
PIG-NAV: KEY INSIGHTS FOR PRETRAINED IMAGE GOAL NAVIGATION MODELS

PREPRINT

Navigation Research Team under MSRA ML Group*
Contact: pushizhang@microsoft.com

March 20, 2025

ABSTRACT

Recent studies have explored pretrained (foundation) models for vision-based robotic navigation, aiming to achieve generalizable navigation and positive transfer across diverse environments while enhancing zero-shot performance in unseen settings.

In this work, we introduce PIG-Nav (Pretrained Image-Goal Navigation), a new approach that further investigates pretraining strategies for vision-based navigation models and contributes in two key areas. Model-wise, we identify two critical design choices that consistently improve the performance of pretrained navigation models: (1) integrating an early-fusion network structure to combine visual observations and goal images via appropriately pretrained Vision Transformer (ViT) image encoder, and (2) introducing suitable auxiliary tasks to enhance global navigation representation learning, thus further improving navigation performance. Dataset-wise, we propose a novel data preprocessing pipeline for efficiently labeling large-scale online video datasets for navigation model training. We demonstrate that augmenting existing open navigation datasets with large-scale internet-sourced navigation videos improves model performance.

Our model achieves an average improvement of 22.6% in zero-shot settings and a 37.5% improvement in fine-tuning settings over existing visual navigation foundation models in two complex simulated environments and one real-world environment. These results advance the state-of-the-art in pretrained image-goal navigation models. Notably, our model maintains competitive performance while requiring significantly less fine-tuning data, highlighting its potential for real-world deployment with minimal labeled supervision.

1 Introduction

Vision-based navigation enables an autonomous agent to navigate toward a specific location or object using visual observations Mayo et al. [2021], Bonin-Font et al. [2008]. This field has gained significant research attention due to its wide-ranging practical applications, including home automation, search-and-rescue missions, and last-mile delivery [Shah et al., 2021a, Szot et al., 2021, Wu et al., 2022, Du et al., 2021].

Recently, researchers have explored pretrained visual navigation foundation models, which serve as generalized navigation frameworks capable of adapting to diverse robotic embodiments and environments [Shah et al., 2023a,b, Sridhar et al., 2024, Zhang et al., 2024a,b]. These models aim to capture prior navigational knowledge from large-scale navigation datasets Karnan et al. [2022], Liu et al. [2024], Yokoyama et al. [2024], allowing them to be deployed to new downstream tasks or fine-tuned with minimal data across different robotic platforms. Existing works have investigated unifying action spaces across various robot embodiments Shah et al. [2023a,b] and training generalized policies for both goal-directed navigation and environment exploration without explicit goal guidance Sridhar et al. [2024].

However, despite these advancements, generalization to novel environments remains a major challenge, particularly when the new environments are out-of-distribution (OOD) compared to the pretraining dataset. Two key limitations contribute to this issue: 1) The design of existing models focuses on the relative information between observations and

*Work in progress.

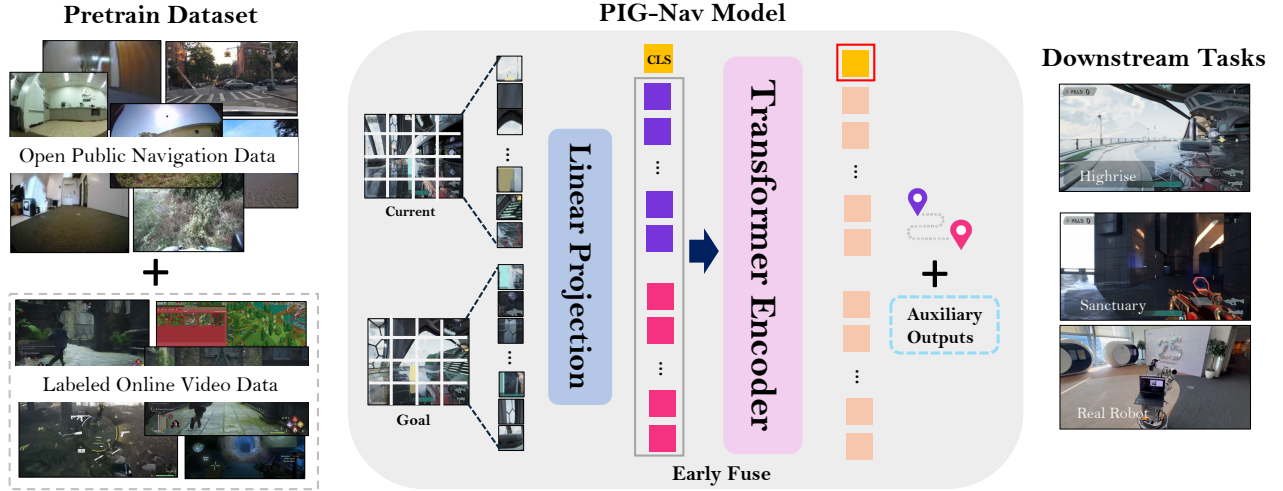


Figure 1: Overview of PIG-Nav.

goals as a whole, failing to capture low-level correspondences. Additionally, they often employ low-capacity encoders that are not initialized with pretrained weights, restricting their ability to learn transferable navigation representations. 2) The datasets used for pretraining are not diverse enough to encompass a broad range of navigation scenarios, limiting the model’s robustness.

To address these challenges, we introduce several key advancements to enhance the capabilities of visual navigation foundation models. First, we propose a simple yet effective model structure that enables fine-grained modeling of the low-level correspondence between images and goals. This is achieved by integrating a novel early-fusion network architecture and a pretrained Vision Transformer (ViT) encoder to enhance representation learning. Second, we introduce auxiliary learning objectives that capture high-level navigation decision-making information, leading to improved representation learning and navigation performance. Furthermore, to address the substantial data requirements of training navigation foundation models, we propose a novel data labeling pipeline that efficiently processes online videos for navigation tasks, complementing existing navigation datasets.

To comprehensively evaluate PIG-Nav, we conduct experiments across two simulation-based game environments and a real-world robotic setting, comparing our approach against multiple baselines. Our results demonstrate that PIG-Nav exhibits superior generalizability, improved downstream fine-tuning performance, and strong potential as a foundation model for navigation tasks.

We demonstrate the following findings through our experiments:

1. PIG-Nav outperforms all baselines, including established visual navigation foundation models such as GNM Shah et al. [2023a], ViNT Shah et al. [2023b], and NoMaD Sridhar et al. [2024], showing an average improvement of 22.6% in navigation success rate for zero-shot generalization settings, and an average improvement of 37.5% for finetuning settings across all tasks.
2. Pretraining PIG-Nav significantly enhances learning efficiency in downstream environments, reducing data requirements by approximately eightfold.
3. The integration of an early-fusion network architecture with a pretrained Vision Transformer (ViT) leads to substantial improvements in navigation performance.
4. Incorporating auxiliary tasks that capture global decision-making information further enhances the performance of the foundation model.
5. The newly labeled online video dataset contributes to the training of navigation foundation models, demonstrating its potential as a valuable resource for future navigation tasks.

2 Related Work

Image-goal navigation requires autonomous agents to navigate from their current location to a target location specified by a reference image. This task is particularly challenging, as it requires agents to establish a robust correspondence between their current observations and the goal image. When the current view does not directly reveal the goal, the agent must actively explore the environment to find relevant cues that guide it toward the goal location Gupta et al. [2017], Pertsch et al. [2020], Chaplot et al. [2020].

Early methods primarily relied on handcrafted features to compare the goal image with the agent’s current view and estimate the direction toward the target. However, these approaches often struggled in complex or dynamic environments, as they were highly sensitive to lighting variations, occlusions, and scene changes. To overcome these limitations, [Kwon et al., 2023] proposed constructing a 3D map of the environment and projecting the goal image onto this map to compute the optimal navigation path [Kwon et al., 2021].

To overcome these limitations, recent research has embraced Deep Reinforcement Learning (DRL) [Yadav et al., 2023, Sun et al., 2024, Kim et al., 2023, Oquab et al., 2023]. DRL-based methods learn end-to-end navigation policies by interacting with simulated environments, using visual observations and goal images to make real-time decisions Zhu et al. [2017], Shi et al. [2019]. However, due to the heavy reliance on simulation environments and overfitting issues associated with DRL-based approaches, there is a growing interest in exploring general pretrained navigation models [Zhang et al., 2018, Radford et al., 2021]. These models leverage large-scale, diverse datasets and high-capacity architectures to achieve robust performance in unseen environments. GNM create a unified model that can operate across different robots and settings, achieving zero-shot deployment. In contrast, ViNT leverages Transformer architecture to fuse features from the current view and the goal image at an early stage, which enhances the learning of fine-grained visual correspondences. Building on ViNT, NoMaD introduces a masked diffusion strategy within a compact Transformer framework, leading to improved performance with lower computational overhead [Radford et al., 2021]. These enhancements aim to improve generalization, data efficiency, and navigation performance in unseen environments.

Our work builds on these advancements by proposing a Vision Transformer (ViT)-based early fusion network that tightly integrates visual features from both the current observation and the goal image. We further augment our architecture with auxiliary tasks to capture global decision-making information, and introduce a novel online video labeling pipeline to address the data requirements. This combination aims to enhance both generalization and data efficiency.

Another active research direction leverages Vision-Language Models (VLM) [Pelluri, 2024], which utilize VLM as a backbone to process both visual inputs and language instructions, guiding agent navigation through multi-modal reasoning [Zhou et al., 2024, Chen et al., 2024, Zhang et al., 2024b, Anderson et al., 2018, Chen et al., 2025, Eftekhar et al., 2024]. These approaches differ from our focus. However, our findings offer valuable insights that could later benefit multi-modal networks.

3 Proposed Model: PIG-Nav

In this section, we provide a comprehensive explanation of our proposed model, PIG-Nav, including its architecture and training methodology, which incorporates auxiliary training tasks. Figure 1 presents an overview of our approach.

Our work begins with the collection of diverse trajectory data, consisting of RGB images along with their corresponding position and orientation information. Additionally, we annotate large-scale online video to further enhance PIG-Nav’s capability. Using these datasets, we pretrain PIG-Nav to learn generalizable navigation representations, allowing it to adapt effectively to a wide range of navigation tasks. After pretraining, we fine-tune the model on environment-specific datasets, optimizing its performance for downstream applications.

Specifically, Section 3.1 introduces the overall architecture of PIG-Nav, while Section 3.2 details the training methodology, including the primary training task and the design of auxiliary tasks. We will elaborate on pertaining datasets in Section 4.

3.1 Model Architecture

Figure 1 illustrates the network architecture of our model. The visual encoder is designed to capture correspondences between the low-level features of the observation and goal images, facilitating effective feature alignment. The current observation and the goal image are first independently processed into patch embeddings before being fused into a unified set of visual tokens. Each patch token is augmented with learnable tokens indicating the observation or goal images, which are then jointly fed into the Vision Transformer (ViT) encoder. We refer to this design as an early-fusion

model architecture, as the transformer encoder directly operates on the low-level features of both the observation and goal images, enabling more effective spatial and semantic integration.

The learnable [CLS] token serves as an additional input to the transformer encoder, enabling the extraction of joint representations from the image inputs. The resulting output embedding of this token is subsequently fed into separate MLP layers as the contextual embeddings to generate waypoint action outputs and auxiliary outputs. The action heads decode the next $N_{\text{waypoint}} = 10$ actions, utilizing a 4-dimensional waypoint action space $[\Delta x, \Delta y, \cos \Delta \psi, \sin \Delta \psi]$, where $\Delta x, \Delta y, \Delta \psi$ are the 2D positional movements and rotations of the agent. The auxiliary prediction tasks are detailed in Section 3.2.

We also propose to leverage ViT encoders that pretrained by powerful self-supervised models including DINOv2 [Oquab et al., 2023] and Masked Auto Encoding (MAE) [He et al., 2022], enabling effective joint processing of observation and goal images. Further discussions on the effects of these pretrained ViT encoders can be found in Section 5.2.4.

3.2 Training Tasks

The primary objective of our model is to predict waypoint actions, enabling agents to determine how to move from their current locations. Additionally, PIG-Nav incorporates auxiliary tasks that provide high-level supervisory signals, guiding low-level navigation actions and leading to more effective trajectory planning. Specifically, we introduce three types of auxiliary tasks: 1) relative pose to the goal position, 2) distance to the goal position, and 3) global path to the goal position. Further calculations and details are provided in supplementary materials.

Waypoint Action Prediction. The main task of PIG-Nav is to predict translation and rotation changes over the next $N_{\text{waypoint}} = 10$ consecutive steps in the provided navigation trajectory based on the current image observation and the goal image observation. This chunk-based action prediction approach has been adopted in recent works [Bachmann and Nagarajan, 2024, Hu et al., 2024] as an improvement over methods that predict only the next immediate action.

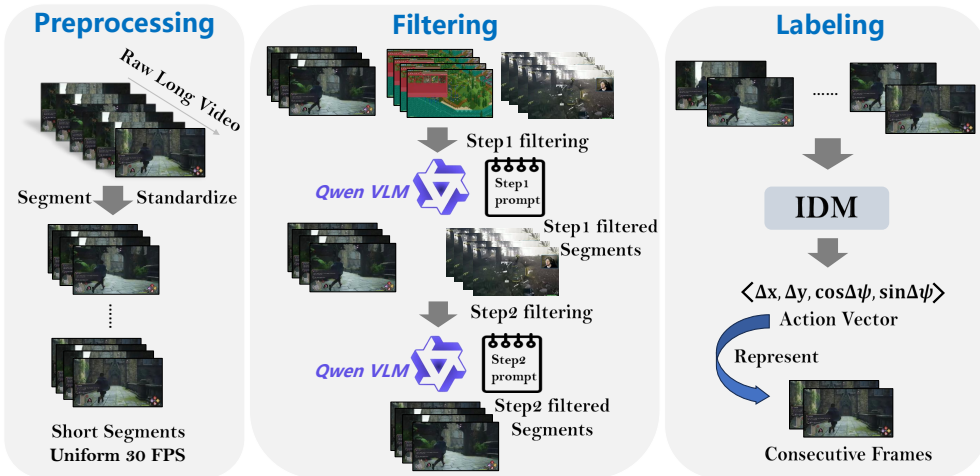


Figure 2: Online Video Data Processing Pipeline.

Relative Pose to Goal. In the relative pose prediction task, the model estimates the pose of the goal image relative to the observation image. This task enables the model to capture global task information by understanding the spatial relationship between the current observation and the goal image.

Navigation Distance Prediction. In the navigation distance prediction task, our model is trained to estimate the total distance required for the agent to navigate from its current state to the goal state. By training to predict the distance, the model learns to predict the connectivity and traversability between different locations in the environment, enabling more efficient path planning.

Global Path Prediction. To further enhance the agent’s ability to predict long-range paths, we introduce the global path prediction task. This task requires the model to predict $N_{\text{global}} = 10$ intermediate waypoints, which are equally spaced in time from the current timestep to the goal image’s timestep.

4 Pretraining Datasets for PIG-Nav

In this section, we present the two components of our PIG-Nav’s pretraining dataset: (1) multiple publicly available navigation datasets, described in Section 4.1, and (2) a newly constructed navigation dataset, derived from diverse online video data, detailed in Section 4.2.

4.1 Open Public Navigation Datasets

We collect a large-scale dataset of real-world navigation trajectories to pretrain our model including GoStanford Hirose et al. [2019], RECON Shah et al. [2021b], CoryHallKahn et al. [2018], Berkeley DeepDrive Yu et al. [2020], SCAND Karnan et al. [2022], TartanDrive Triest et al. [2022], and SACSOn Hirose et al. [2023]. This combined dataset encompasses both teleoperated and autonomous navigation behaviors, collected from various robotic platforms. Additionally, it captures a wide range of motion dynamics, as the data originates from different robots operating in diverse environments, including office buildings, off-road trails, university campuses, and indoor rooms.

To ensure data quality and facilitate stable training, we filtered out data points where position changes exceeded five times the average displacement. Additionally, we follow previous works to transform the data into a unified space, as datasets vary in scale and properties. Further details regarding open public navigation datasets are provided in the supplementary materials.

4.2 Labeled Navigation Datasets from Online Video

To further enhance the generalization and robustness of our PIG-Nav model from a data-centric perspective, we collect a large and diverse set of online 3D game video data. We develop a data processing pipeline that includes preprocessing, filtering, and labeling to ensure high-quality training data for model pretraining.

Figure 2 illustrates our data cleaning and annotation pipeline, which consists of three key stages: preprocessing, cleaning, and labeling. The details of each stage are described below.

Data Source & Pre-processing. We collect over 220,000 long gameplay videos from the Internet, sourced from approximately 2,000 different games, as our original dataset. The average duration of these videos is around 3 hours. To facilitate processing, we segment the original videos into 10-second clips and standardize them to 30 FPS for subsequent processing.

Data Cleaning by VLM. We utilize the open-source vision-language model Qwen2-VL-7B Wang et al. [2024] to classify whether the preprocessed video segments are suitable for inclusion in the navigation dataset. The Qwen2 model takes both video segments and language prompts as input and outputs a binary classification result: *True* (suitable) or *False* (unsuitable), based on the provided prompt instructions.

To ensure high-quality video selection, we apply a two-step filtering process and evaluate the precision, recall, and F1-score at each step. For the first filtering step, we sample total 500 videos from raw video segments randomly and manually label them as *True* or *False*. The Qwen2 model’s initial classification achieves 52% precision and 75% recall, indicating that the first filtering step effectively retains most useful segments, but the precision remains relatively low, suggesting the necessity for further refinement.

In the second filtering process, we experiment with 12 different prompt instructions and select the one yielding the best weighted balance between precision and recall to improve dataset quality. The final prompts we picked result in 85% precision and 68% recall, demonstrating that our two-step filtering strategy effectively constructs a high-quality video dataset for navigation tasks. All prompts we use in the data cleaning process and the statistics are detailed in supplementary materials.

Action Annotation. To annotate navigation actions $[\Delta x, \Delta y, \cos \Delta \psi, \sin \Delta \psi]$ from our filtered video segments, we train an Inverse Dynamics Model (IDM) to generate action labels, which measures the action change between two consecutive video frames. The IDM adopts the same network architecture as PIG-Nav, excluding the waypoint action and auxiliary outputs, and operates by processing two consecutive video frames that capture slight changes in motion. It predicts the corresponding navigation action between these frames using a dedicated action prediction head.

We train the IDM using open navigation datasets described in Section 4.1. To optimize the accuracy of navigation action labels, we experiment with different input frame rates on small dataset batches. Based on these evaluations, we select a configuration where 48 short frame pairs are extracted from each 10-second filtered video for training, and the IDM is trained for 26 hours. Further details regarding the visualizations of labeled video segments can be found in the supplementary materials.

	Easy		Highrise Medium		Hard		Easy		Sanctuary Medium		Hard		Robot 8 Tasks	
	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL
GNM (ZS)	0.86	0.78	0.58	0.47	0.32	0.20	0.33	0.27	0.11	0.05	0.00	0.00	0.00	0.00
ViNT (ZS)	0.82	0.78	0.46	0.43	0.26	0.22	0.41	0.37	0.36	0.32	0.18	0.17	0.13	0.10
NoMaD (ZS)	0.78	0.75	0.42	0.40	0.22	0.19	0.40	0.31	0.42	0.36	0.08	0.05	0.00	0.00
PIG-Nav (ZS)	0.90	0.86	0.72	0.68	0.46	0.38	0.84	0.80	0.48	0.45	0.30	0.26	0.50	0.44
GNM (FT)	0.98	0.92	0.68	0.63	0.66	0.55	0.72	0.67	0.38	0.32	0.20	0.15	0.25	0.22
ViNT (FT)	0.88	0.83	0.66	0.56	0.50	0.42	0.62	0.55	0.38	0.35	0.19	0.16	0.25	0.21
NoMaD (FT)	0.70	0.68	0.34	0.33	0.15	0.14	0.52	0.47	0.18	0.17	0.10	0.09	0.38	0.32
PIG-Nav (FT)	1.00	0.97	1.00	0.94	0.90	0.81	1.00	0.96	0.84	0.75	0.76	0.68	1.00	0.89

Table 1: SR and SPL performance across different environments and settings. All values are rounded to the nearest hundredth.

5 Experiments

We design our experiments to address the following key research questions:

- How does PIG-Nav compare to other pretrained navigation foundation models in terms of zero-shot generalization and fine-tuning performance in simulation and real world environments?
- How does pretraining phase of PIG-Nav enhance the data efficiency of fine-tuning in unseen new environments?
- Does the online video dataset, labeled using our proposed pipeline, improve the pretraining of navigation foundation models?

Additionally, to further justify our design choices and provide insight into developing navigation foundation models, we conduct extensive ablation studies focusing on the following aspects:

- Effect of early fusion – How does early fusion of low-level image features from the observation and goal images impact PIG-Nav’s performance?
- Impact of pretrained ViT initialization – How do self-supervised ViT encoders (e.g., MAE, DINOv2) and ViT model size affect PIG-Nav’s performance?
- Influence of auxiliary tasks – How do auxiliary tasks for predicting global decision-making information contribute to PIG-Nav’s effectiveness?

Next, we present the details of our environments used for testing our models and the experiment setup, and then present the result analysis.

5.1 Experiment Setups

Environment. We conducted evaluation experiments across two simulation environments and one real-world robotic environment. The two simulation environments, Highrise and Sanctuary, are both part of ShooterGame, a representative PC multiplayer First-Person Shooter (FPS) game built on Unreal Engine 4. This game provides a robust simulation framework for evaluating navigation models. Both environments span approximately 10,000 m², featuring large-scale, dynamic environments with numerous obstacles and complex background elements, making them particularly challenging for navigation policies.

For the real-world environment, we deploy our model on a wheeled robot traversing an indoor floor space of approximately 2,000 m², providing a realistic setting to evaluate the model’s performance in physical environments.

Fine-tune Dataset. We collect 2,100 navigation episodes by human players from Highrise and Sanctuary as our downstream task dataset. The average episode length is 34 steps in Highrise and 47 steps in Sanctuary, with approximately 0.7 million and 0.9 million total frames, respectively. For the real-robot dataset, we gather 113 episodes using human teleoperators, with an average episode length of 93 steps and a total of 10,500 frames. In all datasets, human players and teleoperators are instructed to traverse the entire environment, and we subsequently trimmed the trajectories to extract navigation episodes following optimal paths. For all three environments—Highrise, Sanctuary, and the real-robot setting—we split the dataset into 90% for fine-tuning dataset and 10% for validation.

Final Evaluation Task. We randomly sample the start and goal points for evaluation within the corresponding environment. To assess performance under varying conditions, we divide the evaluation into three difficulty levels—Easy, Medium, and Hard—determined by the trajectory length between start point and goal point on ShooterGame. In Highrise, the average values of timesteps required to reach the goal for Easy, Medium, and Hard difficulty levels are (19, 37, 55) respectively. Similarly, in Sanctuary, the corresponding values are (28, 56, 83) timesteps. We sample 50 tasks for each difficulty level in Highrise and Sanctuary. For the robot experiments, we test a total of 8 tasks where the start and goal positions are distributed across the entire indoor floor. The agents are required to reach the goal from their current position within a total of 100 steps.

Evaluation Metrics. We employ two widely used navigation metrics: Success Rate (SR) and Success Weighted by Path Length (SPL), which assess both task completion and navigation efficiency. We define these metrics as:

$$SR = \frac{1}{N} \sum_{i=1}^N S_i, \quad SPL = \frac{1}{N} \sum_{i=1}^N S_i \frac{d_i}{p_i}$$

where S_i equals 1 if navigation is successful in i -th evaluation episode and 0 otherwise, N is the total number of evaluation episodes, d_i represents the shortest distance from the starting position to the goal, and p_i denotes the actual path length taken by the agent during evaluation.

Hyperparameter Settings. Our model utilizes ViT-Base as the image encoder and incorporates multiple MLP heads, resulting in a total of approximately 100 million parameters.

We train the model using the Adam optimizer with a learning rate of 5×10^{-5} and a batch size of 128. The pretraining phase spans 200 epochs, requiring approximately 2 days on a $4 \times$ A100 GPU setup. For fine-tuning, the model is trained for 200 epochs, completing in approximately two days on a single A100 GPU.

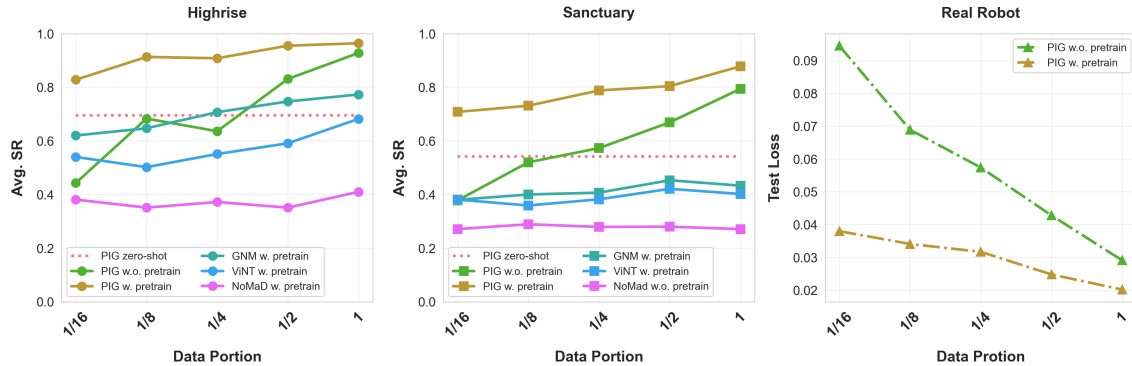


Figure 3: Average SR and Validation Loss performance comparison across different fine-tuning dataset size. Left for Highrise, middle for Sanctuary and right for real robot. Average SR is used for Highrise and Sanctuary measurement, with higher value meaning better model performance. Test Loss is used for real robot measurement, with lower value meaning better model performance.

5.2 Results & Analysis

5.2.1 Comparison with Baselines for Zero-Shot Generalization and Finetuning

In this experiment, we compare PIG-Nav against three baseline methods—GNM, ViNT, and NoMaD—across two key evaluation criteria: generalizability to unseen environments and fine-tuning performance in diverse settings. The results, presented in Table 1, demonstrate that PIG-Nav consistently outperforms all three baselines in both zero-shot and fine-tuned tasks.

Notably, at the Medium and Hard difficulty levels, PIG-Nav surpasses the baselines by an even larger margin. We attribute this to PIG-Nav’s effective model architecture design. The three baselines rely on CNNs to process the concatenated observation and goal images, capturing the relative information as a whole. In contrast, PIG-Nav directly models the spatial relationships among image patches via our designed early-fusion ViT, enabling it to more effectively capture the nuanced connections between the current observation and the goal image. This enhanced contextual understanding allows PIG-Nav to make more informed navigation decisions, particularly in challenging scenarios.

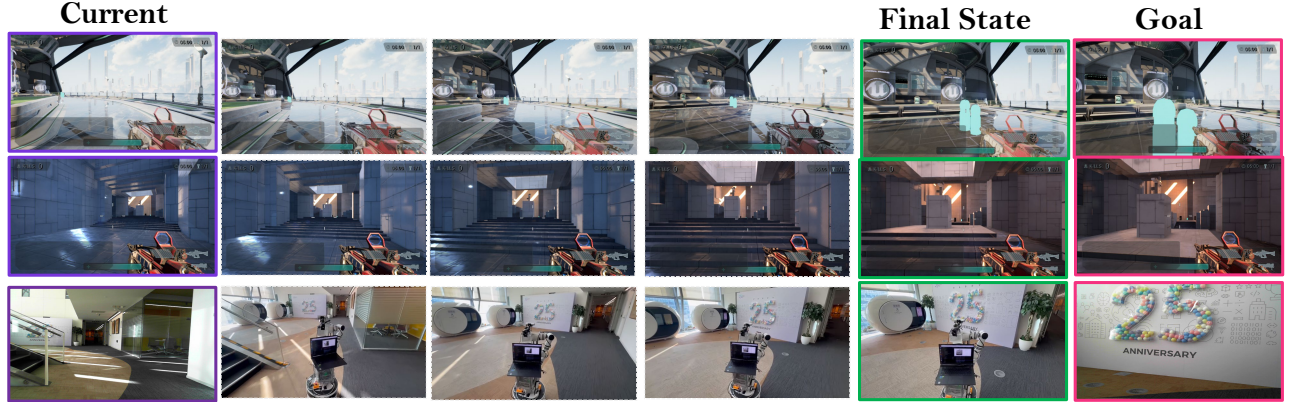


Figure 4: Three evaluation examples of PIG-Nav on three environments: Highrise, Sanctuary, Real Robot.

We present three examples from each environment’s evaluation test in Figure 4. For more case studies, please refer to supplementary materials.

5.2.2 Fine-Tune PIG-Nav on New Environment

In this experiment, we evaluate the fine-tuning efficiency across the three environments. To assess adaptability with limited labeled data, we progressively reduce the amount of fine-tuning data to different levels: 1, 1/2, 1/4, 1/8, and 1/16 of the full dataset. We then measure the model’s performance under each setting, averaging results across three difficulty levels.

As illustrated in Figure 3, performance decreases across nearly all models as the available fine-tuning data is reduced. However, PIG-Nav, with its pretraining phase, demonstrates the most robust performance, maintaining high accuracy even with significantly less fine-tuning data compared to other baselines and PIG-Nav without pretraining. Notably, it reduces data requirements by approximately eightfold, further reinforcing its effectiveness as a navigation foundation model.

Additionally, it is worth highlighting that the zero-shot generalization performance of PIG-Nav surpasses models fine-tuned with 1/16 and 1/8 of the data without pretraining, underscoring its strong generalization capabilities, even in the absence of additional fine-tuning data.

5.2.3 Benefit of Labeled Online Video Data

To assess the impact of our newly labeled online video data, we pretrain PIG-Nav using both this dataset and standard public navigation datasets and compare the average validation loss. As illustrated in Table 2, we find that incorporating additional labeled online video data results in lower validation losses, demonstrating the efficiency of our labeled dataset and its processing pipeline.

	Average Validation Loss
w.o. New Data	0.189
w. New Data	0.164

Table 2: Loss comparison with and without Online Video Data.

5.2.4 Ablation Studies

Early Fuse VS. Non-Early Fuse. To verify the effectiveness of our proposed early-fusion structure, we experiment with two versions of PIG-Nav: early-fusion and non-early-fusion, both combined with an MAE-pretrained ViT image encoder, in the Highrise and Sanctuary environments. The key difference between early-fusion and non-early-fusion is that early-fusion allows cross-attention between the observation and goal images, whereas non-early-fusion processes them separately and later fuses them by contrasting their learned [CLS] tokens from the ViT encoder.

Table 3 presents the performance comparison between these two structures. The results indicate that the early-fusion network architecture achieves better navigation performance in both zero-shot and fine-tuning settings, suggesting that early fusion of low-level features from observation and goal images enhances performance in image-goal navigation.

	Highrise		Sanctuary	
	SR	SPL	SR	SPL
Non-Fuse (ZS)	0.487	0.443	0.280	0.217
Early Fuse (ZS)	0.693	0.640	0.540	0.503
Non-Fuse (FT)	0.927	0.860	0.853	0.770
Early Fuse (FT)	0.967	0.907	0.867	0.797

Table 3: SR and SPL performance on two types of model structure across two environments. Experiments run for zero-shot (first two rows) and fine-tuning (last two rows) settings. Performances are averaged on three difficulty levels.

Integration of Pretrained Visual Encoders. We also tested the effect of various pretrained ViT encoders on PIG-Nav’s performance, with the results presented in Table 4.

The experimental results show that ViT pretrained with Self-Supervised Learning (SSL) methods significantly outperforms models trained from scratch. Additionally, ViT pretrained with MAE demonstrates superior performance compared to ViT trained with DINOv2.

We hypothesize that, although DINOv2 encoder architecture provides high-level semantic representations, MAE focuses more on fine-grained details, making it more effective for image-goal navigation tasks.

	Highrise		Sanctuary	
	SR	SPL	SR	SPL
ViT-raw (ZS)	0.380	0.363	0.227	0.210
ViT-DINOv2 (ZS)	0.600	0.577	0.413	0.393
ViT-MAE (ZS)	0.693	0.64	0.540	0.503
ViT-raw (FT)	0.860	0.803	0.613	0.567
ViT-DINOv2 (FT)	0.953	0.890	0.680	0.637
ViT-MAE (FT)	0.967	0.907	0.867	0.797

Table 4: SR and SPL performance on three types of ViT encoders across two environments. Experiments run for both zero-shot (first two rows) and fine-tuning (last two rows) setting. Performances are averaged on three difficulty levels.

Effect of Auxiliary Loss. We also investigate the effect of each auxiliary loss on the performance of PIG-Nav. The experiments are divided into five different settings. The first only acquires the waypoint information as normal navigation tasks. In the following three settings, we remove the goal, distance and global information separately to show these part effects in auxiliary loss. We show the auxiliary losses are all useful to achieve the success rate of PIG-Nav based on the outcome in Table 5. This experiment also suggests that predicting global decision making information are beneficial to the performance of navigation foundation models.

6 Conclusions

In this paper, we introduce PIG-Nav, a novel pretrained image-goal navigation model that advances state-of-the-art visual navigation foundation models from both model design aspect and data aspect. Our early-fusion network, which integrates pretrained Vision Transformer (ViT) encoders, enhances both efficiency and navigation performance. Additionally, we introduce auxiliary loss functions to further optimize pretraining and fine-tuning, improving the model’s adaptability. Furthermore, we demonstrate that newly labeled online video data can serve as a valuable resource for navigation tasks. Our proposed online video labeling pipeline not only enhances dataset quality for PIG-Nav but also holds potential for broader applications, such as Vision-Language-Action (VLA) models for robotic control. Through extensive experiments in both simulated game environments and real-world robotic settings, we validate the effectiveness of PIG-Nav. These results highlight its potential to enhance autonomous navigation, paving the way for further advancements in robotic perception, decision-making, and generalizable navigation policies.

	Highrise		Sanctuary	
	SR	SPL	SR	SPL
Waypoint Only (ZS)	0.613	0.563	0.447	0.410
No Goal (ZS)	0.627	0.590	0.440	0.417
No Distance (ZS)	0.653	0.623	0.487	0.460
No Global (ZS)	0.680	0.620	0.480	0.450
All (ZS)	0.693	0.640	0.540	0.503
Waypoint Only (FT)	0.953	0.883	0.823	0.747
No Goal (FT)	0.967	0.903	0.827	0.763
No Distance (FT)	0.960	0.900	0.840	0.763
No Global (FT)	0.953	0.897	0.840	0.750
All (FT)	0.967	0.907	0.867	0.797

Table 5: SR and SPL performance with different auxiliary loss designs across two environments. Experiments run for both zero-shot (first two rows) and fine-tuning (last two rows) setting. Performances are averaged in three difficulty levels.

An important direction for future research is to assess PIG-Nav’s generalization across more diverse and complex environments, including dynamic and partially observable settings. Additionally, integrating topological mapping could improve the agent’s ability to perform long-horizon navigation and enhance its exploration capabilities in unseen environments. We also hope that the insights presented in this paper may contribute to the design of VLM-based navigation models.

References

- Bar Mayo, Tamir Hazan, and Ayellet Tal. Visual navigation with spatial attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16898–16907, 2021.
- Francisco Bonin-Font, Alberto Ortiz, and Gabriel Oliver. Visual navigation for mobile robots: A survey. *Journal of intelligent and robotic systems*, 53:263–296, 2008.
- Dhruv Shah, Benjamin Eysenbach, Gregory Kahn, Nicholas Rhinehart, and Sergey Levine. Ving: Learning open-world navigation with visual goals. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13215–13222. IEEE, 2021a.
- Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Singh Chplot, Oleksandr Maksymets, et al. Habitat 2.0: Training home assistants to rearrange their habitat. *Advances in neural information processing systems*, 34:251–266, 2021.
- Qiaoyun Wu, Jun Wang, Jing Liang, Xiaoxi Gong, and Dinesh Manocha. Image-goal navigation in complex environments via modular learning. *IEEE Robotics and Automation Letters*, 7(3):6902–6909, 2022.
- Heming Du, Xin Yu, and Liang Zheng. Vtnet: Visual transformer network for object goal navigation. *arXiv preprint arXiv:2105.09447*, 2021.
- Dhruv Shah, Ajay Sridhar, Arjun Bhorkar, Noriaki Hirose, and Sergey Levine. Gnm: A general navigation model to drive any robot. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7226–7233. IEEE, 2023a.
- Dhruv Shah, Ajay Sridhar, Nitish Dashora, Kyle Stachowicz, Kevin Black, Noriaki Hirose, and Sergey Levine. Vint: A foundation model for visual navigation. *arXiv preprint arXiv:2306.14846*, 2023b.
- Ajay Sridhar, Dhruv Shah, Catherine Glossop, and Sergey 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.
- Jiazhao Zhang, Kunyu Wang, Rongtao Xu, Gengze Zhou, Yicong Hong, Xiaomeng Fang, Qi Wu, Zhizheng Zhang, and He Wang. Navid: Video-based vlm plans the next step for vision-and-language navigation. *arXiv preprint arXiv:2402.15852*, 2024a.
- Jiazhao Zhang, Kunyu Wang, Shaoan Wang, Minghan Li, Haoran Liu, Songlin Wei, Zhongyuan Wang, Zhizheng Zhang, and He Wang. Uni-navid: A video-based vision-language-action model for unifying embodied navigation tasks. *arXiv preprint arXiv:2412.06224*, 2024b.

- Haresh Karnan, Anirudh Nair, Xuesu Xiao, Garrett Warnell, Sören Pirk, Alexander Toshev, Justin Hart, Joydeep Biswas, and Peter Stone. Socially compliant navigation dataset (scand): A large-scale dataset of demonstrations for social navigation. *IEEE Robotics and Automation Letters*, 7(4):11807–11814, 2022.
- Yuanzhi Liu, Yujia Fu, Minghui Qin, Yufeng Xu, Baoxin Xu, Fengdong Chen, Bart Goossens, Poly ZH Sun, Hongwei Yu, Chun Liu, et al. Botanicgarden: A high-quality dataset for robot navigation in unstructured natural environments. *IEEE Robotics and Automation Letters*, 9(3):2798–2805, 2024.
- Naoki Yokoyama, Ram Ramrakhya, Abhishek Das, Dhruv Batra, and Sehoon 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.
- Saurabh Gupta, James Davidson, Sergey Levine, Rahul Sukthankar, and Jitendra Malik. Cognitive mapping and planning for visual navigation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2616–2625, 2017.
- Karl Pertsch, Oleh Rybkin, Frederik Ebert, Shenghao Zhou, Dinesh Jayaraman, Chelsea Finn, and Sergey Levine. Long-horizon visual planning with goal-conditioned hierarchical predictors. *Advances in Neural Information Processing Systems*, 33:17321–17333, 2020.
- Devendra Singh Chaplot, Dhiraj Prakashchand Gandhi, Abhinav Gupta, and Russ R Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. *Advances in Neural Information Processing Systems*, 33:4247–4258, 2020.
- Obin Kwon, Jeongho Park, and Songhwai Oh. Renderable neural radiance map for visual navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9099–9108, 2023.
- Obin Kwon, Nuri Kim, Yunho Choi, Hwiyeon Yoo, Jeongho Park, and Songhwai Oh. Visual Graph Memory With Unsupervised Representation for Visual Navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15890–15899, 2021.
- Karmesh Yadav, Arjun Majumdar, Ram Ramrakhya, Naoki Yokoyama, Alexei Baevski, Zsolt Kira, Oleksandr Maksymets, and Dhruv Batra. Ovrl-v2: A simple state-of-art baseline for imagenav and objectnav. *arXiv preprint arXiv:2303.07798*, 2023.
- Xinyu Sun, Peihao Chen, Jugang Fan, Jian Chen, Thomas Li, and Mingkui Tan. Fgprompt: fine-grained goal prompting for image-goal navigation. *Advances in Neural Information Processing Systems*, 36, 2024.
- Nuri Kim, Obin Kwon, Hwiyeon Yoo, Yunho Choi, Jeongho Park, and Songhwai Oh. Topological semantic graph memory for image-goal navigation. In *Conference on Robot Learning*, pages 393–402. PMLR, 2023.
- Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.
- Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 3357–3364. IEEE, 2017.
- Haobin Shi, Lin Shi, Meng Xu, and Kao-Shing Hwang. End-to-end navigation strategy with deep reinforcement learning for mobile robots. *IEEE Transactions on Industrial Informatics*, 16(4):2393–2402, 2019.
- Chiyuan Zhang, Oriol Vinyals, Remi Munos, and Samy Bengio. A Study on Overfitting in Deep Reinforcement Learning, April 2018.
- Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision, February 2021.
- Nikhilanj Pelluri. Transformers for Image-Goal Navigation, May 2024.
- Gengze Zhou, Yicong Hong, Zun Wang, Xin Eric Wang, and Qi Wu. NavGPT-2: Unleashing Navigational Reasoning Capability for Large Vision-Language Models, September 2024.
- Jiaqi Chen, Bingqian Lin, Xinmin Liu, Lin Ma, Xiaodan Liang, and Kwan-Yee K Wong. Affordances-oriented planning using foundation models for continuous vision-language navigation. *arXiv preprint arXiv:2407.05890*, 2024.
- Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3674–3683, 2018.

- Hong-You Chen, Zhengfeng Lai, Haotian Zhang, Xinze Wang, Marcin Eichner, Keen You, Meng Cao, Bowen Zhang, Yinfei Yang, and Zhe Gan. Contrastive Localized Language-Image Pre-Training, February 2025.
- Ainaz Eftekhari, Luca Weihs, Rose Hendrix, Ege Caglar, Jordi Salvador, Alvaro Herrasti, Winson Han, Eli VanderBil, Aniruddha Kembhavi, Ali Farhadi, et al. The one ring: a robotic indoor navigation generalist. *arXiv preprint arXiv:2412.14401*, 2024.
- Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022.
- Gregor Bachmann and Vaishnavh Nagarajan. The pitfalls of next-token prediction. *arXiv preprint arXiv:2403.06963*, 2024.
- Edward S Hu, Kwangjun Ahn, Qinghua Liu, Haoran Xu, Manan Tomar, Ada Langford, Dinesh Jayaraman, Alex Lamb, and John Langford. Learning to achieve goals with belief state transformers. *arXiv preprint arXiv:2410.23506*, 2024.
- Noriaki Hirose, Fei Xia, Roberto Martín-Martín, Amir Sadeghian, and Silvio Savarese. Deep visual mpc-policy learning for navigation. *IEEE Robotics and Automation Letters*, 4(4):3184–3191, 2019.
- Dhruv Shah, Benjamin Eysenbach, Gregory Kahn, Nicholas Rhinehart, and Sergey Levine. Rapid exploration for open-world navigation with latent goal models. *arXiv preprint arXiv:2104.05859*, 2021b.
- Gregory Kahn, Adam Villaflor, Bosen Ding, Pieter Abbeel, and Sergey Levine. Self-supervised deep reinforcement learning with generalized computation graphs for robot navigation. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 5129–5136. IEEE, 2018.
- Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2636–2645, 2020.
- Samuel Triest, Matthew Sivaprakasam, Sean J Wang, Wenshan Wang, Aaron M Johnson, and Sebastian Scherer. Tartandrive: A large-scale dataset for learning off-road dynamics models. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2546–2552. IEEE, 2022.
- Noriaki Hirose, Dhruv Shah, Ajay Sridhar, and Sergey Levine. Sacson: Scalable autonomous control for social navigation. *IEEE Robotics and Automation Letters*, 9(1):49–56, 2023.
- Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model’s perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

7 Training Tasks for PIG-Nav

Notations. We define the provided navigation trajectory as:

$$\tau = (o_0, o_1, \dots, o_T; p_0, p_1, \dots, p_T)$$

where T represents the total number of steps in the trajectory. The first observation $o_{\text{cur}} = o_0$ corresponds to the current image, while the final observation $o_{\text{goal}} = o_T$ represents the goal image. Each observation o_k is an RGB image that captures the agent’s 90-degree field of view directly in front. Each pose p_k is represented as (x, y, ψ) , where x and y denote the agent’s coordinates, and ψ represents the yaw angle, indicating the agent’s orientation.

Waypoint Action Prediction. The goal of waypoint action prediction is to predict the next $N_{\text{waypoint}} = 10$ consecutive steps in the provided navigation trajectory based on the current image observation o_{cur} and the goal image observation o_{goal} . This chunk-based action prediction approach has been adopted in recent works [Bachmann and Nagarajan, 2024, Hu et al., 2024] as an improvement over methods that predict only the next immediate action.

The waypoint action loss is defined as:

$$L^{\text{waypoint}}(\tau) = \sum_{k=1}^{N_{\text{waypoint}}} D^{\text{pos_yaw}}(f_{\theta}^{\text{waypoint}}(o_{\text{cur}}, o_{\text{goal}})_k, p_{0 \rightarrow k})$$

where the waypoint function $f_{\theta}^{\text{waypoint}}(o_{\text{cur}}, o_{\text{goal}})_k$ denotes the predicted waypoint action generated by PIG-Nav, and $p_{0 \rightarrow k}$ denotes the pose of $t = k$ relative to the initial pose at $t = 0$.

The following position-yaw L_2 metric quantifies the discrepancy between predicted and ground-truth positions and yaw angles, assessing the accuracy of the model’s waypoint predictions.

$$D^{\text{pos_yaw}}(p_1, p_2) = (x_2 - x_1)^2 + (y_2 - y_1)^2 \\ + (\cos(\psi_2) - \cos(\psi_1))^2 + (\sin(\psi_2) - \sin(\psi_1))^2$$

Relative Pose to Goal. In the relative pose prediction task, the model estimates the pose of the goal image relative to the observation image. This task enables the model to capture global task information by understanding the spatial relationship between the current observation and the goal image.

The relative position loss function, which measures the discrepancy between the predicted and ground-truth relative positions and orientations, is defined as:

$$L^{\text{relative}}(o_{\text{cur}}, o_{\text{goal}}; p_0, p_T) \\ = D^{\text{pos_yaw}}(f_{\theta}^{\text{relative}}(o_{\text{cur}}, o_{\text{goal}}), p_{0 \rightarrow T})$$

where $f_{\theta}^{\text{relative}}(o_{\text{cur}}, o_{\text{goal}})$ denotes the predicted relative pose, and $p_{0 \rightarrow T}$ represents the ground-truth relative pose of the goal image with respect to the initial observation.

Navigation Distance Prediction. In the navigation distance prediction task, our model is trained to estimate the total distance required for the agent to navigate from its current state to the goal state. By optimizing this loss, the model learns to infer the connectivity and traversability between different locations in the environment, enabling more efficient path planning.

The navigation distance between the current position and the goal position is defined as:

$$\text{nav_distance}(\tau) = \sum_{k=0}^{T-1} \|(x_{k+1} - x_k, y_{k+1} - y_k)\|_2^2$$

The corresponding navigation distance loss function is formulated as:

$$L^{\text{nav_distance}}(\tau) = (f_{\theta}^{\text{nav_distance}}(o_{\text{cur}}, o_{\text{goal}}) - \text{nav_distance}(\tau))^2$$

where $f_{\theta}^{\text{nav_distance}}(o_{\text{cur}}, o_{\text{goal}})$ represents the model’s predicted navigation distance, and $\text{nav_distance}(\tau)$ is the ground-truth distance computed from the trajectory.

Global Path Prediction. To further enhance the agent’s ability to predict long-range positions, we introduce the global path prediction task. This task requires the model to predict $N_{\text{global}} = 10$ intermediate waypoints, which are equally spaced in time from the current time step to the total trajectory length T . These predicted waypoints generated by PIG-Nav are compared to the ground-truth trajectory to compute the loss function.

The corresponding global path prediction loss is defined as:

$$L^{\text{global}}(\tau) = \sum_{k=1}^{N_{\text{global}}} D^{\text{pos_yaw}}(f_{\theta}^{\text{global}}(o_{\text{cur}}, o_{\text{goal}})_k, p_{0 \rightarrow \lfloor \frac{k \times T}{N_{\text{global}}} \rfloor})$$

where $f_{\theta}^{\text{global}}(o_{\text{cur}}, o_{\text{goal}})_k$ represents the predicted waypoint at step k , and $p_{0 \rightarrow \lfloor \frac{k \times T}{N_{\text{global}}} \rfloor}$ denotes the corresponding ground-truth waypoint in the trajectory.

8 Pretraining Dataset

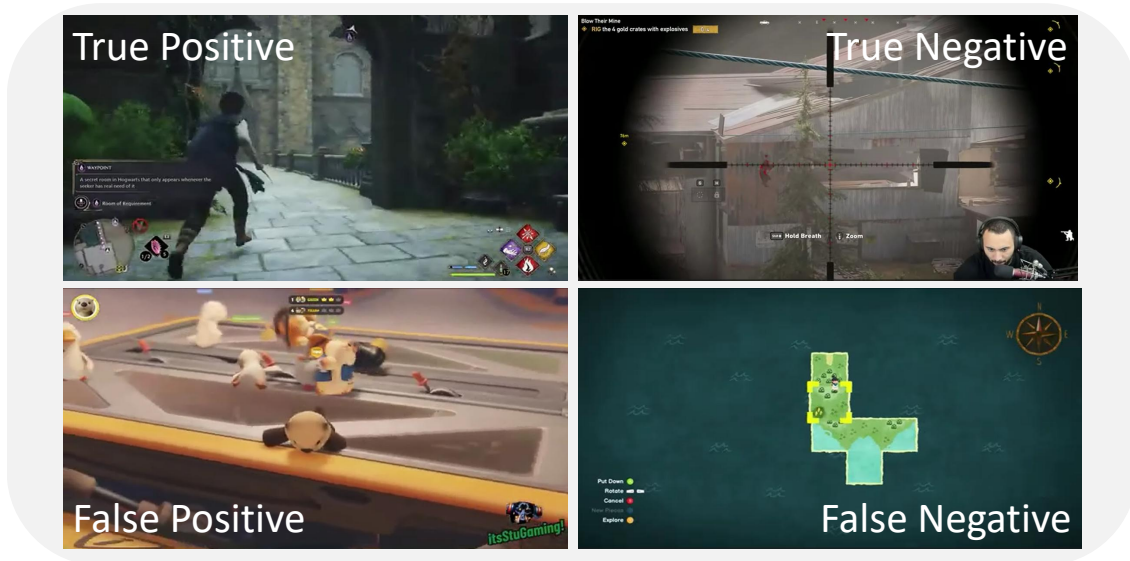


Figure 5: Examples of Qwen2 two steps filtering.

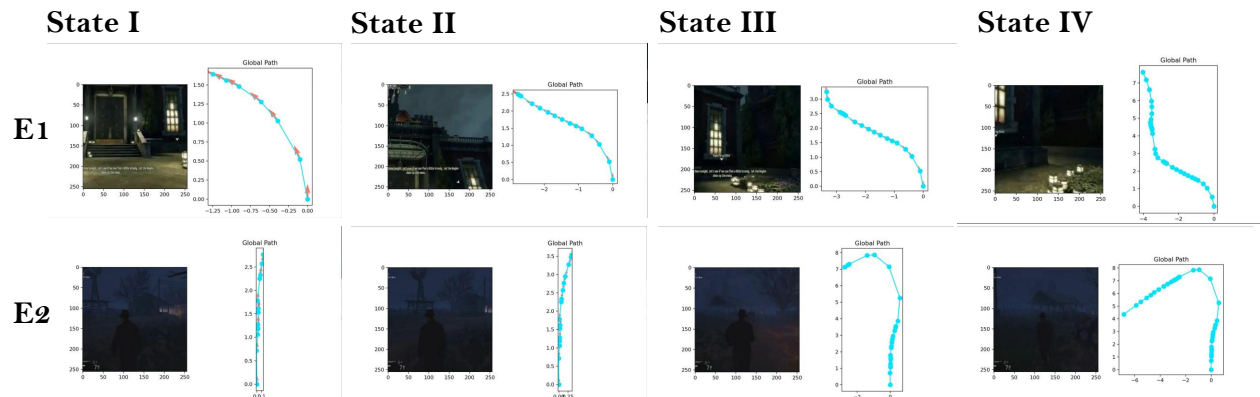


Figure 6: Visualization of IDM labeled data. Four stages from Stage I to Stage II for two environment E1 and E2 are listed in sequence.

8.1 Open Public Navigation Datasets

While we incorporate publicly available datasets, certain data sources remain unavailable compared to those used in [Shah et al., 2023a,b, Sridhar et al., 2024]. However, despite being pretrained on a smaller dataset, our model outperforms existing approaches, demonstrating its effectiveness and robustness.

For existing datasets, two key aspects require careful consideration. First, data cleaning: we identified noise in some datasets, such as abnormally large position changes between consecutive time steps. To ensure data quality and facilitate stable training, we filtered out data points where position changes exceeded five times the average displacement. Second, data normalization: since datasets vary in scale and properties, transforming them into a unified space is crucial for effective training. We experimented with three normalization techniques and selected the one that yielded the best performance. Further details on these methods are provided in the supplementary material.

8.2 Labeling Online Video Data

Filtering. We show the final classification outcome of our two steps filtering by Qwen2 Wang et al. [2024] VLM in Figure 5. In details, Figure 5 show the four types of Qwen2 model classification outputs, including True Positive, False Negative, True Negative and False Negative separately.

Additionally, in the first step filtering, we focus on our navigation tasks and use the prompt *"If this video can be used in the construction of navigation task dataset, then return True, else return False."* For the second step filtering, we test total 12 prompts in Qwen2 with our labeling 500 positive samples, which are the video segments with True label by Qwen2 in classification. The prompt *"If this video does not contain the interaction with game menu and this video measures the movement, then return True, else return False."* performs best in our criterion which combines 80% precision and 20% recall as the final selection metric.

IDM Architecture. For data labeling, We utilize the ViT model architecture as PIG-Nav to encode the change of two consecutive frames and construct an attention head to decode the navigation action to vector representations $[\Delta x, \Delta y, \cos \Delta\psi, \sin \Delta\psi]$. We show the labeled output of our IDM in Figure 6.

We also propose to leverage pre-trained latent action representation encoders based on our final raw video segments firstly to enhance the model abilities that it measures the diverse videos. Then we fine-tune this latent action representation encoders with a 3 layers MLP as its Inverse Dynamics Model to build the relationship between these latent action representations and the low level action representations. We highlight that the pre-trained latent action encoder can capture the similarity among similar action motion in video segments and use the high dimension latent action vector to represent it. We find this structure needs a precise IDM model design to translate the latent action, representing as the high dimensional vector, to the low level action as the navigation action vector.

9 Ablation Study in Details

We explain our ablation studies in details for three difficulty settings in Table 6, Table 7 and Table 8.

Table 6 measures the effectiveness of our proposed early-fusion structure, which performs better than non-fuse in all difficulty settings for both Highrise and Sanctuary environments.

	Easy		Highrise				Easy		Sanctuary			
	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL
Non-Fuse (ZS)	0.74	0.69	0.48	0.45	0.24	0.19	0.56	0.44	0.20	0.15	0.08	0.06
Early Fuse (ZS)	0.90	0.86	0.72	0.68	0.46	0.38	0.84	0.80	0.48	0.45	0.30	0.26
Non-Fuse (FT)	1.00	0.96	0.94	0.88	0.84	0.74	1.00	0.94	0.84	0.75	0.72	0.62
Early Fuse (FT)	1.00	0.97	1.00	0.94	0.90	0.81	1.00	0.96	0.84	0.75	0.76	0.68

Table 6: SR and SPL performance across two environments and three difficulty levels with different model structure. Experiments run for zero-shot (first two rows) and fine-tuning (last two rows) setting. Two types of Model structure include non-early fuse and early fuse. All values are rounded to the nearest hundredth.

In Table 7, we test different ViT encoder structures and find the MAE structure performs best in all three difficulty settings for both Highrise and Sanctuary environments.

	Highrise						Sanctuary					
	Easy		Medium		Hard		Easy		Medium		Hard	
	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL
ViT-raw (ZS)	0.76	0.73	0.26	0.25	0.12	0.11	0.46	0.43	0.14	0.13	0.08	0.07
ViT-DINOv2 (ZS)	0.82	0.81	0.70	0.67	0.28	0.25	0.70	0.67	0.32	0.30	0.22	0.21
ViT-MAE (ZS)	0.90	0.86	0.72	0.68	0.46	0.38	0.84	0.80	0.48	0.45	0.30	0.26
ViT-raw (FT)	1.00	0.96	0.86	0.80	0.72	0.65	0.88	0.83	0.58	0.53	0.38	0.34
ViT-DINOv2 (FT)	1.00	0.97	1.00	0.94	0.86	0.76	0.96	0.92	0.62	0.58	0.46	0.41
ViT-MAE (FT)	1.00	0.97	1.00	0.94	0.90	0.81	1.00	0.96	0.84	0.75	0.76	0.68

Table 7: SR and SPL performance across two environments and three difficulty levels with different pretrained model encoder. Experiments run for both zero-shot (first two rows) and fine-tuning (last two rows) setting. Three model encoders including ViT-raw, ViT-DINOv2 and ViT-MAE apply in experiments. All values are rounded to the nearest hundredth.

We show the effectiveness of our auxiliary loss design in Table 8. Five settings including waypoint only, no goal, no distance, no global and all auxiliary supports are tested for both Highrise and Sanctuary environments.

	Highrise						Sanctuary					
	Easy		Medium		Hard		Easy		Medium		Hard	
	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL
Waypoint Only (ZS)	0.82	0.77	0.70	0.64	0.32	0.28	0.78	0.72	0.42	0.38	0.14	0.13
No Goal (ZS)	0.82	0.79	0.66	0.63	0.40	0.35	0.78	0.76	0.38	0.35	0.16	0.14
No Distance (ZS)	0.86	0.84	0.72	0.69	0.38	0.34	0.82	0.79	0.46	0.43	0.18	0.16
No Global (ZS)	0.88	0.84	0.74	0.68	0.42	0.34	0.80	0.76	0.46	0.42	0.18	0.17
All (ZS)	0.90	0.86	0.72	0.68	0.46	0.38	0.84	0.80	0.48	0.45	0.30	0.26
Waypoint Only (FT)	1.00	0.96	1.00	0.93	0.86	0.76	1.00	0.95	0.84	0.73	0.66	0.56
No Goal (FT)	1.00	0.97	1.00	0.93	0.90	0.81	1.00	0.96	0.82	0.74	0.66	0.59
No Distance (FT)	1.00	0.97	1.00	0.94	0.88	0.79	1.00	0.96	0.84	0.74	0.68	0.59
No Global (FT)	1.00	0.97	1.00	0.94	0.86	0.78	1.00	0.95	0.84	0.72	0.68	0.58
All (FT)	1.00	0.97	1.00	0.94	0.90	0.81	1.00	0.96	0.84	0.75	0.76	0.68

Table 8: SR and SPL performance across two environments and three difficulty levels with different auxiliary loss design. Experiments run for both zero-shot (first two rows) and fine-tuning (last two rows) setting. Five types of design includes waypoint only, without goal, without distance, without global and all. All values are rounded to the nearest hundredth.