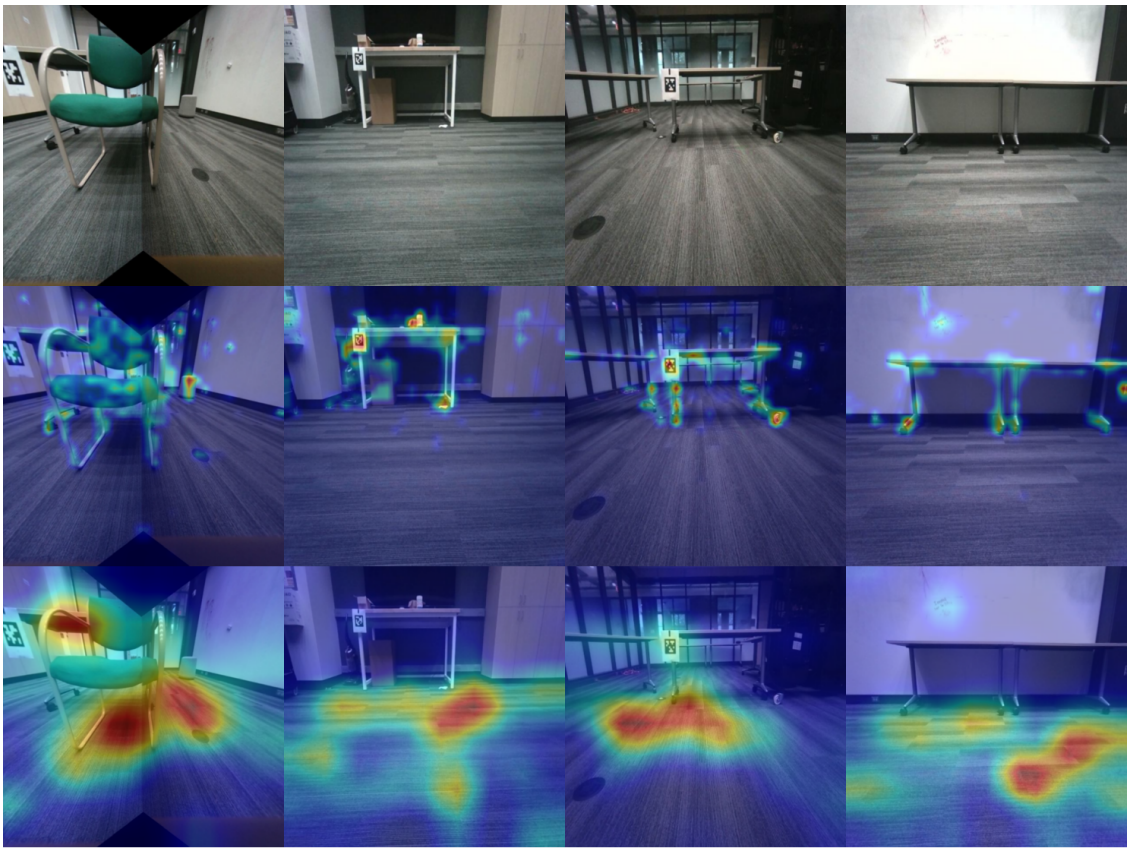


Figure 5.3: DINOv2 heatmap visualizations. **Top**: Raw RGB inputs. **Middle**: Heatmaps generated by attention-rollout (Pre-attention). **Bottom**: Heatmaps after cross-attention (Post-attention).

5.3 Evaluation of Grounded SAM2 Segmentation Reliability

While examining segmentation masks produced by Grounded SAM2 across our dataset, we observed mis-segmentations in the visual frames as illustrated in Figure 5.4, prompting us to systematically quantify these errors and evaluate their impact on navigation performance.

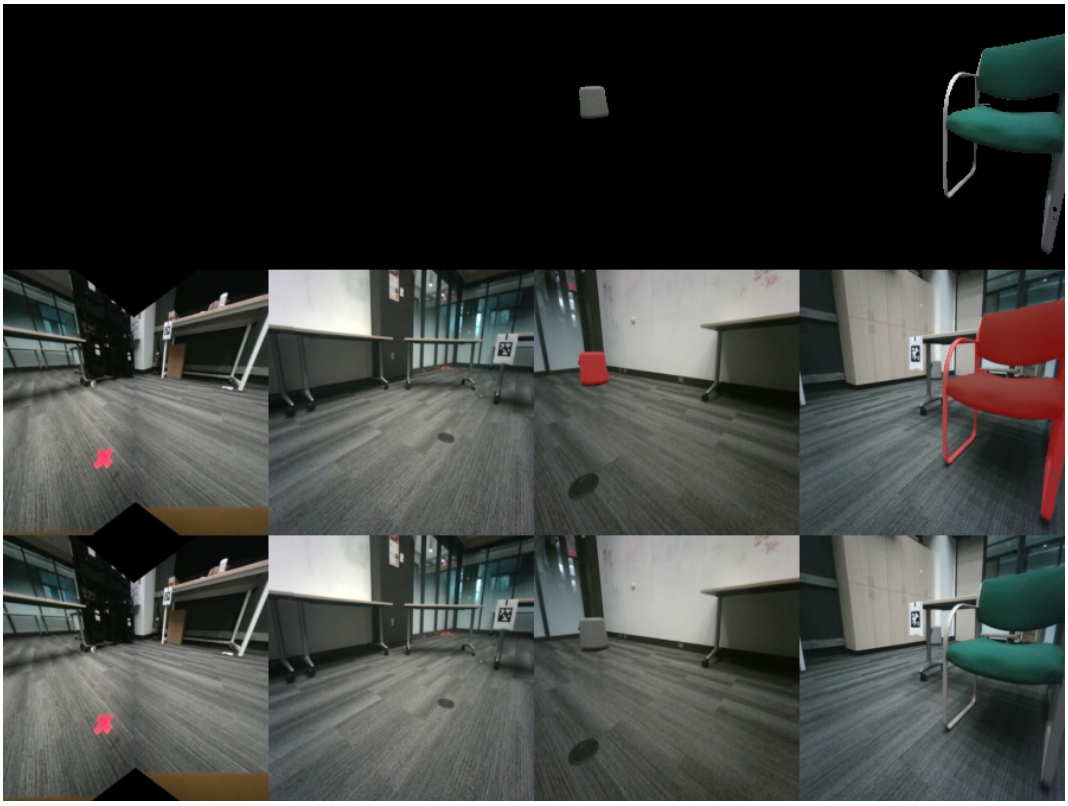


Figure 5.4: Grounded SAM2 mis-segmentation example: Grounded SAM2 segments couch in the image when given the text prompt ”green chair.”

We analyzed segmentation masks from MAP 1 training dataset and observed two main failure cases as shown in Figure 5.5. Wrong object segmentation occurs when Grounded SAM2 identifies incorrect furniture items instead of the target chair, while target object missed represents cases where no segmentation mask for the target object is produced.

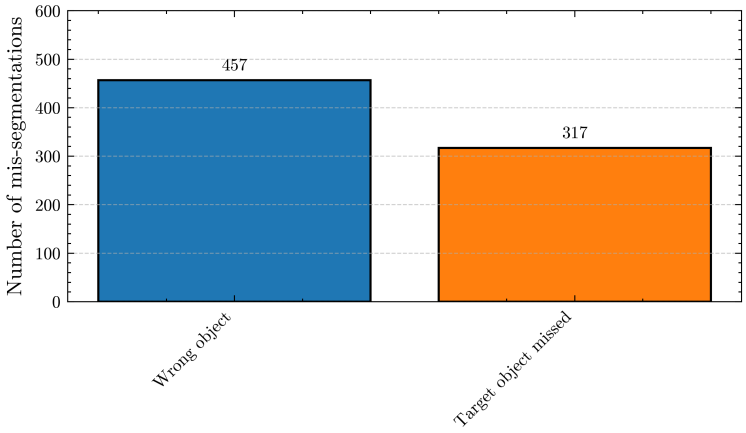


Figure 5.5: Grounded SAM2 mis-segmentation breakdown by category.

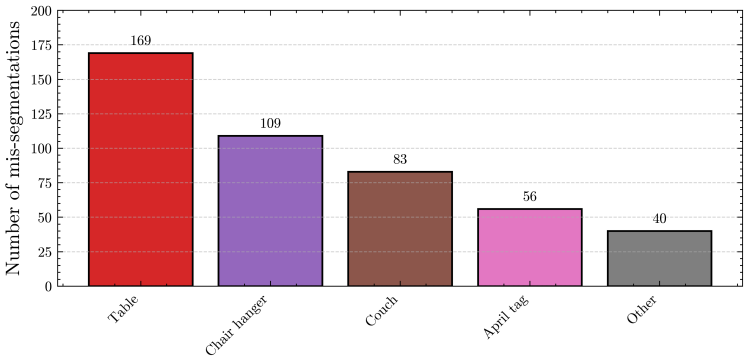


Figure 5.6: Wrong object segmentation sub-category breakdown.

Our findings align with documented limitations of Grounding DINO in open-vocabulary settings. Community evaluations (GitHub Issue #84 [30]) report similar false-positive generation, suggesting that such segmentation errors represent inherent challenges in open-vocabulary detection rather than implementation-specific failures.

Despite producing 774 wrong segmentations (10.6% error rate), Grounded SAM2-MLP maintains reasonable navigation success on tested maps, significantly outperforming ResNet and DINOv2 baselines. Since our architecture processes each frame independently , the robustness likely emerges from the policy's ability to recover at each decision step. When Grounded SAM2 produces an incorrect segmentation, the policy can immediately correct course in the next frame when a valid detection appears.

Chapter 6

Conclusion

This thesis presented an end-to-end imitation learning approach that enables Boston Dynamics SPOT robot to achieve manipulation-ready positioning through visual navigation. By comparing three vision encoders (ResNet18, DINOv2, and Grounded SAM2), we uncovered a fundamental insight: successful local navigation benefits significantly from semantic object understanding. While all architectures achieved comparable performance in a familiar environment, only Grounded SAM2 maintained generalization when the target object is relocated or placed in an unseen environment, achieving 98% and 91% success, respectively compared to complete failure for ResNet18 and DINOv2. This difference reveals that explicit object segmentation provides the semantic grounding necessary for generalizable navigation policies, establishing a practical pathway for deploying mobile manipulators in real-world environments where precise positioning is critical for downstream tasks.

Limitations and Future Work. Our current system achieves partial success in single-category generalization (48 %), indicating room for improvement with more advanced vision encoders that could enhance semantic understanding. The primary limitation stems from the vision encoder's segmentation reliability, with Grounded SAM2 producing 10.6% incorrect detections, predominantly misidentifying semantically similar furniture. Although the policy demonstrates recovery

through frame-by-frame processing, these errors constrain overall navigation consistency. Future directions include developing more robust vision encoders to improve single-category generalization, incorporating prompt-specified approach angles for task-specific manipulation positioning, as well as exploring temporal modeling in the decoder for smoother trajectories.

Bibliography

[1] Naoki Yokoyama et al. “Vlfm: Vision-language frontier maps for zero-shot semantic navigation”. In: *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2024, pp. 42–48.

[2] Devendra Singh Chaplot et al. “Object goal navigation using goal-oriented semantic exploration”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 4247–4258.

[3] Lina Mezghan et al. “Memory-augmented reinforcement learning for image-goal navigation”. In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2022, pp. 3316–3323.

[4] Kuo-Hao Zeng et al. “PoliFormer: Scaling On-Policy RL with Transformers Results in Masterful Navigators”. In: *Conference on Robot Learning*. PMLR. 2025, pp. 408–432.

[5] “A tutorial on visual servo control”. In: *IEEE transactions on robotics and automation* 12.5 (2002), pp. 651–670.

[6] Aaron Hao Tan et al. “Mobile robot regulation with image based visual servoing”. In: *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. Vol. 51807. American Society of Mechanical Engineers. 2018, V05AT07A078.

[7] Abdulrahman Al-Shanoon et al. “Mobile robot regulation with position based visual servoing”. In: *2018 IEEE International Conference on Computational Intelligence and Virtual*

Environments for Measurement Systems and Applications (CIVEMSA). IEEE. 2018, pp. 1–6.

[8] François Chaumette and Seth Hutchinson. "Visual servo control. I. Basic approaches". In: *IEEE Robotics & Automation Magazine* 13.4 (2006), pp. 82–90.

[9] Pushkal Katara et al. "Deepmpcvs: Deep model predictive control for visual servoing". In: *Conference on Robot Learning*. PMLR. 2021, pp. 2006–2015.

[10] Yuanze Wang et al. "NeRF-IBVS: visual servo based on nerf for visual localization and navigation". In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 8292–8304.

[11] Alessandro Scherl et al. "ViT-VS: On the Applicability of Pretrained Vision Transformer Features for Generalizable Visual Servoing". In: *arXiv preprint arXiv:2503.04545* (2025).

[12] Pranjali Pathre et al. "Imagine2Servo: Intelligent Visual Servoing with Diffusion-Driven Goal Generation for Robotic Tasks". In: *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2024, pp. 13466–13472.

[13] Tim Brooks, Aleksander Holynski, and Alexei A Efros. "Instructpix2pix: Learning to follow image editing instructions". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2023, pp. 18392–18402.

[14] Dylan Goetting, Himanshu Gaurav Singh, and Antonio Loquercio. "End-to-end navigation with vision language models: Transforming spatial reasoning into question-answering". In: *arXiv preprint arXiv:2411.05755* (2024).

[15] Soroush Nasiriany et al. "PIVOT: Iterative Visual Prompting Elicits Actionable Knowledge for VLMs". In: *International Conference on Machine Learning*. PMLR. 2024, pp. 37321–37341.

[16] Lingfeng Zhang et al. “Mem2ego: Empowering vision-language models with global-to-ego memory for long-horizon embodied navigation”. In: *arXiv preprint arXiv:2502.14254* (2025).

[17] Arjun Majumdar et al. “Zson: Zero-shot object-goal navigation using multimodal goal embeddings”. In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 32340–32352.

[18] Yimeng Li and Jana Košecka. “Learning view and target invariant visual servoing for navigation”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 658–664.

[19] Kiana Ehsani et al. “Spoc: Imitating shortest paths in simulation enables effective navigation and manipulation in the real world”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 16238–16250.

[20] Nigitha Selvaraj, Alex Mitrevski, and Sebastian Houben. “Are Learning-Based Approaches Ready for Real-World Indoor Navigation? A Case for Imitation Learning”. In: *arXiv preprint arXiv:2507.04086* (2025).

[21] Xiangyun Meng et al. “Aim My Robot: Precision Local Navigation to Any Object”. In: *IEEE Robotics and Automation Letters* (2025).

[22] Shilong Liu et al. “Grounding dino: Marrying dino with grounded pre-training for open-set object detection”. In: *European conference on computer vision*. Springer. 2024, pp. 38–55.

[23] Nikhila Ravi et al. “Sam 2: Segment anything in images and videos”. In: *arXiv preprint arXiv:2408.00714* (2024).

[24] Ashish Vaswani et al. “Attention is all you need”. In: *Advances in neural information processing systems* 30 (2017).

[25] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[26] Maxime Oquab et al. “Dinov2: Learning robust visual features without supervision”. In: *arXiv preprint arXiv:2304.07193* (2023).

[27] Boston Dynamics. *GraphNav Service*. `https://dev.bostondynamics.com/docs/concepts/autonomy/graphnav_service.html`. Version SpotSDK v5.0.0.

[28] Mohammed Bany Muhammad and Mohammed Yeasin. “Eigen-cam: Class activation map using principal components”. In: *2020 international joint conference on neural networks (IJCNN)*. IEEE. 2020, pp. 1–7.

[29] Samira Abnar and Willem Zuidema. “Quantifying attention flow in transformers”. In: *arXiv preprint arXiv:2005.00928* (2020).

[30] IDEA-Research. *GroundingDINO Issue #84: Many false positives*. `https://github.com/IDEA-Research/GroundingDINO/issues/84`.