DDSeVA: Demand-Driven Semantic Value Navigation with Verified Target Acquisition

Anonymous authors
Paper under double-blind review

Abstract

Demand-driven navigation extends embodied navigation beyond predefined object search, requiring an agent to satisfy open-ended human needs in unknown environments. Existing methods typically map demands to concrete target objects and employ vision-language reasoning techniques to achieve this. However, correct demand-object grounding does not guarantee reliable task completion. The agent must search effectively before any target is observed and decide whether a detected candidate provides sufficient evidence to terminate, a challenge we term the detection-to-success gap. To bridge this gap, we propose **DDSeVA**, a closed-loop framework that couples Demand-Driven Semantic Value search with verified target Acquisition. During search, DDSeVA maintains a demand-grounded semantic state and ranks candidate locations according to their expected contribution to task success. During acquisition, each target detection is treated as a hypothesis and verified using temporal consistency, learned reachability estimates, and a prediction of whether `Done` would satisfy the success criterion. Closed-loop evaluation on ProcTHOR in AI2-THOR demonstrates that DDSeVA consistently outperforms existing methods across all four scene-instruction splits. Compared with the current state of the art, DDSeVA improves average navigation success rate relatively by 10.5%, success weighted by path length by 14.6%, and selection success rate by 30.6%. Code and video demos are available at `https://anonymous.4open.science/r/DDSeVA-6675`.

1 Introduction

When instructing embodied agents in indoor environments, users typically express tasks as functional needs rather than precise object categories. For example, a request like "find a place to sit" might be satisfied by a sofa, chair, or bed whose locations are initially unknown. This setting, formalized as **demand-driven navigation (DDN)**, extends beyond navigating to predefined object categories (Chaplot et al., 2020; Ramakrishnan et al., 2022; Majumdar et al., 2022; Dorbala et al., 2022) or following step-by-step routes (Anderson et al., 2018b; Fried et al., 2018; Tan et al., 2019; Hong et al., 2021). To succeed, the agent must continuously coordinate demand interpretation, environment exploration, and verified target acquisition (Wang et al., 2023; Huang et al., 2025). Translating these underspecified demands into reliable navigation behavior is therefore essential for deploying embodied agents in complex indoor environments (Kolve et al., 2017; Deitke et al., 2022; Anderson et al., 2018a).

Early work on DDN learns demand-conditioned mappings between language instructions and potential target objects (Wang et al., 2023). Subsequent studies extend this formulation to multi-object and preference-aware demands (Wang et al., 2024; Li et al., 2025), e.g., CogDDN incorporates vision-language reasoning and cognitive decision-making (Huang et al., 2025). Despite these advances in semantic grounding and task-level reasoning, the *detection-to-success gap* remains insufficiently addressed. A single-frame detection provides a target hypothesis but does not establish that the agent has reached a valid stopping state. Figure 1 illustrates this distinction: an agent detects a demand-relevant sofa but executes `Done` while still far away, outside the required stopping range. In contrast, DDSeVA continues to approach and verify the target, executing `Done` once the target is stably tracked and the predicted termination risk is low. Additionally, given the current demand and accumulated scene evidence, existing methods offer limited guidance on which locations are worth

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

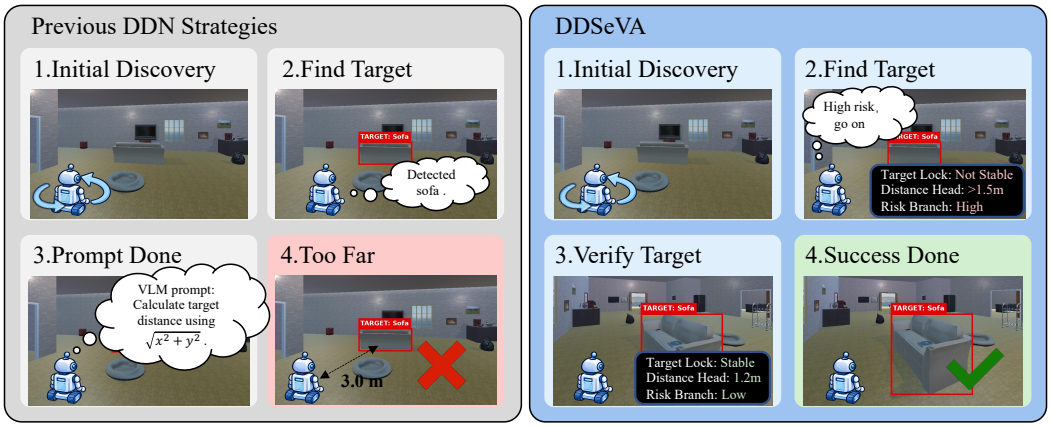


Figure 1: Illustration of the *detection-to-success gap* in demand-driven navigation. Previous DDN strategies may stop after a target-like detection, whereas DDSeVA verifies temporal stability, spatial reachability, and stopping risk before executing `Done`.

exploring. These limitations highlight two coupled decisions that require explicit modeling: *where to search before target discovery, and when the evidence gathered after detection is sufficient for successful termination.*

Addressing these decisions requires an evolving semantic state that integrates demand relevance, scene structure, and exploration history. By preserving contextual cues across observations, this state allows the agent to reason beyond individual frames and reuse evidence from previously explored regions. It should support two complementary capabilities: **success-oriented search**, which evaluates candidate locations by their expected contribution to task completion; and **verified target acquisition**, which assesses whether a detected proposal is temporally stable and spatially reachable for successful termination. As a result, these capabilities frame DDN as closed-loop decision making under partial observability, with the central challenge of converting incomplete search evidence into a reliable stopping state.

To this end, we propose **DDSeVA**, which couples Demand-Driven Semantic Value search with verified target Acquisition. During search, a semantic value model ranks executable candidates using the current demand and accumulated scene evidence to guide exploration. Once a demand-relevant target is detected, DDSeVA treats it as a hypothesis rather than an immediate stopping signal. The acquisition process combines temporal target locking and recovery, distance and reachability estimation, and learned predictions of `Done` success and termination risk. The resulting assessments guide continued approach, local recovery, or termination, connecting search and verification within a unified task-level closed loop.

The main contributions of this paper are summarized as follows:

- We identify and characterize the *detection-to-success gap* in demand-driven navigation, showing that detecting a relevant object does not establish a valid stopping state. This perspective reframes reliable task execution around two coupled processes: success-oriented search before target discovery and success-state verification after detection.
- We propose DDSeVA, a task-level closed-loop framework driven by two explicit mechanisms: a Semantic Value Network that ranks heterogeneous spatial candidates to guide efficient exploration under partial observability, and a verified acquisition module that evaluates visual detections as hypotheses by rigorously assessing their temporal stability, spatial reachability, and termination risk before authorizing task completion.
- Comprehensive experiments on the DDN benchmark demonstrate that DDSeVA consistently outperforms the evaluated baselines across all four scene–instruction environments. Ablation studies further validate the effectiveness of semantic value search and the individual components of verified target acquisition.

2 Related Work

Demand-driven navigation. Demand-driven navigation (DDN) extends object-goal navigation from predefined target categories to natural-language demands, requiring agents to infer which objects can satisfy the underlying needs. The original formulation addresses this ambiguity by learning a demand-conditioned object attribute space that aligns language-derived commonsense attributes with visual object representations (Wang et al., 2023). Subsequent studies broaden the task scope: MO-DDN considers multi-object demand fulfillment and user preferences (Wang et al., 2024), while TP-MDDN investigates long-horizon navigation with multiple preference-aware demands (Li et al., 2025). CogDDN strengthens semantic alignment and decision-making through vision-language reasoning, dual-process thinking, and reflective knowledge accumulation (Huang et al., 2025). These advances improve demand interpretation and task-level reasoning, but the coordination of search before target discovery and verification after detection remains less explicitly addressed. In particular, a semantically relevant detection alone does not establish that the agent has reached a valid stopping state. DDSeVA focuses on this execution-level challenge, coupling demand-conditioned semantic value search with verified target acquisition to connect target discovery with reliable termination.

Semantic exploration and foundation-model navigation. Semantic navigation exploits scene structure and contextual evidence to guide search when target locations are unknown. Object-goal navigation methods use semantic maps, scene priors, object relations, and hierarchical reasoning to guide category-specific exploration (Yang et al., 2018; Chaplot et al., 2020; Pal et al., 2021; Kiran et al., 2022), while learned potential functions provide spatial guidance for target search (Ramakrishnan et al., 2022). Zero-shot and open-vocabulary approaches extend this direction through multimodal goal embeddings, CLIP-based localization, frontier maps, commonsense constraints, and scene-graph prompting to improve generalization beyond fixed target categories (Majumdar et al., 2022; Dorbala et al., 2022; Zhao et al., 2023; Yokoyama et al., 2024; Zhou et al., 2023; Yin et al., 2024). Vision-language navigation addresses instruction following under partial observability, with methods based on progress estimation, backtracking, history modeling, topological planning, and pretraining to improve robustness over long horizons (Anderson et al., 2018b; Fried et al., 2018; Ma et al., 2019a; Ke et al., 2019; Chen et al., 2021; Hao et al., 2020; An et al., 2024). More recently, large language models and vision-language models have enabled explicit reasoning, map-guided prompting, multi-expert deliberation, video-based planning, and semantic-map memory (Long et al., 2024a; Zhou et al., 2024b; Chen et al., 2024; Zhou et al., 2024a; Long et al., 2024b; Zhang et al., 2024; 2025), alongside advances in embodied affordance grounding (Ahn et al., 2022; Driess et al., 2023). While these approaches strengthen semantic understanding and navigation planning, DDSeVA emphasizes the connection between search decisions and successful termination. It learns a shared value space for heterogeneous search candidates and progressively verifies whether detected proposals are temporally stable, spatially reachable, and sufficient for task completion.

3 Problem Formulation

We formulate demand-driven navigation as a partially observable sequential decision-making process in an unknown indoor scene. Let $\mathcal{S}$, $\mathcal{D}$, and $\mathcal{Y}$ denote the sets of scenes, demand instructions, and object categories, respectively. An episode is initialized by $(s, d, \mathbf{x}_0)$, where $s \in \mathcal{S}$ is the scene, $d \in \mathcal{D}$ is a natural-language demand, and $\mathbf{x}_0$ is the initial agent pose. The set of demand-satisfying object categories is defined as $\mathcal{Y}_d = \{y \in \mathcal{Y} \mid G(d, y) = 1\}$, where $G : \mathcal{D} \times \mathcal{Y} \to \{0, 1\}$ is the benchmark-defined demand-object discriminator.

At step t, the agent receives an RGB observation $\mathbf{o}_t$ and chooses an action $a_t \in \{\texttt{MoveAhead}, \texttt{RotateLeft}, \texttt{RotateRight}, \texttt{LookUp}, \texttt{LookDown}, \texttt{Done}\}$. When executing `Done`, the agent must also predict a target bounding box $\hat{\mathbf{b}}_t$. An episode succeeds if there exists a visible demand-satisfying object within a distance threshold $d_{\max}$ whose ground-truth box overlaps the predicted box with an intersection over union (IoU) of at least θ_{iou}:

$$\text{Succ}(\tau, d) = \mathbb{I}[\exists \omega \in \mathcal{I}_\tau^{\text{vis}} : y(\omega) \in \mathcal{Y}_d,\ \text{dist}_{xz}(\mathbf{x}_\tau, \mathbf{p}(\omega)) \le d_{\max},\ \text{IoU}(\hat{\mathbf{b}}_\tau, \mathbf{b}(\omega)) \ge \theta_{\text{iou}}], \quad (1)$$

where $\mathcal{I}_\tau^{\text{vis}}$ is the set of object instances visible at the termination step τ, and $\mathbf{x}_\tau$ represents the agent's final position. For an instance ω, $y(\omega)$ and $\mathbf{b}(\omega)$ denote its object category and ground-truth bounding box, while $\mathbf{p}(\omega)$ denotes its spatial position.

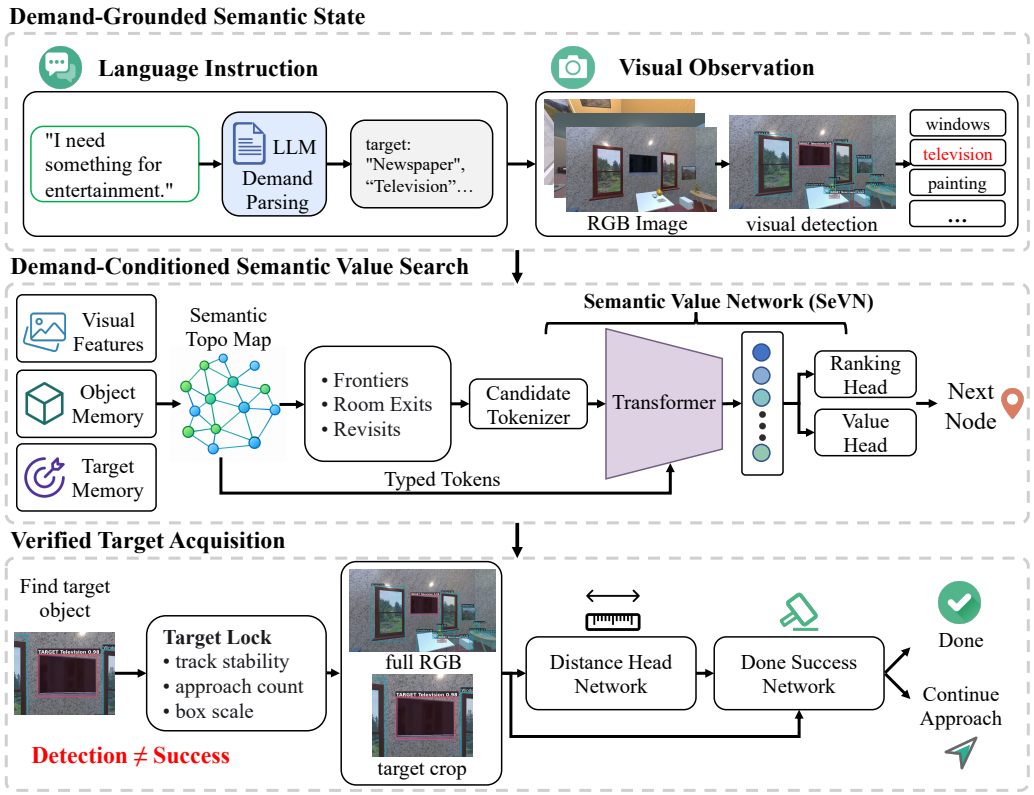


Figure 2: Overview of DDSeVA. The framework first builds a demand-grounded semantic state, then performs demand-conditioned semantic value search before target acquisition, and finally verifies the detected target proposal before executing `Done`.

4 METHOD

As shown in Figure 2, DDSeVA operationalizes demand-driven navigation not as a linear pipeline, but as a continuous hypothesis-generation and verification loop. First, the agent builds a **demand-grounded semantic state** that continuously absorbs online perceptual evidence to bridge abstract human needs with physical environments. Based on this state, **demand-conditioned semantic value search** acts as a spatial hypothesis generator, guiding the agent to explore promising regions and discover potential targets under partial observability. Finally, rather than treating a visual detection as a terminal signal, **verified target acquisition** acts as a rigorous hypothesis tester. It progressively evaluates the temporal stability, spatial reachability, and termination risk of the proposal. Crucially, if a target is rejected, this negative evidence seamlessly updates the semantic state, prompting the agent to resume exploration until task success is robustly confirmed.

4.1 DEMAND-GROUNDED SEMANTIC STATE

To bridge abstract human needs with the physical environment, DDSeVA initially builds an evolving task state that connects user instructions with raw perceptual evidence. Specifically, an adapted Qwen model first distills the natural-language demand d into a compact demand profile $\mathcal{Q}(d)$—a discrete set of object classes capable of satisfying the task. Guided by this semantic objective, the agent continuously extracts visual evidence from the scene using an online YOLO detector. At each step t, the current RGB observation $\mathbf{o}_t$ is processed into a set of visual detections $\mathcal{O}_t$, where each proposal is characterized by its predicted class z_i, bounding box $\mathbf{b}_i$, and confidence score γ_i. Detections whose predicted class matches the demand profile are instantiated as target hypotheses:

$$\mathcal{T}_t = \{(z_i, \mathbf{b}_i, \gamma_i) \in \mathcal{O}_t \mid z_i \in \mathcal{Q}(d)\}\,. \tag{2}$$

These hypotheses trigger verified target acquisition, while all other detections provide contextual cues for scene exploration. Details of Qwen and YOLO adaptation are provided in Appendix C.1.

To support decisions beyond the current frame, DDSeVA aggregates the interaction history and the demand profile into an evolving semantic state:

$$\mathcal{B}_t = \left(\mathcal{N}_t, \mathcal{R}_t, \mathcal{M}_t^{\mathrm{obj}}\right), \tag{3}$$

where $\mathcal{N}_t$, $\mathcal{R}_t$, and $\mathcal{M}_t^{\mathrm{obj}}$ denote node memory, room memory, and object memory, respectively. These components summarize visited locations, explored room regions, observed objects, and their demand relevance. Crucially, when precise instance identities are unavailable, object memories are anchored to their observed spatial nodes and rooms rather than merged merely by object class. This design preserves visually similar objects in different regions as distinct evidence.

Finally, the semantic state $\mathcal{B}_t$ is converted into a set of executable candidates $\mathcal{C}_t$, comprising frontiers, room exits, revisitation nodes, object regions, and the previously extracted target hypotheses $\mathcal{T}_t$. These candidates define the decision alternatives for the subsequent semantic value search and verified target acquisition stages. Additional details of memory update and candidate construction are given in Appendix C.2.

4.2 Demand-Conditioned Semantic Value Search

Before a target is verified, the agent must decide which executable candidate is most likely to lead to eventual task success under partial observability (Wang et al., 2018; Ma et al., 2019b). This is a fundamental challenge because frontiers, room exits, revisitation nodes, object regions, and target hypotheses provide entirely different types of semantic evidence and incur varied spatial costs. DDSeVA introduces the **Semantic Value Network** (SeVN). SeVN innovatively projects these structurally heterogeneous candidates into a unified, demand-conditioned value space, empowering the agent to dynamically evaluate diverse exploration strategies through a single principled metric.

Given the semantic state $\mathcal{B}_t$, SeVN constructs a token sequence $\mathcal{X}_t$ that densely summarizes the current demand, agent node, historical memory components, and the available candidate set $\mathcal{C}_t$. Instead of evaluating candidates in isolation, a Transformer architecture is employed to model the global dependencies across $\mathcal{X}_t$. This allows SeVN to deeply fuse linguistic demand semantics with spatial memory layout, extracting a highly contextualized representation $\mathbf{h}_{t,i}$ for each candidate c_i (Pashevich et al., 2021; Chen et al., 2022). SeVN then maps $\mathbf{h}_{t,i}$ to a ranking logit $\ell_{t,i}$ and an auxiliary semantic value $\hat{v}_{t,i}$:

$$\ell_{t,i} = f_\theta^{\mathrm{rank}}(\mathbf{h}_{t,i}), \qquad \hat{v}_{t,i} = f_\theta^{\mathrm{value}}(\mathbf{h}_{t,i}), \tag{4}$$

where f_θ^{rank} and $f_\theta^{\mathrm{value}}$ are the ranking and value prediction heads, respectively.

Crucially, we design the training supervision to reflect a rigorous cost-benefit analysis of spatial exploration, deriving dense progress signals directly from the environment's underlying topology (Ma et al., 2019a;b). Let $\mathcal{N}_{\mathrm{succ}}$ be the set of success nodes. For a candidate c_i anchored at node n_i, let D_i denote its shortest topological distance to $\mathcal{N}_{\mathrm{succ}}$. We convert D_i into a soft value $v_i = \exp(-D_i/\tau_v)$, where τ_v controls the decay. Furthermore, let D_t be the distance from the agent's current node to $\mathcal{N}_{\mathrm{succ}}$, and κ_i be the path cost to reach n_i. Unlike standard behavioral cloning that mimics surface actions, our offline teacher explicitly optimizes for spatial efficiency by selecting the valid candidate that maximizes the expected distance reduction:

$$i_t^+ = \underset{i:\, c_i \in \mathcal{C}_t^{\mathrm{valid}}}{\arg\max} \left(D_t - \kappa_i - D_i\right), \tag{5}$$

where i_t^+ is the index of the teacher-selected candidate. By systematically rewarding proximity to the goal (D_i) while penalizing the execution overhead (κ_i), this formulation forces the network to learn the optimal trade-off between exploring promising areas and minimizing travel distance. Tie-breaking rules and the treatment of unreachable candidates are provided in Appendix C.3.

SeVN is optimized using a dual-objective loss, coupling a cross-entropy ranking loss for relative candidate prioritization with a masked mean-squared error for absolute value regression:

$$\mathcal{L}_{\mathrm{search}} = -\log \frac{\exp(\ell_{t,i_t^+})}{\sum_{j:\, c_j \in \mathcal{C}_t^{\mathrm{valid}}} \exp(\ell_{t,j})} + \alpha \frac{\sum_{i:\, c_i \in \mathcal{C}_t} m_{t,i} \left(\hat{v}_{t,i} - v_i\right)^2}{\sum_{i:\, c_i \in \mathcal{C}_t} m_{t,i}}, \tag{6}$$

where $m_{t,i} \in \{0, 1\}$ masks invalid value targets, and α balances the two objectives. The auxiliary value regression anchors the ranking logits to true topological distances, significantly stabilizing the representation space. During online inference, where topology-derived labels are unobservable, SeVN autonomously drives the exploration policy: the agent greedily executes the candidate with the highest predicted logit, $c_t^* = \arg\max_{c_i \in \mathcal{C}_t} \ell_{t,i}$. After execution, the semantic state is dynamically updated and the candidate set is regenerated. This forms a robust, closed-loop search process that systematically navigates the unknown environment until a demand-relevant target enters the verification stage.

4.3 Verified Target Acquisition

A demand-relevant detection does not necessarily indicate successful target acquisition: the object may be visually unstable, momentarily occluded, physically unreachable, or simply an incorrect instance. To prevent premature termination, DDSeVA verifies each proposal progressively through a Temporal Target Lock, a Distance Head, and a Done Success Head.

Temporal Target Lock. Because a single-frame detection is highly vulnerable to visual noise and viewpoint shifts, DDSeVA does not use a single-frame detection as a direct termination signal. Instead, the Temporal Target Lock tracks the proposal across consecutive frames to verify its persistence. Let $(z_t^{\text{tar}}, \tilde{\mathbf{b}}_t, \gamma_t)$ denote the category, bounding box, and confidence score of a target proposal at step t. By aggregating the proposal's visibility, bounding-box consistency, and approach history, this module distills fluctuating observations into a robust state vector $\boldsymbol{\lambda}_t$. This state ensures the agent only commits to temporally stable hypotheses, and provides a memory mechanism for local recovery if the target is briefly occluded during the approach.

Distance Head. Once a target is temporally locked, the agent must determine if it is physically actionable. This module estimates the metric distance to the target and classifies its reachability. Given the RGB observation $\mathbf{o}_t$, proposal box $\tilde{\mathbf{b}}_t$, and target category z_t^{tar}, a shared ResNet-18 encoder E extracts visual features from both the full image and a padded target crop:

$$\mathbf{r}_t = \left[E(\mathbf{o}_t) \| E\Big(\text{crop}(\mathbf{o}_t, \tilde{\mathbf{b}}_t)\Big) \| g(\tilde{\mathbf{b}}_t) \| e(z_t^{\text{tar}}) \right], \tag{7}$$

where $g(\tilde{\mathbf{b}}_t)$ is the normalized box geometry and $e(z_t^{\text{tar}})$ is the category embedding. A shared multilayer perceptron processes $\mathbf{r}_t$ into a hidden representation, which is then fed into two separate branches to predict the distance $\hat{d}_t$ and reachability probability ρ_t^{reach}:

$$\hat{d}_t = \exp\big(f_\phi^{\text{dist}}(\mathbf{r}_t)\big) - 1, \qquad \rho_t^{\text{reach}} = \sigma\big(f_\phi^{\text{reach}}(\mathbf{r}_t)\big), \tag{8}$$

where f_ϕ^{dist} and f_ϕ^{reach} denote the branch networks parameterized by ϕ, and $\sigma(\cdot)$ is the sigmoid function.

Done Success Head. Even if a target is stable and reachable, stopping might still fail due to poor viewpoints or bounding-box inaccuracies. To make the final decision, this module predicts whether the current proposal should trigger the `Done` action. It concatenates the visual representation $\mathbf{r}_t$ with a context vector $\mathbf{u}_t$ (which includes the detection confidence, bounding box status, Distance Head outputs, and the target-lock state $\boldsymbol{\lambda}_t$) to form a fused feature $\mathbf{w}_t = [\mathbf{r}_t \| \mathbf{u}_t]$. From this fused feature, two subsequent branches estimate the termination success and risk:

$$\rho_t^{\text{succ}} = \sigma\big(f_\psi^{\text{succ}}(\mathbf{w}_t)\big), \qquad \rho_t^{\text{risk}} = \sigma\big(f_\psi^{\text{risk}}(\mathbf{w}_t)\big), \tag{9}$$

where ρ_t^{succ} measures the likelihood that executing `Done` will satisfy the task, ρ_t^{risk} captures failure risks (e.g., incorrect instances, poor viewpoints, truncated boxes), and ψ denotes the parameters of the Done Success Head.

The policy executes `Done` only when the proposal is temporally stable, spatially reachable, and highly likely to satisfy the task with low termination risk. Crucially, this verification stage is not a terminal dead-end. If the risk-aware gate rejects a proposal due to factors such as persistent unreachability or high termination risk, the target hypothesis is discarded and the semantic state $\mathcal{B}_t$ is updated with this negative evidence. The agent then seamlessly falls back to the semantic value search (Section 4.2) to evaluate alternative candidates, effectively closing the decision-making loop. Training losses, risk-aware gating, and recovery details are provided in Appendices C.4 and C.5.

Table 1: Main results on the four official evaluation splits. All values are percentages, and the best results are shown in bold.

Method	Seen Scene						Unseen Scene					
	Seen Instruction			Unseen Instruction			Seen Instruction			Unseen Instruction		
	NSR	SPL	SSR	NSR	SPL	SSR	NSR	SPL	SSR	NSR	SPL	SSR
Random	5.2	2.6	3	3.7	2.6	2.3	4.8	3.3	2.8	3.5	1.9	1.4
VTN-demand	6.3	4.2	3.2	5.2	3.1	2.8	5.0	3.2	2.8	6.6	4.0	3.3
VTN-CLIP-demand	12.0	5.1	5.7	10.7	3.5	5.0	10.0	3.6	4.0	9.3	3.9	4.0
VTN-GPT*	1.6	0.5	0	1.4	0.4	0.5	1.3	0.2	0.3	0.9	0.4	0.5
ZSON-demand	4.2	2.7	1.9	4.6	3.1	2.0	4.1	2.9	1.2	3.5	2.4	1.1
ZSON-GPT	4.0	1.1	0.3	3.6	1.9	0.3	2.5	0.7	0.2	3.2	0.9	0.2
CLIP-Nav-MiniGPT-4	4.0	4.0	2.0	3.0	3.0	2.0	4.0	3.7	2.0	5.0	5.0	3.0
CLIP-Nav-GPT*	5.0	5.0	4.0	6.0	5.5	5.0	5.5	5.3	4.0	4.0	3.0	2.0
FBE-MiniGPT-4	3.5	3.0	2.2	3.5	3.2	2.0	3.5	3.5	2.0	4.0	4.0	3.5
FBE-GPT*	5.0	4.3	4.3	5.5	5.0	5.5	4.5	4.3	4.5	5.5	5.0	5.5
GPT-3-Prompt*	0.3	0.01	0	0.3	0.01	0	0.3	0.01	0	0.3	0.01	0
MiniGPT-4	2.9	2.0	2.5	2.9	2.0	2.5	2.9	2.0	2.5	2.9	2.0	2.5
DDN	21.5	9.8	7.5	19.3	9.4	4.5	14.2	6.4	5.7	16.1	8.4	6.0
CogDDN	38.3	17.2	29.8	37.5	18.0	28.6	33.3	16.4	27.1	34.5	17.1	27.5
DDSeVA	**42.4**	**21.5**	**39.8**	**44.2**	**21.4**	**41.2**	**37.0**	**18.6**	**34.9**	**35.1**	**17.2**	**31.7**

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

Datasets & Protocol. We evaluate DDSeVA under the official DDN protocol (Wang et al., 2023) in AI2-THOR (Kolve et al., 2017) using scenes from the ProcTHOR dataset (Deitke et al., 2022). Evaluation is conducted on four scene-instruction splits: `seen scene / seen instruction`, `seen scene / unseen instruction`, `unseen scene / seen instruction`, and `unseen scene / unseen instruction`. Each split contains 500 episodes with a maximum length of 100 steps. For DDSeVA and its ablations, we reuse fixed episodes with the same scenes, instructions, and initial poses. See Appendix D for more.

Evaluation Metrics. We report Navigation Success Rate (NSR), Success weighted by Path Length (SPL) (Anderson et al., 2018a), and Selection Success Rate (SSR), following the success criteria of the DDN benchmark (Wang et al., 2023). NSR measures whether the agent executes `Done` with a demand-satisfying object visible within $1.5\,\mathrm{m}$, while SPL further accounts for path efficiency. SSR measures whether the predicted target box matches a visible demand-satisfying instance with IoU at least 0.5 at termination. Higher values are better for all metrics. Formal definitions are provided in Appendix E.1.

5.2 MAIN RESULTS

To comprehensively evaluate DDSeVA, we compare it with three baseline groups: chance-level non-learning methods like Random; general navigation policies adapted to DDN, including VTN (Du et al., 2021), ZSON (Majumdar et al., 2022), CLIP-Nav (Dorbala et al., 2022), FBE, and MiniGPT-4 (Zhu et al., 2024); and state-of-the-art DDN specialists, specifically DDN (Wang et al., 2023) and CogDDN (Huang et al., 2025). We treat the DDN specialists as our primary baselines, using the remaining methods as broader references. Detailed descriptions are provided in Appendix B.

Table 1 reports NSR, SPL, and SSR on the four evaluation splits. DDSeVA achieves the best performance on all splits and across all three metrics, showing consistent gains in navigation success, path efficiency, and target-instance selection. Compared with the strongest baseline, CogDDN, DDSeVA improves the average NSR from 35.9 to 39.7, the average SPL from 17.2 to 19.7, and the average SSR from 28.3 to 36.9. The largest gain appears in SSR, with an absolute improvement of 8.6 percentage points. These comprehensive improvements stem directly from the coupled design of DDSeVA. Specifically, the semantic value search systematically guides the agent toward high-potential regions to enhance path efficiency, while the progressive verification pipeline rigor-

ously prevents premature stops on spurious visual detections, driving the substantial gains in target selection.

Beyond improving navigation success, DDSeVA also converts successful navigation into correct target selection more reliably. We measure this effect by the ratio between Selection Success Rate and Navigation Success Rate, which reflects whether an agent that reaches a successful stopping region can also select the correct demand-satisfying object. As shown in Figure 3, DDSeVA maintains a conversion ratio above 90% across all four scene-instruction splits, while CogDDN remains around 76–81% and DDN is much lower. This result directly supports the motivation of DDSeVA: closing the detection-to-success gap requires more than finding a demand-relevant region. This is crucial because real-world perception is inherently noisy and targets are often physically unreachable. By rigorously verifying each proposal before executing `Done`, the agent can reliably translate ambiguous visual evidence into a correct termination.

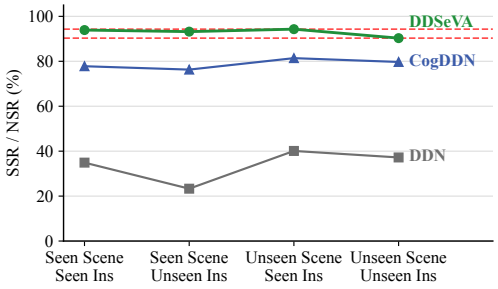


Figure 3: SSR-to-NSR conversion across scene-instruction splits.

5.3 Ablation Study

To systematically validate our architectural design, Table 2 evaluates five variants across the four scene-instruction splits, demonstrating how semantic search and progressive verification jointly close the detection-to-success gap.

The Necessity of Semantic Value Search. The **w/o SeVN** variant replaces the learned candidate valuation with a `frontier_nearest` heuristic. This drastically degrades search efficiency, dropping SPL from 21.5 to 13.6 in the Seen/Seen setting. Such a decline confirms that purely geometric exploration is sub-optimal; the agent must leverage semantic topology to prioritize task-relevant regions over mere physical proximity.

The Critical Role of Verified Acquisition. The **w/o Verified Target Acquisition** variant naively triggers `Done` from single-frame YOLO detections. This causes a catastrophic NSR collapse, plummeting from 42.4 to 14.8 in Seen/Seen and 35.1 to 15.4 in Unseen/Unseen. This failure confirms that merely finding a relevant region is insufficient; explicit verification is mandatory to handle visual false positives, temporary occlusions, and physical unreachability.

Deconstructing the Verification Pipeline. We further isolate the three stages of verified acquisition. **w/o Temporal Target Lock:** Removing the tracker makes the agent vulnerable to visual noise, halving SPL from 21.5 to 10.3 in Seen/Seen because the agent repeatedly loses targets and hesitates. **w/o Distance Head:** Replacing learned 3D reachability with a 2D bounding-box proxy drops NSR by over 11% absolute in unseen scenes, confirming 2D geometry is a flawed proxy for actionable distance. **w/o Done Success Head:** Omitting risk-aware stopping degrades NSR from 35.1 to 32.9 in Unseen/Unseen, showing that even when spatially ready, the agent must semantically assess termination risk to avoid premature stops. Collectively, these components form a progressive filter, ensuring raw visual hypotheses are rigorously transformed into reliable stopping states.

Table 2: Ablation results of DDSeVA under four scene-instruction settings.

Method	Seen Scene						Unseen Scene					
	Seen Ins			Unseen Ins			Seen Ins			Unseen Ins		
	NSR	SPL	SSR	NSR	SPL	SSR	NSR	SPL	SSR	NSR	SPL	SSR
DDSeVA	**42.4**	**21.5**	**39.8**	**44.2**	**21.4**	**41.2**	**37.0**	**18.6**	**34.9**	**35.1**	**17.2**	**31.7**
w/o SeVN	35.7	13.6	33.1	35.0	14.6	32.6	28.7	11.9	27.5	24.5	11.0	22.1
w/o Verified Target Acquisition	14.8	11.3	11.8	16.8	12.6	13.2	12.8	9.1	11.6	15.4	11.8	12.0
w/o Temporal Target Lock	32.7	10.3	31.7	33.9	10.6	30.7	28.3	9.7	25.5	26.1	10.4	23.5
w/o Distance Head	31.2	13.7	28.0	31.7	11.4	27.7	27.1	10.2	24.2	23.4	9.4	19.2
w/o Done Success Head	40.1	18.2	37.3	41.2	17.8	37.6	36.2	17.4	34.1	32.9	16.5	29.3

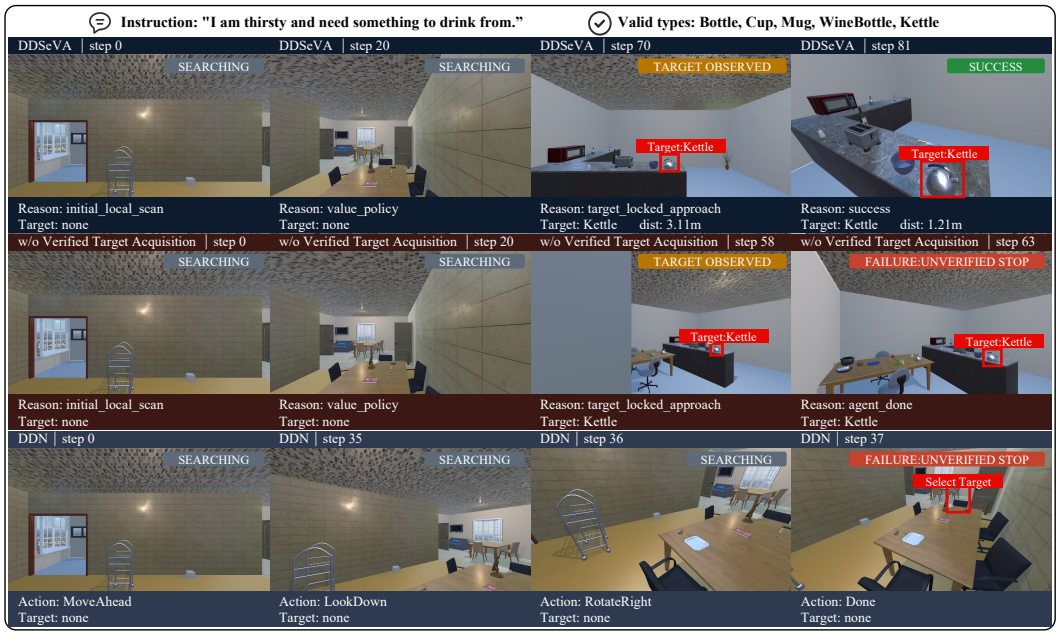


Figure 4: Qualitative target acquisition comparison. DDSeVA successfully approaches and verifies the target before executing `Done`. The unverified variant stops prematurely with an inaccurate bounding box. DDN terminates incorrectly on an unrelated object.

5.4 Qualitative Analysis

Figure 4 illustrates a qualitative comparison driven by the implicit instruction "I am thirsty and need something to drink from." **Full DDSeVA** correctly maps this abstract demand to valid target objects like a kettle. At step 70, it visually detects a kettle on the far kitchen counter from a distance of $3.11\,\mathrm{m}$. Instead of blindly trusting this initial distant detection, DDSeVA engages its verification pipeline, maintaining a temporal lock and physically approaching the target. It executes the final `Done` action only at step 81 when the distance closes to $1.21\,\mathrm{m}$, yielding a precise bounding box and a highly successful episode.

The ablation and baseline behaviors visually highlight the detection-to-success gap. The **w/o Verified Target Acquisition** variant detects the identical kettle at step 70 but triggers a premature `Done` action from across the room, resulting in an unverified stop with a tiny and unusable bounding box. Meanwhile, the baseline **DDN** terminates prematurely at step 37 without identifying a valid target, incorrectly selecting an unrelated object instead. This comparison confirms that visual target discovery is strictly necessary but highly insufficient; rigorous spatial and semantic verification is what ultimately transforms a raw detection into a reliable task completion.

6 Conclusion

We introduced DDSeVA to address the fundamental gap between detecting a demand-relevant object and successfully completing a navigation task. The framework couples demand-conditioned semantic value search with progressive target verification. During exploration, the search module ranks heterogeneous candidates based on their expected contribution to eventual success. Once a potential target is observed, the acquisition module rigorously verifies its temporal stability, spatial reachability, and termination risk before authorizing the `Done` action. Experiments in the AI2-THOR simulator demonstrate that DDSeVA significantly outperforms existing baselines across all scene and instruction settings. While currently evaluated in simulation, this verified closed-loop approach provides a robust foundation for reliable demand-driven navigation in complex environments.

REFERENCES

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

Dong An, Hanqing Wang, Wenguan Wang, Zun Wang, Yan Huang, Keji He, and Liang Wang. Etpnav: Evolving topological planning for vision-language navigation in continuous environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(7):5130–5145, 2024.

Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, et al. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*, 2018a.

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *2018 IEEE/CVF conference on computer vision and pattern recognition*, pp. 3674–3683. IEEE, 2018b.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pp. 213–229. Springer, 2020.

Devendra Singh Chaplot, Dhiraj Prakashchand Gandhi, Abhinav Gupta, and Russ R Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. *Advances in neural information processing systems*, 33:4247–4258, 2020.

Jiaqi Chen, Bingqian Lin, Ran Xu, Zhenhua Chai, Xiaodan Liang, and Kwan-Yee Wong. Mapgpt: Map-guided prompting with adaptive path planning for vision-and-language navigation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9796–9810, 2024.

Shizhe Chen, Pierre-Louis Guhur, Cordelia Schmid, and Ivan Laptev. History aware multimodal transformer for vision-and-language navigation. *Advances in neural information processing systems*, 34:5834–5847, 2021.

Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Cordelia Schmid, and Ivan Laptev. Think global, act local: Dual-scale graph transformer for vision-and-language navigation. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16516–16526. IEEE, 2022.

Matt Deitke, Eli VanderBilt, Alvaro Herrasti, Luca Weihs, Kiana Ehsani, Jordi Salvador, Winson Han, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. Procthor: Large-scale embodied ai using procedural generation. *Advances in neural information processing systems*, 35:5982–5994, 2022.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pp. 4171–4186, 2019.

Vishnu Sashank Dorbala, Gunnar Sigurdsson, Robinson Piramuthu, Jesse Thomason, and Gaurav S Sukhatme. Clip-nav: Using clip for zero-shot vision-and-language navigation. *arXiv preprint arXiv:2211.16649*, 2022.

Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

Heming Du, Xin Yu, and Liang Zheng. Vtnet: Visual transformer network for object goal navigation. *arXiv preprint arXiv:2105.09447*, 2021.

Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. Speaker-follower models for vision-and-language navigation. *Advances in neural information processing systems*, 31, 2018.

Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448, 2015.

Weituo Hao, Chunyuan Li, Xiujun Li, Lawrence Carin, and Jianfeng Gao. Towards learning a generic agent for vision-and-language navigation via pre-training. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13134–13143. IEEE, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. Vln bert: A recurrent vision-and-language bert for navigation. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pp. 1643–1653, 2021.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Yuehao Huang, Liang Liu, Shuangming Lei, Yukai Ma, Hao Su, Jianbiao Mei, Pengxiang Zhao, Yaqing Gu, Yong Liu, and Jiajun Lv. Cogddn: A cognitive demand-driven navigation with decision optimization and dual-process thinking. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 5237–5246, 2025.

Glenn Jocher, Jing Qiu, Mengyu Liu, Shuai Lyu, Fatih Cagatay Akyon, and Muhammet Esat Kalfaoglu. Ultralytics yolo26: unified real-time end-to-end vision models. *arXiv preprint arXiv:2606.03748*, 2026.

Liyiming Ke, Xiujun Li, Yonatan Bisk, Ari Holtzman, Zhe Gan, Jingjing Liu, Jianfeng Gao, Yejin Choi, and Siddhartha Srinivasa. Tactical rewind: Self-correction via backtracking in vision-and-language navigation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6734–6742. IEEE, 2019.

DA Sasi Kiran, Kritika Anand, Chaitanya Kharyal, Gulshan Kumar, Nandiraju Gireesh, Snehasis Banerjee, Ruddra Dev Roychoudhury, Mohan Sridharan, Brojeshwar Bhowmick, and Madhava Krishna. Spatial relation graph and graph convolutional network for object goal navigation. In *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*, pp. 1392–1398. IEEE, 2022.

Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, et al. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017.

Shanshan Li, Da Huang, Yu He, Yanwei Fu, Yu-Gang Jiang, and Xiangyang Xue. Tp-mddn: Task-preferenced multi-demand-driven navigation with autonomous decision-making. *Advances in Neural Information Processing Systems*, 38:84135–84154, 2025.

Yuxing Long, Wenzhe Cai, Hongcheng Wang, Guanqi Zhan, and Hao Dong. Instructnav: Zero-shot system for generic instruction navigation in unexplored environment. *arXiv preprint arXiv:2406.04882*, 2024a.

Yuxing Long, Xiaoqi Li, Wenzhe Cai, and Hao Dong. Discuss before moving: Visual language navigation via multi-expert discussions. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 17380–17387. IEEE, 2024b.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Chih-Yao Ma, Jiasen Lu, Zuxuan Wu, Ghassan AlRegib, Zsolt Kira, Richard Socher, and Caiming Xiong. Self-monitoring navigation agent via auxiliary progress estimation. *arXiv preprint arXiv:1901.03035*, 2019a.

Chih-Yao Ma, Zuxuan Wu, Ghassan AlRegib, Caiming Xiong, and Zsolt Kira. The regretful agent: Heuristic-aided navigation through progress estimation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6725–6733. IEEE, 2019b.

Arjun Majumdar, Gunjan Aggarwal, Bhavika Devnani, Judy Hoffman, and Dhruv Batra. Zson: Zero-shot object-goal navigation using multimodal goal embeddings. *Advances in neural information processing systems*, 35:32340–32352, 2022.

Anwesan Pal, Yiding Qiu, and Henrik Christensen. Learning hierarchical relationships for object-goal navigation. In *Conference on Robot Learning*, pp. 517–528. PMLR, 2021.

Alexander Pashevich, Cordelia Schmid, and Chen Sun. Episodic transformer for vision-and-language navigation. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15922–15932. IEEE, 2021.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Santhosh Kumar Ramakrishnan, Devendra Singh Chaplot, Ziad Al-Halah, Jitendra Malik, and Kristen Grauman. Poni: Potential functions for objectgoal navigation with interaction-free learning. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18868–18878. IEEE, 2022.

Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Foundations and trends® in information retrieval*, 4(1-2):1–174, 2009.

Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 761–769, 2016.

Hao Tan, Licheng Yu, and Mohit Bansal. Learning to navigate unseen environments: Back translation with environmental dropout. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2610–2621, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Hongcheng Wang, Andy Guan Hong Chen, Xiaoqi Li, Mingdong Wu, and Hao Dong. Find what you want: Learning demand-conditioned object attribute space for demand-driven navigation. *Advances in Neural Information Processing Systems*, 36:16353–16366, 2023.

Hongcheng Wang, Peiqi Liu, Wenzhe Cai, Mingdong Wu, Zhengyu Qian, and Hao Dong. Moddn: a coarse-to-fine attribute-based exploration agent for multi-object demand-driven navigation. *Advances in Neural Information Processing Systems*, 37:64176–64214, 2024.

Xin Wang, Wenhan Xiong, Hongmin Wang, and William Yang Wang. Look before you leap: Bridging model-free and model-based reinforcement learning for planned-ahead vision-and-language navigation. In *European Conference on Computer Vision*, pp. 38–55. Springer, 2018.

Brian Yamauchi. A frontier-based approach for autonomous exploration. In *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97.'Towards New Computational Principles for Robotics and Automation'*, pp. 146–151. IEEE, 1997.

Wei Yang, Xiaolong Wang, Ali Farhadi, Abhinav Gupta, and Roozbeh Mottaghi. Visual semantic navigation using scene priors. *arXiv preprint arXiv:1810.06543*, 2018.

Hang Yin, Xiuwei Xu, Zhenyu Wu, Jie Zhou, and Jiwen Lu. Sg-nav: Online 3d scene graph prompting for llm-based zero-shot object navigation. *Advances in neural information processing systems*, 37:5285–5307, 2024.

Naoki Yokoyama, Sehoon Ha, Dhruv Batra, Jiuguang Wang, and Bernadette Bucher. Vlfm: Vision-language frontier maps for zero-shot semantic navigation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 42–48. IEEE, 2024.

Jiazhao Zhang, Kunyu Wang, Rongtao Xu, Gengze Zhou, Yicong Hong, Xiaomeng Fang, Qi Wu, Zhizheng Zhang, and He Wang. Navid: Video-based vlm plans the next step for vision-and-language navigation. *arXiv preprint arXiv:2402.15852*, 2024.

Lingfeng Zhang, Xiaoshuai Hao, Qinwen Xu, Qiang Zhang, Xinyao Zhang, Pengwei Wang, Jing Zhang, Zhongyuan Wang, Shanghang Zhang, and Renjing Xu. Mapnav: A novel memory representation via annotated semantic maps for vlm-based vision-and-language navigation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13032–13056, 2025.

Qianfan Zhao, Lu Zhang, Bin He, Hong Qiao, and Zhiyong Liu. Zero-shot object goal visual navigation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2025–2031. IEEE, 2023.

Gengze Zhou, Yicong Hong, Zun Wang, Xin Eric Wang, and Qi Wu. Navgpt-2: Unleashing navigational reasoning capability for large vision-language models. In *European Conference on Computer Vision*, pp. 260–278. Springer, 2024a.

Gengze Zhou, Yicong Hong, and Qi Wu. Navgpt: Explicit reasoning in vision-and-language navigation with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 7641–7649, 2024b.

Kaiwen Zhou, Kaizhi Zheng, Connor Pryor, Yilin Shen, Hongxia Jin, Lise Getoor, and Xin Eric Wang. Esc: Exploration with soft commonsense constraints for zero-shot object navigation. In *International conference on machine learning*, pp. 42829–42842. PMLR, 2023.

Deyao Zhu, Xiaoqian Shen, Xiang Li, Mohamed Elhoseiny, et al. Minigpt-4: Enhancing vision-language understanding with advanced large language models. In *International Conference on Learning Representations*, volume 2024, pp. 18378–18394, 2024.

APPENDIX

A LIST OF SYMBOLS

For compactness, Table 3 lists only the symbols that carry semantic or modeling meaning in displayed equations. Standard operators, index letters, module names, and purely textual notation are omitted.

Table 3: Notation and description of key symbols.

Notation	Description	Index
$\tau, d, \omega, \mathcal{I}_\tau^{\mathrm{vis}}$	Termination step, demand, instance, and visible instances	Eq. (1)
$\mathcal{Y}_d, y(\omega), \mathbf{b}(\omega), \mathbf{p}(\omega)$	Valid categories; instance category, box, and position	Eq. (1)
$\hat{\mathbf{b}}_\tau, \mathbf{x}_\tau, d_{\max}, \theta_{\mathrm{iou}}$	Predicted box, final position, and distance/IoU thresholds	Eq. (1)
$\mathcal{Q}(d), \mathcal{O}_t, \mathcal{T}_t$	Demand profile, detections, and target hypotheses	Eq. (2)
$z_i, \mathbf{b}_i, \gamma_i$	Predicted class, box, and confidence	Eq. (2)
$\mathcal{B}_t, \mathcal{N}_t, \mathcal{R}_t, \mathcal{M}_t^{\mathrm{obj}}$	Semantic state and node, room, and object memories	Eq. (3)
$\mathbf{h}_{t,i}, \ell_{t,i}, \hat{v}_{t,i}$	Candidate representation, ranking logit, and predicted value	Eq. (4)
$f_\theta^{\mathrm{rank}}, f_\theta^{\mathrm{value}}$	Ranking and value prediction heads	Eq. (4)
$i_t^+, \mathcal{C}_t^{\mathrm{valid}}, c_i$	Teacher index, valid candidate set, and candidate	Eq. (5)
D_t, D_i, κ_i	Current success distance, candidate success distance, and path cost	Eq. (5)
$\mathcal{L}_{\mathrm{search}}, \alpha, m_{t,i}, v_i$	Search loss, value-loss weight, target mask, and target value	Eq. (6)
$\mathbf{r}_t, \mathbf{o}_t, \tilde{\mathbf{b}}_t, z_t^{\mathrm{tar}}$	Proposal feature, observation, target box, and target class	Eq. (7)
E, g, e	Visual encoder, box geometry, and class embedding	Eq. (7)
$\hat{d}_t, \rho_t^{\mathrm{reach}}$	Predicted distance and reachability probability	Eq. (8)
$f_\phi^{\mathrm{dist}}, f_\phi^{\mathrm{reach}}, \phi$	Distance/reachability heads and their parameters	Eq. (8)
$\mathbf{u}_t, \mathbf{w}_t, \rho_t^{\mathrm{succ}}, \rho_t^{\mathrm{risk}}$	Context, fused feature, success probability, and risk probability	Eq. (9)
$f_\psi^{\mathrm{succ}}, f_\psi^{\mathrm{risk}}, \psi$	Success/risk heads and their parameters	Eq. (9)
$\mathcal{L}_{\mathrm{dist}}, \delta_t, y_t^{\mathrm{dist}}, y_t^{\mathrm{reach}}, \lambda_d, \lambda_r$	Distance loss, prediction, labels, and loss weights	Eq. (10)
$\mathcal{L}_{\mathrm{done}}, y_t^{\mathrm{succ}}, y_t^{\mathrm{risk}}, \lambda_s, \lambda_{\mathrm{risk}}$	Done loss, success/risk labels, and loss weights	Eq. (11)
$G_t^{\mathrm{done}}, A_t^{\mathrm{lock}}, A_t^{\mathrm{reach}}, \eta_p, \eta_l, \eta_r$	Done gate, lock/reachability signals, and decision thresholds	Eq. (12)
$N, S_i^{\mathrm{nav}}, \tau_i$	Episode count, navigation-success indicator, and termination step	Eqs. (13)–(14)
$P_i, P_i^\star$	Executed path length and shortest feasible path length	Eq. (15)
$S_i^{\mathrm{sel}}, \hat{\mathbf{b}}_{\tau_i}$	Selection-success indicator and predicted terminal box	Eq. (16)

B BASELINE DETAILS

This appendix describes the baseline configurations used in Table 1.

Random. The Random baseline selects navigation actions without using demand semantics or learned scene representations and provides a chance-level reference.

VTN-demand. The VTN baseline (Du et al., 2021) retains the original VTN navigation pipeline while replacing the target-object vector with a BERT feature (Devlin et al., 2019) of the natural-language demand. The remaining navigation pipeline is unchanged.

VTN-CLIP-demand. Starting from `VTN-demand`, this variant replaces the DETR visual features (Carion et al., 2020) with CLIP visual features (Radford et al., 2021). All other VTN components remain unchanged.

VTN-GPT*. This variant uses the best `VTN-object` checkpoint on the validation set, where `VTN-object` expands the target vocabulary to the 109 DDN object categories and is trained with pre-imitation learning followed by PPO fine-tuning. At the beginning of each episode, GPT-3 (Brown et al., 2020) generates a language-grounded object from the demand. We encode the 109 category names and the generated object with the CLIP text encoder and select the category with the highest cosine similarity to the generated object. The selected category is then provided to the VTN object-goal model. If `Done` fails before the step limit, GPT-3 generates another object and the procedure is repeated.

ZSON-demand. The ZSON baseline (Majumdar et al., 2022) uses its original zero-shot object-goal navigation pipeline with CLIP-based goal embeddings. We select the best `ZSON-object` checkpoint on the validation set, where `ZSON-object` is trained with target images rendered around objects within the DDN navigation distance threshold. For `ZSON-demand`, the category target embedding is replaced with the CLIP text feature of the full demand instruction. The remaining ZSON pipeline is unchanged.

ZSON-GPT. Using the same `ZSON-object` base model, GPT-3 generates a language-grounded object from the demand at the beginning of each episode. The generated object is encoded with the CLIP text encoder and used as the ZSON target. When `Done` fails and steps remain, a new object is generated and the target-selection process is repeated.

CLIP-Nav-MiniGPT-4. The CLIP-Nav baseline (Dorbala et al., 2022) uses CLIP-based visual localization and navigation. Because DDN instructions are high-level demands rather than step-by-step routes, GPT-3 or MiniGPT-4 is queried at each step only to determine whether the current observation contains an object satisfying the demand. In this variant, the query model is MiniGPT-4.

CLIP-Nav-GPT*. This variant uses the same CLIP-Nav navigation component but queries GPT-3 at each step to determine whether the current observation contains an object satisfying the demand.

FBE-MiniGPT-4. The frontier-based exploration baseline constructs occupancy and exploration maps and selects the nearest unexplored node at each step (Yamauchi, 1997). MiniGPT-4 determines whether an object visible in the current observation satisfies the demand.

FBE-GPT*. This variant uses the same frontier-based exploration policy and queries GPT-3 for demand satisfaction at each step.

GPT-3-Prompt*. This baseline directly prompts GPT-3 to select the next action from the DDN action space. The prompt contains the task success criterion, the available actions, the demand, the visible object list with bounding boxes and distances, and the action history. When selecting `Done`, the model is additionally asked to output the selected object.

MiniGPT-4. MiniGPT-4 (Zhu et al., 2024) receives the current RGB image, the demand instruction, the visible object list, and the previous actions. It is prompted to select one action from the DDN action space and to identify the target object when selecting `Done`. Unlike the adapted navigation baselines, this configuration does not use a separately trained navigation policy.

DDN. DDN (Wang et al., 2023) learns a demand-conditioned object attribute space that maps a natural-language demand to potential object categories and uses the resulting semantic representation for navigation. We use DDN as the primary demand-object grounding baseline without an explicit success-verification stage.

CogDDN. CogDDN (Huang et al., 2025) augments demand matching with vision-language reasoning, dual-process decision making, and reflective knowledge accumulation. Its heuristic process handles exploration and target approach, while its analytic process uses visual reasoning and accumulated knowledge to revise decisions. We use CogDDN as the most directly related cognitive decision-making baseline.

C Additional Method Details

The main text presents DDSeVA at the level of its decision abstraction: demand-grounded state construction, semantic value search, and verified target acquisition. This appendix follows the same order as Sections 4.1–4.3 and supplies the implementation details omitted for space. The goal is to make the closed-loop system reproducible without changing the claims or decision flow described in the main paper.

C.1 Demand and Perception Adapters

DDSeVA instantiates its semantic state (Section 4.1) using two adapted perception modules. First, a Qwen-based demand parser translates the natural-language instruction into a discrete set of target object classes. Second, a YOLO-based detector extracts per-frame visual evidence from RGB observations. To strictly prevent data leakage, both modules are trained exclusively on seen scenes and

seen instructions; no unseen evaluation episodes are used for adaptation. Detailed data construction and training setups are provided in Appendix D.1.

Demand-Profile Cache. To avoid the computational latency of repeated large-model queries, Qwen is invoked only once before navigation begins. Its JSON output is parsed, canonicalized against the simulator's `objectType` vocabulary, and stored as the demand profile $\mathcal{Q}(d)$. For robustness against unseen instructions, this profile optionally merges Qwen predictions with nearest-neighbor evidence from a BM25 retrieval library (Robertson & Zaragoza, 2009). Regardless of this retrieval augmentation, $\mathcal{Q}(d)$ functions as a fixed set of target categories loaded at the start of the episode.

Online Detector Interface. During active navigation, the YOLO adapter processes each RGB observation to extract object classes, bounding boxes, confidence scores, and box-area statistics. DDSeVA applies a dual-threshold strategy: a standard confidence threshold of 0.10 for general contextual objects, and a relaxed threshold of 0.05 for categories matching the demand profile $\mathcal{Q}(d)$. This asymmetric thresholding intentionally favors recall for potential targets. Any resulting false positives are systematically filtered later by the verified target acquisition modules (i.e., temporal locking, distance estimation, and `Done` success verification).

C.2 Semantic State and Candidate Construction

Memory Update. The online semantic state $\mathcal{B}_t$ is continuously updated after each observation. It comprises three structured components: (1) *Node memory* logs local pose information, visitation counts, geometric features (e.g., yaw coverage, doorway evidence, frontier status), and object-type histograms. (2) *Room memory* aggregates these nodes into higher-level representations, tracking room types, structural layouts, and regional object distributions. (3) *Object memory* records detection metadata (e.g., category, confidence, bounding boxes, observation recency) and demand relevance. Crucially, to disambiguate visually identical objects across different areas, object records are spatially anchored to their observed nodes and rooms rather than grouped merely by category.

Candidate Schema. DDSeVA structures each search alternative as an executable candidate c_i. To avoid redundant mathematical tuples, each candidate is simply parameterized by its semantic type, anchor node n_i, path cost κ_i, and type-specific metadata. The semantic generator produces five candidate categories: frontiers for exploring unknown space, room exits for topological transfer, revisitation nodes for area sweeps, object regions for contextual search, and target hypotheses for verified acquisition. During physical execution, these semantic candidates map to concrete navigation subgoals (e.g., moving through a portal or approaching a target). However, this physical mapping does not alter the SeVN ranking process, which operates strictly on the abstract candidate parameters.

Candidate Validity Constraints. A candidate is deemed valid only if its anchor node is currently reachable with a finite path cost. Furthermore, object and target hypotheses must satisfy temporal recency constraints, and target candidates must strictly align with the cached demand profile $\mathcal{Q}(d)$. During training, invalid candidates are masked in the ranking loss, and those lacking finite topological distances are excluded from the auxiliary value regression.

C.3 SeVN Tokenization and Offline Teacher

Candidate-Centric Tokenization. SeVN converts the semantic state into a heterogeneous token sequence $\mathcal{X}_t$, which comprises demand, node, room, object, and candidate tokens. Instead of relying on a complex hierarchy, each token $\mathbf{x}_j \in \mathcal{X}_t$ is constructed through a straightforward summation of embeddings. Specifically, it combines discrete identity embeddings (e.g., token type, object category, room type) with continuous feature projections (e.g., normalized spatial offsets, graph distances, path costs, and bounding-box geometry). Additionally, a bag-of-words vector is incorporated to capture local object distributions. These fused tokens are then contextualized by the SeVN Transformer described in Section 4.2.

Offline Teacher and Tie-Breaking. The offline teacher derives dense supervision purely from the environment's underlying topology. For each valid candidate c_i, the builder computes its distance D_i to the nearest success node, its soft value $v_i = \exp(-D_i/\tau_v)$, and its expected distance reduction $\Delta_i = D_t - \kappa_i - D_i$. To unambiguously select the optimal ranking target i_t^+, the teacher applies a lexicographic tie-breaking rule that prioritizes candidates based on: (1) maximum distance reduction

Δ_i, (2) minimum remaining distance D_i, (3) highest soft value v_i, and (4) minimum path cost κ_i. A deterministic candidate identifier resolves any remaining exact ties without introducing task-specific bias. Crucially, these topology-derived labels are strictly confined to offline dataset construction and are never exposed to the agent during online inference.

C.4 Training Details for Verified Target Acquisition

Temporal Target Lock. Instead of treating detections independently, the Temporal Target Lock maintains a persistent proposal-level state. It records a comprehensive set of statistics categorized into temporal persistence (e.g., observation counts, lost steps), spatial approach (e.g., first/last nodes, low-distance counts), and visual consistency (e.g., bounding-box area ratios). By accumulating this history, the module filters out spurious single-frame detections, forcing the agent to gather sustained, active evidence before triggering `Done`. To handle transient occlusions, the lock remains active for a bounded window, enabling local recovery based on the last known bearing and node.

Distance Head. The Distance Head is jointly optimized for continuous metric distance estimation and binary reachability classification. As illustrated in Figure 5, the architecture fuses visual features—extracted from both the full RGB observation and the target crop via a shared ResNet-18 backbone—with normalized box geometry and category embeddings. The concatenated feature representation $\mathbf{r}_t$ is then processed by a shared MLP before splitting into two dedicated prediction networks: a distance branch (f_ϕ^{dist}) and a reachability branch (f_ϕ^{reach}). Let δ_t denote the raw log-distance prediction (prior to the exponential transform in Section 4.3), and let y_t^{dist} and $y_t^{\text{reach}} \in \{0, 1\}$ represent the ground-truth metric distance and reachability target, respectively. The training objective combines a Smooth L1 loss (Girshick, 2015) and a Binary Cross-Entropy (BCE) loss:

$$\mathcal{L}_{\text{dist}} = \lambda_d \, \text{SmoothL1}(\delta_t, \log(1 + y_t^{\text{dist}})) + \lambda_r \, \text{BCE}(\rho_t^{\text{reach}}, y_t^{\text{reach}}), \tag{10}$$

where λ_d and λ_r are the corresponding loss weights. The log-space distance regression stabilizes gradient updates at near ranges, while the reachability branch yields the discrete spatial authorization required by the final stopping gate.

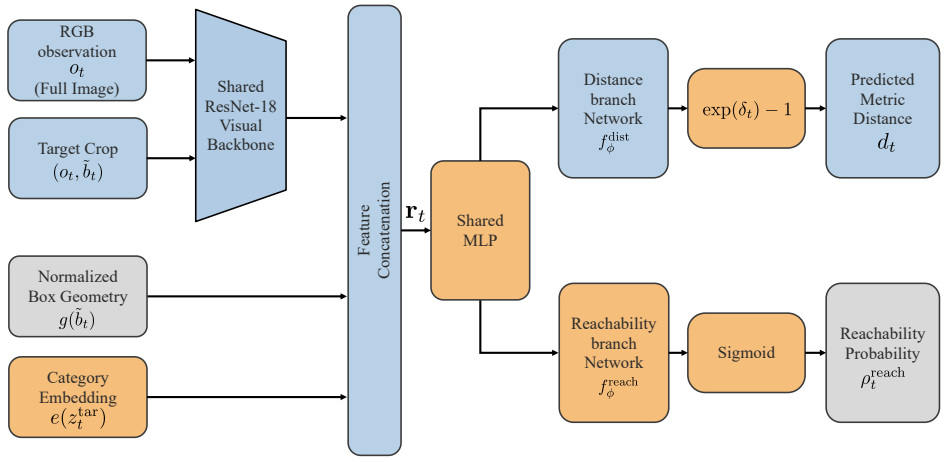


Figure 5: Architecture of the Distance Head. Visual features from the full image and target crop are extracted via a shared ResNet-18 and concatenated with normalized bounding-box geometry and category embeddings. This fused representation $\mathbf{r}_t$ is processed by a shared MLP before branching into log-space metric distance regression and binary reachability classification.

Done Success Head. This head is supervised by whether executing `Done` on the current proposal would objectively satisfy the task's success criteria. It inherits the visual encoding architecture of the Distance Head but expands its input with a rich context vector, incorporating detection confidence, bounding-box integrity, distance estimates, and the Temporal Target Lock statistics. The network is trained with a dual-branch BCE objective to predict both termination success and failure risk:

$$\mathcal{L}_{\text{done}} = \lambda_s \, \text{BCE}(\rho_t^{\text{succ}}, y_t^{\text{succ}}) + \lambda_{\text{risk}} \, \text{BCE}(\rho_t^{\text{risk}}, y_t^{\text{risk}}), \tag{11}$$

where y_t^{succ} is the ground-truth success label, and λ_s and λ_{risk} are the balancing weights. Crucially, the risk target y_t^{risk} emphasizes mined hard-negative proposals—objects that appear visually plausible but would trigger a false stop. By pairing a success estimator with a dedicated risk suppressor, this dual-branch design explicitly bridges the detection-to-success gap.

C.5 Inference Gate and Recovery

Risk-Aware Termination Gate. At inference time, DDSeVA executes `Done` only when multiple verification signals reach a strict consensus. Let A_t^{lock} indicate that the proposal has accumulated sufficient temporal stability and approach evidence, and let A_t^{reach} indicate that the Distance Head (or a size-aware geometric proxy) accepts the target as spatially reachable. The final termination decision is governed by a risk-aware gate:

$$G_t^{\text{done}} = \mathbb{I}\left[A_t^{\text{lock}} \wedge A_t^{\text{reach}} \wedge \left(\rho_t^{\text{succ}} \geq \eta_p \vee \left(\rho_t^{\text{succ}} \geq \eta_l \wedge \rho_t^{\text{risk}} \leq \eta_r\right)\right)\right], \tag{12}$$

where η_p is the primary success threshold, η_l is a relaxed success threshold, and η_r defines the maximum acceptable risk. This logic ensures that while highly confident success predictions can trigger termination directly, marginal (relaxed) success predictions are heavily gated by the risk branch to prevent false stops.

Local Acquisition and Recovery. If a target is visible but fails the termination gate, the agent seamlessly transitions into a local acquisition phase rather than immediately reverting to global search. During this phase, the agent actively approaches the locked target using bearing-consistent viewpoints, or holds its position briefly if no better approach exists. The lock is abandoned only after accumulating bounded evidence of failure, such as persistent unreachability, lack of bounding-box growth, or prolonged target loss. If the target is only temporarily lost, the policy prioritizes short-horizon recovery via local scanning before fully discarding the hypothesis and resuming semantic search. Finally, auxiliary safeguards (e.g., initial sweeping, size-aware thresholds, and specific portal execution rules) act strictly as execution stabilizers. They preserve the integrity of the core pipeline and do not interfere with the learned SeVN supervision or the acquisition-head training objectives.

D Training Data Construction

The previous appendix describes how the trained components are assembled during inference. This section specifies how their training examples and supervision are constructed. Unless stated otherwise, training data are generated from seen-scene training houses, seen-instruction labels, or their paired DDN training episodes. Simulator topology, object metadata, depth, and instance annotations are used only to construct offline labels. These signals are removed from the corresponding model inputs and are unavailable to the online policy.

D.1 Demand Profile and Object Detection Data

Demand profile data. The demand-profile dataset is built from the seen-instruction portion of the DDN instruction bundle. Each record contains a natural-language demand, a shuffled ProcTHOR `objectType` vocabulary, and a strict JSON target whose field `target_object_candidates` contains the detector-recognizable object classes that satisfy the demand. Episode target locations, house states, and scene observations are not included in the Qwen input. We create eight prompt variants per instruction and split the data at the instruction level with validation ratio 0.15 and seed 0, so variants of the same instruction never appear in both splits. Qwen2.5-VL-7B-Instruct is adapted as a text-only parser with LoRA on the attention and MLP projection modules, using rank 16, alpha 32, dropout 0.05, bfloat16 precision, batch size 4, gradient accumulation 4, learning rate 1×10^{-4}, and 8 epochs. After fine-tuning, we generate profiles with temperature 0, maximum length 160, and batch size 4. The outputs are parsed, canonicalized against the allowed object vocabulary, and cached before navigation. For the final DDN-aware cache, Qwen candidates are combined with a BM25 retrieval library built only from seen-instruction labels; the cache keeps at most 16 candidates per demand, uses top-5 retrieved neighbors, and preserves Qwen candidates under the default conflict policy.

Object detection data. The detector dataset is constructed from seen-scene training houses rendered in AI2-THOR/ProcTHOR. Raw frames are sampled from reachable poses with camera yaws in

$\{0°, 90°, 180°, 270°\}$ and horizons in $\{-30°, 0°, 30°\}$. Visible objects are extracted from simulator instance boxes and metadata, filtered by the detector vocabulary, and converted to the normalized YOLO format (c_x, c_y, w, h) with object-type labels. The default raw grounding set uses 100 training houses with 200 frames per house and 20 validation houses with 100 frames per house. To improve recall for small and long-tail objects, we apply frame-level oversampling to underrepresented classes rather than changing their box annotations. Classes with at most 250 original training instances are oversampled toward approximately 600 instances, with at most eight duplicate copies of a frame. The detector is initialized from YOLO26 and trained for 150 epochs at image size 640, batch size 64, seed 0, AdamW optimizer (Loshchilov & Hutter, 2017), initial learning rate 1×10^{-4}, final learning-rate ratio 0.01, and mosaic closing in the last 10 epochs. The detector labels specify object identity and localization only. Whether a detection is relevant to the current demand is determined online by the cached demand profile and is subsequently verified by the acquisition module.

D.2 Semantic Value Search Data

Each Semantic Value Network (SeVN) example is an episode state containing the current semantic map, the candidate set, and topology-derived supervision. We generate episodes in AI2-THOR/ProcTHOR by sampling DDN-compatible house–instruction pairs from the training split. At each sampled state, the semantic map is updated, frontiers, room exits, revisitation nodes, object regions, and target-related candidates are generated, and one training record is stored. The raw teacher rollouts may use simulator-grounded semantic observations to obtain stable pretraining states. During leakage-safe preparation, target locations, target-membership indicators, and success-node identifiers are removed from the model input, and target-hypothesis candidates are excluded from the prepared search input. At online inference, the same state schema is populated from the adapted detector and the accumulated semantic memory.

For supervision, valid demand object instances are converted into a success region using the task success-distance criterion, which is 1.5 m in the default configuration. The corresponding reachable nodes form the set of success nodes. For each candidate, the data builder computes its shortest topological distance D_i to this region, its soft value $v_i = \exp(-D_i/\tau_v)$, and its path cost κ_i from the current node. The default value temperature is $\tau_v = 6.0$. The ranking target is the candidate with the largest success-distance improvement $D_t - \kappa_i - D_i$, followed by deterministic tie breaking using D_i, v_i, and κ_i. Value, distance, improvement, and path-efficiency fields are stored in the raw records, while the main SeVN objective uses the ranking target and the value-regression target defined in Section 4.2. The prepared dataset is split by episode, with the default validation ratio set to 10%, so temporally adjacent states from one rollout cannot appear in both partitions. Success nodes and all topology-derived labels are used only while constructing these offline records and are not exposed during online inference.

D.3 Target Acquisition Data

The acquisition datasets are designed around the detection-to-success gap. They contain not only successful target views, but also visually plausible proposals that should not trigger `Done`. This makes the verification heads learn the difference between recognizing a target-like object and reaching a valid stopping state.

Distance Head data. The Distance Head dataset combines proposals collected from online YOLO rollout frames with target-centric oracle samples. For the latter, the simulator places the agent at reachable positions around valid target instances and renders multiple distance bins, yaw jitters, and camera horizons. The default distance bins cover $[0, 0.75]$, $(0.75, 1.2]$, $(1.2, 1.5]$, $(1.5, 1.8]$, $(1.8, 2.5]$, and $(2.5, 4.0]$ meters. For each proposal, the metric-distance label is obtained from the depth statistics inside its matched target box, using a $p10$ depth estimate by default and an agent–object XZ distance fallback when valid depth is unavailable. The reachability label is determined from the target's XZ distance and the actionable success range. Near positives, small targets, and boundary negatives just outside the success range are repeated during training to emphasize the spatial boundary at which detection becomes actionable. The train/validation split is performed by episode and stratified to retain both reachable and non-reachable cases.

Done Success Head data. Done Success examples are generated by replaying navigation frames in the simulator and running the YOLO detector on the replayed RGB observations. For each frame,

simulator metadata identifies valid target instances that are visible and within the success distance, and their ground-truth boxes are matched to each detector proposal by intersection-over-union. A proposal is labeled as a positive only when its object type is valid for the episode and its best IoU with a visible success target exceeds 0.5. Consequently, a demand-relevant category at an insufficient distance, an incorrect instance, or a poorly aligned box remains a negative example. The dataset explicitly includes `Done` steps, metric-success-visible steps, visible-target steps, and mined hard-negative cases (Shrivastava et al., 2016). This replay construction exposes near-target failures and false-stop cases that would be missed by collecting only successful terminal frames. Train and validation partitions are separated by episode and stratified so that both positive and negative episodes are represented in each partition. The proposal context stored with each example includes detector confidence, box geometry and clipping status, Distance Head outputs, target-lock statistics, target size group, and initial local-scan state.

Risk labels. When the risk branch is enabled, its labels are constructed separately from the success labels. Successful `Done` proposals are assigned risk label 0. Risk-positive examples are selected from unsuccessful but visually plausible proposals, including mined hard negatives, wrong proposals selected by the policy, near-target negatives, active `Done` steps that fail the metric criterion, and proposals with high predicted success or reachability evidence. The selective rule therefore does not mark every negative proposal as risky; it concentrates supervision on cases likely to cause a false stop. The risk branch is trained with these labels and is used to reject ambiguous proposals in the relaxed-success region of the inference gate.

E Experiment

E.1 Evaluation Metric Definitions

We provide the formal definitions of the three evaluation metrics used in Section 5.1. Let N denote the number of evaluation episodes. For episode i, let $S_i^{\text{nav}} \in \{0,1\}$ indicate navigation success, and let $S_i^{\text{sel}} \in \{0,1\}$ indicate selection success at termination. Let P_i be the executed path length and $P_i^\star$ be the shortest path length from the initial pose to any valid success region. We set the maximum actionable distance to $d_{max} = 1.5\,\text{m}$ and the bounding-box threshold to $\theta_{\text{iou}} = 0.5$.

We define navigation success as

$$S_i^{\text{nav}} = \mathbb{I}\left[\exists\, \omega \in \mathcal{I}_{\tau_i}^{\text{vis}} : y(\omega) \in \mathcal{Y}_{d_i}, \text{dist}_{xz}(\mathbf{x}_{\tau_i}, \mathbf{p}(\omega)) \le d_{\max}\right], \tag{13}$$

where τ_i is the termination step, $\mathcal{I}_{\tau_i}^{\text{vis}}$ is the set of visible object instances at termination, and $\mathcal{Y}_{d_i}$ is the set of demand-satisfying categories for instruction d_i.

Navigation Success Rate (NSR). NSR measures whether the agent stops within the required navigation distance of a visible demand-satisfying object:

$$\text{NSR} = \frac{1}{N}\sum_{i=1}^{N} S_i^{\text{nav}}. \tag{14}$$

Success weighted by Path Length (SPL). SPL further accounts for path efficiency among successful navigation episodes:

$$\text{SPL} = \frac{1}{N}\sum_{i=1}^{N} S_i^{\text{nav}} \frac{P_i^\star}{\max(P_i, P_i^\star)}. \tag{15}$$

This metric assigns higher scores to successful episodes whose executed paths are closer to the shortest feasible paths.

Selection Success Rate (SSR). SSR measures whether the agent not only reaches a valid stopping region, but also selects the correct target instance. Given the predicted bounding box $\hat{\mathbf{b}}_{\tau_i}$ at termination and the ground-truth box $\mathbf{b}(\omega)$ of a visible demand-satisfying object, selection success is defined as

$$S_i^{\text{sel}} = S_i^{\text{nav}} \cdot \mathbb{I}\left[\text{IoU}(\hat{\mathbf{b}}_{\tau_i}, \mathbf{b}(\omega)) \ge \theta_{\text{iou}}\right]. \tag{16}$$

SSR is then computed as

$$\mathrm{SSR} = \frac{1}{N} \sum_{i=1}^{N} S_i^{\mathrm{sel}}. \tag{17}$$

NSR evaluates whether the agent reaches a demand-satisfying object, SPL evaluates the efficiency of successful navigation, and SSR evaluates whether the final selected target is correct. Together, these metrics separate navigation, efficiency, and target-selection reliability.

E.2 Implementation Details

The demand parser is adapted from Qwen2.5-VL-7B-Instruct (Bai et al., 2025) using LoRA (Hu et al., 2021). It is fine-tuned for 8 epochs with LoRA rank 16, learning rate 1×10^{-4}, and bfloat16 precision. The detector is initialized from YOLO26 (Jocher et al., 2026) and trained for 150 epochs at 640×640 resolution. The semantic value model is a 4-layer Transformer encoder (Vaswani et al., 2017) with hidden dimension 256, 8 attention heads, and auxiliary value-loss weight 0.1. The Distance Head and Done Success Head both use a ResNet-18 (He et al., 2016) visual backbone with 224×224 full-image and target-crop inputs. All evaluation runs use seed 42. During online evaluation, the YOLO confidence threshold is 0.10, the demand-target threshold is 0.05, the NMS IoU threshold is 0.70, and at most 300 detections are retained per frame.

E.3 Efficiency Analysis

Runtime efficiency is important for demand-driven navigation because the agent must repeatedly perceive, decide, and act in a closed loop. We therefore evaluate the episode-level wall-clock time of DDSeVA and CogDDN. Since CogDDN does not release official code, we reproduce its core decision pipeline according to the paper description and measure its runtime under the same simulator protocol. All timings are measured on the same machine and include online perception, policy decision, target acquisition, and environment interaction. Offline training and data construction are excluded. For DDSeVA, the demand profile is cached before navigation, and online inference uses the object detector, SeVN, Distance Head, and Done Success Head.

Table 4: Runtime efficiency comparison. We report average wall-clock time per episode.

Method	Implementation	Avg. Time / Episode	Relative Speedup
CogDDN	Reproduced from paper	581.93 s	$1.0\times$
DDSeVA	Ours	13.63 s	$42.7\times$

As shown in Table 4, the reproduced CogDDN pipeline requires 581.93 seconds per episode on average, while DDSeVA requires only 13.63 seconds. This corresponds to a $42.7\times$ speedup and a 97.7% reduction in episode-level runtime. The difference mainly comes from the decision paradigm. CogDDN relies on heavy vision-language reasoning and reflective decision making during closed-loop execution, whereas DDSeVA converts demand grounding, semantic search, and target acquisition into lightweight learned modules. These results show that the performance gain of DDSeVA is not obtained by increasing online inference cost. Instead, DDSeVA improves closed-loop reliability while remaining substantially more efficient for embodied deployment.

E.4 Diagnosing Residual Failures

E.4.1 Failure Mode Analysis

To understand the remaining errors of DDSeVA, we analyze unsuccessful evaluation episodes across the four scene–instruction splits. Each failed episode is assigned to one dominant failure mode according to its trajectory evidence. Table 5 summarizes the resulting distribution.

A total of $1{,}219$ unsuccessful episodes are included in the analysis. The largest search-and-approach failures occur before the agent obtains a valid stopping opportunity. In 280 episodes (23.0%), no demand-relevant target proposal is detected. In another 252 episodes (20.7%), the agent detects a target but never reaches a valid success view. Together, these two categories account for 43.7% of all

Table 5: Failure modes among unsuccessful evaluation episodes. Each episode is assigned to one dominant failure mode.

Stage	Failure Mode	Episodes	Main Cause
Search	Target not detected	280 (23.0%)	Insufficient search or cross-room transfer
Approach	Target detected, no valid view	252 (20.7%)	Incomplete approach or viewpoint failure
Approach	Success distance, not visible	99 (8.1%)	Viewpoint or target-box mismatch
Stopping	Valid view, missed `Done`	232 (19.0%)	Missed stopping window
Stopping	Invalid `Done`	356 (29.2%)	Premature stop or wrong target box

failures, suggesting that insufficient exploration, cross-room transfer, and post-detection approach remain major bottlenecks.

The remaining failures occur after target-related evidence has appeared. In 99 episodes (8.1%), the agent reaches the target-distance range but fails to obtain a metric-visible view. In 232 episodes (19.0%), a valid stopping window appears but the policy misses the `Done` action. Finally, 356 episodes (29.2%) end with an invalid `Done`, usually due to premature stopping, an insufficient viewpoint, or an incorrect target box. These cases show that reaching a target region is still not equivalent to successful termination.

Overall, the failure distribution points to two residual bottlenecks. Before acquisition, the agent may fail to search far enough or transfer to the room containing the target. After target evidence appears, the agent may still need additional approach and verification before termination is reliable. This motivates the following step-budget analysis.

E.4.2 Step-Budget Sensitivity

We next test whether limited episode horizon contributes to the search-related failures above. We vary the maximum number of allowed steps while keeping all learned modules, thresholds, and decision rules unchanged. The official evaluation setting uses 100 steps, and we compare it with budgets of 125, 150, and 200 steps.

As shown in Figure 6, increasing the step budget consistently improves Navigation Success Rate and Selection Success Rate across all four settings. When the budget increases from 100 to 200 steps, the average Navigation Success Rate rises from 39.7% to 47.0%, and the average Selection Success Rate rises from 36.9% to 43.6%. The gains are especially clear in unseen scenes, where additional steps give the agent more opportunities to enter new rooms, recover from uncertain observations, and obtain valid target views. This confirms that a substantial portion of the remaining failures is caused by limited search coverage rather than only by incorrect final stopping decisions.

Figure 7 explains how the failure distribution changes under larger budgets. From 100 to 200 steps, episodes with no detected target decrease from 280 to 171, and episodes with a detected target but no valid success view decrease from 252 to 162. Thus, the additional budget mainly alleviates long-horizon search and approach failures. At the same time, invalid `Done` failures increase from 356 to 423. This reflects a redistribution of failures rather than a contradiction: more episodes move beyond search and enter target acquisition, where viewpoint quality, target-box selection, and stopping reliability still determine success.

E.4.3 Success–Efficiency Trade-off in Verified Navigation

The step-budget study also reveals a gap between success recovery and path efficiency. When the maximum budget increases from 100 to 200 steps, the average NSR and SSR increase by 7.3 and 6.7 percentage points, respectively, while the average SPL increases only from 19.7% to 20.9%. Additional steps therefore help DDSeVA recover more successful episodes, but these recovered trajectories often require longer paths.

This pattern points to a success–efficiency trade-off in demand-driven navigation. Larger budgets mainly help difficult episodes where the target is far away, located in another room, or requires repeated observations before termination. In such cases, the agent may need to explore additional

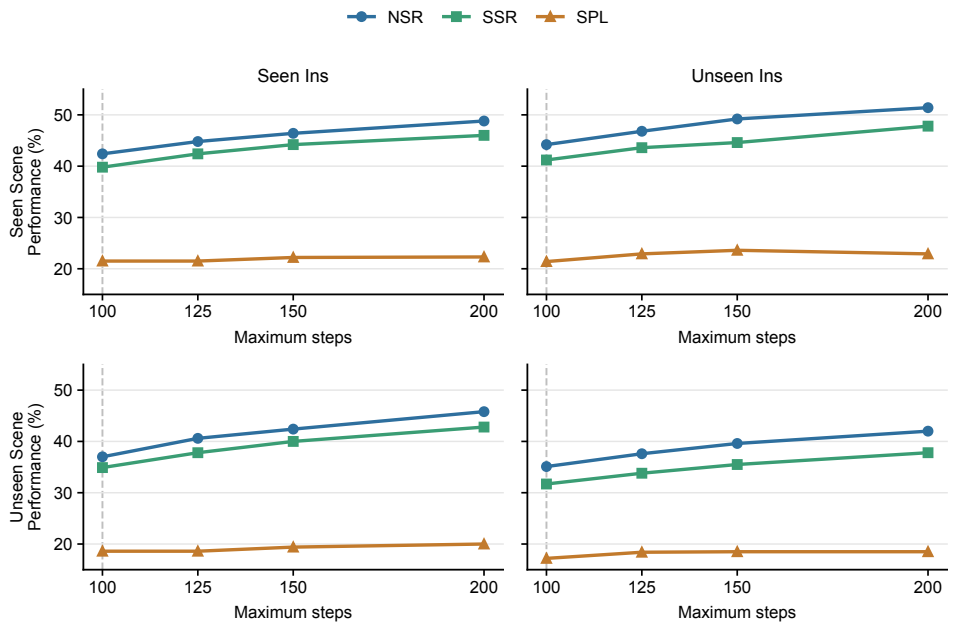


Figure 6: Step-budget sensitivity of DDSeVA under four scene–instruction settings. Increasing the maximum number of steps mainly improves Navigation Success Rate and Selection Success Rate, while Success weighted by Path Length grows more slowly.

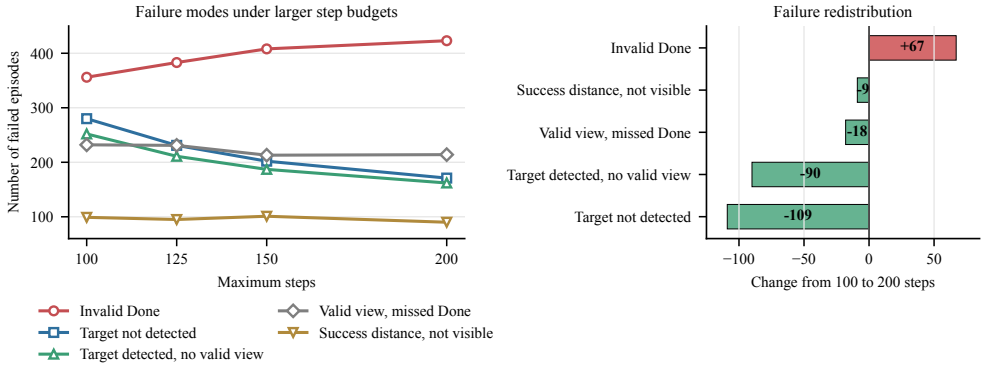


Figure 7: Failure redistribution under larger step budgets. Larger budgets substantially reduce search-related failures, including episodes where the target is never detected or detected without a valid success view. Invalid `Done` failures increase because more episodes progress from search into target acquisition and termination.

regions, revisit uncertain locations, approach the target from a better viewpoint, and verify whether the final proposal satisfies the stopping condition. These behaviors reduce premature termination and improve target selection, but they also increase the executed path length and limit SPL gains.

This observation suggests an important direction for future work: efficient verification. Reliable policies should not only determine whether a detected target can support `Done`, but also decide how much additional evidence is worth collecting. Future demand-driven agents may need uncertainty-aware stopping, cost-sensitive target acquisition, and search policies that jointly optimize target discovery, verification confidence, and path length. The next challenge is therefore to close the detection-to-success gap with fewer unnecessary movements.

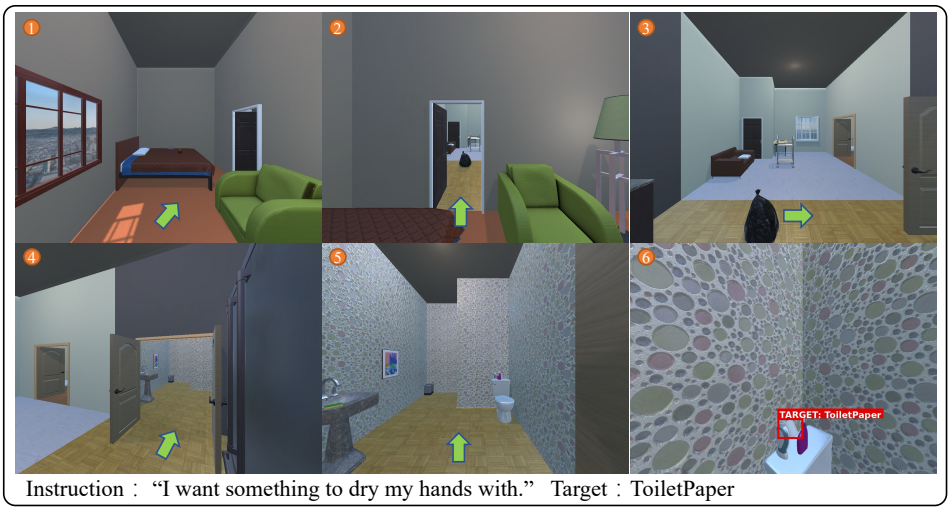


Figure 8: Successful long-horizon cross-room search. Given the demand "I want something to dry my hands with," DDSeVA leaves the initial bedroom, crosses the shared living area, enters the bathroom, and finally localizes the demand-relevant target `ToiletPaper`. The numbered frames show the progression from initial exploration to cross-room transfer and final target acquisition.

E.5 Qualitative Case Studies

We provide qualitative trajectories to illustrate how DDSeVA handles different closed-loop navigation situations. The following case focuses on long-horizon search when the demand-relevant target is outside the initial room.

Long-Horizon Cross-Room Search. Figure 8 shows a successful trajectory for the demand "I want something to dry my hands with." The demand does not explicitly specify an object category, and the benchmark target, `ToiletPaper`, is not visible in the initial bedroom. DDSeVA first leaves the starting room, crosses the shared living area, and transfers to the bathroom. After entering the bathroom, it continues local exploration until the target becomes clearly visible. The agent then approaches the target and terminates with the correct target localization. This example demonstrates that demand-driven navigation requires more than matching the current view to a target hypothesis. The agent must preserve the demand objective across rooms and sustain a long-range search until sufficient evidence for target acquisition becomes available.

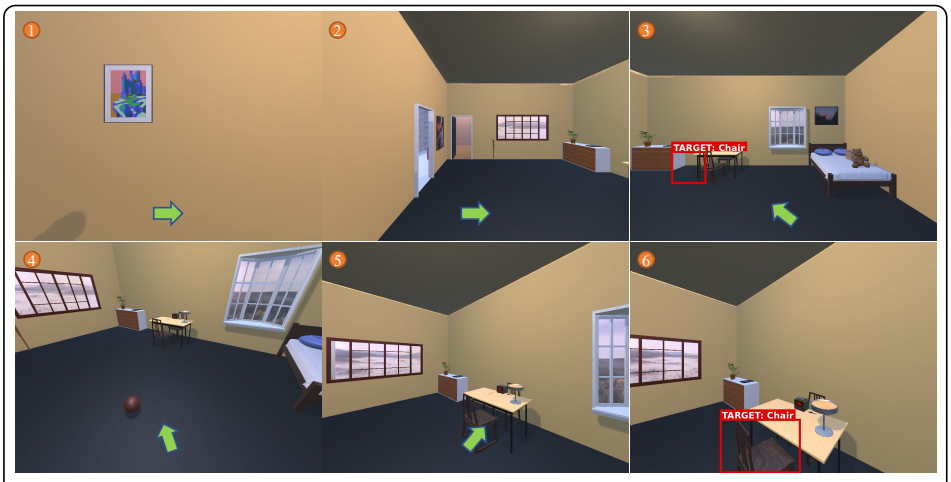
Recovery from Temporary Target Occlusion. Figure 9 illustrates a successful trajectory under intermittent target visibility. The demand is "I need a basin to clean and wash things in," whose corresponding target is `Sink`. The target is first detected through the doorway in frame 2, but becomes occluded as the agent changes its viewpoint and is not detected in the following frames. Instead of discarding the target hypothesis or terminating from a single observation, DDSeVA maintains the acquisition process and performs local recovery. Once the sink reappears in frame 5, the agent resumes its approach and obtains a close, well-localized view in frame 6 before completing the task.

Navigation under Intermittent Target Detection. Figure 10 illustrates a successful trajectory under unstable target recognition. The instruction is "I need a step to reach higher places," and the corresponding demand-relevant target is `Chair`. The target is detected in frame 3, but the detection disappears after the agent changes its viewpoint in frames 4 and 5. The disappearance does not indicate that the target is absent: the chair remains in the local scene, but its visual evidence is temporarily insufficient for the detector. DDSeVA therefore continues local navigation instead of terminating or abandoning the target hypothesis. After repositioning, the target is detected again in frame 6, allowing the agent to approach it and complete the task. This example demonstrates that reliable target acquisition must tolerate intermittent detection and maintain a target hypothesis across short periods of uncertain visual evidence.



Instruction: “I need a basin to clean and wash things in.” Target: Sink

Figure 9: Recovery from temporary target occlusion. Given the demand “I need a basin to clean and wash things in,” DDSeVA initially detects the target `Sink`, temporarily loses it because of viewpoint-induced occlusion, and then reacquires it through local recovery. The final frame shows a close and correctly localized target view before successful termination.



Instruction: “I need a step to reach higher places.” Target:Chair

Figure 10: Navigation under intermittent target detection. Given the instruction “I need a step to reach higher places,” DDSeVA detects the demand-relevant target `Chair` in frame 3, temporarily loses the detection after a viewpoint change in frames 4 and 5, and reacquires the target in frame 6. The trajectory illustrates how temporal target locking and local recovery support navigation under unstable visual recognition.

F Discussion and Future Work

While DDSeVA demonstrates significant improvements in bridging the detection-to-success gap, it also opens up several promising directions for future research in demand-driven navigation.

Efficiency and Long-Horizon Exploration. As revealed in our step-budget analysis, rigorously verifying a target requires accumulating multi-view evidence, which inherently trades path efficiency (SPL) for higher success rates. An exciting future direction is to develop cost-sensitive verification mechanisms that dynamically determine the optimal stopping point to minimize redundant observations. Furthermore, integrating large foundation models for high-level topological planning or

incorporating 3D scene graphs could further optimize long-horizon search and cross-room exploration in massively complex environments.

End-to-End Grounding and Real-World Embodiment. Currently, DDSeVA employs a modular design that leverages adapted vision-language models for demand interpretation and visual detection. A compelling next step is to design an end-to-end architecture that seamlessly unifies open-vocabulary perception with our verified acquisition pipeline, thereby entirely removing the dependency on predefined object vocabularies. Finally, extending DDSeVA with robust state estimation to handle noisy odometry and dynamic obstacles on physical robotic platforms remains a critical milestone for real-world deployment.

G AI USE STATEMENT

In this study, we utilized generative AI solely to refine the language, polish the phrasing, and improve the readability of our initial drafts. We did not employ AI tools for any core research activities, such as conceptualizing the framework, designing the architecture, conducting experiments, analyzing data, or deriving scientific conclusions. We have carefully reviewed all AI-assisted revisions, manually verifying that the polished text faithfully preserves our original methodological accuracy and scientific intent without introducing any misrepresentations. We assume full responsibility for the entirety of the final manuscript, including any text refined with the aid of generative AI.

H ETHICS STATEMENT

In accordance with the ICLR Code of Ethics, we are committed to the responsible development of embodied AI. While demand-driven navigation frameworks like DDSeVA hold substantial promise for assistive and domestic robotics, transitioning these systems to real-world environments entails critical ethical considerations. Our current research relies exclusively on publicly available simulation platforms, precluding the collection of any sensitive or human-subject data. However, physical deployment introduces inherent risks, particularly concerning visual privacy from continuous environmental perception and the potential for large language models to manifest societal biases during demand interpretation. We explicitly condemn the misuse of autonomous navigation for unauthorized surveillance or malicious applications. To promote transparency, reproducibility, and responsible progress, we will open-source our code and models, encouraging the community to collaboratively address these safety and fairness challenges.

I REPRODUCIBILITY STATEMENT

To facilitate reproducibility, we formalize the task and terminal success criterion in Section 3 and describe the DDSeVA architecture and closed-loop inference procedure in Sections 4.1–4.3. Implementation details for semantic-state construction, semantic value search, target locking, acquisition heads, and risk-aware termination are provided in Appendix C. The construction of training data for all adapted and learned components, including the leakage-safe training splits, is described in Appendix D, while the training configurations and inference hyperparameters are summarized in Appendix E.2. The evaluation protocol and metric definitions are specified in Section 5.1 and Appendix E.1, and the baseline configurations are documented in Appendix B. Anonymous source code and video demonstrations are available at `https://anonymous.4open.science/r/DDSeVA-6675`.