VLD: Visual Language Goal Distance for Reinforcement Learning Navigation

Lazar Milikic[1,2] Manthan Patel[1] Jonas Frey[1,3,4]
[1]ETH Zurich [2]EPFL [3]Stanford University [4]UC Berkeley
lazar.milikic@epfl.ch patelm@ethz.ch jonfrey@stanford.edu

Abstract

Training end-to-end policies from image data to directly predict navigation actions for robotic systems has proven inherently difficult. Existing approaches often suffer from either the sim-to-real gap during policy transfer or a limited amount of training data with action labels. To address this problem, we introduce Vision–Language Distance (VLD) learning, a scalable framework for goal-conditioned navigation that decouples perception learning from policy learning. Instead of relying on raw sensory inputs during policy training, we first train a self-supervised distance-to-goal predictor on internet-scale video data. This predictor generalizes across both image- and text-based goals, providing a distance signal that can be minimized by a reinforcement learning (RL) policy. The RL policy can be trained entirely in simulation using privileged geometric distance signals, with injected noise to mimic the uncertainty of the trained distance predictor. At deployment, the policy consumes VLD predictions, inheriting semantic goal information—"where to go"—from large-scale visual training while retaining the robust low-level navigation behaviors learned in simulation. We propose using ordinal consistency to assess distance functions directly and demonstrate that VLD outperforms prior temporal distance approaches, such as ViNT and VIP. Experiments show that our decoupled design achieves competitive navigation performance in simulation while supporting flexible goal modalities, providing an alternative and, most importantly, scalable path toward reliable, multimodal navigation policies.

1. Introduction

Scalable training on diverse, real-world(-like), internet-scale datasets has emerged as one of the most promising approaches for learning broadly capable navigation policies. Recent advances in embodied AI have shown that policies trained on large-scale multimodal data exhibit robustness and generalization that narrow, task-specific approaches fail to achieve [4]. This highlights the importance of pursuing navigation research under conditions that better mirror the real world, rather than limiting it to controlled lab settings [26, 50].

Traditional imitation learning and expert demonstration pipelines offer a straightforward pathway for training navigation agents [2]; however, since expert-labeled actions—typically in the form of future trajectories represented as 2D or 2.5D waypoints—are expensive to collect, they remain fundamentally constrained by the scarcity of annotated trajectories [46]. Initial large-scale efforts, such as GNM [39], ViNT [40], and NoMaD [41], have collected on the order of 200 hours of demonstrations across multiple robotic embodiments, illustrating the promise of cross-embodiment learning through curated multi-robot datasets. Nevertheless, these datasets still exhibit strong *straight-walking bias* (the majority of labeled actions correspond to simply moving forward), raising doubts about whether annotation-heavy pipelines can scale to broadly generalizable navigation. Moreover, such approaches largely overlook embodiment-specific signals and rely solely on image inputs, thereby precluding the use of proprioception, which is often crucial for effective navigation.

Reinforcement learning (RL) enables agents to develop robustness through interaction and exploration. However, due to its sample inefficiency in real-world environments, RL training for navigation heavily depends on simulators. Collecting diverse environmental assets and simulating photorealistic images with corresponding physical interactions remain open research challenges, hindering the effective transfer of visual navigation policies from simulation to reality [3, 26, 35, 50, 52].

In this work, we propose an alternative framework that combines the strengths of imitation learning and RL while avoiding their annotation and sim-to-real bottlenecks. Rather than conditioning policies on raw visual inputs, we decouple the problem into two stages: (i) training a scalable, self-supervised *distance-to-goal* predictor on internet-scale video data, and (ii) training RL policies in simulation that consume this distance signal directly. Unlike expert-labeled

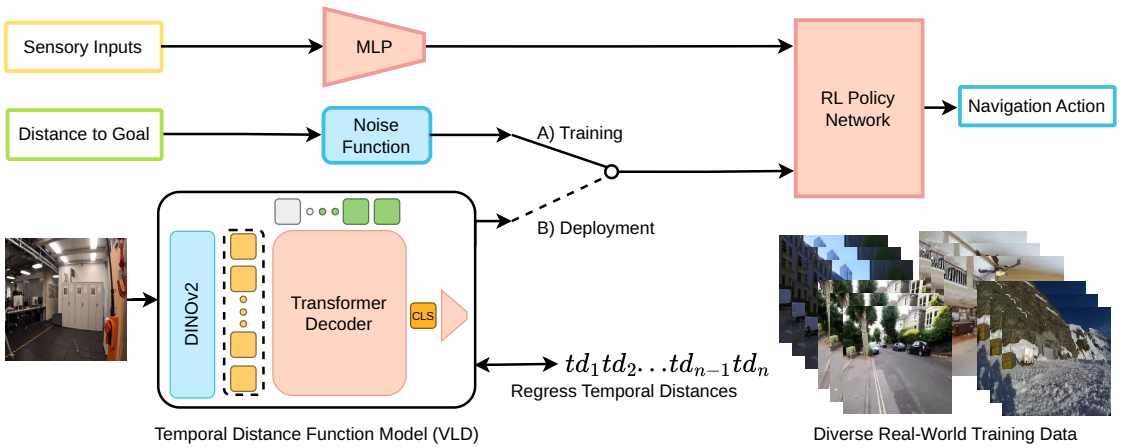


Figure 1. **Overview of our framework.** (A) Training stage: we separately train a temporal *Vision–Language Distance (VLD)* function on diverse real-world and synthetic video datasets, and an RL navigation policy in simulation using geometric distance-to-goal signals with injected noise to mimic real predictor uncertainty. (B) Deployment stage: the trained RL policy consumes predictions from the learned VLD model—specified by either image or text goals—to navigate in simulated and real-world environments.

trajectories, the distance signal—defined as the number of temporal steps from the current view to a goal observation—can be obtained from sequential video without annotation, making it both scalable and domain-agnostic.

We curate 3,000 hours of diverse trajectories from synthetic environments [35], internet-scale walking videos [26], and embodied datasets [9, 12, 17], spanning a wide variety of indoor and outdoor scenes. This design lets large-scale video learning provide the perception of *where* to navigate, while RL in simulation focuses on training a sensory–motor control policy for *how* to navigate, without relying on vision and instead using geometric signals that transfer robustly from simulation to the real world [22]. In this way, distance prediction emerges as a central self-supervised signal for general navigation.

To broaden applicability, we extend the problem to the more general setting of *vision–language goal navigation*. In practice, specifying a navigation target via natural language is often more convenient than providing a reference image, especially in real-world deployments where an image may not be available. To enable this, we bootstrap a subset of our data with vision–language models [23], generating textual goal descriptions paired with video trajectories [14]. This allows us to train a multimodal distance predictor that can operate with either image or text goals. We refer to this function as the **V**ision–**L**anguage **D**istance (VLD), which estimates the distance between the current egocentric view and a goal specified in either modality.

On the control side, to reflect the inherent uncertainty of distance-function predictions, we expose the RL policy during simulation training to noisy distance signals generated by a lightweight MLP trained to mimic the error distribution of our learned predictor. This prepares the policy for the noisy predictions it will encounter at deployment.

Together, these components form the foundation of our framework for scalable, multimodal, and transferable navigation policies. Our main contributions are: (i) a scalable framework for training **vision–language distance (VLD) predictors** on internet-scale real-world data, outperforming prior designs for temporal distance functions such as ViNT [40] and VIP [27]; (ii) a **novel evaluation methodology** for temporal distance functions, introducing *ordinal consistency* as the first principled way to evaluate distance predictors in isolation rather than only through downstream policy success; and (iii) a **navigation framework** that decouples RL policy training in simulation from perception learning on internet-scale data, combining robust RL control with scalable semantic understanding.

2. Related Work

2.1. Temporal Distance Functions

Temporal distance has evolved from graph-based heuristics to scalable self-supervised learning. ViNG regressed temporal steps between observation pairs with negative mining and planned over an experience graph [37], but lacked closed-loop RL policies for robustness. Building on this idea, ViNT and NoMaD applied Transformer encoders over observation and goal embeddings, supervising both distance and short-horizon actions [40, 41]. While validating distance as a useful signal, these approaches primarily treated it as an auxiliary to the policy, leaving open the question of how well the distance estimator generalizes in isolation.

VIP reframed representation learning as goal-conditioned value prediction, where Euclidean distance in latent space acts as a reward [27]. Quasimetric RL formal-

ized multi-goal optimal values as quasimetric distances, improving sample efficiency across state- and image-based tasks [29, 44]. Yet most evaluations still emphasize downstream success, rather than direct validation of the distance function itself.

Motivated by these gaps, we adopt pairwise supervision with negative mining but design a Transformer *decoder* that queries distance from the goal signal. Unlike prior work, we treat the distance estimator as a first-class product, and replace mean-squared error with an inlier–outlier Gaussian mixture NLL [30], yielding calibrated predictions and uncertainty estimates that provide a reliability signal for downstream policies. Furthermore, we introduce a direct evaluation methodology for distance functions based on *ordinal consistency*, which tests whether predicted distances rank pairs in the same order as ground-truth distances—providing a principled way to validate distance estimators in isolation rather than only through downstream policy success.

2.2. Navigation Policies

Goal-conditioned RL trains policies conditioned on state, image, or language goals [25]. While point-goal tasks are largely solved [45], image- and language-goal navigation remain challenging due to reliance on simulators and sim-to-real gaps [48, 52]. To mitigate this, we train RL policies with privileged geometric distances in simulation and replace them at deployment with predictions from our learned VLD, reducing dependence on hard-to-obtain simulator photorealism for navigation training [7]. Related distillation and privileged-information approaches [10, 21, 28] differ in that we directly substitute a raw privileged value with a learned predictor, offering a simple yet effective sim-to-real bridge.

Imitation learning from demonstrations [6, 26, 38–41] has improved generalization through curated datasets, but remains limited by annotation costs. Our approach avoids action labels entirely, scaling supervision via self-supervised distance learning on internet-scale video.

2.3. Multimodal Navigation

The emergence of vision–language–action (VLA) models [4, 19] has pushed toward tightly coupling perception, language understanding, and control in a single foundation model. While this integration achieves impressive generality, it comes with a high computational cost and limited scalability. In contrast, our approach explicitly *decouples* perception from control: we extend the temporal distance function with CLIP-based text encoding, enabling a **Vision–Language Distance (VLD)** predictor that handles both image and language goals. This design maintains the practicality of multimodal goals while simplifying scaling, since perception can benefit from internet-scale pretraining without requiring full-scale language models.

3. Method

3.1. Temporal Distance Function

We define the temporal distance function as a mapping from the current egocentric observation and a goal specification (either visual or textual) to the predicted number of steps required for an agent to reach the goal under an optimal policy. Formally, given a current observation o_t and a goal description g, the temporal distance function $\mathcal{T}_d(o_t, g)$ estimates the expected temporal horizon between the two. Unlike action-labeled supervision, temporal distance is naturally available from sequential video or trajectory data, making it a scalable, annotation-free signal that can be mined from large-scale video datasets.

3.1.1. Evaluation Methodology

Our goal is not to obtain perfectly accurate predictions of temporal distance, which is inherently ambiguous and often impossible to infer exactly from a single view (e.g., distance from a bedroom to a kitchen may vary drastically across apartments). Instead, what matters is that the learned function provides a *useful ordering*: it should reliably distinguish between near and far goals—for instance, recognizing that from a bedroom view, the kitchen is much closer than the city park—even if the absolute values are imprecise.

We evaluate this principal property with *ordinal consistency*, which measures whether predicted distances decrease as the agent approaches the goal and increase when it moves away. This is quantified using *Kendall's* τ, a rank correlation coefficient, lying in the range $[-1, 1]$, where $\tau = 1$ indicates perfect agreement, $\tau = -1$ indicates complete disagreement, and $\tau = 0$ corresponds to random ordering [18]. It is fully *scale-invariant*, capturing correlation between predicted and ground truth distances, and aligning with our focus on relative rather than absolute accuracy.

3.1.2. Model architecture

Encoders. RGB observations are encoded with DINOv2 (small) [32]. Text goals are processed by a CLIP text encoder (ViT-B/32) [33], projected into the same dimensional space as DINOv2-small tokens. Both encoders remain frozen in all experiments.

Goal tokens. Image and text tokens are concatenated as $G = [G_{\text{text}}; G_{\text{img}}]$, with modality and positional embeddings added. Missing modalities are masked, enabling the model to support image-only, text-only, or joint goals.

Decoder and outputs. An L-layer Transformer decoder attends from observation queries to the goal memory G, yielding outputs Z. From the CLS token, we predict:

$$\hat{t} = \mathrm{ReLU}(\mathrm{MLP}_t(Z_{\text{CLS}})), \qquad \hat{c} = \sigma(\mathrm{MLP}_c(Z_{\text{CLS}})),$$

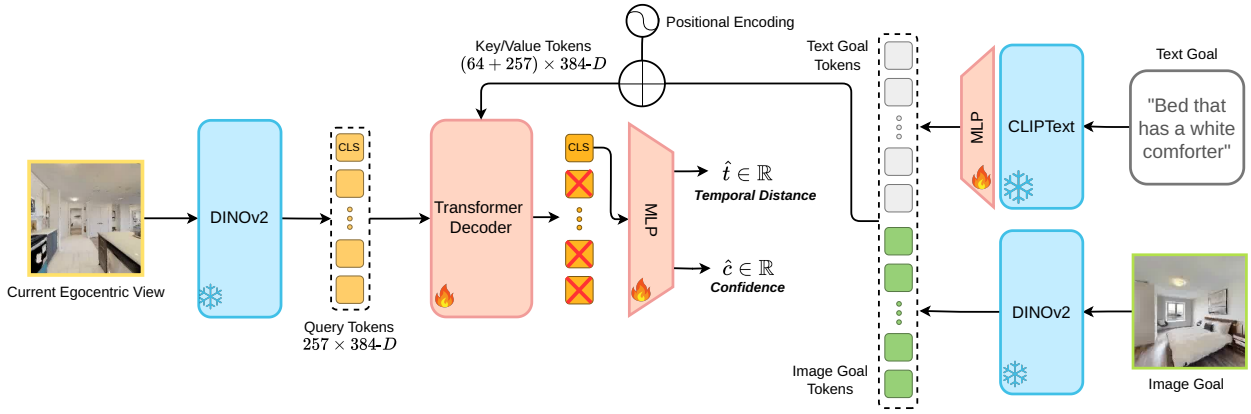


Figure 2. **Vision–Language Distance (VLD) architecture.** Egocentric observations and a goal (image and/or text) are encoded into tokens using frozen backbones (DINOv2 for images, CLIP for text). Text tokens are projected into the DINOv2 embedding space so that both modalities lie in the same space as the observation tokens. A Transformer decoder attends from observation queries to goal tokens, and the CLS output is used by MLP heads to predict temporal distance and a confidence score.

where $\hat{t} \geq 0$ is the predicted temporal distance and $\hat{c} \in [0, 1]$ is a confidence score. The confidence aligns naturally with our Gaussian mixture NLL loss (Section 3.1.3), providing a reliability signal for downstream policies.

3.1.3. Training Procedure

We frame distance prediction as a (self-)supervised regression problem. For each trajectory we sample observation pairs o_i and o_j such that $i \leq j$ and the $td = j - i$ is their temporal distance, where $td \leq td_{\max}$ for a fixed $td_{\max}$, forming the dataset of triplets $\{(o_i, o_j, j - i)\}_{i,j}$. The training algorithm is described in detail in Appendix A.

Because exact distance prediction is inherently noisy and often ambiguous (as discussed in Section 3.1.1), we train the model to produce both a prediction $\hat{t}_n$ and an associated confidence $\hat{c}_n \in [0, 1]$. Intuitively, the confidence allows the model to downweight uncertain cases (e.g., long-horizon pairs with little to no visual overlap) while still utilizing more reliable near-goal predictions.

Concretely, we model the likelihood of the ground-truth temporal distance td_n given the prediction $\hat{t}_n$ as a *inlier–outlier Gaussian mixture* [30]: with probability $\hat{c}_n$, the error is drawn from a tight "reliable" Gaussian with variance σ_R^2, and with probability $1 - \hat{c}_n$, it comes from a broader "outlier" Gaussian with variance σ_O^2, where $\sigma_R \ll \sigma_O$. Then the training objective is the negative log-likelihood (NLL) over a batch of N samples:

$$\mathcal{L} = -\frac{1}{N} \sum_{n=1}^{N} \log \left[\hat{c}_n \frac{1}{\sqrt{2\pi}\,\sigma_R} \exp\!\left(-\frac{(td_n - \hat{t}_n)^2}{2\sigma_R^2}\right) + (1 - \hat{c}_n) \frac{1}{\sqrt{2\pi}\,\sigma_O} \exp\!\left(-\frac{(td_n - \hat{t}_n)^2}{2\sigma_O^2}\right) \right]. \tag{1}$$

This inlier–outlier Gaussian mixture NLL formulation has two benefits: (i) it encourages calibrated uncertainty estimates that can be propagated to downstream policies, and (ii) it reduces the penalty for inherently ambiguous predictions, preventing the model from overfitting to noisy supervision.

3.1.4. Training Data

To train our distance function, we combine three complementary data sources: (i) synthetic trajectories from simulators, (ii) "in-the-wild" internet-scale videos, and (iii) embodiment datasets collected on real robots. A detailed summary of the dataset and its collection is provided in Appendix B.

3.2. Navigation Policy

We formulate navigation as a variant of the *point-goal navigation* task: the agent is placed in a 3D environment with an unknown map and must reach a goal location specified by its geometric distance from the agent's current position. During training, this distance-to-goal signal is provided directly by the simulator; at deployment, it is replaced by predictions from our learned VLD model. This setup enables the RL agent to learn generalizable navigation behaviors independently of simulator photorealism, while grounding its decision-making in a scalar signal that can also be inferred from real-world images. The limitation of this formulation is that the policy cannot directly rely on visual information to infer its global position—rather, it must reason implicitly about spatial structure through the distance signal alone.

3.2.1. Policy Network

The policy receives two types of inputs: (i) **sensory information**, providing minimal obstacle awareness and internal agent state, and (ii) the **distance-to-goal** and its **confidence**. During training, the distance signal comes from the simu-

lator with injected noise; at deployment, it is replaced by VLD predictions (see Section 3.2.2 and Section 4.2).

Inputs are fed into an LSTM core that maintains temporal memory. The LSTM output is processed by two MLP heads: a *policy head* that outputs navigation actions and a *value head* used for PPO optimization [36]. The noise-injected distance signal during training encourages robustness and mimics the uncertainty patterns of the VLD model.

3.2.2. Noise Function

A key challenge in our framework is that the learned distance function is inherently noisy and ambiguous. Training a policy purely on perfect, noise-free distances from the simulator would therefore produce brittle behaviors that fail at deployment. To address this, we perturb the privileged distance signal during RL training, so that the policy experiences inputs with a noise structure resembling what it will later encounter from the VLD predictor.

Our main noise injection approach is *geometric overlap noise*. Since the simulator provides a geometric distance-to-goal while the VLD predictor estimates temporal distance, the two are only partially correlated. Moreover, the dominant source of prediction error arises from the degree of visual overlap between the agent's view and the goal. Thus, by conditioning perturbations on overlap features and learning to map geometric distances into temporally realistic errors, we produce noise that more faithfully reflects the structure of prediction uncertainty encountered in practice.

To this end, we design a learned noise model based on *geometric overlap features*: for each observation and goal pair, we extract a 13-dimensional feature vector:

1. **Projection success ratio (PSR).** To quantify visual overlap between the current and goal views, we define the projection success ratio (PSR), which measures the fraction of pixels in the goal image that can be geometrically projected into the current view with depth-consistent matches. A higher PSR indicates greater visual overlap (details in Appendix D).
2. **Relative camera rotation.** The relative rotation $R_{\text{rel}} \in SO(3)$, flattened to 9 parameters [11].
3. **Relative translation.** The relative translation $t_{\text{rel}} \in \mathbb{R}^3$, encoded with the symmetric logarithm

$$\text{SymLog}(x) \;=\; \text{sign}(x)\,\log\big(1+\alpha|x|\big), \qquad (2)$$

   which compresses large displacements while preserving directionality.

These features are passed to a lightweight MLP with two output heads: one predicting a distribution over *distance-noise bins* and the other over *confidence-noise bins*. Training is formulated as a classification problem, where labels are derived from the VLD's predicted distance and confidence values on held-out image pairs, discretized into bins. We optimize the model using cross-entropy loss.

During **RL policy training**, the trained geometric overlap noise MLP infers probability vectors $p^{(td)} \in \mathbb{R}^{B_d}$ and $p^{(c)} \in \mathbb{R}^{B_c}$ over distance and confidence bins. A bin b is first drawn via multinomial sampling, then the final value is sampled from a Gaussian centered at the bin midpoint with standard deviation proportional to the bin width:

$$\hat{z} \sim \mathcal{N}\big(\tfrac{l_b+u_b}{2},\ \tfrac{u_b-l_b}{6}\big)\,, \qquad (3)$$

where $[l_b, u_b]$ is the bin interval and the $\sigma = \frac{u_b-l_b}{6}$ standard deviation ensures that the sampled point is within the bin with $\approx 99.7\%$ probability. This produces smooth, non-discrete samples while respecting the predicted uncertainty.

Ultimately, the model outputs a noisy pair $(\hat{d}, \hat{c})$ conditioned on overlap and relative pose, thereby reproducing structured error patterns of the VLD predictor: larger errors and lower confidence when the goal is out of view or behind the agent, and tighter, higher-confidence estimates when visual alignment is strong.

We discuss an alternative formulation of distance noise, along with additional design considerations, in Appendix K.

3.2.3. Reward Function

For policy training, we directly extend the *Zero-Exploration Reward (ZER)* formulation [1] to explicitly encourage the agent to orient toward the goal. At each step t, the agent's overlap with the goal is quantified by the projection success ratio $psr_t \in [0,1]$. Let $psr_t^{\max}$ denote the maximum overlap achievable at the current position if the agent were rotated optimally, and let psr_s be a success threshold. The reward is then given as

$$\begin{aligned} r_t \;&= (d_{t-1} - d_t) \;-\; \gamma\,(1 - psr_t) \\ &\quad +\; \mathbb{1}[\,d_t < d_s \;\wedge\; psr_t^{\max} > psr_s\,]\,R_s, \end{aligned} \qquad (4)$$

where d_t is the true geometric distance-to-goal at time t, γ is a step penalty, d_s is a success margin, and R_s is a success bonus applied when the agent enters within d_s of the goal.

4. Experiments

4.1. Distance Function Experiments

We design experiments to demonstrate that our distance function provides a meaningful measure for navigation by producing predictions that correlate with ground-truth distances. In particular, we focus on verifying that the predicted distances have the potential to serve as a reliable navigation signal even in cases where there is no visual overlap between the agent's current observation and the goal image.

4.1.1. Baselines

We compare VLD against two representative families of image-goal distance function approaches that differ in how they estimate temporal distances.

Table 1. Ordinal consistency (Kendall's τ) on Habitat datasets.

Model	HM3D (↑)			Gibson (↑)		
	20	50	100	20	50	100
ViNT	0.40	0.35	0.28	0.53	0.41	0.39
ViNT-Tuned	0.67	0.56	0.48	0.78	0.65	0.64
VIP	0.29	0.19	0.17	0.39	0.30	0.29
VIP-Nav	0.55	0.45	0.38	0.65	0.51	0.50
VIP-DINOv2	0.42	0.45	0.40	0.62	0.52	0.51
VLD (synthetic)	0.81	**0.71**	**0.62**	**0.84**	**0.74**	**0.71**
VLD (real-world)	0.10	0.07	0.07	0.12	0.10	0.09
VLD (all)	**0.82**	0.70	0.61	0.84	0.73	0.71

The first family is based on ViNT [40]. It provides a strong baseline due to its training on diverse real-world datasets and a comparable parameter count to our model ($\sim$31M). We evaluate both the publicly released pretrained ViNT and a fine-tuned variant on our data (ViNT-Tuned).

The second family follows VIP [27]. The original VIP model employs a ResNet-50 backbone [13] and was primarily developed for manipulation tasks. For a fair comparison, we evaluate the released VIP model, a retrained navigation-oriented version (VIP-Nav), and a DINOv2-enhanced variant (VIP-DINOv2) where the ResNet encoder is replaced with a frozen DINOv2 [32].

More details about the architecture and training setup of the baseline models are available in Appendix E.

4.1.2. Dataset

For VLD training with the hyperparameters described in Appendix F, we consider three configurations: (i) using only synthetic Habitat data, (ii) using real-world trajectories from in-the-wild and embodiment datasets, and (iii) combining all available sources. This setup enables us to evaluate both the benefits of large-scale real-world diversity and the robustness of models trained solely in simulation. The distance functions are evaluated independently for each dataset on held-out validation trajectories.

4.1.3. Ordinal Consistency Evaluation

We evaluate ordinal consistency using Kendall's τ, as described in Section 3.1.1. This metric measures whether the predicted distances preserve the correct ordering of progress toward the goal, independent of absolute values. In our setup, trajectories are constructed such that the ground-truth temporal distance to the goal strictly decreases by design; thus, a perfect distance function would produce a monotonically decreasing sequence of predicted distances.

To assess model performance under varying levels of difficulty, we compute ordinal consistency at multiple temporal horizons corresponding to different degrees of visual overlap between the current view and the goal. Specifically, we evaluate on clipped trajectory segments with maximum start-to-goal distances of 20, 50, and 100 steps, as well as on the 100–20 range where overlap is minimal or absent. Shorter horizons (e.g., 0–20) capture settings with strong field-of-view overlap, while longer or offset horizons (e.g., 100–20) evaluate consistency when the goal lies outside the agent's visual field.

Results on Habitat Datasets. We first evaluate models in synthetic Habitat environments. Goals correspond to the final viewpoint of each trajectory. The ordinal consistency (Kendall's τ) results for horizons of 20, 50, and 100 steps are reported in Table 1.

VLD models trained on Habitat data—either alone or combined with real-world data—substantially outperform all baselines. In contrast, the model trained only on real-world videos collapses to a near-constant output, effectively predicting the midpoint of the temporal horizon and yielding Kendall's τ near zero. This failure reflects the unstructured nature of internet-scale walking trajectories, where motion is not consistently directed toward a goal and often includes turns, pauses, and stochastic exploration. The structured, goal-oriented trajectories in Habitat provide the key supervisory signal necessary for learning meaningful temporal distance relationships. All baselines fall considerably short of VLD.

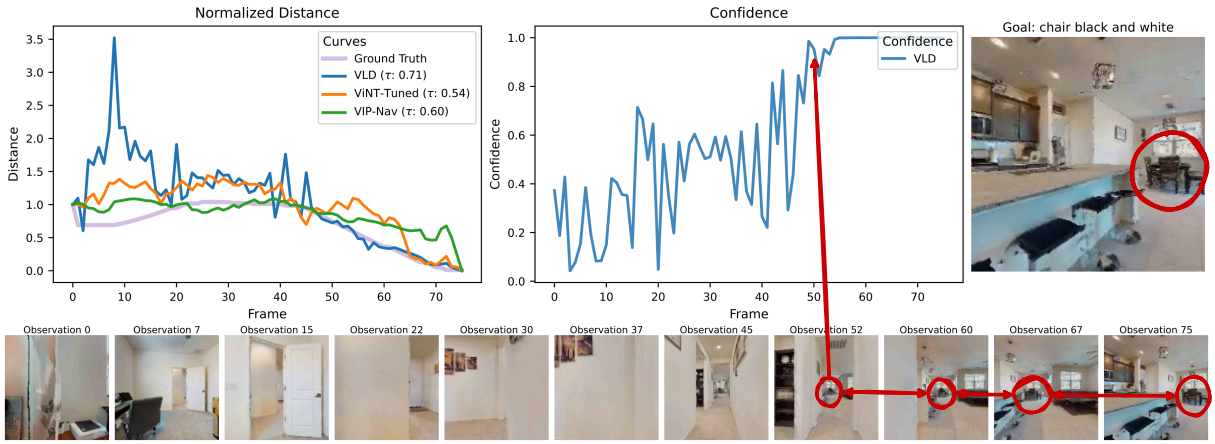
Results on Real-World Datasets. We next evaluate on real-world datasets consisting of in-the-wild videos and embodiment trajectories. We report results in Table 2.

The VLD model trained on both synthetic and real-world data achieves the strongest results. Interestingly, the synthetic-only model is consistently second-best, indicating that structured simulator trajectories generalize well even to real-world outdoor footage.

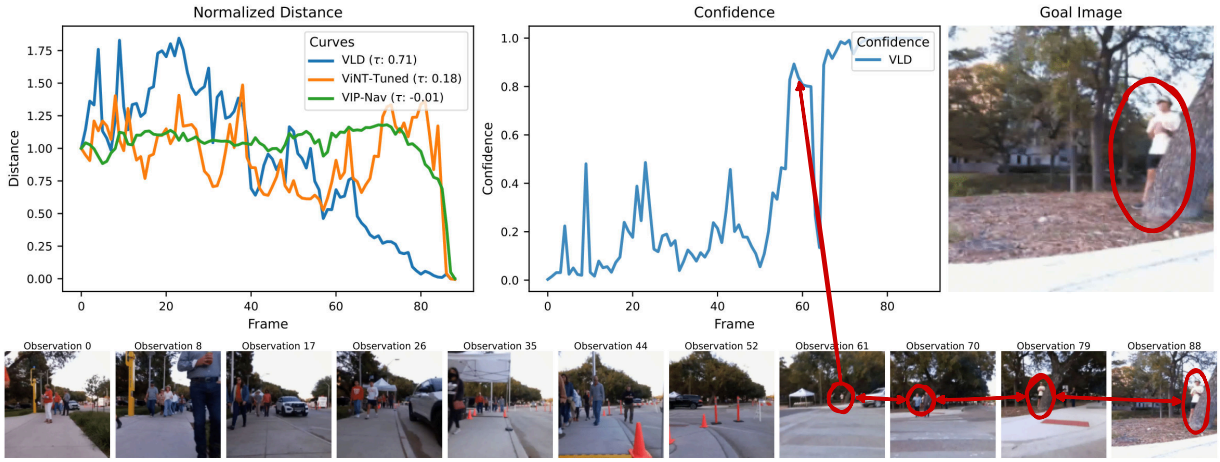
Text-Goal Results. We evaluate language-conditioned distance prediction on HM3D by bootstrapping text descriptions from object-goal annotations (Appendix B). Results are reported in Table 3. When using image goals, VLD achieves the strongest performance, and adding text alongside images leads to nearly identical results. In practice, the model naturally prioritizes the visual signal when available, and the additional text conditioning neither helps

Table 2. Ordinal consistency (Kendall's τ) on real-world trajectories. VLD (all) generalizes best across bothW settings.

Model	In-the-wild (↑)		Embodiment (↑)	
	50	100	50	100
ViNT	0.35	0.25	0.41	0.32
ViNT-Tuned	0.40	0.29	0.48	0.37
VIP	0.28	0.20	0.43	0.33
VIP-Nav	0.32	0.23	0.46	0.39
VIP-DINOv2	0.42	0.30	0.52	0.44
VLD (synthetic)	0.44	0.31	0.58	0.48
VLD (real-world)	0.23	0.18	0.16	0.14
VLD (all)	**0.69**	**0.61**	**0.73**	**0.63**



(a) **Image-goal Habitat example (VLD, ViNT-Tuned, VIP-Nav).** As the agent progresses, all models produce declining distance estimates, but VLD tracks the ground-truth trend more closely, achieving the highest Kendall's τ score. Around frame 50, visual cues from the goal image (the table area) enter the field of view, leading to an increase in VLD confidence and a drop in its predicted distance. Baselines either react later or exhibit weaker monotonic alignment, illustrating VLD's stronger ordinal consistency.



(b) **Real-world trajectory example (VLD: text, image, multimodal).** Similarly to Figure 3a, around frame 60, the person circled in the goal image becomes (barely) visible in the observation (see frames 61, 70, 79, 88). Despite the minimal visual footprint, VLD detects this overlap, producing a sharp confidence increase and a corresponding drop in predicted distance. This highlights VLD's sensitivity to subtle goal-relevant cues and its ability to express uncertainty meaningfully compared to baselines.

Figure 3. **Ordinal consistency analysis on image-goal examples.** For each trajectory, models compute distances *independently* at every time step using the last frame as the goal. **Top rows:** normalized distance curves with associated Kendall's τ values (left), VLD confidence evolution (middle), and the goal image or goal text (right). **Bottom rows:** the sequence of agent observations along the trajectory. Across both examples, VLD exhibits strong monotonic alignment with ground truth and meaningful confidence behavior, while baselines either drift or fail to reflect appearance-based changes as reliably.

nor harms performance in a meaningful way. When using text-only goals, performance drops noticeably, which is expected since natural language descriptions are inherently less precise than images in specifying spatial targets, and our text labels are automatically generated and therefore coarse. However, the text-only VLD model still achieves ordinal consistency well above random and remains comparable to the image-based baselines even after those baselines are fine-tuned. This suggests that VLD is able to form semantically grounded navigation distances from text alone, despite the weaker supervision signal.

Long-Horizon No-Overlap Consistency. We additionally evaluate ordinal consistency in settings where the agent's current observation and the goal image have little to no visual overlap. To do so, we compute Kendall's τ over clipped trajectory segments in the 50–20 and 100–20 ranges, where the agent is still far from the goal and direct correspondence is weak. On **HM3D**, VLD achieves

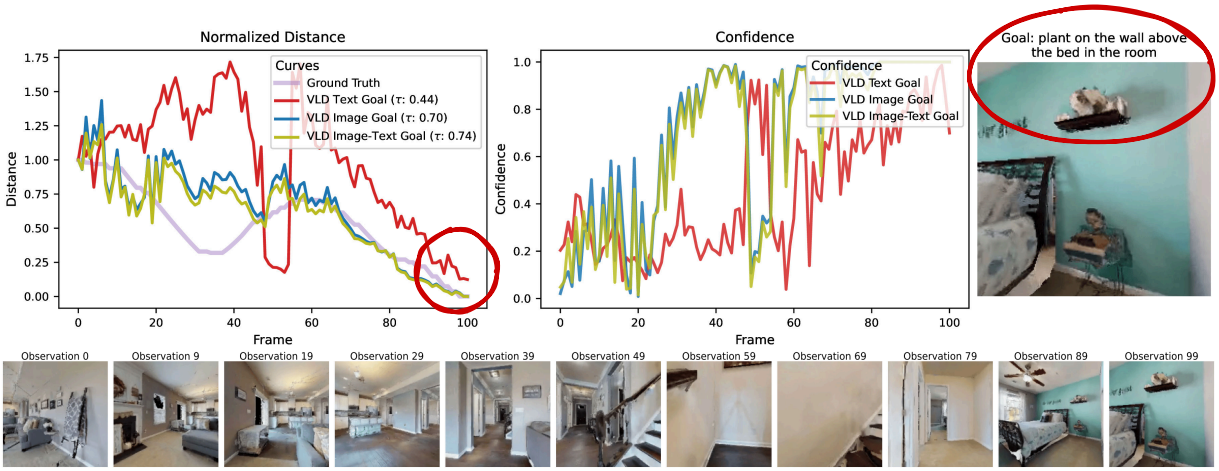


Figure 4. **Ordinal consistency analysis for text-goal Habitat example (VLD: text, image, multimodal).** VLD computes distances *independently* at every time step using the last frame or text prompt as the goal. **Top row:** normalized distance predictions with corresponding Kendall's τ scores (left), VLD confidence curve (middle), and the goal image (right). **Bottom row:** the sequence of observations along the trajectory. The automatically bootstrapped (Appendix B) text description ("plant on the wall above the bed in the room") provides only partial and even occasionally ambiguous guidance. Nonetheless, text-only VLD predictions follow the global decreasing trend and remain broadly aligned with the image-goal variants, albeit with higher noise—as expected for semantic-only supervision. As the agent approaches the room containing the goal, confidence increases and predicted distances fall smoothly, though not exactly to zero—an expected outcome for semantic (non-pixel-aligned) goal specifications. The multimodal version closely tracks the image-only-goal predictions, demonstrating that linguistic cues can reinforce—but do not override—visual distance estimation.

Table 3. Ordinal consistency (Kendall's τ) on text-specified goals (HM3D). Combining image and text recovers near image-only performance.

Model	20 (↑)	50 (↑)	100 (↑)
ViNT	0.40	0.35	0.28
ViNT-Tuned	0.67	0.56	0.48
VIP	0.29	0.19	0.17
VIP-Nav	0.55	0.45	0.38
VIP-DINOv2	0.42	0.45	0.40
VLD (image)	0.81	0.70	0.61
VLD (text)	0.49	0.49	0.44
VLD (image+text)	**0.81**	**0.71**	**0.62**

$\tau = 0.40$ at 50–20 and $\tau = 0.35$ at 100–20 (compared to $\tau = 0.71$ and $\tau = 0.62$ under full-horizon evaluation), indicating that ordinal structure is largely preserved even without overlapping views. On **in-the-wild** trajectories, VLD reaches $\tau = 0.42$ and $\tau = 0.45$ for 50–20 and 100–20, respectively, while on **embodiment** data it obtains $\tau = 0.52$ and $\tau = 0.46$. These results show that VLD continues to provide a meaningful navigation signal even when the goal is visually out of view, demonstrating robustness to long-horizon perception gaps.

Additional ordinal consistency experiments, including ablations of the VLD setup, are provided in Appendix G.

Qualitative results. Across all datasets, we observe a consistent pattern: ordinal consistency is highly reliable whenever even a small amount of visual overlap exists between the current observation and the goal. This holds across both Habitat trajectories and real-world sequences (see Figures 3 and 4, with additional examples in Appendix I). The image-goal examples (Figure 3a and Figure 3b) illustrate how VLD closely follows the ground-truth monotonic trend and reacts promptly when goal-relevant cues enter the field of view, outperforming ViNT-Tuned and VIP-Nav. Similarly, the text-goal example (Figure 4) shows that text-only predictions remain coherent—albeit noisier—and that multimodal conditioning effectively aligns with image-based performance.

In scenarios where the goal lies completely outside the field of view, confidence scores drop markedly and the predicted distances become noisier, reflecting true perceptual uncertainty rather than model failure. This motivates analyzing whether VLD still provides a meaningful navigation signal even when no visual overlap is present. To that end, we introduce a *distance accuracy* metric that compares relative distances between pairs of observations without goal image visual overlap. Full details and results are provided in Appendix J.

4.2. Application: Using VLD for Navigation Policies

The reinforcement learning policies are trained and evaluated in the Gibson environment [47]. Gibson is disjoint from HM3D, meaning that the navigation policy is deployed in a fully unseen environment.

We train point-goal navigation policies using structured

Table 4. Navigation performance on Gibson (validation). "Swap" indicates replacing (noised) ground-truth distance with VLD/ViNT/VIP at deployment. We report success rate (SR↑) and success weighted by path length (SPL↑).

Policy Configuration	**SR (↑)**	**SPL (↑)**
Privileged Training (GT Distance)		
GT Distance (no noise)	**0.9577**	**0.6103**
GeoNoise	0.9091	0.5547
GeoNoise + Confidence	0.8994	0.5809
Trained Directly on VLD		
Policy trained end-to-end on VLD	0.5664	0.4227
Swap: Replace Distance with VLD/ViNT/VIP at Deployment		
VLD + (GeoNoise)	**0.7314**	**0.3995**
VLD + (GeoNoise + Confidence)	0.6821	0.3860
ViNT-Tuned + (GeoNoise)	0.6046	0.2555
VIP-Nav + (GeoNoise)	0.2787	0.1148
External Image-Based Nav Baselines (reported)		
OVRL-V2 [49]	0.820	0.587

geometric-overlap noise (Section 3.2.2). The policy additionally receives lightweight obstacle-awareness features (via a frozen pretrained EfficientNet-B0 encoder [42]), GPS displacement, and proprioception. Notably, many successful sim-to-real RL pipelines rely on training policies purely from geometric signals [51]; we follow a more Habitat-friendly setup here for simplicity, but this approach can be readily adapted to geometry-only exteroception. We also evaluate a policy trained end-to-end directly on VLD predictions. Results are summarized in Table 4, with noise function ablations discussed in Appendix K, and trajectory visualizations and analyses provided in Appendix L.

Quantitative Results. The privileged policy trained with ground-truth distance nearly solves the task, confirming that accurate distance-to-goal is a strong navigation signal. Training a policy directly on VLD predictions achieves moderate success but is significantly less efficient, requiring way more computational resources and time to converge.

The key result is that policies trained on *noisy* ground-truth distances transfer effectively to VLD at deployment: simply swapping the distance signal yields a success rate of 0.73, a $\sim$17% relative drop from respective privileged performance. This demonstrates that geometric-overlap noise induces robustness to the characteristic error patterns of VLD, enabling strong zero-shot replacement.

Interestingly, the confidence-aware policy performs slightly worse. We hypothesize that this is due to compounding uncertainty: the policy encounters both (i) noise in the distance predictions and (ii) noise in the associated confidence values. This suggests that confidence is valuable during training, but may compound uncertainty in decision-making once the distance estimates become model-predicted rather than simulator-provided.

In addition, Table 4 compares how different distance predictors perform when swapped into the same navigation policy at deployment. Although the noise model was calibrated to VLD, the policy can also operate with alternative distance functions such as ViNT-Tuned or VIP-Nav, allowing us to assess their suitability for control directly. The trend is consistent: VLD provides a more informative and stable signal, leading to the highest success rate and SPL among all swapped variants. ViNT-Tuned is partially compatible—highlighting the robustness induced by geometric-overlap noise—but still underperforms VLD by a substantial margin, while VIP-Nav degrades navigation sharply. These outcomes mirror the ranking observed in the distance-prediction benchmarks: predictors with higher ordinal consistency and more reliable distance estimates yield stronger downstream navigation behavior.

Finally, we compare our approach against leading ImageNav systems such as OVRL-V2 [49]. Unlike OVRL-V2, which learns policies directly from full image observations, our approach relies on a *single scalar* goal signal complemented by lightweight perception. Despite this substantial difference in supervision, the swapped policy achieves competitive navigation performance, with only $\sim$9% drop in SR. While we observe a decrease of $\sim$20% in SPL, this is expected since the robot needs to move around more to determine the direction in which the distance is decreasing.

5. Conclusion

Our work takes a step toward bridging the gap between large-scale perception models and embodied action. While the information exchange between the VLD and RL policy is limited as VLD reduces rich visual input to a single distance value—potentially overlooking subtle semantic cues—the proposed VLD framework delegates the responsibility of safe local navigation to the RL policy, while abstracting high-level object goals and text conditioning into a scalable learning objective. With sufficiently large-scale pretraining, such models may eventually capture the subtleties required for robust goal understanding, turning this seemingly minimal signal into a useful foundation. While we ultimately do not argue for abstracting visual navigation as a single scalar, we showcase that abstracting the task description into a single scalar — or a history of scalar values — for the policy leads to the emergence of interesting search and navigation strategies. Furthermore, although we demonstrated transfer to a different environment in simulation, we have not yet validated its effectiveness in real-world robotic applications. We hypothesize that the sim-to-real gap may be smaller in this case because the visual domain discrepancies only affect a scalar value. However, as evident from our simulation results, the limited information will result in suboptimal navigation behavior compared to a policy that has access to the full visual cues.

References

[1] Ziad Al-Halah, Santhosh K. Ramakrishnan, and Kristen Grauman. Zero experience required: Plug & play modular transfer learning for semantic visual navigation. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17010–17020, 2022. 5

[2] Mayank Bansal, Alex Krizhevsky, and Abhijit S. Ogale. Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst. *ArXiv*, abs/1812.03079, 2018. 1

[3] Jiaqi Chen, Jonas Frey, Ruyi Zhou, Takahiro Miki, Georg Martius, and Marco Hutter. Identifying terrain physical parameters from vision - towards physical-parameter-aware locomotion and navigation. *IEEE Robotics and Automation Letters*, 9(11):9279–9286, 2024. 1

[4] An-Chieh Cheng, Yandong Ji, Zhaojing Yang, Zaitan Gongye, Xueyan Zou, Jan Kautz, Erdem Biyik, Hongxu Yin, Sifei Liu, and Xiaolong Wang. Navila: Legged robot vision-language-action model for navigation. In *Robotics: Science and Systems (RSS)*, 2025. 1, 3

[5] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, 2014. Association for Computational Linguistics. 24

[6] Felipe Codevilla, Matthias Miiller, Antonio López, Vladlen Koltun, and Alexey Dosovitskiy. End-to-end driving via conditional imitation learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, page 1–9. IEEE Press, 2018. 3

[7] Alejandro Escontrela, Justin Kerr, Arthur Allshire, Jonas Frey, Rocky Duan, Carmelo Sferrazza, and Pieter Abbeel. Gaussgym: An open-source real-to-sim framework for learning locomotion from pixels. *arXiv preprint arXiv:2510.15352*, 2025. 3

[8] Zhicheng Feng, Xieyuanli Chen, Chenghao Shi, Lun Luo, Zhichao Chen, Yun-Hui Liu, and Huimin Lu. Image-goal navigation using refined feature guidance and scene graph enhancement, 2025. 26

[9] Jonas Frey, Turcan Tuna, Lanke Frank Tarimo Fu, Cedric Weibel, Katharine Patterson, Benjamin Krummenacher, Matthias Müller, Julian Nubert, Maurice Fallon, Cesar Cadena, and Marco Hutter. Boxi: Design Decisions in the Context of Algorithmic Performance for Robotics. In *Proceedings of Robotics: Science and Systems*, Los Angeles, United States, 2025. 2, 14

[10] Jiawei Fu, Yunlong Song, Yan Wu, Fisher Yu, and Davide Scaramuzza. Learning deep sensorimotor policies for vision-based autonomous drone racing. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5243–5250, 2023. 3

[11] Andreas René Geist, Jonas Frey, Mikel Zhobro, Anna Levina, and Georg Martius. Learning with 3d rotations, a hitchhiker's guide to SO(3). In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. 5

[12] Kristen Grauman, Andrew Westbury, Eugene Byrne, Vincent Cartillier, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Devansh Kukreja, Miao Liu, Xingyu Liu, Miguel Martin, Tushar Nagarajan, Ilija Radosavovic, Santhosh Kumar Ramakrishnan, Fiona Ryan, Jayant Sharma, Michael Wray, Mengmeng Xu, Eric Zhongcong Xu, Chen Zhao, Siddhant Bansal, Dhruv Batra, Sean Crane, Tien Do, Morrie Doulaty, Akshay Erapalli, Christoph Feichtenhofer, Adriano Fragomeni, Qichen Fu, Abrham Gebreselasie, Cristina Gonzalez, James Hillis, Xuhua Huang, Yifei Huang, Wenqi Jia, Weslie Khoo, Jachym Kolar, Satwik Kottur, Anurag Kumar, Federico Landini, Chao Li, Yanghao Li, Zhenqiang Li, Karttikeya Mangalam, Raghava Modhugu, Jonathan Munro, Tullie Murrell, Takumi Nishiyasu, Will Price, Paola Ruiz Puentes, Merey Ramazanova, Leda Sari, Kiran Somasundaram, Audrey Southerland, Yusuke Sugano, Ruijie Tao, Minh Vo, Yuchen Wang, Xindi Wu, Takuma Yagi, Ziwei Zhao, Yunyi Zhu, Pablo Arbelaez, David Crandall, Dima Damen, Giovanni Maria Farinella, Christian Fuegen, Bernard Ghanem, Vamsi Krishna Ithapu, C. V. Jawahar, Hanbyul Joo, Kris Kitani, Haizhou Li, Richard Newcombe, Aude Oliva, Hyun Soo Park, James M. Rehg, Yoichi Sato, Jianbo Shi, Mike Zheng Shou, Antonio Torralba, Lorenzo Torresani, Mingfei Yan, and Jitendra Malik. Ego4D: Around the World in 3,000 Hours of Egocentric Video . *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (01): 1–32, 2021. 2, 14

[13] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2015. 6

[14] Noriaki Hirose, Catherine Glossop, Dhruv Shah, and Sergey Levine. Omnivla: An omni-modal vision-language-action model for robot navigation, 2025. 2

[15] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, 1997. 24

[16] Jakob Hollenstein, Sayantan Auddy, Matteo Saveriano, Erwan Renaudo, and Justus Piater. Action noise in off-policy deep reinforcement learning: Impact on exploration and performance. *Transactions on Machine Learning Research*, 2022. Survey Certification. 24

[17] 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*, 2022. 2, 14

[18] M G Kendall. A new measure of rank correlation. *Biometrika*, 30(1-2):81–93, 1938. 3

[19] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024. 3

[20] Jacob Krantz, Stefan Lee, Jitendra Malik, Dhruv Batra, and Devendra Singh Chaplot. Instance-specific image goal navigation: Training embodied agents to find object instances, 2022. 13

[21] Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. Rma: Rapid motor adaptation for legged robots. 2021. 3

[22] Joonho Lee, Marko Bjelonic, Alexander Reske, Lorenz Wellhausen, Takahiro Miki, and Marco Hutter. Learning robust autonomous navigation and locomotion for wheeled-legged robots. *Science Robotics*, 9(89):eadi9641, 2024. 2

[23] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: bootstrapping language-image pre-training with frozen image encoders and large language models. In *Proceedings of the 40th International Conference on Machine Learning*. JMLR.org, 2023. 2, 14

[24] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Manfred Otto Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *CoRR*, abs/1509.02971, 2015. 24

[25] Minghuan Liu, Menghui Zhu, and Weinan Zhang. Goal-conditioned reinforcement learning: Problems and solutions. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 5502–5511. International Joint Conferences on Artificial Intelligence Organization, 2022. Survey Track. 3

[26] Xinhao Liu, Jintong Li, Yicheng Jiang, Niranjan Sujay, Zhicheng Yang, Juexiao Zhang, John Abanes, Jing Zhang, and Chen Feng. Citywalker: Learning embodied urban navigation from web-scale videos. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 6875–6885, 2025. 1, 2, 3, 14

[27] Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. VIP: towards universal visual reward and representation via value-implicit pre-training. *CoRR*, abs/2210.00030, 2022. 2, 6, 15, 16, 17, 18

[28] Takahiro Miki, Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning robust perceptive locomotion for quadrupedal robots in the wild. *Science Robotics*, 7(62):eabk2822, 2022. 3

[29] Vivek Myers, Chongyi Zheng, Anca Dragan, Sergey Levine, and Benjamin Eysenbach. Learning temporal distances: contrastive successor features can provide a metric structure for decision-making. In *Proceedings of the 41st International Conference on Machine Learning*. JMLR.org, 2024. 3

[30] Andrew Naish-guzman and Sean Holden. Robust regression with twinned gaussian processes. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2007. 3, 4

[31] David Nistér, Oleg Naroditsky, and James R. Bergen. Visual odometry. *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, 1:I–I, 2004. 13

[32] Maxime Oquab, Timothée Darcet, Theo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Wojciech Galuba, Mike Rabbat, Mido Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2023. 3, 6

[33] 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. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, pages 8748–8763. PMLR, 2021. 3

[34] Santhosh Kumar Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alexander Clegg, John M Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, Manolis Savva, Yili Zhao, and Dhruv Batra. Habitat-matterport 3d dataset (HM3d): 1000 large-scale 3d environments for embodied AI. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021. 13

[35] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A Platform for Embodied AI Research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 1, 2, 13

[36] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347, 2017. 5

[37] 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)*, page 13215–13222. IEEE, 2021. 2, 13, 17

[38] Dhruv Shah, Benjamin Eysenbach, Nicholas Rhinehart, and Sergey Levine. Rapid exploration for open-world navigation with latent goal models. In *Proceedings of the 5th Conference on Robot Learning*, pages 674–684. PMLR, 2022. 3

[39] Dhruv Shah, Ajay Sridhar, Arjun Bhorkar, Noriaki Hirose, and Sergey Levine. GNM: A General Navigation Model to Drive Any Robot. In *International Conference on Robotics and Automation (ICRA)*, 2023. 1

[40] Dhruv Shah, Ajay Sridhar, Nitish Dashora, Kyle Stachowicz, Kevin Black, Noriaki Hirose, and Sergey Levine. ViNT: A foundation model for visual navigation. In *7th Annual Conference on Robot Learning*, 2023. 1, 2, 6, 14, 16, 17, 26

[41] Ajay Sridhar, Dhruv Shah, Catherine Glossop, and Sergey Levine. NoMaD: Goal Masked Diffusion Policies for Navigation and Exploration. *arXiv pre-print*, 2023. 1, 2, 3

[42] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *ArXiv*, abs/1905.11946, 2019. 9, 14

[43] Tongzhou Wang and Phillip Isola. Improved representation of asymmetrical distances with interval quasimetric embeddings. In *Workshop on Symmetry and Geometry in Neural*

Representations (NeurReps) at Conference on Neural Information Processing Systems (NeurIPS), 2022. Proceedings Track. 16

[44] Tongzhou Wang, Antonio Torralba, Phillip Isola, and Amy Zhang. Optimal goal-reaching reinforcement learning via quasimetric learning. In *International Conference on Machine Learning*. PMLR, 2023. 3, 17, 18

[45] Erik Wijmans, Abhishek Kadian, Ari Morcos, Stefan Lee, Irfan Essa, Devi Parikh, Manolis Savva, and Dhruv Batra. DD-PPO: learning near-perfect pointgoal navigators from 2.5 billion frames. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. 3

[46] Tim Windecker, Manthan Patel, Moritz Reuss, Richard Schwarzkopf, Cesar Cadena, Rudolf Lioutikov, Marco Hutter, and Jonas Frey. Navitrace: Evaluating embodied navigation of vision-language models. *Preprint submitted to arXiv*, 2025. Currently a preprint on arXiv (arXiv:2510.26909). Awaiting peer review and journal submission. 1

[47] Fei Xia, Amir R. Zamir, Zhi-Yang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson env: real-world perception for embodied agents. In *Computer Vision and Pattern Recognition (CVPR), 2018 IEEE Conference on*. IEEE, 2018. 8, 14

[48] Ziyang Xie, Zhizheng Liu, Zhenghao Peng, Wayne Wu, and Bolei Zhou. Vid2sim: Realistic and interactive simulation from video for urban navigation. *Preprint*, 2024. 3

[49] 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*, abs/2303.07798, 2023. 9

[50] Guang-Zhong Yang, Jim Bellingham, Pierre E Dupont, Peer Fischer, Luciano Floridi, Robert Full, Neil Jacobstein, Vijay Kumar, Marcia McNutt, Robert Merrifield, Bradley J Nelson, Brian Scassellati, Mariarosaria Taddeo, Russell Taylor, Manuela Veloso, Zhong Lin Wang, and Robert Wood. The grand challenges of science robotics. *Sci. Robot.*, 3(14): eaar7650, 2018. 1

[51] Chong Zhang, Jin Jin, Jonas Frey, Nikita Rudin, Matías Mattamala, Cesar Cadena, and Marco Hutter. Resilient legged local navigation: Learning to traverse with compromised perception end-to-end. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 34–41, 2024. 9

[52] Shaoting Zhu, Linzhan Mou, Derun Li, Baijun Ye, Runhan Huang, and Hang Zhao. Vr-robo: A real-to-sim-to-real framework for visual robot navigation and locomotion. *IEEE Robotics and Automation Letters*, 10(8):7875–7882, 2025. 1, 3

Appendix

A. Distance Function Training

The training, summarized in Algorithm 1, is framed as a self-supervised regression problem with both positive and negative triplets. The positive examples $(o_i, o_j, td = j - i)$ are formed by sampling o_i and o_j along the same trajectory such that $i \leq j$ and the $td = j - i$ is their temporal distance, where $td \leq td_{\max}$. For hard negatives, following ViNG's negative mining strategy [37], we pair observations from different trajectories and assign the maximum distance $td_{\max}$. This exposes the model to cross-trajectory and cross-scene mismatches, thereby improving the robustness of the learned distance function in recognizing when a goal seems off or is very far away.

We further highlight the role of *negative mining* through the example shown in Figure 5. When computing the distance between two observations—one taken in a standard indoor kitchen and the other onboard a spaceship—we expect the model to assign a value close to $td_{\max}$, reflecting that these views are entirely unrelated. In practice, comparing models trained with and without negative mining shows a sharp contrast. The model trained with negative mining predicts a distance of approximately 102, slightly exceeding its $td_{\max} = 100$, which is desirable behavior as it confidently signals that the model believes that the two images are very far apart. In contrast, the model trained without negative mining predicts a distance of only 50, essentially the midpoint of the range. Moreover, the negative-mining-trained model produces a confidence score close to 1.0, while the model without negative mining yields only 0.13, confirming that negative mining is essential for enabling the distance function to decisively recognize mismatched or out-of-distribution goals.

B. Dataset Collection

Training data for the distance function is collected from three kinds of sources: (i) synthetic trajectories from simulators, (ii) "in-the-wild" internet-scale videos, and (iii) embodiment datasets collected on real robots. A summary is provided in Table 5.

(a) Agent's Observation from a Kitchen (b) Image from a Spaceship

Figure 5. **Negative mining teaches the model to separate unrelated scenes.** Given two observations from completely different environments (left: kitchen; right: spaceship), the desired behavior is to output a distance close to $td_{\max}$, indicating that the images cannot correspond to nearby states on any trajectory. Only the model trained with negative mining learns this behavior, predicting a high distance with high confidence; the model trained without it collapses toward an uncertain mid-range prediction.

Algorithm 1 Distance function training

Require: Trajectories $\{\tau^{(k)} = (o_1^{(k)}, a_1^{(k)}, o_2^{(k)}, a_2^{(k)}, \dots)\}_{k=1}^{K}$, max horizon $td_{\max}$, negative sampling probability p_{neg}, parametrized distance model $\mathcal{T}_\theta(o, g)$, loss function $\mathcal{L}$

Construct positives: $\mathcal{D}_+ \leftarrow \{(o_i^{(k)}, o_j^{(k)}, td = j - i) \mid 1 \leq i \leq j, \ , \ j - i \leq td_{\max} k \in [1..K]\}$
Construct negatives: $\mathcal{D}_- \leftarrow \{(o_i^{(k)}, o_j^{(\ell)}, td = td_{\max}) \mid k \neq \ell\}$
Initialize distance model $\mathcal{T}_\theta(o, g) \rightarrow (\hat{t}, \hat{c})$

while not converged **do**
  if Uniform$(0, 1) < p_{\text{neg}}$ **then** ▷ negative step
    Sample mini-batch $\mathcal{B} \sim \mathcal{D}_-$
  else ▷ positive step
    Sample mini-batch $\mathcal{B} \sim \mathcal{D}_+$
  end if
  Compute predictions $\{(\hat{t}_n, \hat{c}_n)\} \leftarrow \{\mathcal{T}_\theta(o_n, g_n) \mid (o_n, g_n, td_n) \in \mathcal{B}\}$
  Compute loss: $\mathcal{L}(\theta) \leftarrow$ LossFunction$(\{(\hat{t}_n, \hat{c}_n, td_n)\}_{n \in \mathcal{B}})$
  Update parameters: $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}(\theta)$
end while
return trained distance function $\mathcal{T}_\theta$

Since these datasets span different embodiments and capture rates, the average velocity of agents varies considerably. To normalize the effective temporal spacing, we adjust the frame rate so that the distance covered between two consecutive frames is approximately 0.25 m. This matches the step size of the Habitat agent and ensures geometric consistency across sources. When velocity metadata is unavailable, we estimate it using a simple visual odometry method [31].

The collection procedure for each data type:

1. **Synthetic data.** We use Habitat [35] with the HM3D scenes [34], which provide well-structured trajectories together with specific goal object annotations and corresponding image goals [20]. We exploit this structure by generating trajectories with the *ShortestPathFollower*. To make goal images more meaningful, we extend the follower: instead of stopping at the exact point-goal location, the agent continues until the goal object is visible and then rotates to orient toward it before issuing the stop action. This ensures that the final goal view clearly depicts the object of interest.
   To further enrich this data, we bootstrap text goals.

Table 5. Datasets used for training and validating the temporal distance function. We report average velocity, total hours, dataset type, and the frame rate used for generating temporal distance supervision.

Dataset	Avg. Velocity	Total Hrs	Type
HM3D	2.5 m/s	1150h	Synthetic
YouTube	0.5 m/s	1720h	In-the-wild
Ego4D	0.5–1.5 m/s	50h	In-the-wild
SCAND	1.5–2 m/s	9h	Embodiment
Grand Tour	1 m/s	5h	Embodiment
CityWalker	1.5 m/s	15h	Embodiment
	Total	2949h	

HM3D provides object labels at goal viewpoints; using these, we prompt BLIP [23] to generate a goal description given the image of the goal and a simple template (e.g., "the {goal object} is"). This procedure yields 475,562 trajectories from 145 training scenes and 2,271 trajectories from 35 validation scenes (after filtering out very short routes). We additionally include 994 validation trajectories from Gibson [47] as a fully held-out test dataset.

2. **"In-the-wild" videos.** Following CityWalker [26], we curate 1,720 hours of YouTube videos of humans walking across different outdoor settings, seasons, and times of day. Sources of these videos are listed in Section C. We also add 50 hours of diverse sequences from Ego4D [12], filtered to scenarios such as "street walking", "indoor navigation", and "jogging/cycling". No further preprocessing is applied beyond adjusting the frame rate. For both sources, we hold out roughly 10% of trajectories for validation.
3. **Embodiment datasets.** We include several smaller but very high-quality datasets collected on physical robots. SCAND [17] provides trajectories from Spot and Jackal across indoor and outdoor settings. The GrandTour [9] spans more than 49 environments, including indoor, urban, natural, and mixed settings collected with a quadrupedal robot. We also incorporate CityWalker's own robot-collected subset [26]. These datasets are limited in scale but highly diverse, and we upsample them during training. Again, around 10% of trajectories are reserved for validation.

C. Walking Data Video Sources

For the in-the-wild data component, we follow the CityWalker dataset [26] and use the same YouTube walking video sources. The playlists are publicly available and can be accessed at:

- Day Walking Tours
- Sunset Walking Tours
- Rainy Walking Tours
- Night City Walking Tours
- Snow Days

These playlists comprise almost 2000 hours of human walking footage captured across diverse urban and natural environments, recorded at various times of day and under varying weather conditions. They serve as the source for our internet-scale "in-the-wild" trajectories.

D. Projection Success Ratio (PSR)

The projection success ratio (PSR) provides a geometric proxy for visual overlap between a goal view and the agent's current egocentric view. It is defined as the fraction of valid goal pixels that project consistently into the current depth image:

$$\text{PSR} = \frac{1}{N}\sum_{p=1}^{N} \mathbf{1}\Big(\left| Z_p^{\text{curr-pred}} - Z_{\pi(p)}^{\text{curr}} \right| < \tau \Big), \quad (5)$$

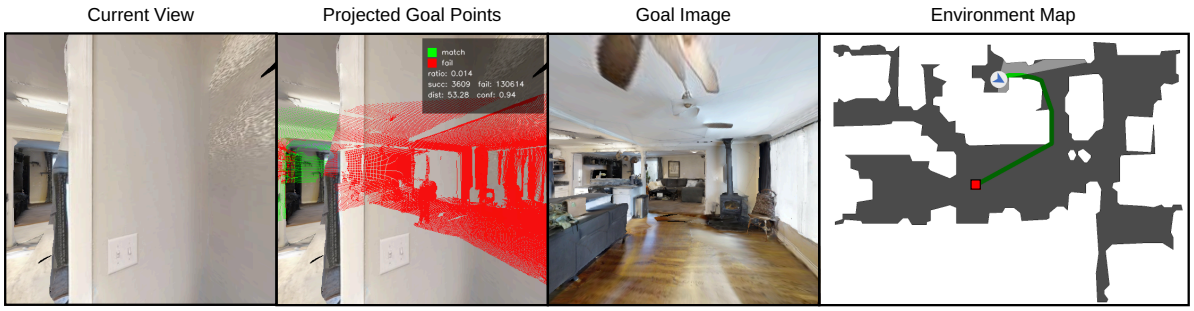
where N is the number of valid depth pixels in the goal image, $Z_p^{\text{curr-pred}}$ is the predicted depth of pixel p after unprojection from the goal frame and transformation into the current camera frame, $\pi(p)$ is its projection into the current image plane using calibrated intrinsics, and $Z_{\pi(p)}^{\text{curr}}$ is the observed depth at the corresponding pixel in the current view. The threshold τ controls the tolerance for depth consistency.

Figure 6 illustrates PSR in practice. With no overlap, nearly all projected pixels fail to match (yielding PSR ≈ 0). With partial overlap, a subset of projections are consistent, resulting in an intermediate PSR. With full overlap, most goal pixels successfully project, producing a high PSR ≈ 1.

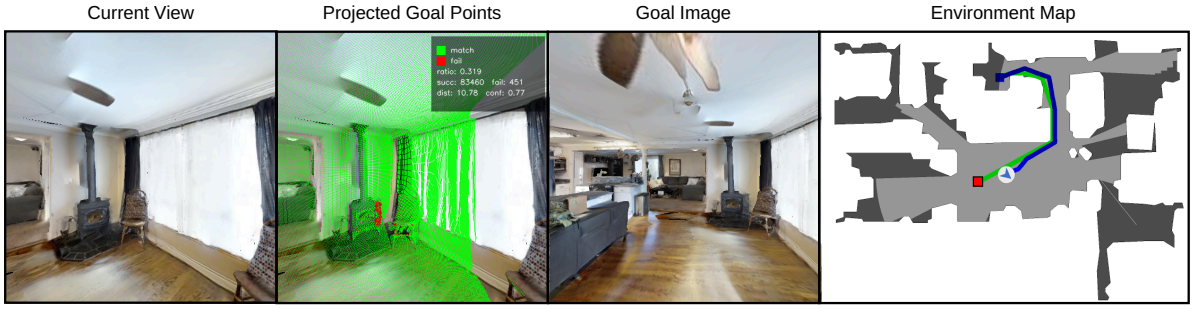
E. Alternative Distance Function Architectures

To test the robustness of our design choices, we evaluated several alternative architectures for distance prediction, summarized in Figures 8 and 9. These span both encoder-based and embedding-distance approaches.

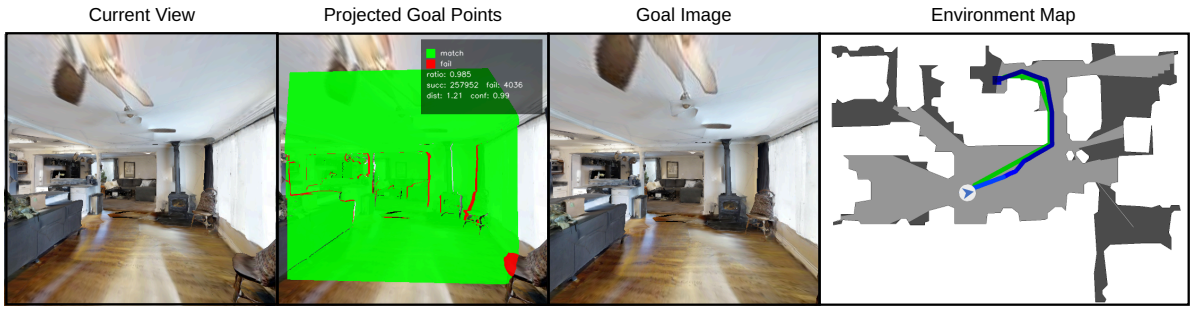
(i) ViNT-style baseline. We directly adopt the ViNT architecture [40], where observation and goal tokens are concatenated and processed jointly by a Transformer encoder (see Figure 7). In contrast to our use of frozen internet-scale encoders (DINOv2 and CLIP), ViNT relies on EfficientNet backbones [42] that are trained from scratch. For consistency with our setting, we remove the action-prediction head and retain only the temporal distance output. We evaluate both the released ViNT model (pre-trained on their curated datasets) and variants trained on our data. This serves



(a) **No overlap.** The goal view is occluded (e.g., by a wall), yielding a near-zero projection success ratio.



(b) **Partial overlap.** A subset of projected points land consistently, producing an intermediate projection success ratio.



(c) **Full overlap.** Most projected points are consistent, resulting in a high projection success ratio.

Figure 6. **Projection success ratio (PSR) visualization.** Each row shows (left) the current view, (second from the left) the current view with projected goal points, and (second from the right) the goal image (right). In the middle panel, the legend reports: PSR r, the number of successfully matched points (green) and failed matches (red), and the geometric-overlap noise model's predicted distance $\hat{d}$ and confidence $\hat{c}$.

as a natural baseline, since ViNT also predicts temporal distance as an auxiliary signal, though its backbone and training setup differ substantially from ours.

(ii) Encoder over joint tokens. As shown in Figure 8, we experiment with an encoder-only Transformer applied to the concatenation of observation and goal embeddings. Unlike our decoder-based design, which retains the full token sequence from DINOv2 and CLIP, here each image (or text sequence) is reduced to a single pooled embedding from its backbone. These embeddings are then concatenated, augmented with positional and modality encodings, and processed symmetrically by a Transformer encoder. The output tokens are then averaged, and the resulting embedding is used to regress distance and confidence. This design allows us to isolate the effect of our decoder-based query–memory formulation versus pooled joint encoding.

(iii) VIP-style embedding distances. Inspired by VIP [27], we also test architectures where observation and goal images are encoded independently, and temporal distance is predicted as a function of their embeddings (Figure 9). Concretely, we experiment with simple ℓ_2 norms as well as learned functions such as a linear layer with ReLU. Since the prediction head in this setup is relatively weak and lacks the capacity of a decoder with cross-attention, the choice of image encoder becomes especially critical. We therefore evaluate both frozen and

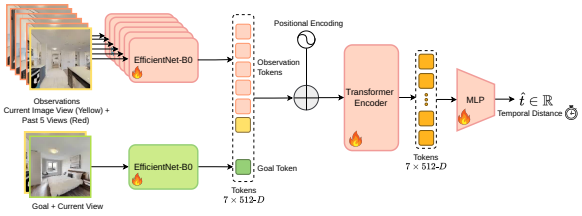


Figure 7. **Adapted ViNT Model Architecture [40].** We modify the original ViNT model by removing the action prediction head and retaining only the temporal distance prediction branch. The model encodes past observations and goal images with two EfficientNet-B0 encoders, fuses the resulting features into a sequence of tokens, and processes them with a Transformer to predict the (temporal) distance to the goal.

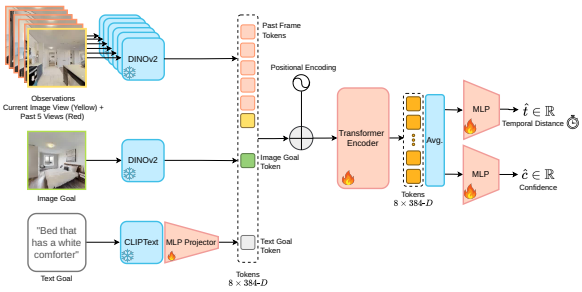


Figure 8. **Encoder-based alternative.** Observation and goal tokens are concatenated and processed jointly with a Transformer encoder, following the ViNT design but using frozen DINOv2 and CLIP encoders.

trainable encoders, ranging from lightweight CNNs to DINOv2. In practice, we restrict these baselines to image goals, as they already struggle to model distances reliably, and extending them to text proved unfeasible.

(iv) Quasimetric neural networks. Finally, we consider using the distance head with a quasimetric neural network (QNN) [43], which enforces quasimetric constraints in the embedding space by design. This guarantees desirable geometric properties such as asymmetry and triangle inequality, aligning with our QRL-inspired objective (Section H). We embed this into the VIP-style architecture of Figure 9.

F. VLD Architecture and Training Setup

Tables 6 and 7 summarize the hyperparameters of our Single-View VLD model and training setup. We set the maximum temporal distance to $td_{\max} = 100$ steps, corresponding to approximately 25 meters in Habitat, and train using the inlier–outlier Gaussian mixture NLL loss, with the reliable and outlier variances fixed at $\sigma_R^2 = 4$ and $\sigma_O^2 = 40$,

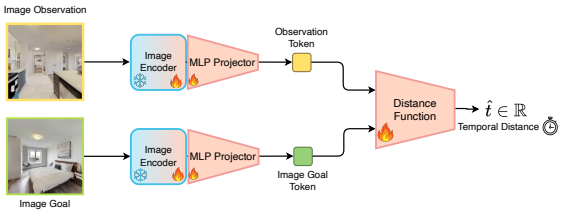


Figure 9. **VIP-style alternatives.** Observations and goals are encoded separately via some image encoder that can either be frozen or trainable, and distance is computed directly in the embedding space (via ℓ_2, linear, or quasimetric neural networks).

Table 6. Hyperparameters for the Single-View VLD model.

VLD Model	
Total # Parameters	97M
Trainable # Parameters	11M
Image Encoder	DINOv2-Small
Image Input Res.	224×224
Image Token Dimension	384
Text Encoder	CLIPText (ViT-B/32)
Max Text Sequence Length	64
Text Token Dimension	512
Attn. Hidden Dim.	384
# Decoder Layers	4
# Attention Heads	8
Activation Function	GELU

respectively.

Although the model has a total of 97M parameters, only 11M are trainable. The remainder comes from the frozen vision encoder (DINOv2-Small, ∼22M parameters) and the text encoder (CLIPText, ∼64M parameters). This means that for image-goal distance prediction, the model effectively relies on only ∼33M parameters in total, making it directly comparable in scale to existing baselines such ViNT [40] and VIP [27].

The Multi-View variant differs only in that it includes a temporal pooling module consisting of a single self-attention layer across $T = 5$ past frames (Figure 10), which adds roughly 2M trainable parameters. Unless otherwise stated, all experiments reported under "VLD" refer to the Single-View variant as our default comparison model.

G. Distance Function Ablations

Beyond the experiments with training data presented in Section 4.1.3 in the main paper body, which showed that VLD benefits from data scaling laws, we perform additional ablations to examine two key design choices: (i) the impact of using multi-view versus single-view inputs, and (ii) the effect of different learning objectives.

Table 7. Training setup for the Single-View VLD model.

Training	
# Steps	2M
Batch Size	64
Learning Rate	5×10^{-5}
Optimizer	AdamW
LR Schedule	Cosine
Warmup Steps	10K
Max temporal dist. ($td_{\max}$)	100
Neg. mining prob. (p_{neg})	0.05
Conf. "reliable" var. (σ_R^2)	4
Conf. "outlier" var. (σ_O^2)	40
Compute Resources	4× RTX 3090
Training Walltime	120 h

Table 8. **Single-view vs. multi-view inputs.** Ordinal consistency (Kendall's τ, ↑ higher is better) of VLD models on Habitat datasets. We report results at 20, 50, and 100-step horizons.

Model	HM3D (↑)			Gibson (↑)		
	20	50	100	20	50	100
VLD (single)	0.82	0.70	0.62	0.84	0.73	0.71
VLD (multi)	**0.82**	**0.71**	**0.63**	**0.86**	**0.75**	**0.73**

G.1. Multi-view inputs

Drawing inspiration from ViNT [40], which leverages multiple past frames for improved spatial reasoning, we extend our VLD architecture to support multi-view inputs, shown in Figure 10. The model independently encodes a sequence of recent observations using DINOv2, applies self-attention across temporal tokens, and averages the resulting representations before decoding them into the final scalar distance prediction.

Results on Habitat datasets. Table 8 compares single-view and multi-view VLD models. The multi-view variant achieves slightly higher ordinal consistency across all horizons and both Habitat datasets. However, the improvements are relatively small compared to the additional computational cost, suggesting that temporal context offers modest robustness gains while the single-view design remains a more efficient choice for most downstream settings.

G.2. Learning objectives

We further compare our Gaussian mixture negative log-likelihood (NLL) objective against several alternatives: (i) mean squared error (MSE), (ii) the implicit time-contrastive loss used in VIP [27], and (iii) the QRL loss [44]. Full formulations are provided in Appendix H.

Results on Habitat datasets. Table 9 summarizes the results. The Gaussian mixture NLL outperforms all alternatives, demonstrating that explicit confidence modeling helps the network down-weight ambiguous examples and focus on reliable pairs. In contrast, QRL exhibits unstable behavior—tending to saturate and under-predict large distances. These results confirm that our Gaussian mixture NLL provides the most stable and robust supervision signal for learning spatially consistent distance functions.

Table 9. **Comparison of learning objectives.** Ordinal consistency (Kendall's τ, ↑ higher is better) of VLD models trained with different objectives on Habitat datasets. Results shown for 20, 50, and 100-step horizons.

Objective	HM3D (↑)			Gibson (↑)		
	20	50	100	20	50	100
VLD (NLL)	**0.82**	**0.70**	**0.61**	**0.84**	**0.73**	**0.71**
VLD (MSE)	0.71	0.67	0.61	0.78	0.70	0.68
VLD (QRL)	0.17	0.20	0.16	0.06	0.07	0.07
VLD (VIP)	0.73	0.53	0.43	0.79	0.59	0.56

H. Training objectives

Although our main experiments rely on the Gaussian mixture NLL, we also experimented with alternative objectives that allow for self-supervised training and are conceptually attractive because they link the temporal distance function more directly to (i) reward learning from large-scale video (VIP) [27] and (ii) geometric consistency of value functions (QRL) [44].

As a baseline, following ViNG and ViNT, we first considered the simplest formulation: minimizing mean squared error (MSE) between predicted and ground-truth temporal distances [37, 40]. This approach is straightforward and often surprisingly effective, but it does not explicitly encourage the distance function to acquire useful geometric or representational structure. We therefore explored objectives designed specifically for temporal distance learning.

Value-Implicit Pretraining (VIP). VIP [27] was introduced as a way to learn temporal distance functions from large-scale human video by framing representation learning as an *implicit goal-conditioned RL problem*. Rather than regressing absolute distances, VIP trains a visual encoder such that the distance to the goal is reflected in the similarity of embeddings. Concretely, the learning objective is designed such that initial and goal frames are pulled closer together in embedding space, while intermediate frames are repelled through a temporal-difference style objective (see Algorithm 2). This produces embeddings that are smooth over time and encode a notion of goal progress.

The appeal of VIP in our context is that it provides a temporally consistent signal that does not depend on explicit action labels, aligning well with our emphasis on large-scale video pretraining. However, it is worth noting that VIP has been predominantly evaluated in manipulation domains, where image overlap between successive frames is

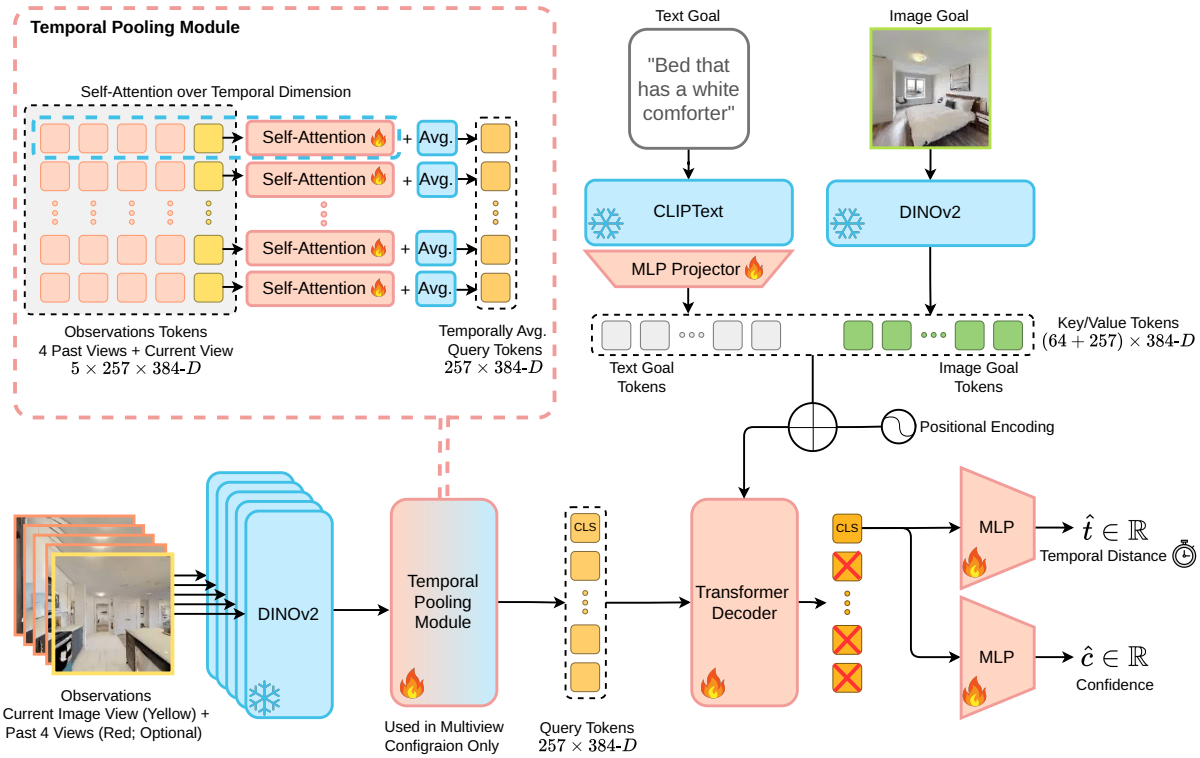


Figure 10. **Multiview Vision–Language Distance (VLD) architecture.** Similar to the single-view variant (Figure 2), but processes a short temporal window of past views. Each frame is encoded with DINOv2, followed by self-attention across temporal tokens before aggregation and decoding.

high and the visual signal remains relatively consistent. In contrast, navigation often involves long horizons with little or no visual overlap, so we wished to test whether VIP's benefits transfer to this more challenging setting.

Quasimetric RL (QRL). Inspired by Quasimetric RL (QRL) [44], we also considered training the distance function to satisfy *quasimetric* properties. A quasimetric is a generalization of a metric that allows asymmetry (i.e., $d(x, y) \neq d(y, x)$) while still respecting the triangle inequality. QRL shows that in multi-goal RL, the optimal value function is always a quasimetric, making these geometric constraints both natural and theoretically well-founded.

Formally, let $d_\theta(o, g) \in [0, D_{\max}]$ denote the predicted temporal distance from observation o to goal g. The QRL objective optimizes a Lagrangian that (i) *spreads* random pairs apart up to a margin $D_{\max}$ and (ii) enforces *local one-step consistency* on consecutive frames, without requiring any action labels (self-supervised learning objective):

Algorithm 2 Value-Implicit Pretraining (VIP) [27]

Require: Offline videos $D = \{(o_1^i, \ldots, o_{h_i}^i)\}_{i=1}^N$, visual encoder ϕ

1: **for** number of training iterations **do**
2: Sample sub-trajectories $\{o_t^i, \ldots, o_k^i, o_{k+1}^i, \ldots, o_T^i\}$ from D
3: Compute loss:

$$\mathcal{L}(\phi) = \frac{1}{B}\sum_{i=1}^{B} \|\phi(o_t^i) - \phi(o_T^i)\|_2^2 + \log \frac{1}{B}\sum_{i=1}^{B} \exp\left(\|\phi(o_k^i) - \phi(o_T^i)\|_2 - \gamma\|\phi(o_{k+1}^i) - \phi(o_T^i)\|_2\right)$$

4: Update ϕ using SGD: $\phi \leftarrow \phi - \alpha_\phi \nabla\mathcal{L}(\phi)$s
5: **end for**

$$\min_\theta \max_{\lambda \geq 0} \underbrace{\mathbb{E}_{o\sim p_{\text{state}},\, g\sim p_{\text{goal}}}\left[\,\psi\big(D_{\max} - d_\theta(o,g)\big)\,\right]}_{\text{global spreading}} + \lambda\left(\underbrace{\mathbb{E}_{(o,o')\sim p_{\text{trans}}}\left[\big(\text{relu}(d_\theta(o,o') - c)\big)^2\right]}_{\text{local one-step constraint}} - \varepsilon^2\right). \tag{6}$$

where $\psi(x) = \text{softplus}(x)$, c is the per-step cost (equal to the temporal distance between o and o'), and $\varepsilon > 0$ is a slack tolerance. In practice, we parameterize $\lambda = \text{softplus}(\rho)$, clamp it to a maximum value (to prevent degenerate solutions where the model becomes overly conservative), and apply a gradient-reversal trick on ρ to implement the $\max_{\lambda \geq 0}$ update.

The *global spreading* term encourages random (o, g) pairs to lie near the maximum margin $D_{\max}$, ensuring broad separation of states. The *local constraint* enforces that the predicted distance between neighboring observations[1] does not exceed the known step cost c, implemented via a squared hinge penalty that only activates when $d_\theta(o, o') > c$.

Intuitively, this formulation drives the model to behave like a *cost-to-go function*: distances accumulate along paths, preserve local step costs, and cannot "shortcut" around the triangle inequality. Unlike explicit regression to ground-truth temporal distances, the QRL objective indirectly shapes the function through geometric constraints and value consistency, potentially yielding a more robust and transferable inductive structure for navigation.

I. Ordinal Consistency: Qualitative Results

Procedure. For a set of held-out trajectories, we *independently* compute predicted distances at each frame for every model under evaluation, using the last frame as the goal image (and an optionally bootstrapped text prompt). For each trajectory, we form a time series of normalized distances per model and compute Kendall's τ against ground-truth temporal distance, yielding an ordinal consistency score per trajectory. We additionally render qualitative plots overlaying (i) predicted distance curves, (ii) ground-truth distance (normalized), (iii) model confidence (when available), and (iv) the goal image with optional bootstrapped text.

Interpreting the qualitative trajectories. These plots visualize the characteristic pattern that the predicted distance steadily decreases as the agent progresses along the trajectory, demonstrating that the learned distance function is *sensible and spatially consistent* even when intermediate observations have limited or no visual overlap with the goal.

Habitat examples (Figure 11). Consider an example trajectory from the shortest-path follower in Habitat. VLD predictions closely follow the ground-truth linear decrease in distance as the agent moves toward the goal, whereas baselines (e.g., VIP-Nav or ViNT-Tuned) tend to drift more often and fail to align with the true monotonic trend. When the goal image enters the agent's field of view, we observe spikes in confidence that coincide with sharp drops in predicted distance—precisely the expected behavior when the model detects visual cues from the goal. This illustrates that VLD not only encodes a reliable spatial prior but also expresses *uncertainty* in a meaningful, interpretable way.

Real-world examples (Figures 12–13). Across diverse in-the-wild scenes, we observe the same trend: predicted distances decrease consistently with motion toward the goal. Even when the goal is not directly visible, VLD maintains ordinal consistency and produces smooth, interpretable distance curves. Moments when the goal comes into partial or full view correspond to spikes in confidence and improved accuracy—mirroring the synthetic Habitat setting and demonstrating strong sim-to-real coherence.

Text-goal examples (Figure 14). Finally, comparing VLD under image-only, text-only, and multimodal goals reveals that all modalities exhibit the desired monotonic distance behavior, though the text-based variant is naturally noisier. Even when the goal description perfectly matches the observation, the predicted distance does not collapse exactly to zero—an expected outcome, reflecting semantic rather than pixel-level alignment. The multimodal model tracks the image-goal variant closely, indicating that VLD effectively integrates linguistic cues while preserving ordinal structure.

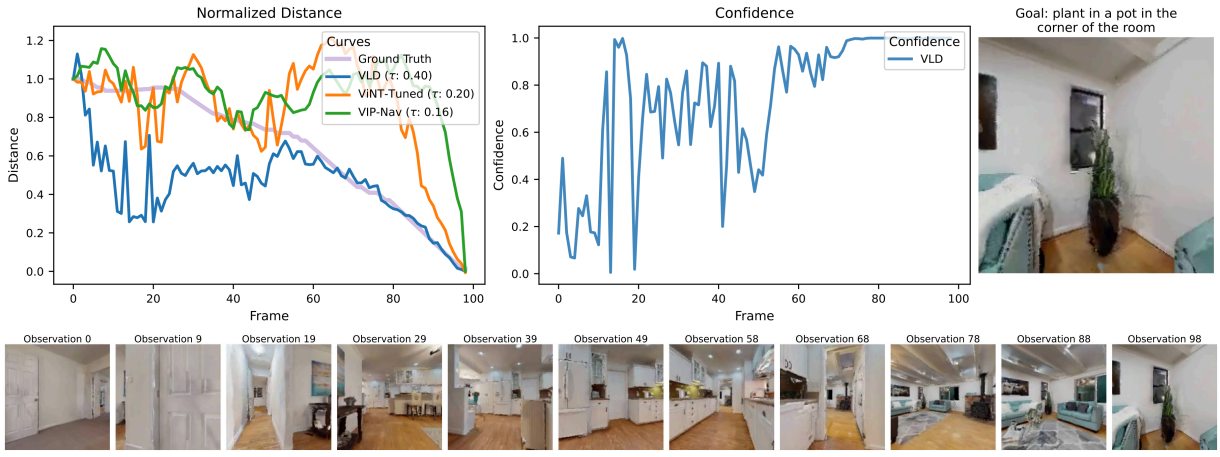
Videos. In addition to the figures presented in this section, we also include videos illustrating how the predicted distance evolves. These visualizations provide richer context by showing the agent's current observation at each step alongside the corresponding distance and confidence estimates. The videos are available at the `RL Distance Navigation` website.

J. Distance Accuracy Evaluation

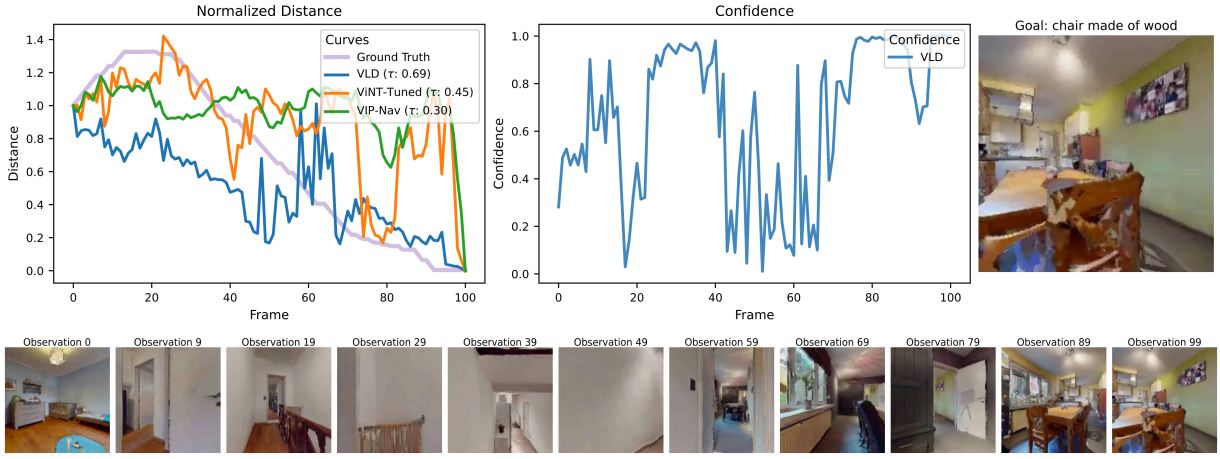
To complement ordinal consistency and more directly assess whether VLD provides a meaningful navigation signal when the goal is visually out of view, we introduce a *distance accuracy* metric that compares relative distances between pairs of observations.

For each validation episode in HM3D, we sample 100 viewpoints located in the same room as the goal, while explicitly selecting viewpoints and orientations such that the goal is *not* visible. We also sample 100 viewpoints located outside that room (see Figure 15). Pairing these viewpoints yields three comparison categories: (i) **in-in** (both viewpoints in the goal room), (ii) **out-in** (one in the goal room, one outside), and (iii) **out-out** (both outside the goal room). For each category, we form all possible pairwise comparisons of temporal distance to the goal, and compute the frac-

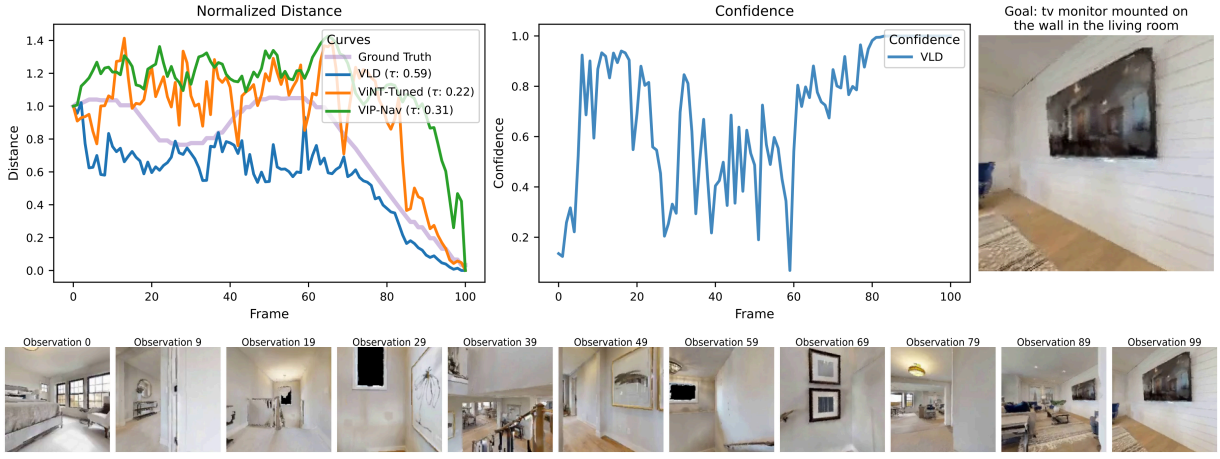
[1] o' is sampled such that the distance between o and o' is less than a set margin, $c \leq d_{\text{local}}$.



(a) Habitat Example 1

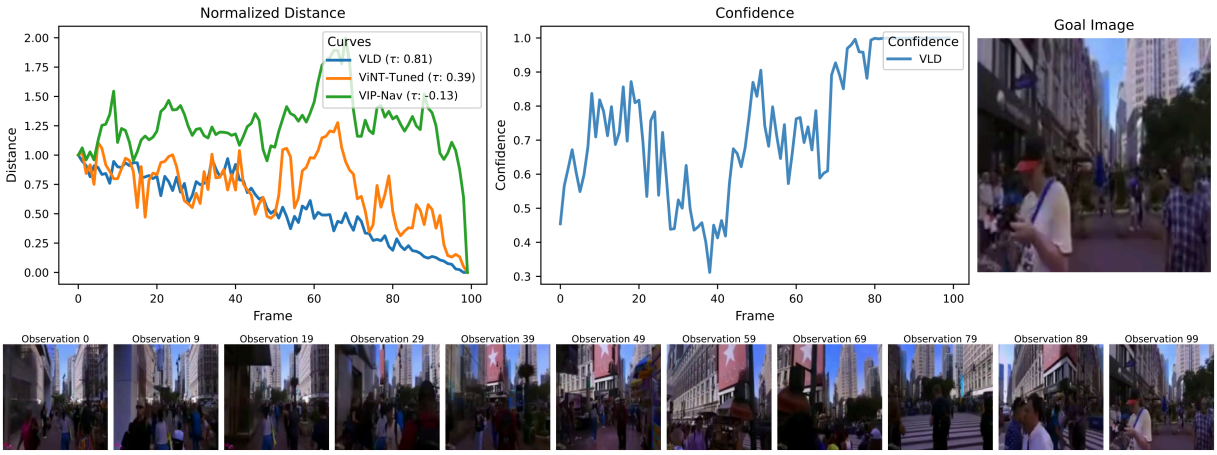


(b) Habitat Example 2

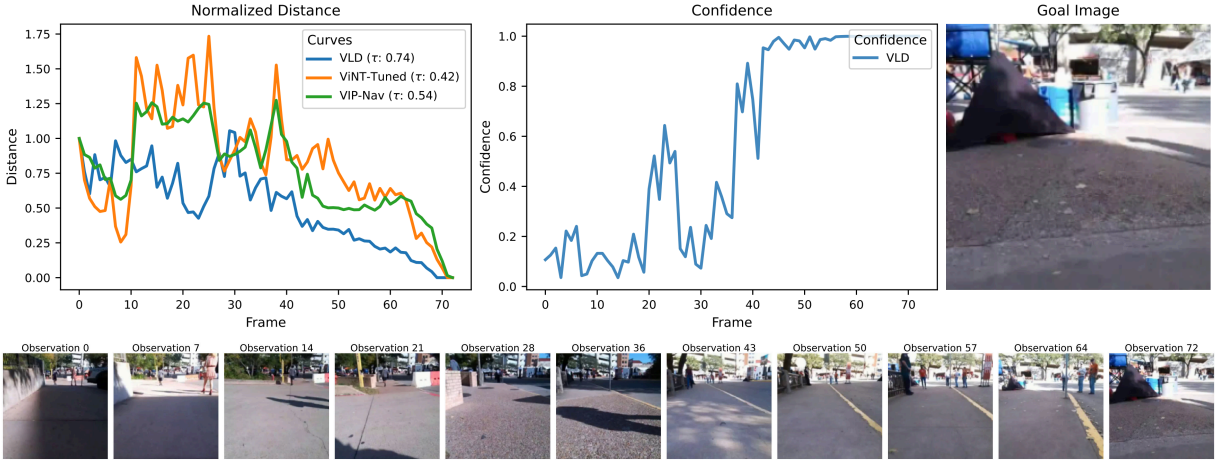


(c) Habitat Example 3

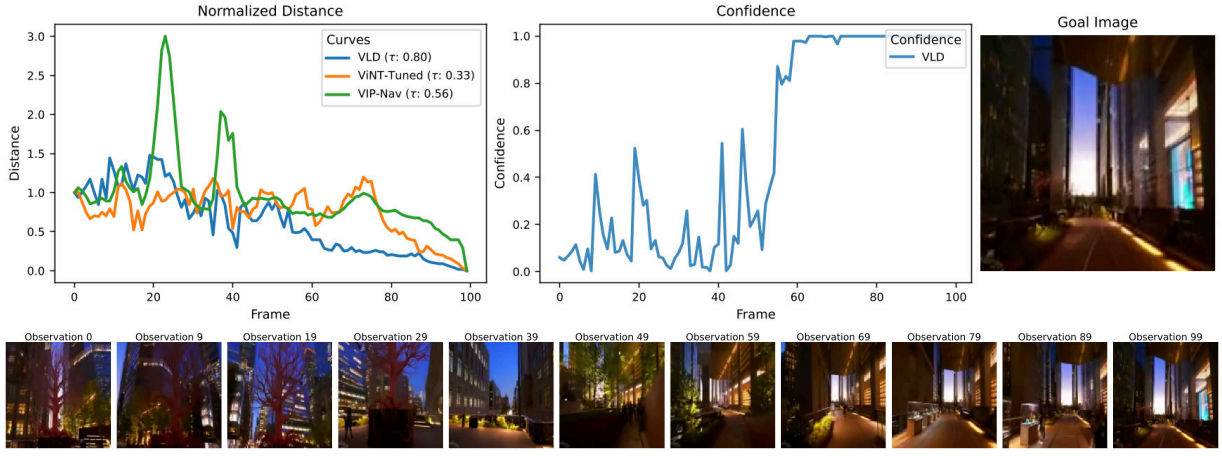
Figure 11. **Habitat qualitative trajectories.** Each top row shows a full trajectory plot (normalized predicted distances, GT distance, and Kendall's τ) with all models overlaid (VLD, ViNT-Tuned, VIP-Nav). Each bottom row displays the sequence of observations along the respective trajectory.



(a) Real-world Example 1

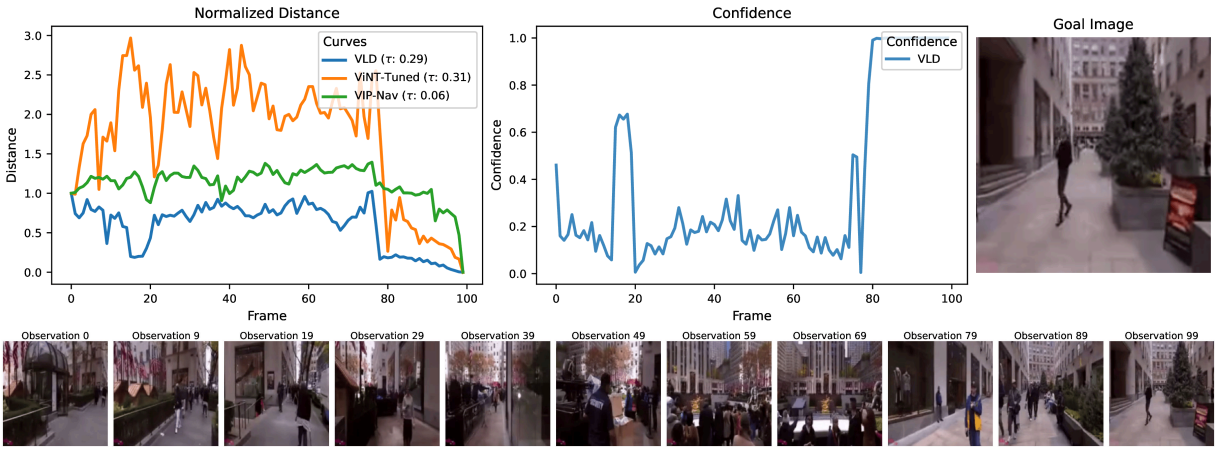


(b) Real-world Example 2

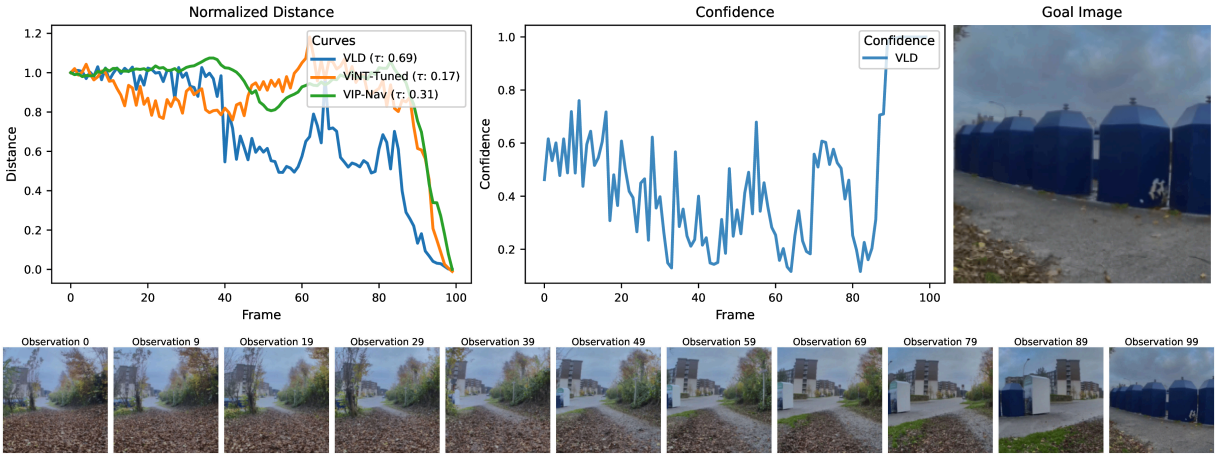


(c) Real-world Example 3

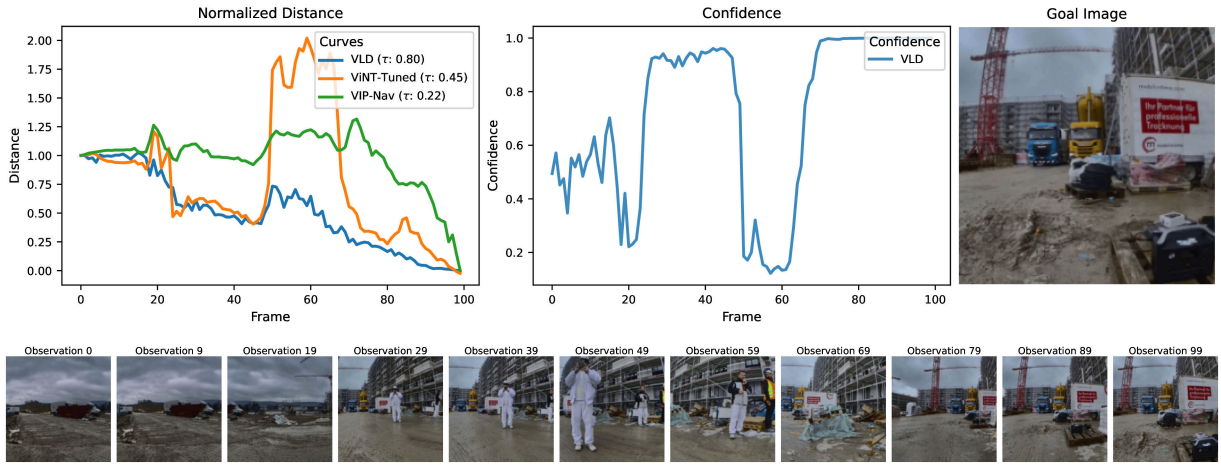
Figure 12. **Real-world qualitative trajectories (Part 1).** The top rows of examples 1–3 show diverse "in-the-wild" and embodiment scenes with normalized predicted distances and Kendall's τ overlaid for all models (VLD, ViNT-Tuned, VIP-Nav). Each bottom row displays the sequence of observations along the respective trajectory.



(a) Real-world Example 4

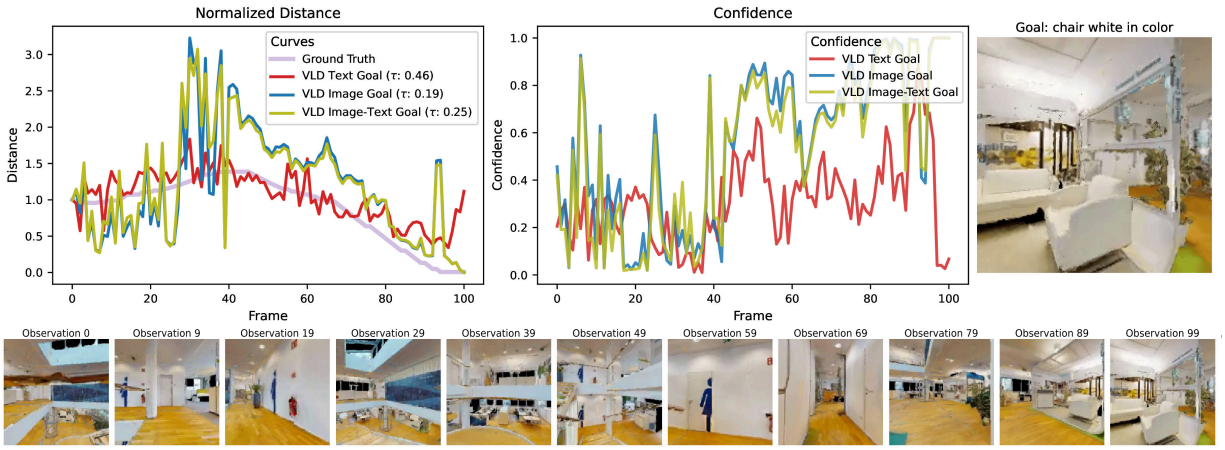


(b) Real-world Example 5

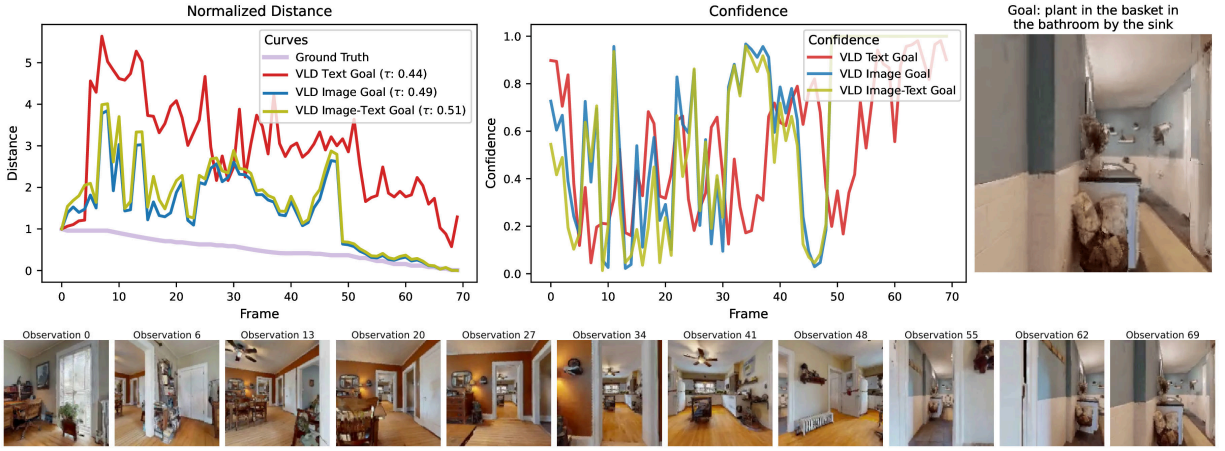


(c) Real-world Example 6

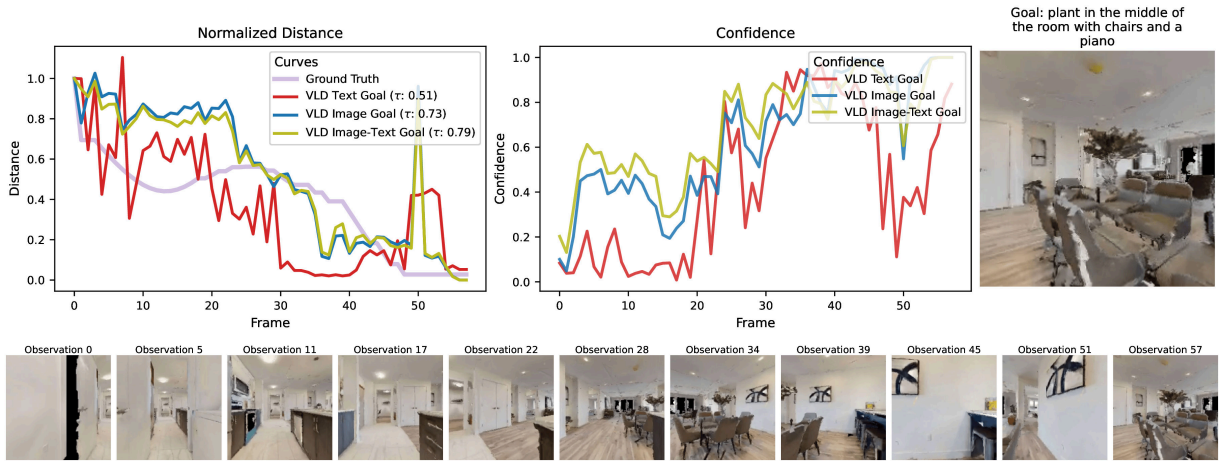
Figure 13. **Real-world qualitative trajectories (Part 2).** The top rows of examples 4–6 further demonstrate that VLD maintains stable ordinal behavior and decreasing predicted distances across visually diverse scenes. The bottom rows display the sequence of observations along the respective trajectory. Together with Fig. 12, these illustrate consistent real-world generalization.



(a) Text-goal Example 1 (VLD: image / text / multimodal)



(b) Text-goal Example 2 (VLD: image / text / multimodal)



(c) Text-goal Example 3 (VLD: image / text / multimodal)

Figure 14. **HM3D text-goal qualitative trajectories.** Each top row compares VLD with image-only, text-only, and multimodal (image+text) conditioning. Each bottom row displays the sequence of observations along the respective trajectory.

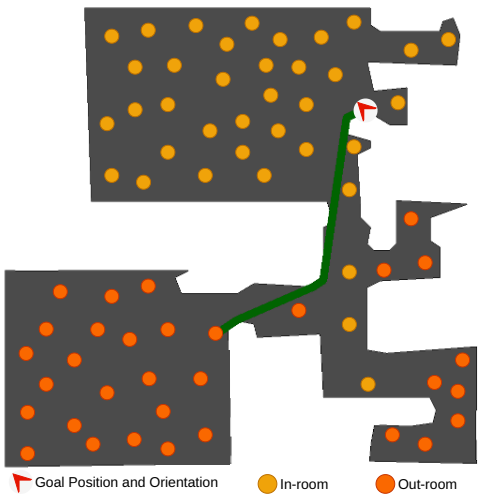


Figure 15. **Evaluating distance accuracy via spatial point sampling.** For each scene, we sample points both *inside* and *outside* the room containing the goal (red). Following the criterion illustrated here, a point is considered *in-room* if there exists at least one camera orientation at that location from which the goal image exhibits non-zero visual overlap; otherwise, it is labeled *out-room*. *A notable side effect of this definition is that certain hallway locations in this figure, adjacent to the goal room, may also be classified as "in-room," simply because the goal becomes marginally visible from specific orientations at those positions.*

tion of pairs for which VLD correctly predicts which viewpoint is closer according to ground-truth geodesic distance.

Results are summarized in Table 10. Despite the absence of any direct visual overlap with the goal in all three categories, VLD is able to identify the closer viewpoint well above random, indicating that the model learns a spatially coherent representation that provides meaningful directional guidance even when the goal is not visible. This complements the long-horizon ordinal consistency results, suggesting that VLD can gradually guide the agent toward the goal over extended navigation, reducing uncertainty as new observations are acquired.

We emphasize that this comparison is particularly challenging because of the geometry of indoor environments. First, in most training videos, the agent intentionally moves toward the final goal, meaning that upon entering a room, they typically observe objects or structures that also appear in the goal image. This creates a strong visual bias that is not present in our evaluation setup. Second, the difficulty is compounded by the geometry of indoor environments. Even when the agent stands in the same room as the goal image was captured, our evaluation forces it to consider orientations where the goal is not visible. In many realistic layouts, looking away from the goal often exposes features of other connected rooms—sometimes more prominently

Table 10. Distance accuracy on HM3D validation scenes. The metric reports the fraction of pairwise comparisons in which VLD correctly identifies the closer viewpoint relative to the goal.

Category	**Accuracy (%, ↑)**
In–In (both in goal room)	58.61
Out–In (one in goal room)	65.48
Out–Out (neither in goal room)	53.32

than the features of the room that actually contains the goal. For example, a goal image taken in a living room may face one direction, while the opposite orientation reveals an adjacent kitchen rather than the living room itself. As a result, the "same-room but opposite orientation" case can be surprisingly ambiguous and visually misleading, making the viewpoint-distance estimation task substantially harder than it may appear.

K. Distance Noise Models Ablation

K.1. Ornstein-Uhlenbeck (OU) Distance Noise

Besides geometric overlap noise (Section 3.2.2), we experiment with one more distance-noising method. Since recurrent policies (LSTMs, GRUs) can inherently filter out i.i.d. Gaussian noise [5, 15], we do not use this most straightforward approach for applying noise. Instead, we introduce temporally correlated noise based on *Ornstein–Uhlenbeck (OU) process*, a type of "red noise" widely used in RL exploration [16, 24]. For each episode, we maintain a noise state updated as:

$$\epsilon_t = \alpha\,\epsilon_{t-1} + \sigma\,\xi_t, \quad \xi_t \sim \mathcal{N}(0,1),$$

where $\alpha = 0.9$ and $\sigma = 0.1$ are fixed hyperparameters controlling correlation and magnitude. To mimic unpredictable prediction failures, we additionally inject occasional spike perturbations. Concretely, at each timestep we sample a Bernoulli indicator $b_t \in \{0,1\}$ with probability 0.05 of being active, and—conditioned on a spike occurring—a Gaussian amplitude $\Psi_t \sim \mathcal{N}(0,4)$. The final noise is therefore

$$\epsilon'_t = \epsilon_t + b_t\,\Psi_t, \qquad b_t \sim \text{Bernoulli}(0.05),\ \Psi_t \sim \mathcal{N}(0,4).$$

For more realistic behavior, the noise signal is applied to the ground-truth *geometric distance-to-goal* (d) provided by the simulator, conditioned as follows:

$$\hat{d} = \max\big(0,\ d + \epsilon'_t\big), \qquad |\epsilon'_t| \le f(d) = \exp\big(\sqrt{d}\big) - 1,$$

where $f(d)$ decreases with d, reducing the admissible noise magnitude near the goal. This prevents unrealistic negative distances or excessively large errors when the agent is close to the target.

Confidence values are derived directly from the noise level, as by design, confidence should decrease when distance predictions are unreliable. In other words, the higher

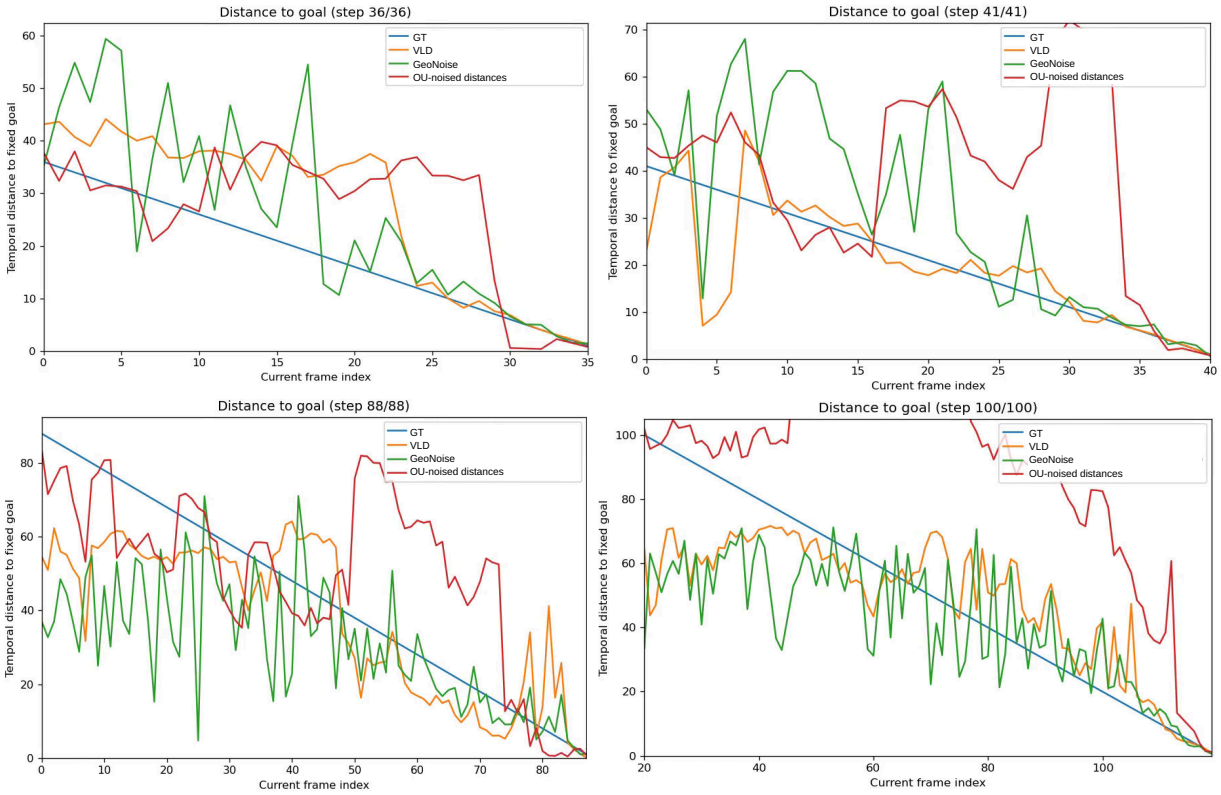


Figure 16. Four example trajectories illustrating the behavior of different noise models. Ground-truth (GT) temporal distance (a linearly decreasing signal) is shown in blue. VLD predictions are shown in orange. The GeoNoise—the learned geometric-overlap noise model—produces the curves shown in green, while OU-noised distances are shown in red, obtained by applying an Ornstein–Uhlenbeck process directly to the privileged ground-truth distances.

the noise magnitude applied to the ground truth, the lower the confidence should be provided to the policy. We simulate this by first computing a proxy:

$$c' = \exp(-\kappa|\epsilon'_t|),$$

where κ is a scaling constant, and then perturbing c' with an independent OU process (ϕ) to avoid trivially revealing the injected noise to the policy:

$$\hat{c} = \text{clip}(c' + \phi_t,\ 0,\ 1).$$

The final observation to the policy is thus the pair $(\hat{d}, \hat{c})$ of *noisy geometric distance-to-goal* and noisy confidence, producing uncertainty patterns aligned with the behavior expected from the VLD model trained under the mixture NLL objective.

K.2. Noise Ablation Results

Table 11 summarizes the comparison between Ornstein–Uhlenbeck (OU) noise and our geometric-overlap noise model. OU noise does not significantly hinder policy learning when used during RL training; in fact, its performance remains close to the noise-free baseline. However, as illustrated in Figure 16, OU noise fails to capture the distribution of the VLD predictions.

In contrast, the GeoNoise model not only mimics the statistical distribution of VLD outputs more faithfully but also induces policy behaviors that are crucial for downstream navigation driven by a vision(-language)-based distance function. Policies trained with GeoNoise learn to perform periodic 360° scans at key locations—a behavior aligned with the fact that VLD's predicted distance can vary dramatically with orientation, depending on whether goal cues are visible. This enables the policy to actively seek informative viewpoints and makes the VLD signal meaningfully actionable during deployment (see Appendix L).

Practically, this difference is substantial: although both noise models approximate the marginal variance of VLD, only GeoNoise replicates *the structural nature* of VLD errors. Policies trained under OU noise, therefore, lack the exploration strategy needed to exploit VLD predictions, re-

Table 11. Navigation performance on Gibson (validation). "Swap" indicates replacing (noised) ground-truth distance with VLD at deployment. We report success rate (SR↑) and success weighted by path length (SPL↑).

Policy Configuration	SR (↑)	SPL (↑)
Privileged Training (GT Distance)		
GT Distance (no noise)	**0.9577**	**0.6103**
GeoNoise	0.9091	0.5547
GeoNoise + Confidence	0.8994	0.5809
OU	0.9452	0.6459
OU + Confidence	0.9726	0.7209
Swap: Replace Distance with VLD at Deployment		
VLD + (GeoNoise)	**0.7314**	**0.3995**
VLD + (GeoNoise + Confidence)	0.6821	0.3860
VLD + (OU)	0.1145	0.0400
VLD + (OU + Confidence)	0.4708	0.3150

sulting in poorer transfer. By contrast, policies trained with GeoNoise transfer significantly better, and this benefit extends even to other distance functions such as ViNT (see Section 4.2), indicating that the learned overlap-conditioned noise serves as a generally useful proxy for image-based distance prediction models.

L. Navigation Policy: Qualitative Results

We visualize representative rollouts from the navigation policy when deployed with VLD. These examples illustrate how the agent (i) explores previously unseen spaces, (ii) uses VLD feedback to prioritize promising directions, (iii) detects partial visual cues from the goal, and (iv) converges to the target. We also show characteristic failure modes.

Videos. Additionally, full-trajectory visualizations are provided at the `RL Distance Navigation` website, showing how policies behave throughout entire rollouts.

L.1. Successful Navigation

Exploration behavior. When entering an unfamiliar room, the agent typically performs a brief 360-degree scan—often multiple partial rotations—before committing to a direction. VLD encourages movement toward headings with lower predicted distances: as the agent rotates, it evaluates several candidate viewpoints, compares predicted distances, and then proceeds along the direction with the minimum value. See Figure 17 for an illustration.

Detecting and approaching the goal. In successful rollouts, the agent eventually encounters partial visual overlap with the goal image. At this moment, VLD confidence sharply increases while the predicted distance drops, giving the policy a decisive signal to commit to the correct direction. Figure 18 shows a sequence illustrating this behavior.

L.2. Failure Cases

To better characterize how and why navigation attempts fail, we categorize all unsuccessful episodes into three primary failure modes. Table 12 summarizes their relative frequencies. The majority of failures arise from time-outs (62%), followed by near-match cases (29%), and ambiguous or uninformative goal images (9%).

Table 12. Distribution of navigation failure modes. Percentages are computed over all unsuccessful episodes.

Failure Type	Percentage (%)
Time-out failure	62
Near-match failure	29
Bad or uninformative goal	9

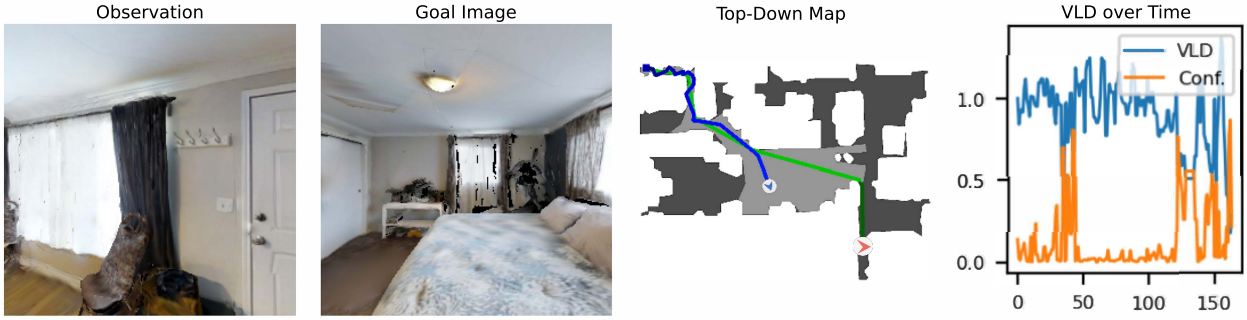
We visualize representative examples of these failure modes in Figure 19.

(1) Time-out failures. Most failures arise from trajectories where the agent becomes stuck exploring visually similar but spatially incorrect regions. Equipping the agent with explicit memory—either short- or long-term—of previously visited locations, for example via a topological map or episodic graph [8, 40], would likely mitigate such failures (see Figure 19a). However, memory augmentation is outside the scope of this work, as our focus is on evaluating VLD as a standalone navigation signal.

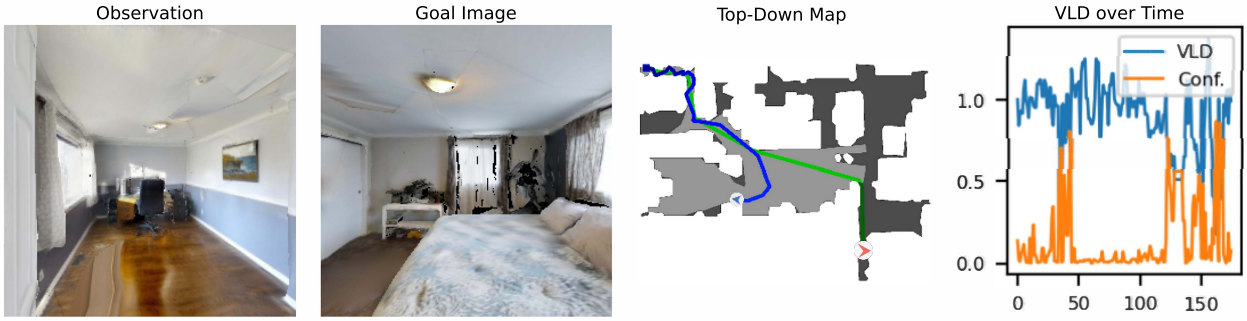
(2) Bad or uninformative goal images. A second failure class stems from task design in the Gibson environment. As goal viewpoints are sampled from random orientations, some goal images are highly ambiguous or visually uninformative (Figure 19b). In these cases, even an ideal distance function would struggle, as the target may not correspond to a distinctive or semantically meaningful observation.

(3) Near-match failures. A third failure mode originates from the properties of the distance function itself. In these *near-match failures*, the agent encounters a view that appears deceptively similar to the goal image even though the true goal is still physically distant (Figures 19c–19d). However, because the policy relies solely on a scalar distance signal—which is naturally biased toward feature overlap—the agent may incorrectly conclude that it has reached the goal and stop prematurely.

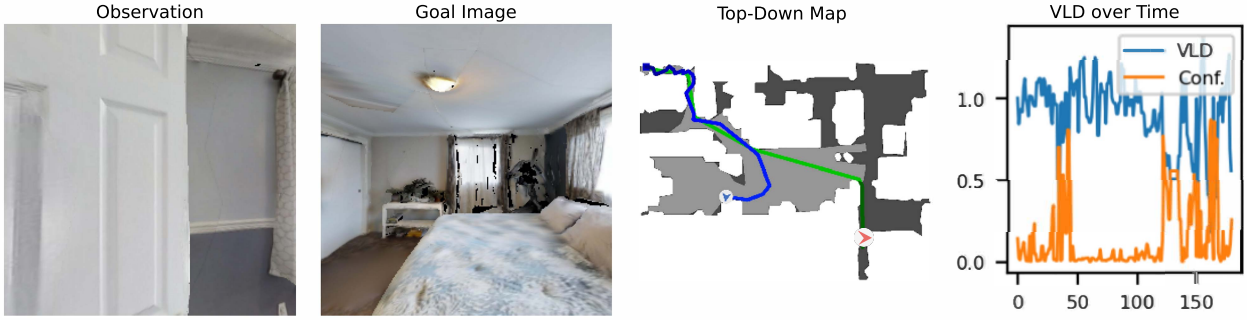
Often, the target object is fully visible from the agent's location, but from a different perspective. Depending on the navigation task specification, one could increase the success radius threshold such that these situations can be classified as successful (which could be justified since in these cases the agent has reached a position where the goal content is clearly seen), which could additionally boost the success rate of our method.



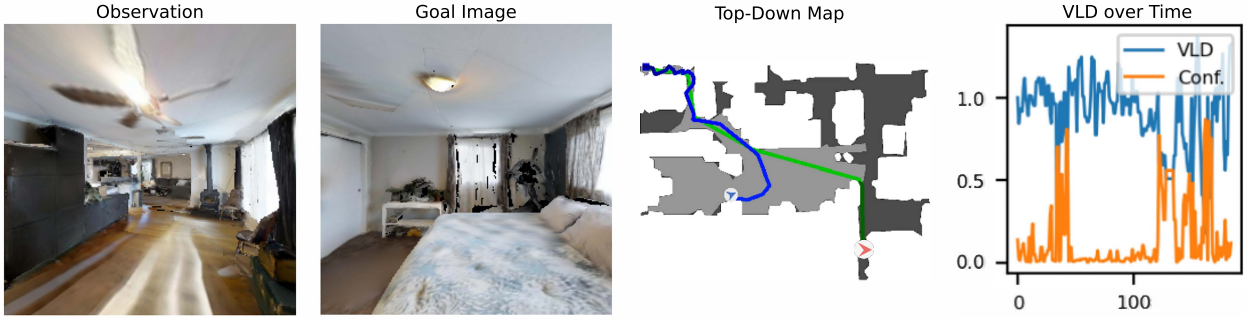
(a) The agent enters a new space with no immediate cues toward the goal.



(b) It performs an initial sweep, collecting VLD predictions across multiple headings.

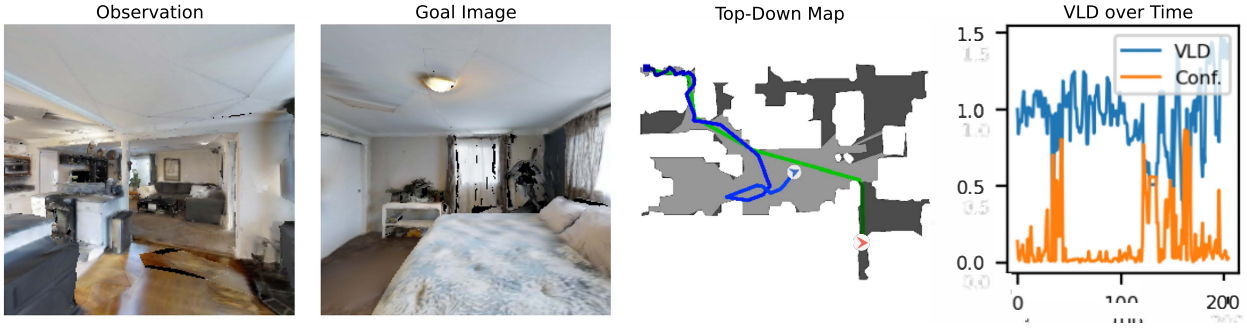


(c) A full rotation provides a broader set of viewpoints for comparing predicted distances.

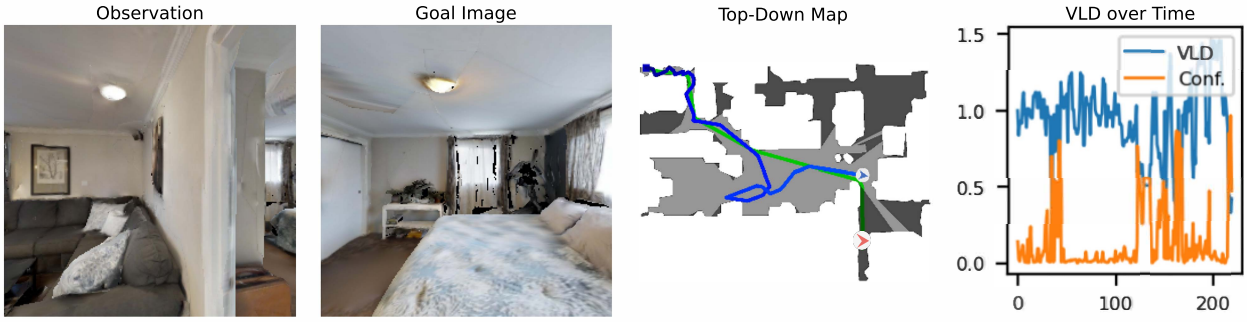


(d) The agent commits to the direction with the lowest VLD-predicted distance and continues moving.

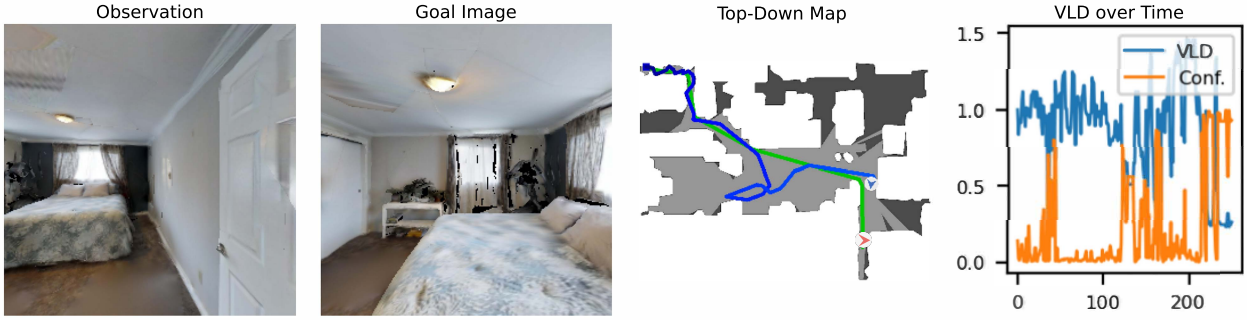
Figure 17. **Exploration driven by VLD.** The policy uses VLD as a scalar navigation signal, rotating and evaluating multiple viewpoints before selecting an exploration direction. Each step shows: (i) the agent's current observation (blue arrow) and the goal position and orientation (red arrow), (ii) the goal image, (iii) a top-down map with the taken trajectory in blue (green shows the optimal path), and (iv) the plot of VLD-predicted distance and confidence over past steps.



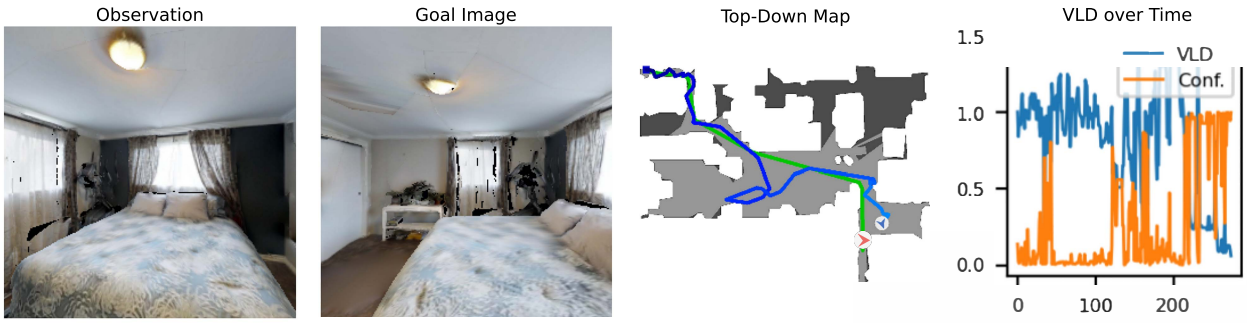
(a) The agent continues exploring, guided by VLD predictions.



(b) First partial cue from the goal appears; VLD confidence spikes and predicted distance drops.

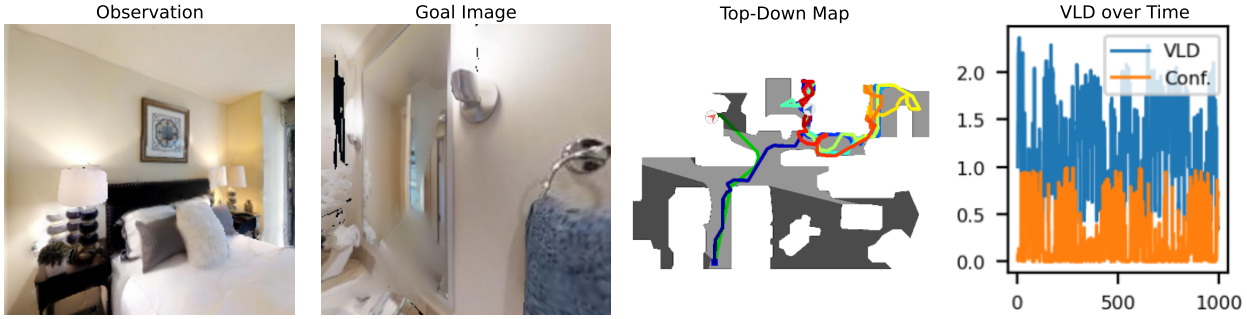


(c) The agent commits to the identified direction; distance decreases sharply.

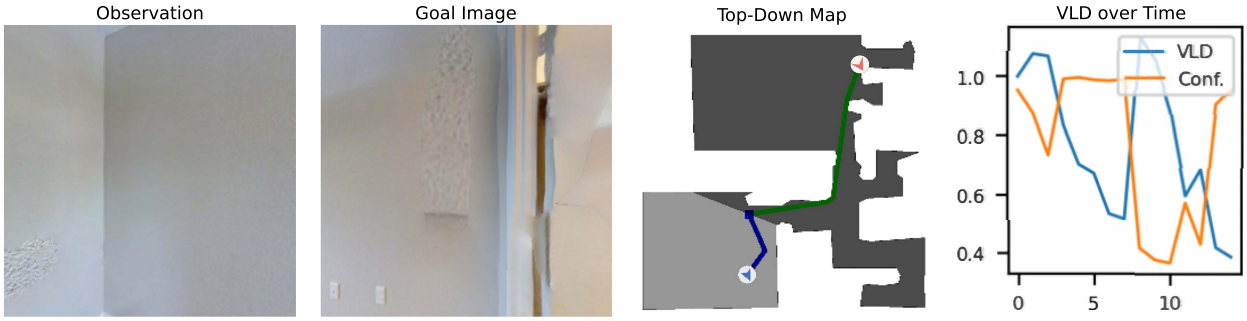


(d) The goal object is reached and the episode terminates successfully.

Figure 18. **Successful navigation.** Once partial visual overlap occurs, VLD's confidence and distance predictions tighten, enabling rapid convergence to the goal. Each step shows: (i) the agent's current observation (blue arrow) and the goal position and orientation (red arrow), (ii) the provided goal image, (iii) a top-down map with executed (blue) and optimal trajectories (green), and (iv) VLD distance and confidence plots.

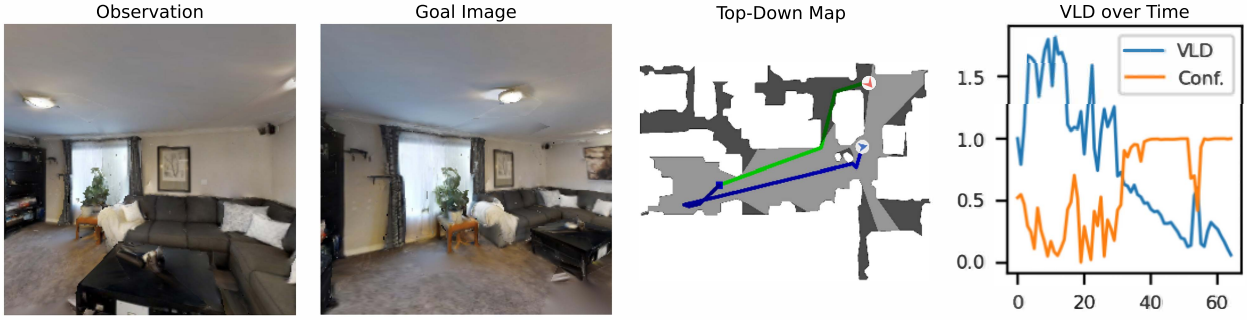


(a) Timeout: the agent explores but never finds a direction with sufficiently decreasing VLD distance.



(b) Ambiguous goal image induces misalignment between the intended and perceived target.

(c) Near-match failure: the observation looks deceptively similar to the goal image, but the true goal is physically distant.



(d) Another near-match failure: visual similarity misleads the policy despite incorrect spatial alignment.

Figure 19. **Failure cases.** Most failures arise from visually deceptive near-matches, ambiguous goal definitions, or insufficient memory and capacity for long-horizon exploration. Each example shows: (i) the agent's current observation (blue arrow) and the goal position and orientation (red arrow), (ii) the goal image, (iii) the top-down map with the executed (gradient color from dark blue at the start, to red color at the steps close to max steps limit) and optimal paths (green), and (iv) the VLD distance/confidence trace.