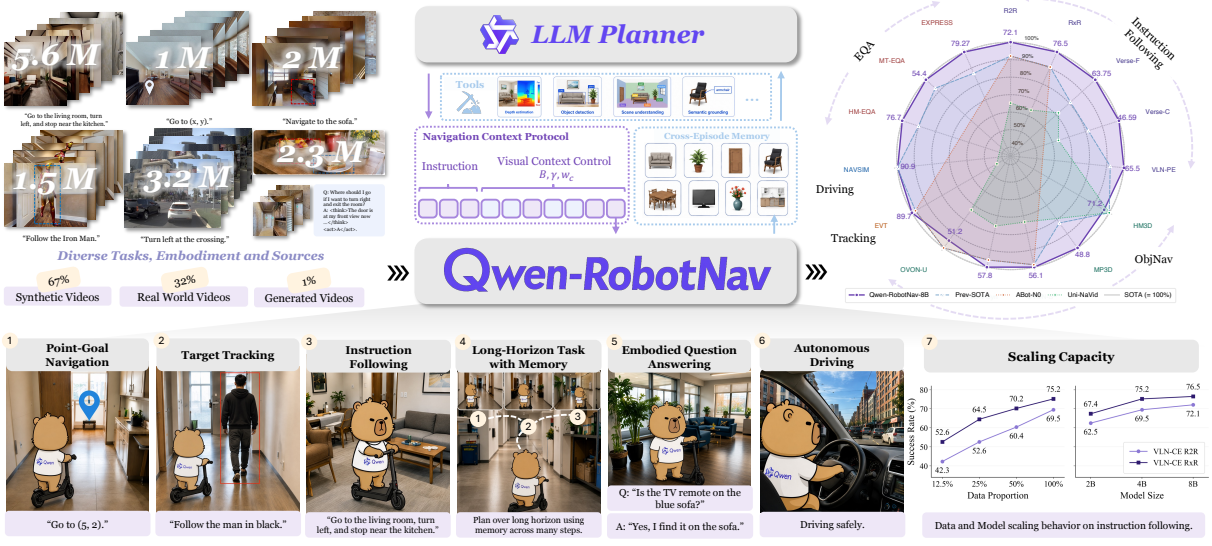
Qwen-RobotNav Technical Report: A Scalable Navigation Model Designed for an Agentic Navigation System

Qwen Team

https://qwen.ai/blog?id=qwen-robotnav
https://github.com/QwenLM/Qwen-RobotNav

Abstract

Agentic navigation systems require a base navigation model whose observation strategy can be externally reconfigured at inference time, because instruction following, object search, target tracking, and autonomous driving share the same perception-planning backbone yet demand fundamentally different strategies for consuming the visual stream. We present Qwen-RobotNav, a scalable navigation model built on Qwen3-VL that addresses it through a parameterised interface with two complementary dimensions: multiple task modes that select the navigation behaviour, and controllable observation parameters (e.g., token budget, per-camera weights) that govern how visual history is encoded. With training-time randomization over all parameters, Qwen-RobotNav is robust to any inference-time configuration requiring zero architectural modification to the Qwen3-VL backbone. We train Qwen-RobotNav on 15.6M samples; co-training with vision-language data prevents the collapse into reactive action-sequence mappers observed in trajectory-only training. The parameterised interface also makes Qwen-RobotNav a natural building block for agentic systems: for long-horizon scenarios, an upper-level planner decomposes goals into sub-tasks and dynamically switches Qwen-RobotNav's task mode and context strategy mid-episode, composing complex behaviours from repeated calls to the same model. Extensive experiments show that Qwen-RobotNav sets new state-of-the-art results across major navigation benchmarks, achieving 76.5% success rate on VLN-CE RxR, 90.0% tracking rate on EVT-Bench, and 91.4 PDMS on NAVSIM. Beyond these standalone results, an agentic navigation system built with **Qwen-RobotNav set a new state of the art on Embodied Question Answering, improving over the best prior method by 10.8% on HM-EQA and 15.4% on EXPRESS-Bench while requiring 77% fewer navigation steps**. The model exhibits favourable scaling from 2B to 8B parameters, with joint multi-task training developing a shared spatial-planning substrate that transfers across task families, and demonstrates strong zero-shot generalisation to real-world robots across diverse environments.



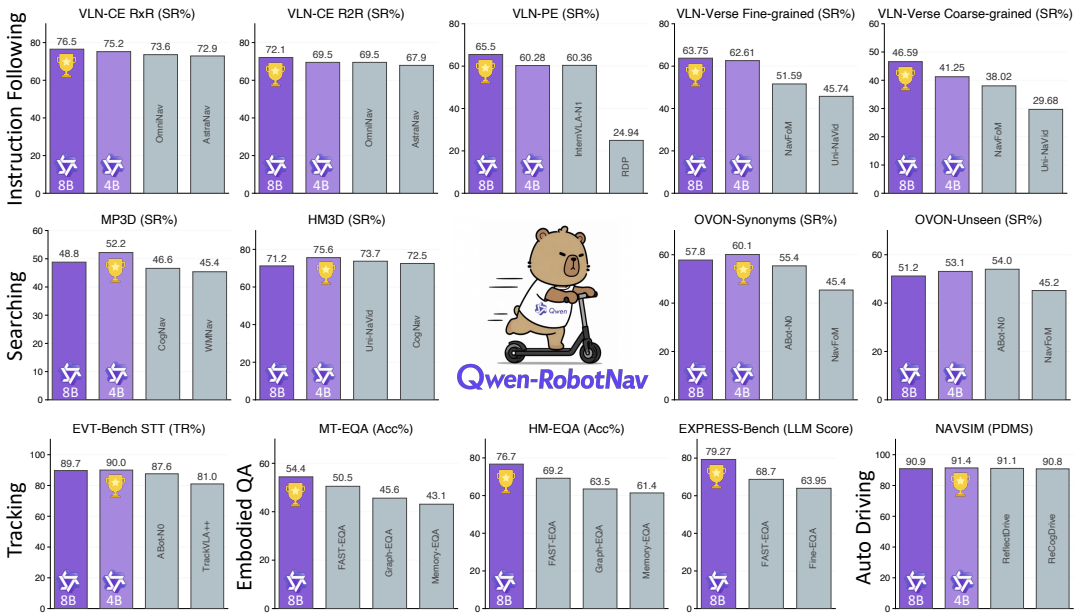


Figure 1: **Benchmark summary.** Across instruction following, object search, target tracking, embodied question answering, and autonomous driving, Qwen-RobotNav-4B and Qwen-RobotNav-8B achieve state-of-the-art or competitive performance against specialist and navigation foundation model baselines. Trophy icons mark the best result in each benchmark group.

1 Introduction

Embodied navigation spans a remarkably diverse family of tasks, including instruction following (Anderson et al., 2018; Krantz et al., 2020; Ku et al., 2020; Yin et al., 2025a), point-goal navigation (Savva et al., 2019; Wijmans et al., 2020), object searching (Batra et al., 2020; Yokoyama et al., 2024b;a; Yin et al., 2025b), target tracking (Zhong et al., 2024; Wang et al., 2025b; Liu et al., 2025), and autonomous driving (Caesar et al., 2020; Sun et al., 2020), each coupling perception with purposeful movement through open-world environments. Recent unified navigation models (Zhang et al., 2024a; 2025b;a; AMAP CV Lab, 2025; Cheng et al., 2025; Wang et al., 2026; Cai et al., 2025; Zhou et al., 2025a; Zhong et al., 2025; Yin et al., 2026) and navigation-oriented evaluation suites (Lin et al., 2025; Zhao et al., 2025) have demonstrated that a single architecture can handle multiple task families.

However, complex real-world scenarios, such as Embodied Question Answering (Das et al., 2018; Majumdar et al., 2024; Ren et al., 2024; Yang et al., 2025b; Zhang et al., 2026), demand more: the navigation model must serve as a core module within larger agentic systems, where an outer planner dynamically orchestrates navigation capabilities to accomplish long-horizon, multi-step goals. This requires not only multi-task capability, but also a controllable interface whose observation strategy can be reconfigured by an external agent at inference time. Currently, no existing model meets this requirement: NavFoM (Zhang et al., 2025a) uniformly sub-samples frames; ABot-N0 (AMAP CV Lab, 2025) retains only a sliding window, each embedding a single assumption about which observations matter that cannot be adjusted at deployment time. The root cause is that different navigation tasks demand fundamentally different temporal context and memory strategies. Instruction following requires retaining observations spanning dozens of prior steps to re-reference distant landmarks (An et al., 2024; Wang et al., 2024; Wei et al., 2025b;a), whereas target tracking is governed by the most recent few frames and treats stale history as noise (Zhong et al., 2024; Wang et al., 2025b; Liu et al., 2025). Object searching further complicates the picture by shifting its own requirements within a single episode, from broad global memory during exploration to tight recency-focused attention during approach (Yokoyama et al., 2024a; Yin et al., 2024; Kuang et al., 2024; Cao et al., 2024). No single fixed context strategy can serve this full spectrum, let alone adapt mid-episode.

We introduce Qwen-RobotNav, a scalable navigation model built on Qwen3-VL (Yang et al., 2025a) that reframes the central challenge of multi-task navigation as observation context modelling rather than architecture design. Qwen-RobotNav formulates all tasks as unified waypoint trajectory prediction and exposes a parameterised interface with two complementary configuration dimensions. First, multiple task modes (VLN, PointNav, ObjNav, Tracking) allow an upper-level planner to select the navigation behaviour appropriate for each sub-goal. Second, controllable observation parameters, including visual

token budget, temporal decay, and per-camera importance weighting, govern how the model consumes the observation stream, enabling dynamic adjustment of the context strategy at inference time without task-specific retraining. Training-time randomisation over all parameters ensures that Qwen-RobotNav generalises to any inference-time configuration without task-specific tuning: the model never trains at a fixed setting, so it cannot overfit to one regime. Camera identity, temporal order, and embodiment type are communicated entirely through natural-language tags and prompt preambles, requiring zero architectural modification to the Qwen3-VL backbone; supporting a new platform requires only a new prompt template, not new parameters.

The parameterised interface makes Qwen-RobotNav a natural building block for agentic navigation systems. For long-horizon scenarios such as Embodied Question Answering, where prior systems rely on explicit reasoning, exploration, and memory modules (Zhou et al., 2024b;a; Long et al., 2024; Ren et al., 2024; Saxena et al., 2024; Yang et al., 2025b; Zhang et al., 2026), we deploy Qwen-RobotNav within a two-tier hierarchical system. An upper-tier planner (Qwen3.6-Plus) decomposes complex goals into sub-goals and dispatches configurable navigation calls, each specifying a task mode, a sub-goal instruction, and an observation configuration; Qwen-RobotNav serves as the reactive executor, predicting waypoint trajectories at high frequency. The planner can dynamically reconfigure Qwen-RobotNav's task mode and context strategy mid-episode, and the two tiers communicate exclusively through natural language, keeping the system modular and extensible. To support long-horizon reasoning, the system maintains a two-level memory: compact single-episode memory summarising each navigation rollout, and a persistent cross-episode memory that accumulates durable conclusions such as searched regions, candidate object locations, and rejected hypotheses, enabling effective context compression over extended episodes.

To train Qwen-RobotNav, we curate **15.6M** samples comprising navigation trajectory planning data (85%) spanning five task families (instruction following, point-goal navigation, object searching, target tracking, and autonomous driving) across diverse embodiments, and navigation-related vision-language reasoning data (15%). Co-training with vision-language data preserves the language understanding that underpins Qwen-RobotNav's natural-language camera and temporal tags and mitigates the degradation of open-world perception that can occur when vision-language models are adapted only to action prediction (Zhou et al., 2024a; Zhang et al., 2024a; 2025b; Cheng et al., 2025).

Without any task-specific fine-tuning, Qwen-RobotNav achieves state-of-the-art results across diverse navigation benchmarks. On VLN-CE, Qwen-RobotNav-8B attains **72.1%** SR on R2R and **76.5%** SR on RxR Val-Unseen, surpassing NavFoM by 10.4% and 12.1% SR respectively. On EVT-Bench, Qwen-RobotNav achieves the highest tracking rate (**90.0%** TR) among all evaluated methods. Strong cross-embodiment generalisation is also observed on HM3Dv2 object-goal navigation (**75.6%** SR) and NAVSIM autonomous driving (**91.4** PDMS). The model exhibits favourable scaling properties: performance improves from 2B to 8B parameters, with particularly pronounced gains on long-horizon reasoning tasks. Equipped with the agentic system, Qwen-RobotNav sets new state-of-the-art results on three EQA benchmarks (HM-EQA, MT-EQA, and EXPRESS-Bench), and further demonstrates zero-shot transfer to real-world robots across diverse environments.

2 Navigation Model

Task Formulation. We consider a general mobile navigation setting (Zhang et al., 2025a; AMAP CV Lab, 2025; Zhou et al., 2025a) in which an embodied agent receives a textual instruction $\mathcal{L}$ and a sequence of multi-view observations $\mathbf{I}_{1:T}^{1:N} \in \mathbb{R}^{H\times W\times 3}$ captured from N cameras at T timesteps. Given these inputs, the navigation policy π must predict a waypoint trajectory

$$\mathcal{W} = \{(x_k, y_k, \theta_k)\}_{k=1}^{K}, \tag{1}$$

where K=8 waypoints each encode a 2D position (x_k, y_k) and heading θ_k. A central challenge is that T grows online as navigation proceeds, N varies across robot platforms, and the total visual token count scales as $\mathcal{O}(T \cdot N)$, rapidly exceeding the context budget of any LLM backbone without a principled compression strategy. Moreover, different tasks impose fundamentally different demands on how observations should be modelled: target tracking requires only a short recency window of high-resolution frames to maintain lock on a moving object, whereas object-goal navigation must retain long-horizon episode history to recall previously explored regions and avoid redundant revisits. A single, task-agnostic observation encoding therefore cannot serve all navigation tasks well, motivating the task-adaptive strategy described in Section 2.2.

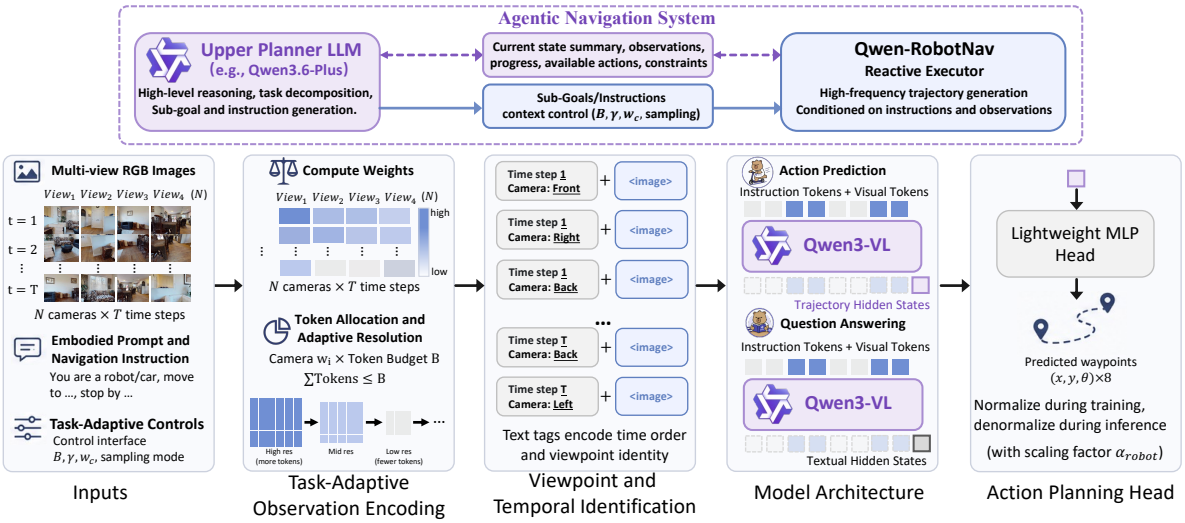


Figure 2: **Qwen-RobotNav architecture.** *Top:* In the agentic navigation system, an upper planner LLM decomposes long-horizon goals into sub-goals and controls Qwen-RobotNav through task-adaptive context parameters such as token budget B, temporal decay γ, camera weights w_c, and frame sampling mode. *Bottom:* Qwen-RobotNav receives multi-view RGB observations, an embodied prompt, and a navigation instruction; allocates visual tokens across cameras and timesteps; inserts natural-language temporal and viewpoint tags; and feeds the resulting visual-language sequence into the Qwen3-VL backbone. A lightweight MLP action head maps trajectory hidden states to eight waypoint predictions (x, y, θ).

2.1 Model Architecture

Qwen-RobotNav inherits the Qwen3-VL architecture (Yang et al., 2025a) and augments it with a lightweight action head for trajectory regression. The overall system consists of three components:

- **Vision Encoder.** The vision encoder, inherited from Qwen3-VL, is built on a SigLIP-2 (Tschannen et al., 2025) Vision Transformer with native dynamic-resolution support via 2D-RoPE (Chen et al., 2025a), enabling the token allocation strategy (Section 2.2) to freely rescale each camera view to its allocated pixel budget. A two-layer MLP patch merger compresses s_m^2 adjacent spatial tokens into a single vector aligned to the LLM hidden dimension, and the DeepStack mechanism (Meng et al., 2024) injects visual tokens from multiple ViT layers into early LLM layers, preserving multi-level visual representations critical for spatial reasoning.
- **Language Backbone.** The LLM, also inherited from Qwen3-VL, processes the concatenation of visual tokens (organised by the strategy described in Sections 2.2 and 2.3) and language tokens from $\mathcal{L}$. Through large-scale vision-language pretraining, the backbone has already acquired rich world knowledge, spatial understanding, and cross-modal reasoning capabilities that transfer directly to navigation.
- **Action Head.** A lightweight 4-layer MLP maps the LLM's final hidden state to K=8 waypoints, each with 3 degrees of freedom (x_k, y_k, θ_k), yielding a 24-dimensional output (Section 2.5). By keeping the action head minimal, the bulk of spatial reasoning remains within the LLM, allowing the model to benefit from the pretrained language backbone's world knowledge when planning trajectories in unseen environments.

2.2 Task-Adaptive Observation Encoding

Navigation is inherently a sequential decision-making process under partial observability: the agent cannot see the full environment at once and must accumulate spatial knowledge from a stream of egocentric observations. How much history to retain, and at what fidelity, depends fundamentally on the task's decision structure. Tasks that require *plan verification*, such as instruction following, demand global episode memory so the agent can re-reference previously observed landmarks against the instruction; tasks driven by *reactive pursuit*, such as target tracking, depend almost entirely on the most recent frames to maintain a tight perception loop around the moving target. Fixed context strategies, such as uniform sampling (Zhang et al., 2024a; 2025b;a) or sliding windows (AMAP CV Lab, 2025), commit to a single

Algorithm 1 Task-Adaptive Observation Encoding

Require: Observations $\mathbf{I}_{1:T}^{1:N}$, instruction $\mathcal{L}$, config $\Phi{=}(B,\ \gamma,\ \{w_c\},\ m,\ b_{\min},\ b_{\max})$
Ensure: Token sequence fed to the LLM backbone

$\mathbf{I}' \leftarrow \textsc{SampleFrames}(\mathbf{I}_{1:T}^{1:N},\ m,\ T')$ ▷ $T'{\le}T$ frames
$\omega_t \leftarrow \exp\big(\gamma \cdot t\ /\ (T'{-}1)\big),\ t{=}0,\dots,T'{-}1$
$\mathbf{W}[t,c] \leftarrow \omega_t \cdot w_c$ for all (t,c) ▷ Eq. 3
$\mathbf{A} \leftarrow \textsc{ConstrainedAlloc}(\mathbf{W},\ B,\ b_{\min},\ b_{\max})$
for each frame (t,c) **do**
    $\hat{\mathbf{I}}'_{t,c} \leftarrow \textsc{Resize}(\mathbf{I}'_{t,c},\ \ \mathbf{A}[t,c] \times (p \cdot s_m)^2)$
end for
$\mathbf{V} \leftarrow \textsc{VisionEncode}(\hat{\mathbf{I}}')$ ▷ ViT + patch merger
$\mathbf{S} \leftarrow \textsc{Interleave}(\mathbf{V},\ \text{viewpoint/temporal tags})$ ▷ Sec. 2.3
return Concat($\mathbf{S}$, Tokenize($\mathcal{L}$))

function $\textsc{ConstrainedAlloc}(\mathbf{W},\ B,\ b_{\min},\ b_{\max})$
    $\mathbf{A} \leftarrow b_{\min} \cdot \mathbf{1}_{T' \times N};\quad \mathcal{U} \leftarrow$ all cells
    repeat
        Distribute $B - \sum\mathbf{A}$ to $\mathcal{U}$ proportionally to $\mathbf{W}$
        Clamp cells exceeding $b_{\max}$; remove from $\mathcal{U}$
    until no cell exceeds $b_{\max}$
    return $\mathbf{A}$
end function

regime and cannot adapt across tasks without retraining. Qwen-RobotNav instead provides a unified, parameterised interface with four orthogonal control axes:

Parameter	Symbol	Range	Effect
Visual token budget	B	2048–4096	Total tokens across all cameras and timesteps
Temporal decay	γ	$[1,3]$	Recency bias toward the latest frames
Camera weights	w_c	per-robot defaults	Per-camera importance multiplier
Frame sample mode	-	`random` / `latest`	History coverage vs. recency window

Token Allocation Algorithm. Given T timesteps and N cameras, we first sub-sample or select $T'{\le}T$ frames according to the frame sample mode, and then compute a temporal weight for each retained frame:

$$\omega_t = \begin{cases} 1, & T' = 1, \\ \exp\left(\gamma \cdot \frac{t}{T'-1}\right), & T' > 1, \end{cases} \quad t = 0,\dots,T'{-}1, \tag{2}$$

so that $\gamma{=}0$ recovers uniform weighting and larger γ allocates disproportionately more tokens to recent frames (at $\gamma{=}2$ the most recent frame receives $\approx 7.4\times$ the budget of the oldest), as visualised in Figure 3(a). A joint weight matrix is then formed as:

$$\mathbf{W}[t,c] = \omega_t \cdot w_c, \quad t \in \{0,\dots,T'{-}1\},\ c \in \{1,\dots,N\}, \tag{3}$$

where w_c is a per-camera importance weight that reflects the asymmetric information density across viewpoints: the forward-facing camera, which captures the richest actionable cues such as obstacles, navigable paths, and goal landmarks, is assigned the highest weight, while the rear camera receives the lowest as it primarily provides contextual redundancy (e.g. $w_c{=}[2.0, 1.0, 0.5, 1.0]$ for front, right, back, left). During training, w_c is randomly sampled from its own range (e.g. the front camera from $\mathcal{U}[1.5, 2.5]$); see Section 2.6 for full randomisation details.

For a given token budget B and the frame (t,c) at time step t and camera c, token allocation proceeds in three stages: (1) each cell (t,c) receives a minimum floor of $b_{\min}$ tokens; (2) the remaining budget $B - T'Nb_{\min}$ is distributed proportionally to $\mathbf{W}$; (3) any cell exceeding the per-image ceiling $b_{\max}$ releases its surplus, which is redistributed iteratively until stable. We require $T'Nb_{\min} \le B \le T'Nb_{\max}$; if a sampled configuration violates this constraint, B is clipped to the feasible interval before allocation. The allocated token count for each image then determines its pixel resolution through the patch size and spatial merge size of the vision encoder, and the image is rescaled to this pixel budget while preserving the original aspect ratio. Note that this token allocation is an empirically proven heuristic to expose a unified interface to control the token context of the model, which is also used in our agentic system (Section 3). We believe this strategy could be further improved by a more principled token allocation algorithm. The

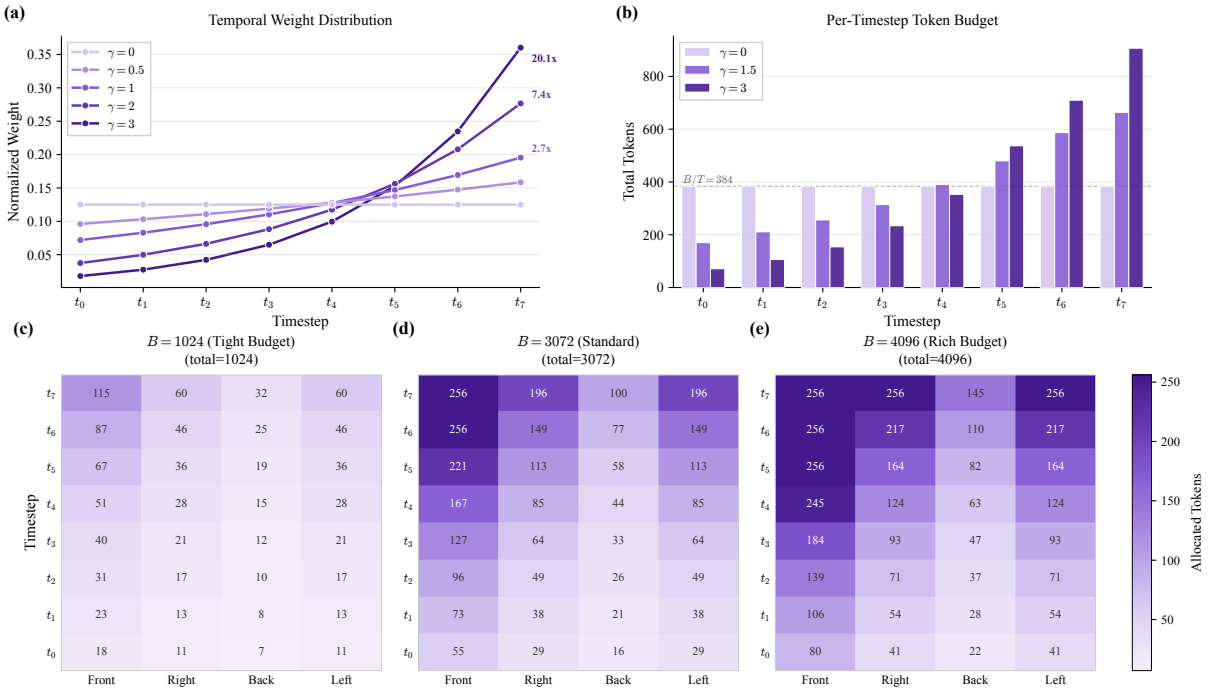


Figure 3: **Visualization of task-adaptive observation encoding.** **(a)** Normalized temporal weights $\omega_t = \exp(\gamma \cdot t/(T'-1))$ for varying decay factors γ when $T'>1$; annotations show the newest-to-oldest weight ratio. **(b)** Resulting per-timestep token budget (summed across all cameras) under a fixed total budget B=3072 with camera weights w_c=[2.0, 1.0, 0.5, 1.0] for front, right, back, and left views. The dashed line marks the uniform baseline B/T'. **(c, d, e)** Token allocation matrices across timesteps and cameras for three budget levels (B=1024, 3072, 4096) at fixed γ=2.0, with per-image constraints $b_{\min}$=4 and $b_{\max}$=256. Higher γ concentrates tokens on the most recent frames, while larger B increases overall fidelity until the per-image ceiling $b_{\max}$ saturates the highest-weight cells.

full procedure is summarised in Algorithm 1. Figure 3(c, d, e) illustrates how the constrained allocation distributes tokens across timesteps and cameras under three representative budget levels.

Training-Time Randomisation. At training time, all observation hyperparameters are independently randomised per sample: $\gamma \sim \mathcal{U}[1,3]$, $B \sim \mathcal{U}[2048, 4096]$, each camera weight w_c sampled from its own range (e.g. the front camera from $\mathcal{U}[1.5, 2.5]$), and the per-image token floor and ceiling $b_{\min} \sim \mathcal{U}_{\mathbb{Z}}[1, 8]$, $b_{\max} \sim \mathcal{U}_{\mathbb{Z}}[128, 256]$. This ensures the model is never trained at a fixed configuration and supports robust generalisation across configurations within the randomised training range, with moderate extrapolation beyond it, without task-specific tuning.

2.3 Viewpoint and Temporal Identification

After encoding, visual tokens from different cameras and timesteps are indistinguishable to the LLM: the backbone has no built-in mechanism to associate a token with "front camera, step 12" versus "left camera, step 3". Without explicit identity signals, the model cannot learn camera-specific or temporally-ordered representations, limiting its ability to reason about spatial layout or episode history.

Qwen-RobotNav resolves this by interleaving natural-language viewpoint and timestep tags with the visual tokens before they enter the LLM. Concretely, each timestep group is introduced by a `Time step` t header, followed by each camera's name and its corresponding <image> token. For example, a two-step, six-camera input is serialised as:

```
Time step 0 Front View ⟨image⟩ Front Right View ⟨image⟩ . . . Front Left View ⟨image⟩
Time step 1 Front View ⟨image⟩ . . .
```

Camera identity and temporal order are thus communicated entirely through ordinary vocabulary tokens already present in Qwen3-VL, requiring zero architectural modification. The model's existing open-world language grounding handles viewpoint semantics naturally: words like "Front", "Left", and "Back Right" carry spatial meaning that the pretrained LLM has already internalised from its pretraining corpus. An alternative representation is to specify each camera by its numeric azimuth (e.g. "right 90 degrees");

however, we find that descriptive names yield slightly better performance, likely because they carry richer semantic associations that the language model can leverage for spatial reasoning.

2.4 Embodiment-Aware Prompt Design

Since Qwen-RobotNav operates across fundamentally different physical platforms, it must distinguish between embodiments such as indoor mobile robots and autonomous vehicles. Rather than encoding embodiment information through learned embeddings or separate model heads, Qwen-RobotNav communicates the agent's physical identity through a natural-language preamble in the system prompt. For example, an indoor robot sample begins with "`Imagine you are a` **`robot`** `programmed for` **`navigation tasks`**", whereas an autonomous driving sample begins with "`Imagine you are a` **`car`** `programmed for` **`autonomous driving`**".

This embodiment preamble acts as a *task prior*: by identifying itself as a "robot" or a "car", the model recruits different subsets of its pretrained world knowledge (e.g., indoor spatial layouts versus traffic rules and road geometry) to inform trajectory prediction. The text-based approach is also inherently extensible: supporting a new platform, such as a drone, a wheeled robot, or a quadruped robot, requires only defining a new prompt template, with no architectural changes or additional parameters.

2.5 Action Planning

The action planning module maps the LLM's final hidden state $\mathbf{E}^{\mathcal{A}} \in \mathbb{R}^d$ to a waypoint trajectory. We use a lightweight 4-layer MLP head with hidden dimension 512 and GELU activations. The 24-dimensional output encodes K=8 waypoints each with 3 DOF (x_k, y_k, θ_k). Waypoints are normalised to $[-1, 1]$ during training using per-dataset scale factors computed as the 99th percentile of each coordinate across all training trajectories, and training minimises the MSE loss between predicted and ground-truth trajectories. At inference, the predicted waypoints are de-normalised using the same scale factors.

2.6 Training Strategy

Training Objective. Qwen-RobotNav is trained with a composite loss that jointly optimises trajectory regression and vision-language alignment:

$$\mathcal{L} = \mathcal{L}_{\text{traj}} + \lambda \, \mathcal{L}_{\text{VL}}, \tag{4}$$

where $\mathcal{L}_{\text{traj}} = \|\hat{\mathcal{W}} - \mathcal{W}^*\|_2^2$ is the MSE loss between predicted and ground-truth waypoints (active only on navigation trajectory samples), and $\mathcal{L}_{\text{VL}}$ is the standard next-token prediction loss on navigation-related vision-language reasoning samples. The two objectives share the same forward pass; λ is set to 1.0 in all experiments.

Configuration Randomisation. A central principle of Qwen-RobotNav's training is that *no observation configuration is fixed*. At each training step, all continuous control parameters of the task-adaptive interface (B, γ, w_c, $b_{\min}$, $b_{\max}$; ranges detailed in Section 2.2) are independently sampled per sample, and the frame sample mode alternates between `random` and `latest` with equal probability.

By exposing the model to a broad combinatorial space of these parameters during training, Qwen-RobotNav generalises across configurations within the randomised training range without task-specific tuning. In particular, training with both `random` (global coverage) and `latest` (recency window) frame sampling in equal proportion supports zero-shot context-strategy switching at deployment time, a property exploited by the agentic system described in Section 3.

Co-training. The training corpus mixes 85% navigation trajectory planning data with 15% navigation-related vision-language reasoning data (Section 4). Datasets are sampled at the batch level using per-dataset rates defined in a dataset registry, ensuring balanced exposure across all navigation task types (instruction following, point-goal navigation, object searching, target tracking, and autonomous driving) within every training epoch. The vision-language component prevents catastrophic forgetting of Qwen3-VL's open-world perception capabilities: models trained on navigation trajectory data alone tend to collapse toward reactive action sequences and lose general-purpose spatial reasoning, while co-training preserves the rich language grounding that underpins Qwen-RobotNav's zero-shot generalisation to unseen environments and instruction styles.

Optimisation Details. Qwen-RobotNav is initialised from the pretrained Qwen3-VL checkpoint and fine-tuned end-to-end. We use the AdamW optimiser (Loshchilov &Hutter, 2019) with β_1=0.9, β_2=0.95, and weight decay 10^{-2}. A cosine learning rate schedule is applied with a linear warm-up over the first 3% of training steps; the peak learning rate is 2×10^{-5} for the vision encoder and LLM backbone and

1×10^{-4} for the action head. Gradient norms are clipped to 1.0. The 8B model is trained with a global batch size of 256 for a total of 2,816 H100 GPU hours.

3 Qwen-RobotNav for Agentic Navigation

3.1 Overview

The navigation model described in Section 2 is designed not only for standalone benchmark evaluation, but also for deployment as a navigation module inside general-purpose embodied agents (Zhou et al., 2024b; Long et al., 2024). In this setting, the upper-level agent should be able to decide not only *what* navigation sub-goal to execute, but also *which navigation task mode*, *which observation configuration*, and *when to query auxiliary visual evidence* should be used for the current phase of the task. For example, a long-horizon search episode may alternate between instruction following, object-goal search, point-to-point movement, and target tracking, while also changing how much visual history should be retained.

We therefore expose Qwen-RobotNav as an agent-ready navigation model through a lightweight tool interface. At each navigation step, the upper-level planner provides a sub-goal instruction $\mathcal{L}_i$, selects a task mode τ_i, and specifies an observation configuration Φ_i. The task mode τ_i describes the navigation behavior to be invoked, such as `VLN`, `PointNav`, `ObjNav`, or `Tracking`. The configuration $\Phi_i = (B, \gamma, \{w_c\}, m, b_{\min}, b_{\max})$ follows the notation in Section 2.2: B controls the total visual-token budget, γ controls the recency bias, $\{w_c\}$ specifies camera weights, m specifies the frame sample mode, and $b_{\min}, b_{\max}$ define the per-image allocation constraints. Given the selected task mode and configuration, Qwen-RobotNav predicts a waypoint trajectory $\mathcal{W}_i$ in the same form as Eq. (1).

Besides navigation calls, the planner may issue auxiliary vision-tool calls over the current observations or stored key frames. In our interface, these tools include object detection, scene understanding, and semantic grounding. They provide additional evidence for deciding the next sub-goal or verifying candidate observations, but they do not predict waypoints and do not replace Qwen-RobotNav as the core navigation executor.

The resulting system uses Qwen-RobotNav as the core navigation executor. The upper-level planner is responsible for decomposing long-horizon goals, selecting task modes and context configurations, querying auxiliary visual evidence when needed, and reasoning over the evidence returned by previous navigation calls. A lightweight harness connects the two sides: it converts planner decisions into Qwen-RobotNav calls, and converts completed navigation rollouts into compact trajectory evidence for subsequent planning. This keeps the agentic layer simple while allowing it to access the controllable interfaces already built into Qwen-RobotNav.

3.2 Agent-Facing Qwen-RobotNav Interface

The key agent-facing property of Qwen-RobotNav is that its navigation behavior is externally configurable at inference time. This configurability comes from two complementary navigation interfaces.

First, Qwen-RobotNav supports multiple navigation task modes. In `VLN` mode, the model follows natural-language route instructions and grounds them in visual observations. In `PointNav` mode, it moves toward a specified spatial target or waypoint-like goal. In `ObjNav` mode, it searches for an object category or instance using accumulated visual evidence. In `Tracking` mode, it prioritizes recent observations to maintain lock on a moving or recently observed target. These modes are not separate navigation policies; they are different task interfaces to the same Qwen-RobotNav model.

Second, each navigation call can specify an observation configuration Φ. The model section has already described the full task-adaptive observation encoding algorithm in Section 2.2, so here we only emphasize how it is used by the planner. The planner can increase B and use a weaker recency bias when the current sub-goal needs broader episode history, such as object search or revisiting a previously explored region. It can instead use a smaller B, larger γ, and `latest` frame sampling when the sub-goal is local and reactive, such as approaching a visible object or tracking a moving target. In practice, w_c, $b_{\min}$, and $b_{\max}$ are usually kept at platform-level defaults, while B, γ, and m are the main controls exposed to the planner.

A navigation call can therefore be written abstractly as:

$$\mathcal{W}_i = \texttt{nav\_qwennav}\big(\mathcal{L}_i,\ \tau_i,\ \Phi_i\big), \tag{5}$$

where $\mathcal{L}_i$ is the current sub-goal, τ_i is the selected task mode, and Φ_i is the observation configuration. This interface gives the planner access to Qwen-RobotNav's internal context controls without requiring architectural changes, task-specific fine-tuning, or a separate routing model. The same model weights can be used across different task phases; only the tool-call arguments change.

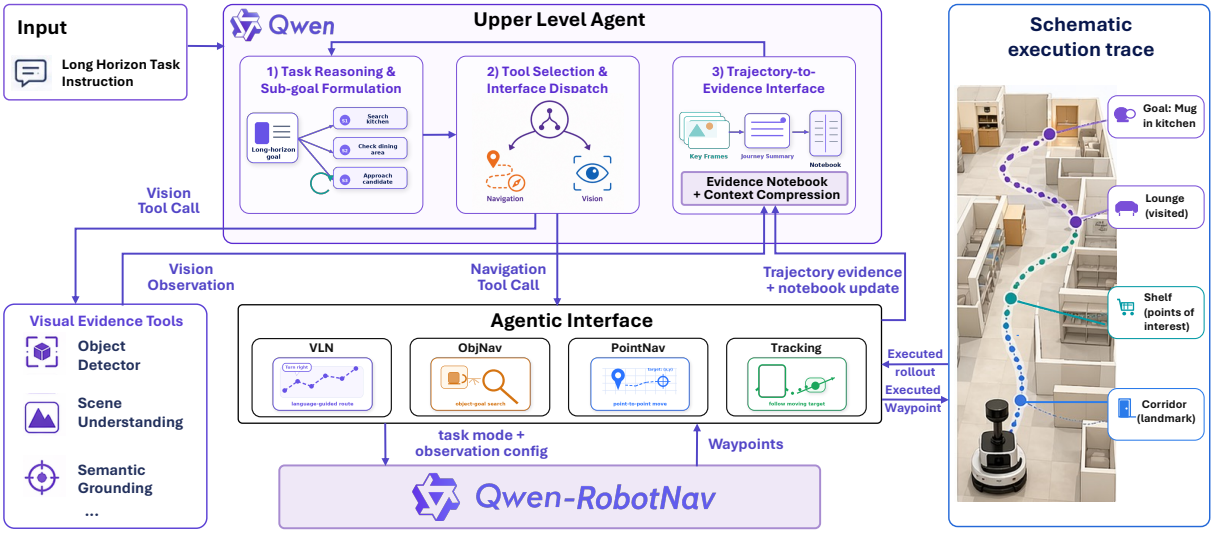


Figure 4: **Qwen-RobotNav for agentic navigation.** An upper-level planner decomposes a long-horizon task into sub-goals and dispatches either auxiliary vision-tool calls or Qwen-RobotNav navigation calls. Each navigation call is parameterized by a sub-goal instruction $\mathcal{L}_i$, a task mode τ_i, and an observation configuration Φ_i. Qwen-RobotNav uses the selected task mode and configuration to predict waypoints $\mathcal{W}_i$, which are executed in the environment. The harness converts the resulting rollout into source-indexed key frames, trajectory summaries, and evidence-notebook updates, providing compact context for the next planning step.

For example, when searching for an object in a large indoor scene, the planner may call Qwen-RobotNav in `ObjNav` mode with a larger token budget and history-covering frame sampling. After a candidate object is observed, the planner can switch to `Tracking` or local `PointNav` behavior with a more recency-focused configuration. This illustrates the main role of the agent-facing interface: it allows the upper-level planner to compose long-horizon behavior from repeated, configurable calls to the same navigation foundation model.

3.3 Navigation Harness: Trajectory Evidence and Context Compression

The interface above controls how Qwen-RobotNav is called. The remaining question is how the result of each navigation call is returned to the planner. A raw Qwen-RobotNav rollout contains dense egocentric observations, sampled history frames, low-level control traces, and a predicted waypoint trajectory. Passing this full stream back into the planner dialogue would quickly exhaust the context budget. On the other hand, returning only a success flag would discard the evidence needed for future planning. We therefore use a lightweight trajectory-to-evidence interface.

For each navigation call, the harness converts the executed rollout into a compact evidence record:

```
{
  subgoal: "Search the kitchen area for a mug",
  task_mode: ObjNav,
  config: Φ_i (main controls: B, γ, m),
  progress: "entered kitchen, checked countertop and dining table",
  salient: ["sink", "countertop", "round table", "no mug observed"],
  outcome: "target not found",
  key_frames: [#18, #31]
}
```

This record is not intended to replace visual evidence. Instead, it acts as a source-indexed summary of what happened during the navigation segment. The textual fields tell the planner what was attempted, where the agent moved, what was observed, and how the sub-goal ended. The key-frame identifiers point back to stored visual observations that can be retrieved later if the planner needs to re-inspect a scene or verify a candidate target.

The harness maintains a compact evidence notebook for long-horizon reasoning. The notebook stores durable conclusions such as searched regions, candidate object locations, rejected hypotheses, landmark cues, and layout assumptions. Unlike raw dialogue history, notebook entries are designed to survive

context compression. Later entries may revise earlier beliefs, but the update history remains auditable. A typical entry might be:

> `[step 47]` Kitchen entered and searched; countertop and dining table checked. No mug observed. Corridor shelf remains a possible candidate region from key frame #12.

Together, trajectory evidence and the evidence notebook form a two-level memory. By default, the planner reasons over compact textual records and notebook entries. When text is insufficient, it can retrieve source images through visual recall using the stored key-frame identifiers. This keeps the planner context concise while preserving access to detailed visual evidence.

The harness can also expose auxiliary vision-tool calls, such as object detection, scene understanding, and semantic grounding. These tools support the planner's situational awareness, but they do not replace Qwen-RobotNav as the core navigation model. Their role is to provide additional evidence channels around the same planner–Qwen-RobotNav loop: the planner selects a task mode and configuration, Qwen-RobotNav executes the navigation segment, and the harness returns compact trajectory evidence for the next decision.

In summary, the agentic layer does not introduce a separate navigation policy. Its main function is to make Qwen-RobotNav's existing task modes and observation controls accessible to an upper-level planner, and to make completed rollouts usable for long-horizon reasoning through trajectory evidence, notebook memory, and context compression.

4 Data

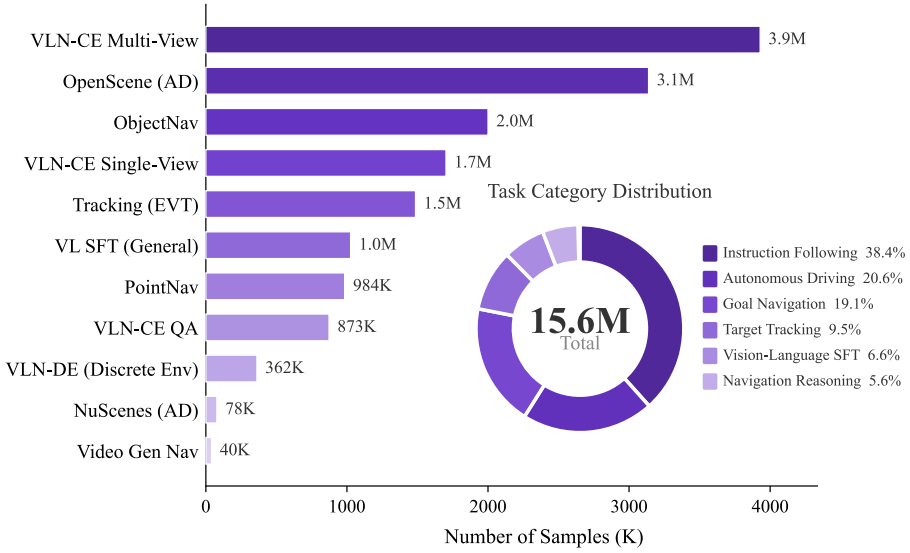


Figure 5: **Training data distribution.** *Left:* Per-dataset sample counts across all navigation trajectory and vision-language sources. *Right:* Aggregated distribution over task categories, totalling 15.6M training samples.

4.1 Navigation Trajectory Planning

A key design principle of Qwen-RobotNav is to train on a deliberately broad spectrum of navigation tasks rather than specialising in any single paradigm. We structure the trajectory planning corpus around four capability dimensions that broaden the operational demands placed on the model: grounding language or geometric guidance into executable motion, exploring partially observed spaces to locate targets (Zheng et al., 2025; 2024; Peng et al., 2025), interacting with dynamic agents whose future states must be anticipated in real time, and planning under high-speed, multi-agent, safety-critical dynamics in open-world environments. These dimensions roughly span increasing environmental uncertainty, temporal dependence, and embodiment diversity, but they are not mutually exclusive in practice. Rather, more complex navigation regimes often reuse simpler competencies while introducing additional perception,

planning, and control requirements. Jointly training across this spectrum encourages the model to develop a shared spatial-planning substrate that generalises beyond any single task family. The full training corpus comprises 15.6M samples; Figure 5 summarises the per-dataset and per-category distributions.

Guided by these dimensions, we select five concrete task families as the instantiation of our training corpus:

- **Instruction following** and **point-goal navigation** jointly ground guided navigation by pairing rich procedural language with compact directional or coordinate-based commands, thereby spanning the full granularity spectrum of task specification.
- **Object-goal navigation** realises the exploration dimension, requiring the agent to build implicit spatial maps and hypothesise target locations without step-by-step instructions.
- **Active target tracking** introduces dynamic interaction, demanding real-time motion anticipation and re-identification of moving persons in crowded scenes.
- **Autonomous driving** realises cross-embodiment planning under high-speed, multi-agent traffic dynamics with stringent safety constraints, an operational regime that shares the same underlying spatial-planning principles as indoor navigation yet demands substantially different perception range, action granularity, and decision latency.

These five tasks are chosen because they are *mutually non-redundant*: each exercises a distinct combination of perception, planning, and control that the others leave undertrained, while together they provide sufficient coverage for these capability dimensions to reinforce one another during joint optimisation.

4.1.1 Instruction Following

Instruction following represents the language-rich form of guided navigation. The agent must interpret natural language instructions alongside egocentric visual observations and plan waypoint trajectories to reach the described destination while adhering to the procedural milestones specified in the instruction. This supervision trains fine-grained language-to-control grounding: spatial relations, landmark references, and temporal ordering in the instruction must be translated into executable motion. We construct **5.63M** instruction-following training samples from two VLN-CE benchmarks (Krantz et al., 2020), building on Matterport3D (Chang et al., 2017) scenes.

VLN-CE R2R (1,491K). The Room-to-Room benchmark (Krantz et al., 2020) provides approximately 10K continuous navigation clips in indoor environments, each paired with a concise goal-oriented instruction. We unroll each ground-truth trajectory with teacher forcing and record RGB observations in both single-camera (front only) and multi-camera (front, left, right, and rear) configurations at every discrete step, yielding 1,491K training samples after data balancing across view configurations, instruction refinement, and image quality enhancement.

VLN-CE RxR (4,140K). The Room-Across-Room benchmark (Ku et al., 2020) significantly expands R2R in both scale and complexity, featuring longer paths, richer spatial relationships, and multilingual instructions with dense landmark references. Following the same teacher-forcing protocol, we extract 4,140K training samples from approximately 20K continuous clips across view configurations and augmentation variants.

All instruction-following samples from both sources undergo the image quality enhancement and instruction refinement pipelines described in Section 4.2.1: simulator renders are transformed into photorealistic images via Qwen-Image-Edit (Wu et al., 2025) style transfer, and each instruction is paraphrased into multiple linguistically diverse variants using an LLM (Yang et al., 2025a); the sample counts above include all resulting augmentation variants.

4.1.2 Point Goal Navigation

Point-goal navigation provides the language-sparse counterpart to instruction following. The agent must reach a specified target given only its current visual observation and a compact task specification, without detailed natural language guidance. Whereas instruction-following data emphasise procedural language grounding, point-goal data isolate geometric path planning, local obstacle avoidance, and smooth goal approach from minimal coordinate or command inputs. We generate **984K** point-goal navigation training samples from Matterport3D (Chang et al., 2017) and HM3D (Ramakrishnan et al., 2021) scenes using the Habitat simulator (Savva et al., 2019), and organise them into a curriculum from primitive motion grounding to long-horizon path search.

Coordinate-based point goals (922K). The agent receives the target position as a numeric coordinate in

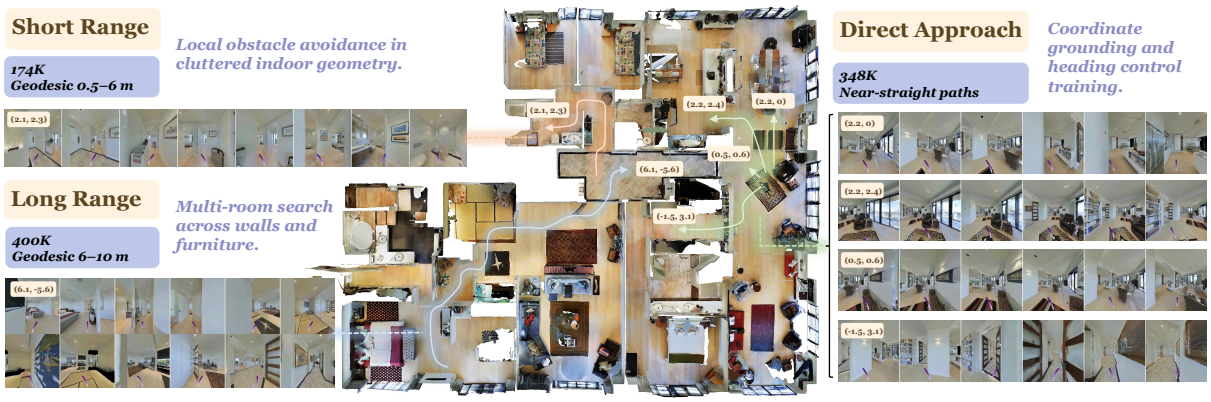


Figure 6: Visualization of the three coordinate-based point-goal navigation categories. **Direct Approach** (348K) targets near-straight paths along canonical egocentric directions. **Short Range** (174K) introduces local obstacle avoidance within cluttered indoor geometry. **Long Range** (400K) requires multi-room path search over extended horizons, navigating around walls and furniture.

its egocentric frame together with its current pose, distance, and bearing angle to the goal. We allocate this subset according to difficulty rather than sampling uniformly.

- **Direct-approach point goals** (348K). Targets are placed along canonical egocentric directions or on an integer grid within $[-5, 5] \times [-5, 5]$ m, and paths whose geodesic-to-euclidean ratio exceeds 1.15 are rejected. This group corresponds to simplified point-of-interest navigation in which the target can be reached by a near-straight trajectory, focusing supervision on basic coordinate grounding, heading control, and smooth approach behaviour.
- **Short-range point goals** (174K; geodesic distance 0.5–6 m). These episodes introduce cluttered indoor geometry and local obstacle avoidance while keeping the destination within a limited spatial neighbourhood. Because they are relatively local and the visual evidence is often sufficient to infer a feasible detour, a moderate sample budget is sufficient.
- **Long-range point goals** (400K; geodesic distance 6–10 m). These episodes receive the largest allocation because they frequently lack direct line of sight to the target and require the agent to search over alternative routes, navigate around walls and furniture, and plan across rooms over extended horizons.

This distribution implements a curriculum in which the model first learns reliable coordinate-to-motion grounding, then local collision avoidance, and finally the more difficult path-searching behaviour needed for non-myopic navigation.

Command-based motion primitives (62K). Instead of explicit coordinates, the agent receives a textual motion primitive. *Parameterised commands* specify both action and magnitude (*e.g.*, "*Move forward 2.0 meters*", "*Turn left 90 degrees*"), while *bare commands* provide only the direction without a numeric target (*e.g.*, "*Move forward*", "*Turn left*"), requiring the model to infer an appropriate displacement from visual context. This subset is intentionally small because such commands primarily anchor basic action semantics, such as moving forward by a specified distance or rotating by a specified angle; extensive repetition yields limited additional path-planning diversity compared with coordinate-based long-range episodes.

Point-goal navigation (PointNav) also serves as a controlled baseline for studying navigation competence because the goal specification is simple and the required behaviour can be evaluated without the confounding effect of complex language. This simplicity makes PointNav particularly suitable for controlled observation-configuration and motion-dynamics augmentation; the corresponding front-view-only training protocol and simulator perturbations are described in Section 4.2.1.

Each training sample contains a uniformly sampled set of history frames, the task specification (coordinates or command), and an 8-waypoint future trajectory. Within 1.5 m of the goal, deceleration trajectories with linearly decreasing step sizes are generated to teach smooth stopping behaviour. Forward steps are subsampled at a 45% inclusion rate to rebalance the action distribution, while turns and stop actions are always retained.

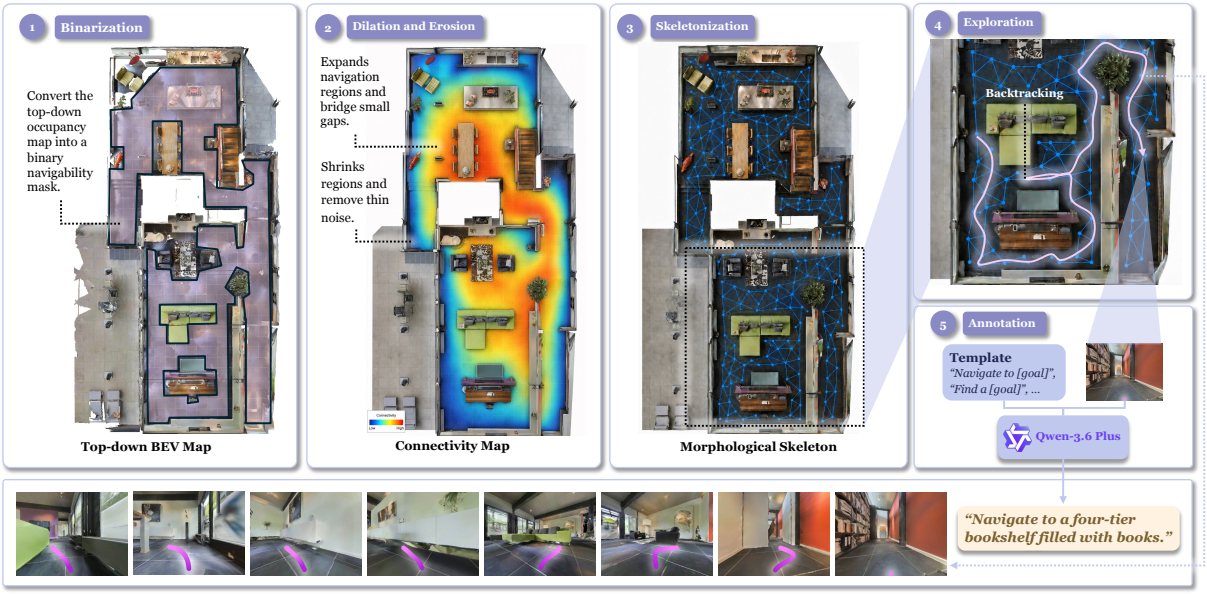


Figure 7: Object-goal navigation data generation pipeline. **(1)** The top-down occupancy map is binarised into a navigability mask. **(2)** Morphological dilation and erosion expand navigable regions and remove thin noise. **(3)** Skeletonisation extracts the medial-axis graph of the navigable space. **(4)** An exploration trajectory is generated by traversing the skeleton with backtracking at dead ends. **(5)** A VLM annotates the goal object at the terminal viewpoint, producing open-vocabulary goal specifications embedded in diverse instruction templates.

4.1.3 Object-Goal Navigation

Object-goal navigation addresses goal-directed exploration under partial observability. The agent receives a semantic object category or open-vocabulary object description, rather than a complete route, and must search the environment until an instance of the target is reached. This setting exercises capabilities that are difficult to induce from route-following data alone: maintaining an implicit memory of explored regions, using scene priors to hypothesise likely object locations, and revising the route when the target is absent from the current field of view.

We construct **2,000K** object-goal navigation training samples from Matterport3D (Chang et al., 2017) and HM3D (Ramakrishnan et al., 2021) reconstructed scenes, including open-vocabulary object annotations from HM3D-OVON (Yokoyama et al., 2024b).

Skeleton-based exploration trajectories. A key challenge is that standard shortest-path followers produce straight-to-goal trajectories that do not reflect realistic search behaviour: in practice, an agent exploring for an unseen object must traverse corridors, inspect rooms, and backtrack from dead ends before locating the target. We therefore generate trajectories using a skeleton-based exploration strategy over the scene's navigable space. Given the simulator's top-down occupancy map, we first extract the largest connected component of the navigable area and then apply morphological skeletonisation (Zhang &Suen, 1984) to reduce it to a one-pixel-wide medial-axis graph. Short spurious branches are pruned to suppress redundant bifurcation points. On this skeleton graph, a goal location is sampled at random, and the agent starts from a random navigable position. The path is planned by traversing the skeleton: at each junction the agent randomly selects a branch; upon reaching a dead end it backtracks and explores an alternative branch, continuing until the branch containing the goal is reached. Because the skeleton lies along the medial axis of navigable regions, the resulting paths maintain a safe distance from walls and obstacles. The raw skeleton path is then smoothed via cubic-spline interpolation: waypoints sampled at regular intervals along the skeleton are fitted with a parametric cubic spline, and the final trajectory points are sampled from this spline at a fixed step size of 0.25 m, producing smooth, physically plausible motion.

VLM-based goal annotation. Each trajectory requires an associated object goal. Rather than relying on a fixed category taxonomy, we adopt a VLM-in-the-loop annotation strategy that produces open-vocabulary goals by construction. At the terminal point of a generated trajectory, the agent is oriented towards a random direction and the resulting egocentric image is submitted to a vision-language model, which is asked to identify a salient, reachable object in the scene. If the VLM confirms the presence of a suitable target, the trajectory is retained and the object name returned by the VLM is used as the

goal specification; otherwise the trajectory is discarded. Because the VLM is free to name any visible object without a predefined label set, the resulting goal vocabulary is inherently open and covers a long tail of everyday objects beyond the fixed categories of standard ObjectNav benchmarks. The goal is then embedded into diverse natural-language instruction templates (*e.g.*, "*navigate to the {goal_object}*", "*find and reach the {goal_object}*") to provide linguistic variation during training. Each training sample contains uniformly sampled history frames, the task specification, and an 8-waypoint future trajectory, following the same format used by the other navigation task families. The same observation-configuration and image-quality augmentations described in Section 4.2.1 are applied, including camera height and field-of-view randomisation.

4.1.4 Target Tracking

Active visual tracking extends the trajectory corpus from static goal reaching to dynamic interaction. The agent must identify a person described by a natural language query (*e.g.*, "*Follow the man in the blue t-shirt*"), maintain pursuit in crowded indoor environments, and keep an appropriate following distance while avoiding obstacles. Compared with object-goal navigation, this setting introduces time-critical requirements including motion anticipation, occlusion handling, target re-identification, and balancing pursuit with safety constraints. We collect **1,486K** training samples from the EVT-Bench dataset (Wang et al., 2025b), which covers diverse indoor scenes populated with hundreds of digital avatars whose appearance and motion are controlled independently. Following the benchmark protocol (Wang et al., 2025b), we focus on the *Single Target Tracking* (STT) split, where a unique target is specified per episode and no distractors are introduced.

Each training sample contains the current egocentric observation, a short history of past frames, the textual target description, and the ground-truth future trajectory. We additionally apply the same image-quality and observation-configuration augmentations used in the instruction-following data to improve cross-scene and cross-sensor generalisation.

4.1.5 Autonomous Driving

Autonomous driving is incorporated as a cross-embodiment source of trajectory supervision rather than as a separate top-level capability class (Hu et al., 2023; Li et al., 2025c; Peng et al., 2026; Liang et al., 2026). It requires an agent to reason over complex, dynamic, and safety-critical traffic environments and to predict feasible future trajectories under diverse road geometries, traffic rules, and agent interactions. Unlike indoor embodied navigation, driving scenarios involve higher-speed motion, structured lane topology, long-range perception, and dense multi-agent dynamics. These differences stress the same spatial-planning substrate under a substantially different operating regime, exposing the model to long-horizon planning, interaction-aware motion prediction, and safety-constrained control. To equip the model with transferable planning capabilities across both embodied and vehicle-centric settings, we incorporate large-scale driving data into training and cast autonomous driving as a unified waypoint prediction problem conditioned on multimodal observations. We construct the autonomous driving training set from two complementary sources: nuScenes (78K) (Caesar et al., 2020) and OpenScene (3,138K) (Contributors, 2023). Together, these datasets provide diverse urban scenes, rich sensor observations, map-aware traffic contexts, and a broad spectrum of driving behaviours ranging from routine lane following to complex maneuvers such as turning, merging, yielding, and obstacle avoidance. By integrating them into a common trajectory-planning format, we obtain approximately **3.2M** autonomous-driving trajectory-supervision instances. Here, the reported number refers to the total number of annotated supervision variants rather than the number of distinct raw driving trajectories. Specifically, a single underlying trajectory may be instantiated into multiple conditioning variants, depending on whether navigation instructions, ego-state information, and historical ground-truth trajectory priors are provided. This design allows the same driving behavior to be observed under different levels of prior information, thereby encouraging the model to learn trajectory planning that is robust to heterogeneous inputs and transferable across different settings.

Each supervision instance is centered on predicting the ground-truth future trajectory at the current frame, conditioned on multimodal observations and a configurable set of auxiliary priors. All instances include multi-view camera observations, while different annotation variants may additionally provide navigation instructions, the current ego-vehicle state, and/or a short history of past ground-truth trajectories. Thus, the autonomous-driving data is not treated as a single fixed-input prediction task, but as a family of trajectory-planning tasks with different available context. This unified yet flexible representation enables the model to reason over visual scene context, vehicle motion, temporal dynamics, and high-level route intent when available, while also learning to remain effective when some priors are absent.

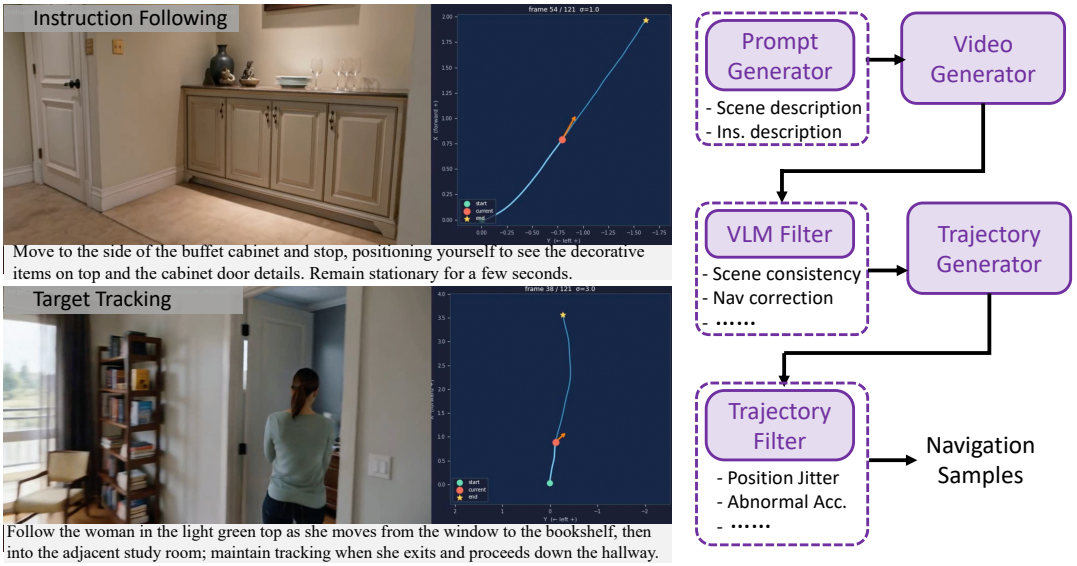


Figure 8: **Autogenerated navigation data pipeline.** *Right:* A large language model first generates paired video prompts and navigation instructions; a text-to-video model then synthesises first-person egocentric videos, which are filtered by a vision-language model for quality before a monocular depth-and-pose estimator extracts 2-D trajectories; a final kinematic filter removes physically implausible samples. *Left:* Two example outputs covering instruction following (top) and target tracking (bottom), each showing a generated video frame alongside the recovered bird's-eye-view trajectory.

4.2 Autogenerated Navigation Data with a Video Generator

While simulator-derived trajectories provide precise ground-truth supervision, they are constrained by the availability of reconstructed 3-D assets and exhibit a persistent visual domain gap relative to real-world observations. To complement the simulator corpus with diverse, photorealistic training data that does not require 3-D scene reconstruction, we introduce a fully automated pipeline that converts text-to-video (T2V) generations into navigation trajectory samples. The pipeline covers two core capabilities, *instruction following* and *target tracking*, and produces **40K** training samples through five sequential stages illustrated in Figure 8.

- **Stage 1: Prompt and instruction generation.** A large language model generates paired outputs for each sample: a detailed video prompt describing a first-person navigation scenario, and a corresponding natural-language navigation instruction. Prompts are drawn from a scene-complexity matrix spanning 12 scene categories (*e.g.*, home, office, shopping mall, hospital, park, street) and 7 interaction complexities (*e.g.*, obstacle avoidance, target occlusion and reappearance, crowd navigation, sudden direction changes), yielding 29 distinct combinations that ensure broad environmental and behavioural coverage.
- **Stage 2: Text-to-video synthesis.** Each video prompt is fed to a T2V model, which renders a first-person egocentric navigation video of approximately five seconds. The resulting clips depict the agent moving through the described scene with realistic lighting, textures, and dynamic elements such as pedestrians, without requiring any 3-D assets or physics simulation.
- **Stage 3: VLM-based video quality filtering.** A vision-language model evaluates every generated video against capability-specific quality dimensions. For instruction-following videos the assessment covers scene consistency, navigation correctness, goal arrival, stopping behaviour, motion continuity, and collision avoidance; target-tracking videos are additionally evaluated on target visibility, pursuit behaviour, and identity consistency across frames. Only videos that pass all applicable criteria are retained.
- **Stage 4: Trajectory extraction.** A monocular depth-and-pose estimation model recovers per-frame camera-to-world poses from each retained video. The estimated poses are converted from camera coordinates into the robot-centric ground-plane frame, producing a 2-D trajectory $[\,x,\ y,\ \text{yaw}\,]$ per frame, where x and y denote forward and lateral displacement in metres and yaw is the heading angle in radians, all relative to the first frame.
- **Stage 5: Trajectory quality filtering.** A rule-based kinematic filter removes trajectories that exhibit physically implausible characteristics, including near-zero total displacement, excessive positional or

Figure 9: Visual comparison between original Habitat simulator renders (top) and Qwen-Image-Edit (Wu et al., 2025) (bottom).

heading jitter, single-step teleportation, abnormal acceleration, and high-frequency noise detected via spectral analysis. Each check uses a tunable threshold, and a trajectory is discarded if it triggers any single criterion.

The resulting dataset provides trajectory supervision grounded in photorealistic, T2V-generated imagery across a wide variety of real-world-like scenes, complementing the simulator-derived corpus with greater visual diversity and reduced domain gap.

4.2.1 Navigation Data Augmentation

To enhance robustness and data diversity, we apply task-dependent augmentation strategies to simulator-derived navigation data. Instruction refinement is restricted to language-conditioned trajectories, whereas visual refinement, observation-configuration perturbation, and speed variation are applied to simulator-derived trajectory samples when supported by the source data.

- **Instruction refinement.** We employ an LLM-based paraphrasing pipeline (Yang et al., 2025a) that first deduplicates instructions by trajectory identifier and then generates multiple paraphrased variants per unique instruction. The generation prompt instructs the model to preserve all spatial directions and relative landmarks, vary landmark nouns with synonyms or hypernyms, diversify sentence structure and action verbs, and correct any grammatical noise, so that each variant reads as if written by a different annotator. Three variants are generated per unique instruction, effectively tripling the linguistic diversity of the corpus.
- **Image quality enhancement.** The sim-to-real visual gap remains a major bottleneck for transferring simulation-trained policies to real environments. We build a scalable refinement pipeline based on Qwen-Image-Edit (Wu et al., 2025), which applies prompt-guided diffusion-based style transfer to every rendered observation. Given a source image and a natural-language editing prompt (*e.g.*, "*Convert the rendered image into a photorealistic photograph while preserving spatial layout*"), the model generates a visually refined version that retains the original geometric layout while exhibiting more realistic textures, lighting, and material appearance.
- **Observation and camera augmentation.** Camera height is uniformly sampled in $[0.5, 1.5]$ m, horizontal field of view (HFoV) in $[90°, 120°]$, and image aspect ratio between 2:1 and 4:3, so the model learns to navigate under diverse sensor configurations without task-specific retraining. For PointNav, we additionally perturb the robot's initial heading and egocentric viewpoint, and explicitly construct front-view-only variants in which the model receives only the front camera observation rather than panoramic inputs. This setting matches deployment scenarios where only a forward-facing camera is available.
- **Speed augmentation.** We generate trajectory data at multiple speed regimes to expose the model to diverse motion dynamics. The *standard-speed* variant uses the default Habitat action discretisation

(0.25 m forward steps, 15° turns). The *low-speed* variant replays the same planned trajectories at finer temporal granularity with randomised sub-step sizes (0.05 to 0.25 m), while PointNav trajectories are further replayed under varied motion scales to improve robustness across robot actuation speeds.

4.3 Vision Language Data

Robust action prediction rests on perceptual and reasoning capabilities that cut across all navigation task families and cannot be acquired reliably from trajectory supervision alone. Recognising objects in cluttered scenes, reading text on signs, reasoning about spatial layouts, comparing multiple views, and interpreting novel visual contexts are prerequisites for navigation in unseen environments. We co-train navigation trajectories with approximately **1.0M** general vision-language samples and **873K** navigation-specific reasoning samples, organised into two complementary groups that collectively maintain and strengthen the perceptual substrate required for the VLM-to-VLA transfer.

General vision-language data (∼1.0M). We incorporate training samples spanning eight complementary categories: visual question answering (∼669K; VQAv2 (Goyal et al., 2017), DVQA (Kafle et al., 2018), FigureQA (Kahou et al., 2018), CLEVR (Johnson et al., 2017), *etc*.), image captioning (∼6K), visual grounding (∼178K; RefCOCO (Kazemzadeh et al., 2014), COCO (Lin et al., 2014), Objects365 (Shao et al., 2019)), instruction following (∼30K), multi-image reasoning and comparison (∼38K), general visual question answering (∼76K), object and landmark recognition (∼16K), and STEM problem solving (∼14K). Rather than serving merely as a regulariser against catastrophic forgetting, these samples preserve and extend the in-the-wild visual understanding that underpins downstream action prediction—object and landmark recognition, spatial reasoning, text parsing, and multi-image analysis. All samples follow a unified conversation format and are co-trained with navigation data, so that this perceptual substrate remains intact as the model acquires embodied competence.

Navigation-specific reasoning (873K). Navigation reasoning is inherently domain-specific: deciding whether to turn left at a hallway intersection demands spatial understanding that differs qualitatively from general visual question answering. We extend NavGPT-2 (Zhou et al., 2024a) and construct two complementary reasoning formats from existing VLN trajectory data. *Free-form QA* targets key decision points within each trajectory, the start step, the final step, and transition points where the dominant action category switches between forward motion and turning. Each sample's 8-step future trajectory is classified into one of twelve fine-grained action classes (six pure turns: slight, medium, and large in both left and right directions: four forward-with-turn combinations, straight-ahead movement, and stop) and cast as a question–answer pair, training the model to articulate spatial reasoning in natural language before committing to actions. *Structured multi-perspective reasoning* extends this idea to multi-view contextual analysis. For each selected sample, the agent's historical front-view observations (up to eight uniformly sampled frames) and current four-view panoramic images (front, right, back, left) are combined with the navigation instruction, ground-truth action label, and trajectory statistics into a structured prompt. A vision-language model then generates a structured reasoning chain comprising four components:

- **History reasoning.** A first-person narrative summarising the journey so far based on historical views.
- **Scene analysis.** A description of what is visible in each of the four current views.
- **Instruction progress.** An assessment of completed and remaining sub-goals relative to the original instruction.
- **Action reasoning.** A concise reasoning chain concluding with the predicted action and a confidence score.

Each sample is subsequently decomposed into four independent question–answer pairs, yielding separate fine-tuning signals for history comprehension, scene understanding, progress tracking, and action prediction. By requiring the model to reconstruct its trajectory context, analyse the current scene, and assess instruction progress before deriving an action, both formats enforce a systematic reasoning process that serves as an inductive bias for the VLM-to-VLA transfer: the language-mediated reasoning, once internalised, acts as a transferable scaffold that improves both QA accuracy and the quality of predicted action trajectories.

Discrete multi-round navigation data (362K). To complement the continuous-environment trajectory data, we incorporate approximately 362K training samples derived from discrete VLN datasets by re-formulating graph-based navigation trajectories into a multi-round, multi-image conversation format. Following the VLN-MME evaluation framework (Zhao et al., 2025), each navigation step is formulated as a single-step action prediction task cast in a multiple-choice question format: the agent receives four perspective-view images (front, right, back, left) extracted from the panoramic observation rendered by the Matterport3D Simulator (Chang et al., 2017), together with a set of candidate next-viewpoints

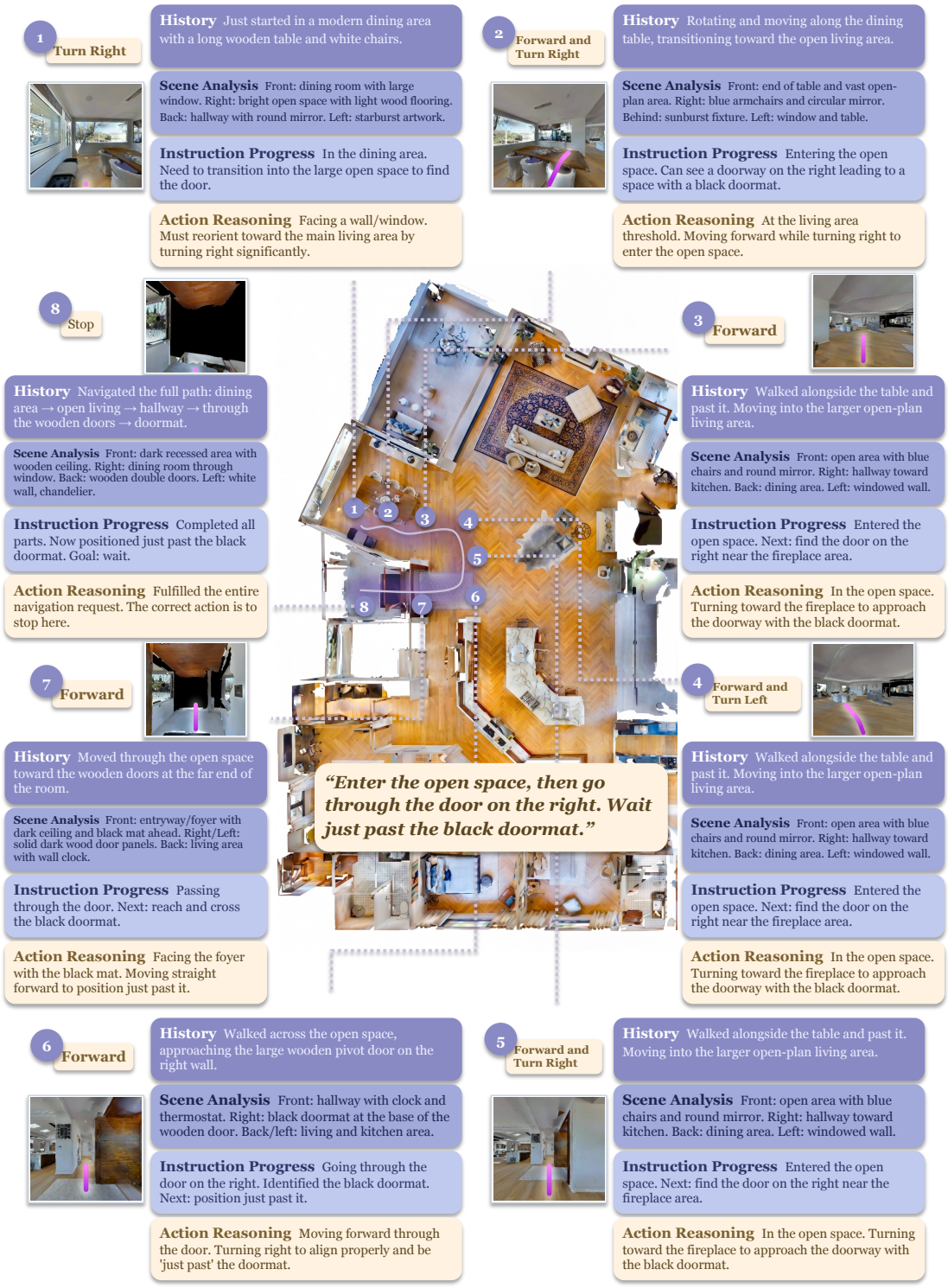


Figure 10: Visualization of structured multi-perspective reasoning along a complete navigation trajectory. Given the instruction "*Enter the open space, then go through the door on the right. Wait just past the black doormat*," the agent executes eight sequential steps (numbered on the floor plan). At each step, a four-component reasoning chain is produced: **History** summarises the journey so far, **Scene Analysis** describes the current multi-view observations, **Instruction Progress** tracks completed and remaining sub-goals, and **Action Reasoning** derives the next action. This structured decomposition is used to generate navigation-specific reasoning supervision for VLM-to-VLA transfer.

annotated with visual markers as answer options, and selects the correct action among them. Rather than rendering observations in the Habitat simulator, we use the Matterport3D Simulator, which produces higher-definition photorealistic panoramic views directly from the original 3-D scans. Unlike

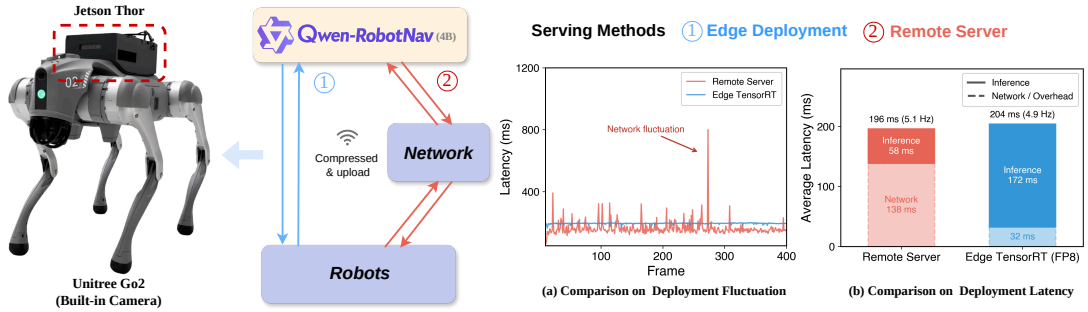


Figure 11: Deployment architecture and latency comparison of Qwen-RobotNav-4B on Unitree Go2. Left: remote-server and edge deployment pipelines, where edge inference runs on Jetson Thor with FP8 quantization and TensorRT acceleration. Right: latency evaluation, including (a) per-frame latency fluctuations and (b) average latency breakdown for both deployments.

the continuous-environment action data, which are decomposed into independent single-step samples, this discrete formulation allows each trajectory to be maintained as a complete multi-turn dialogue, preserving the full decision history within the conversation context and enabling the model to leverage prior turns for spatial reasoning without requiring an explicit history encoding module. Through this pipeline, we incorporate additional VLN datasets beyond R2R and RxR: CVDN (Thomason et al., 2020), which introduces dialog-conditioned navigation requiring multi-turn linguistic grounding; SOON (Zhu et al., 2021), which targets situated object-oriented navigation with fine-grained spatial goal specifications; REVERIE (Qi et al., 2020), which pairs remote referring expressions with object localisation; and SRDF (Wang et al., 2025c), which bootstraps large-scale augmented trajectories via a self-refining data flywheel across diverse indoor environments. This approach enables us to significantly scale the number of training trajectories and their linguistic diversity while preserving the multi-turn reasoning capabilities that single-step formulations sacrifice.

5 Experiments

5.1 Deployment

For real-world deployment, we explore both cloud-based and on-device inference for Qwen-RobotNav. In the cloud setting, the robot uploads compressed observations and text instructions over the network to a remote server for trajectory prediction, and receives the planned actions for execution. This setup supports larger models with strong inference capability and faster server-side inference, but introduces communication overhead and relies on network stability. In the on-device setting, inference runs directly on the robot via an NVIDIA Jetson Thor, eliminating transmission delay and improving robustness, at the cost of being constrained by onboard compute and bandwidth resources.

Taking network transmission into account, the remote-server deployment still achieves lower average end-to-end latency than on-device inference, with 196 ms (5.1 Hz) versus 204 ms (4.9 Hz). The trade-off is higher sensitivity to network conditions, leading to larger latency variance and occasional spikes. By eliminating network transmission, on-device inference provides more consistent latency and is therefore more robust for latency-sensitive tasks such as tracking. Overall, remote-server inference is faster on average, whereas on-device inference is more stable and reliable in real-world deployment.

5.2 Evaluation

We evaluate Qwen-RobotNav on three standard embodied navigation benchmarks: vision-and-language navigation in continuous environments (VLN-CE), open-vocabulary object-goal navigation (OVON), and active visual tracking (EVT-Bench). Results are compared against two recent navigation foundation model baselines (NavFoM (Zhang et al., 2025a) and ABot-N0 (AMAP CV Lab, 2025)) as well as established specialist methods.

5.2.1 Vision-and-Language Navigation

VLN-CE. Table 1 reports VLN-CE R2R and RxR Val-Unseen results under both monocular and panoramic observation settings. Under the panoramic setting, Qwen-RobotNav-8B achieves **72.1%** SR and **66.6%**

Table 1: **VLN-CE Val-Unseen results** under monocular and panoramic observation settings. **Bold**: best within each setting; underline: second best.

Method	R2R Val-Unseen				RxR Val-Unseen			
	NE↓	OS↑	SR↑	SPL↑	NE↓	nDTW↑	SR↑	SPL↑
Monocular								
NaVid (Zhang et al., 2024a)	5.72	49.2	41.9	36.5	5.72	–	45.7	38.2
Uni-NaVid (Zhang et al., 2025b)	5.58	53.3	47.0	42.7	6.24	–	48.7	40.9
NaVILA (Cheng et al., 2025)	5.22	62.5	54.0	49.0	6.77	58.8	49.3	44.0
StreamVLN (Wei et al., 2025b)	4.98	64.2	56.9	51.9	6.22	61.9	52.9	46.0
DualVLN (Wei et al., 2025a)	**4.05**	70.7	64.3	58.5	4.58	**70.0**	61.4	51.8
InternVLA-N1 (Cai et al., 2025)	4.83	63.3	58.2	54.0	5.91	65.3	53.5	46.1
Qwen-RobotNav-4B	4.22	**73.6**	**66.9**	**60.5**	**4.15**	68.6	71.3	61.5
Qwen-RobotNav-8B	4.36	72.7	65.7	59.6	4.16	69.9	**73.4**	**63.5**
Panoramic								
NavFoM (Zhang et al., 2025a)	4.61	72.1	61.7	55.3	4.74	65.8	64.4	56.2
ABot-N0 (AMAP CV Lab, 2025)	3.78	70.8	66.4	63.9	3.83	–	69.3	60.0
OmniNav (Hu et al., 2025)	3.74	74.6	69.5	66.1	3.77	–	73.6	62.0
AstraNav-World (Hu et al., 2026)	3.86	73.9	67.9	65.4	3.82	–	72.9	61.5
Qwen-RobotNav-4B	3.80	77.2	69.5	63.6	3.80	71.9	75.2	65.0
Qwen-RobotNav-8B	**3.53**	**78.5**	**72.1**	**66.6**	**3.58**	**72.5**	**76.5**	**65.7**

Table 2: **Navigation foundation model comparison on VLNVerse (Lin et al., 2025) test split.** TL: Trajectory Length; NE: Navigation Error; SR: Success Rate; OSR: Oracle Success Rate; SPL: Success weighted by Path Length. **Bold**: best; underline: second best.

Method	Fine-grained					Coarse-grained				
	TL	NE↓	OSR↑	SR↑	SPL↑	TL	NE↓	OSR↑	SR↑	SPL↑
InternVLA-N1 (Cai et al., 2025)	9.23	5.68	38.69	28.95	25.00	4.00	5.71	23.32	17.51	16.54
Uni-NaVid (Zhang et al., 2025b)	12.35	4.11	67.02	45.74	26.91	10.33	5.05	42.70	29.68	13.47
NavFoM (Zhang et al., 2025a)	8.58	4.13	69.68	51.59	32.40	6.52	4.93	45.93	38.02	23.15
Qwen-RobotNav-4B	7.99	3.05	70.36	62.61	56.22	7.39	4.40	51.03	41.25	37.37
Qwen-RobotNav-8B	8.16	**3.00**	**72.81**	**63.75**	**57.93**	7.68	**4.08**	**55.97**	**46.59**	**41.54**

SPL on R2R, surpassing NavFoM by 10.4% SR and ABot-N0 by 5.7% SR. On the longer-horizon RxR benchmark, Qwen-RobotNav-8B reaches **76.5%** SR and **72.5** nDTW, outperforming NavFoM by 12.1% SR and ABot-N0 by 7.2% SR. Under the monocular setting, Qwen-RobotNav remains competitive despite the significantly reduced field of view. Qwen-RobotNav-4B achieves 66.9% SR and 60.5% SPL on R2R, surpassing the strongest monocular baseline DualVLN by 2.6% SR and 2.0% SPL. On RxR, Qwen-RobotNav-8B reaches 73.4% SR and 63.5% SPL, outperforming DualVLN by 12.0% SR and 11.7% SPL, demonstrating particularly strong gains on long-horizon instructions. The consistent improvements across both observation settings validate the effectiveness of task-adaptive token allocation and the generality of Qwen-RobotNav: it delivers state-of-the-art performance regardless of the input modality, whether operating with a single forward-facing camera or a full panoramic observation.

VLNVerse. Table 2 reports results on VLNVerse (Lin et al., 2025), a large-scale benchmark that unifies previously fragmented VLN tasks into a single evaluation framework and replaces teleporting "ghost" agents with full-kinematics embodied locomotion driven by a physics engine. We evaluate under both fine-grained and coarse-grained instruction settings, thereby probing whether an agent can follow detailed step-by-step guidance as well as interpret high-level goal descriptions. Qwen-RobotNav-8B achieves **63.75%** SR and **57.93%** SPL on the fine-grained split, surpassing NavFoM by 12.2% SR and 25.5% SPL, and Uni-NaVid by 18.0% SR and 31.0% SPL. On the more challenging coarse-grained split, where instructions omit procedural detail and require the agent to infer intermediate waypoints, Qwen-RobotNav-8B reaches **46.59%** SR and **41.54%** SPL, outperforming NavFoM by 8.6% SR and 18.4% SPL. Notably, the SPL margins are substantially larger than the SR margins in both settings, indicating that Qwen-RobotNav not only reaches the goal more often but follows markedly more efficient paths. The 4B variant also surpasses all baselines, confirming that the performance advantage is not merely a consequence of model scale.

VLN-PE. Table 3 further evaluates on VLN-PE (Wang et al., 2025a) using the flash controller, which decouples high-level navigation planning from low-level locomotion control and thereby provides a more direct assessment of spatial reasoning quality. Qwen-RobotNav-8B achieves **65.50%** SR and **61.19%**

Table 3: **VLN-PE results with flash controller (R2R Val-Unseen) (Wang et al., 2025a).** TL: Trajectory Length; NE: Navigation Error; FR: Fall Rate; OS: Oracle Success Rate; SR: Success Rate; SPL: Success weighted by Path Length. **Bold**: best; underline: second best.

Method	TL	NE↓	FR↓	OS↑	SR↑	SPL↑
Seq2Seq (Krantz et al., 2020)	19.24	8.27	**0.22**	43.05	15.74	9.70
CMA (Krantz et al., 2020)	40.21	31.24	**0.22**	45.06	20.94	14.06
RDP (Wang et al., 2025a)	15.12	6.98	0.30	42.54	24.94	17.54
InternVLA-N1 (Cai et al., 2025)	10.11	4.13	0.45	67.63	60.36	54.93
Qwen-RobotNav-4B	10.39	4.24	3.83	72.70	60.28	55.24
Qwen-RobotNav-8B	9.17	**3.73**	4.05	**72.99**	**65.50**	**61.19**

Table 4: **Closed-vocabulary object-goal navigation on MP3D and HM3Dv2.** †: uses depth or odometry. **Bold**: best; underline: second best.

Method	MP3D		HM3Dv2	
	SR↑	SPL↑	SR↑	SPL↑
VLFM† (Yokoyama et al., 2024a)	36.4	17.5	52.5	30.4
OpenFMNav† (Kuang et al., 2024)	37.2	15.7	52.5	24.1
SG-Nav† (Yin et al., 2024)	40.2	16.0	54.0	24.9
TriHelper† (Zhang et al., 2024b)	–	–	56.5	25.3
WMNav† (Nie et al., 2025)	45.4	17.2	58.1	31.2
CogNav† (Cao et al., 2024)	46.6	16.1	72.5	26.2
Uni-NaVid (Zhang et al., 2025b)	–	–	73.7	**37.1**
Qwen-RobotNav-4B	**52.2**	16.0	**75.6**	30.6
Qwen-RobotNav-8B	48.8	**17.7**	71.2	33.0

SPL, surpassing InternVLA-N1 by 5.1% SR and 6.3% SPL, while also attaining the lowest navigation error (**3.73** m) and the highest oracle success rate (**72.99%**) among all methods. Qwen-RobotNav-4B performs comparably to InternVLA-N1 in SR (60.28% vs. 60.36%) but achieves higher oracle success (72.70% vs. 67.63%), suggesting stronger goal-proximate planning even at smaller model capacity.

5.2.2 Object-Goal Navigation

MP3D & HM3Dv2. Table 4 reports closed-vocabulary ObjectNav results on MP3D and HM3Dv2. Notably, most prior methods in this space rely on depth sensors or odometry (marked with †), whereas Qwen-RobotNav operates from RGB observations alone. Despite this disadvantage, Qwen-RobotNav-4B achieves **52.2%** SR on MP3D, surpassing all depth-based baselines including CogNav (46.6%) and WMNav (45.4%). On HM3Dv2, Qwen-RobotNav-4B reaches **75.6%** SR with a distance-to-goal of only **1.72 m**, surpassing even the vision-only Uni-NaVid (73.7%) and establishing a new state of the art on this benchmark. The 8B variant achieves the best SPL on MP3D (**17.7%**) and competitive SPL on HM3Dv2 (33.0%), indicating more efficient paths at larger model scale.

HM3D-OVON. Table 5 reports open-vocabulary results on HM3D-OVON, where the agent must locate objects described by free-form category names rather than a fixed label set. Qwen-RobotNav-4B achieves **57.7%** / **60.1%** / 53.1% SR on the Seen / Synonyms / Unseen splits, attaining the best success rate on two of three splits and ranking a close second on Unseen (53.1% vs. ABot-N0's 54.0%). Importantly, Qwen-RobotNav uses only a single forward-facing camera in this evaluation, whereas ABot-N0 consumes panoramic multi-view observations that provide 360° scene coverage and substantially reduce the need for active exploration—yet Qwen-RobotNav still surpasses it on both Seen and Synonyms by 2.4% and 4.7% SR respectively. The 8B variant also outperforms ABot-N0 on these two splits while achieving higher SPL (28.5% / 28.8%), reflecting more efficient goal approach at larger model capacity. The lower SPL of Qwen-RobotNav relative to NavFoM and ABot-N0 reflects a reach-first exploration behaviour: the skeleton-based training trajectories encourage thorough room-by-room search, which improves goal-finding rate but at the cost of longer paths.

5.2.3 Active Visual Tracking

Table 6 reports single-target tracking performance on EVT-Bench. Qwen-RobotNav achieves the highest tracking rates among all methods (**90.0%** TR for 4B and 89.7% TR for 8B), surpassing ABot-N0 (87.6%)

Table 5: **Open-vocabulary object-goal navigation on HM3D-OVON (Yokoyama et al., 2024b).** †: uses depth or odometry. **Bold**: best; underline: second best.

Method	Seen		Synonyms		Unseen	
	SR↑	SPL↑	SR↑	SPL↑	SR↑	SPL↑
VLFM† (Yokoyama et al., 2024a)	35.2	18.6	32.4	17.3	35.2	19.6
DAgRL+OD† (Yokoyama et al., 2024b)	38.5	21.1	39.0	21.4	37.1	19.8
MTU3D† (Zhu et al., 2025)	55.0	23.6	45.0	14.7	40.8	12.1
Uni-NaVid (Zhang et al., 2025b)	41.3	21.1	43.9	21.8	39.5	19.8
NavFoM (Zhang et al., 2025a)	40.1	27.1	45.4	32.6	45.2	**31.9**
ABot-N0 (AMAP CV Lab, 2025)	55.3	**32.1**	55.4	**33.2**	**54.0**	30.5
Qwen-RobotNav-4B	**57.7**	24.4	**60.1**	25.1	53.1	20.9
Qwen-RobotNav-8B	56.1	28.5	57.8	28.8	51.2	24.0

Table 6: **Active visual tracking on EVT-Bench (Wang et al., 2025b) (Single Target split, single-view).** SR: Success Rate; TR: Tracking Rate; CR: Collision Rate. †: uses GroundingDINO; ‡: uses SoM+GPT-4o. **Bold**: best; underline: second best.

Method	TR↑	CR↓	SR↑
IBVS† (Gupta et al., 2016)	56.2	3.75	42.9
PoliFormer† (Zeng et al., 2024)	15.5	40.1	4.67
EVT (Zhong et al., 2024)	39.1	42.5	24.4
EVT‡ (Zhong et al., 2024)	49.9	40.5	32.5
Uni-NaVid (Zhang et al., 2025b)	39.5	41.9	25.7
TrackVLA (Wang et al., 2025b)	78.6	**1.65**	85.1
TrackVLA++ (Liu et al., 2025)	81.0	2.10	86.0
NavFoM (Zhang et al., 2025a)	80.5	–	85.0
ABot-N0 (AMAP CV Lab, 2025)	87.6	8.54	**86.9**
Qwen-RobotNav-4B	**90.0**	6.40	77.4
Qwen-RobotNav-8B	89.7	5.70	78.6

by 2.4%, NavFoM (80.5%) by 9.5%, and the dedicated tracker TrackVLA++ (81.0%) by 9.0%. Qwen-RobotNav-8B also achieves the lowest collision rate among generalist models (5.70% CR), substantially below ABot-N0 (8.54%). However, the success rate of Qwen-RobotNav (77.4% / 78.6%) trails that of ABot-N0 (86.9%) and TrackVLA++ (86.0%), which are specifically optimised for tracking tasks. We hypothesise that the broader multi-task training of Qwen-RobotNav introduces a trade-off where the model maintains tighter following behaviour (superior TR) while being more conservative in declaring episode success.

5.3 Embodied Question Answering

Table 7 summarises embodied question answering results on HM-EQA, MT-HM3D, and EXPRESS-Bench. Our Qwen-RobotNav-based variants outperform prior methods across all three benchmarks. In particular, **Qwen3.6-Plus+Qwen-RobotNav** achieves the best overall performance, reaching **76.7** SR on HM-EQA, **54.4** SR on MT-HM3D, and **79.27** LLM Score on EXPRESS-Bench, consistently surpassing recent strong baselines such as FAST-EQA and Memory-EQA. Compared with FAST-EQA, Qwen3.6-Plus+Qwen-RobotNav improves absolute performance by 7.5 points on HM-EQA, 3.9 points on MT-HM3D, and 10.57 points on EXPRESS-Bench. It also reduces normalized equivalent steps on HM-EQA and MT-EQA and improves EXPRESS-Bench E_{path} by 4.71 points, indicating that the learned executor supports more targeted physical exploration. These results suggest that the multimodal reasoning and navigation capabilities of Qwen-RobotNav transfer effectively to embodied question answering, enabling both stronger scene exploration and more accurate answer prediction. Furthermore, the substantial improvement in navigation efficiency achieved by Qwen-RobotNav enables our system to reach target locations using significantly fewer equivalent steps. This further demonstrates its viability as an effective physical-world tool for general-purpose multimodal agents.

Table 7: **Performance comparison on embodied question answering benchmarks.** Results are compared against prior state-of-the-art methods on HM-EQA, MT-HM3D, and EXPRESS-Bench. *: result reported by prior work. †: results are on the full A-EQA split. **Bold**: best; underline: second best.

Method	HM-EQA		MT-EQA		EXPRESS-Bench	
	Acc. (↑)	Steps (↓)	Acc. (↑)	Steps (↓)	LLM Score (↑)	E_{path} (↑)
Explore-EQA (Ren et al., 2024)	58.4	0.52	36.2*	0.64	–	–
Graph-EQA (Saxena et al., 2024)	63.5	0.20	45.6*	0.45	–	–
Memory-EQA (Li et al., 2026a)	61.4	0.40	43.1	0.41	–	–
Fine-EQA (Jiang et al., 2025)	56.0	0.54	–	–	63.95	25.58
3D-Mem (Yang et al., 2025b)	50.4	0.63	–	–	–	–
FAST-EQA (Zhang et al., 2026)	69.2	0.65	50.5	0.52	68.7	29.25
Qwen3.5-Plus+QwenNav-8B	74.1	0.17	52.1	0.22	77.66	31.73
Qwen3.6-Plus+QwenNav-8B	**76.7**	**0.15**	**54.4**	**0.19**	**79.27**	**33.96**

Table 8: **Performance comparison on NAVSIM navtest using closed-loop metrics.** NC: Navigation Compliance; DAC: Drivable Area Compliance; TTC: Time-to-Collision; Comf.: Comfort; EP: Ego Progress; PDMS: PDM Score. †: uses additional LiDAR information. ‡: our model evaluated without historical ego-status prior. **Bold**: best; underline: second best.

Method	NC↑	DAC↑	TTC↑	Comf.↑	EP↑	PDMS↑
Human	100	100	100	99.9	87.5	94.8
Constant Velocity	68.0	57.8	50.0	100	19.4	20.6
Ego Status MLP	93.0	77.3	83.6	100	62.8	65.6
UniAD (Hu et al., 2023)	97.8	91.9	92.9	100	78.8	83.4
TransFuser† (Chitta et al., 2022)	97.7	92.8	92.8	100	79.2	84.0
PARA-Drive (Weng et al., 2024)	97.9	92.4	93.0	99.8	79.3	84.0
LAW (Li et al., 2024)	96.4	95.4	88.7	99.9	81.7	84.6
DRAMA† (Yuan et al., 2024)	98.0	93.1	94.8	100	80.1	85.5
Hydra-MDP++ (Li et al., 2025a)	97.6	96.0	93.1	100	80.4	86.6
DiffusionDrive†	98.2	96.2	94.7	100	82.2	88.1
WoTE† (Li et al., 2025b)	98.5	96.8	94.9	99.9	81.9	88.3
Hydra-NeXt† (Li et al., 2025d)	98.1	97.7	94.6	100	81.8	88.6
VADv2 (Jiang et al., 2024)	98.3	97.4	95.7	100	82.3	89.3
GoalFlow† (Xing et al., 2025)	98.4	98.3	94.6	100	85.0	90.3
DrivingGPT (Chen et al., 2025b)	98.9	90.7	94.9	95.6	79.7	82.4
NavFoM (Zhang et al., 2025a)	97.7	93.5	92.3	100	79.6	84.3
AutoVLA (Zhou et al., 2025b)	98.4	95.6	98.0	99.9	81.9	89.1
ReCogDrive (Li et al., 2025c)	97.9	97.3	94.9	100	**87.3**	90.8
ReflectDrive (Li et al., 2026b)	97.7	**99.3**	93.5	100	86.9	91.1
Qwen-RobotNav-4B‡	96.4	90.9	89.0	99.9	75.2	79.5
Qwen-RobotNav-8B‡	95.9	91.3	88.4	100	75.5	79.5
Qwen-RobotNav-4B	**99.8**	97.5	**98.5**	99.9	84.4	**91.4**
Qwen-RobotNav-8B	**99.8**	96.9	98.2	99.9	84.2	90.9

5.4 Autonomous Driving

NAVSIM. Table 8 reports trajectory planning results with closed-loop metrics on NAVSIM navtest. During evaluation, we provide the ground-truth trajectories of the previous three frames in the prompt as historical priors, allowing the model to condition its prediction on short-term ego-motion context. Qwen-RobotNav achieves strong performance among vision-language driving models. In particular, Qwen-RobotNav-4B reaches **91.4** PDMS, outperforming NavFoM by 7.1 points, AutoVLA by 2.3 points, ReCogDrive by 0.6 points, and ReflectDrive by 0.3 points. It also achieves highly competitive safety-related scores, with **99.8** NC and **98.5** TTC, while maintaining strong DAC and EP performance. Qwen-RobotNav-8B obtains similarly strong results, reaching 90.9 PDMS with **99.8** NC and 98.2 TTC. Compared with the variants evaluated without historical ego-status priors, both models obtain a substantial gain of more than 11 points in PDMS, highlighting the importance of short-term trajectory history for closed-loop driving prediction.

As shown in Figure 12, the qualitative NAVSIM visualization further illustrates the closed-loop behavior of Qwen-RobotNav in a left-turn scenario. The model leverages multi-view observations together

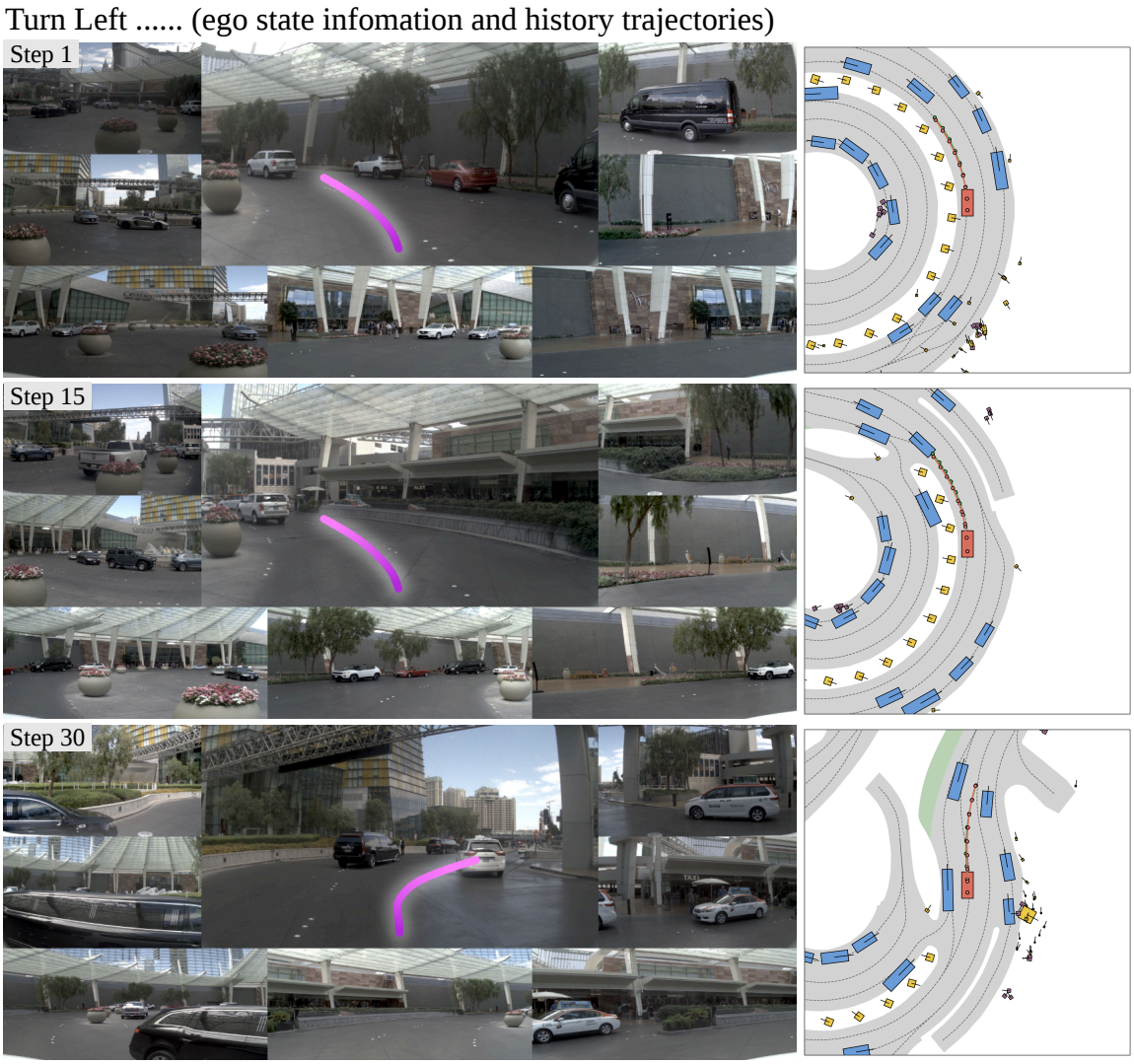


Figure 12: **Qualitative closed-loop planning visualization on NAVSIM.** We visualize a representative left-turn case in an annular road scene. For each timestep, the figure shows the multi-view camera observations, the predicted future trajectory overlaid on the front-view image, and the corresponding BEV scene. Qwen-RobotNav produces temporally consistent curved trajectories from Step 1 to Step 30, progressively completing the left turn while remaining aligned with the drivable lane.

Table 9: **AlpaSim closed-loop evaluation on the PhysicalAI-AV NuRec dataset.** Results are evaluated on 920 scenarios using at-fault closed-loop metrics, where close encounters are counted only when the ego vehicle is deemed responsible, excluding rear-end close encounters. **Bold**: best; underline: second best.

Method	Close Encounter Rate↓ (%)	Off-Road Rate↓ (%)	AlpaSim Score↑
Alpamayo-R1-0.5B	9.0	19.0	0.35
Alpamayo-R1-10B	**4.0**	**16.0**	**0.72**
Qwen-RobotNav-4B	22.0	34.0	0.15
Qwen-RobotNav-8B	22.0	27.0	0.17

with ego-state and historical trajectory information to predict a smooth turning trajectory. Across different timesteps, the predicted path adapts to the changing scene context, follows the curvature of the annular road, and gradually transitions from entering the turn to aligning with the outgoing lane. This demonstrates that Qwen-RobotNav can maintain temporally coherent planning behavior under multi-view driving observations.

AlpaSim. Table 9 presents closed-loop evaluation on the PhysicalAI-AV NuRec dataset using AlpaSim. This benchmark evaluates driving performance with at-fault metrics, where close encounters are counted

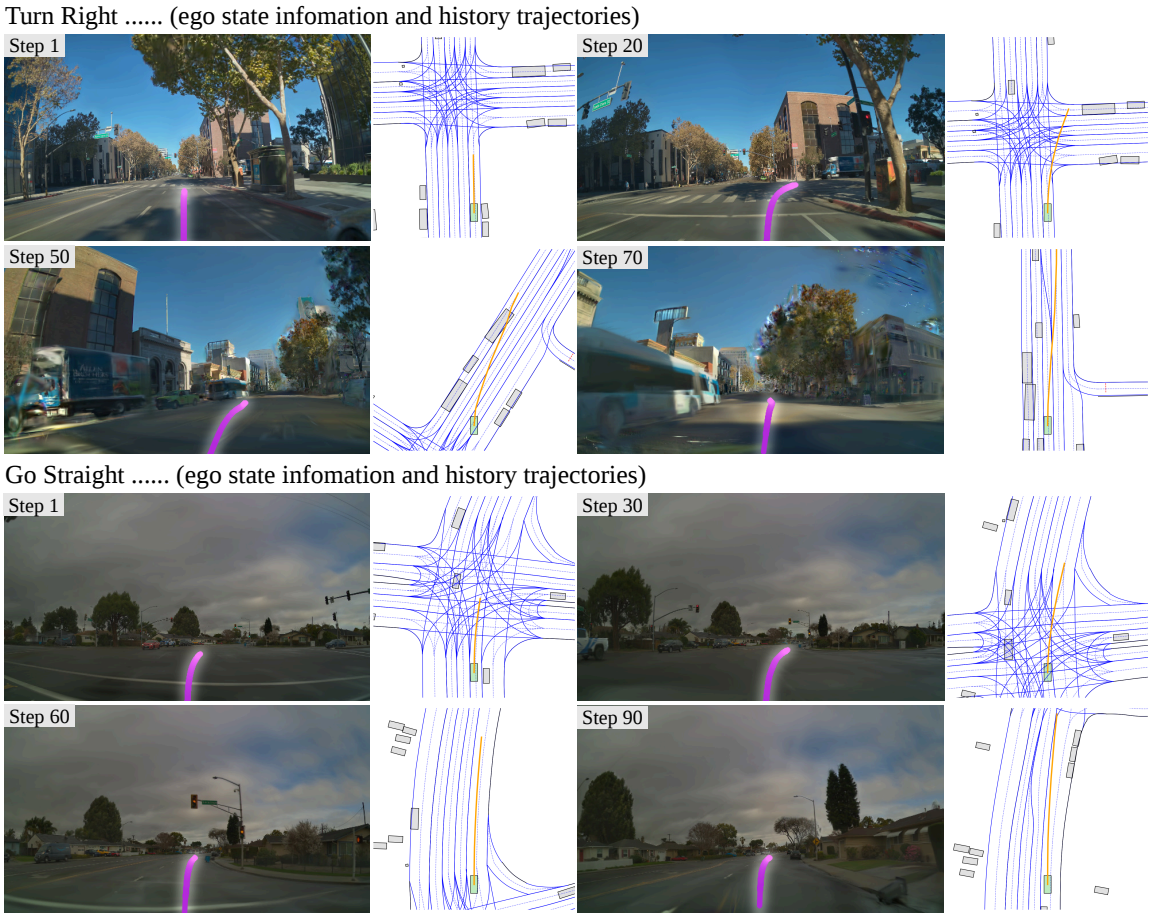


Figure 13: **Qualitative zero-shot closed-loop simulation visualization on AlpaSim.** We show two representative cases from the PhysicalAI-AV NuRec dataset. In the first right-turn case, Qwen-RobotNav slows down and proceeds straight after approaching an intersection, then performs a right turn while keeping away from the road boundary, and finally continues forward along the outgoing lane. In the second straight-driving case, the ego vehicle starts from a stopped state after the traffic light turns green and safely passes through a complex intersection while avoiding surrounding traffic participants.

only when the ego vehicle is deemed responsible. We include AlpaSim as an additional autonomous driving benchmark to examine closed-loop safety and long-horizon driving robustness beyond NAVSIM.

Importantly, Qwen-RobotNav is evaluated in a zero-shot setting on AlpaSim: the model is directly rolled out in the simulator without AlpaSim-specific training or closed-loop adaptation on the PhysicalAI-AV NuRec scenarios. Therefore, this benchmark mainly measures the out-of-domain transfer ability of a general vision-language navigation model under long-horizon autonomous driving simulation. Although Qwen-RobotNav still lags behind the Alpamayo-R1 models that are designed for this evaluation setting, it achieves non-trivial zero-shot closed-loop performance. Moreover, increasing the model scale from 4B to 8B improves the off-road rate from 34.0% to 27.0% and the AlpaSim score from 0.15 to 0.17, suggesting that larger backbones provide better closed-loop stability and scene-level generalization under this challenging transfer setting.

Figure 13 provides qualitative examples of Qwen-RobotNav in zero-shot AlpaSim closed-loop simulation. In the right-turn scenario, the ego vehicle first approaches the intersection cautiously and proceeds straight at a reduced speed, then executes a right turn while avoiding the road boundary, and finally aligns with the outgoing lane to continue driving forward. In the intersection scenario, the vehicle correctly reacts to the green traffic light, starts from a stopped state, and proceeds through a complex multi-agent intersection without colliding with other vehicles. These cases suggest that, despite the remaining quantitative gap on aggregate AlpaSim metrics, Qwen-RobotNav can still exhibit meaningful reactive planning behaviors when transferred zero-shot to challenging closed-loop driving scenes.

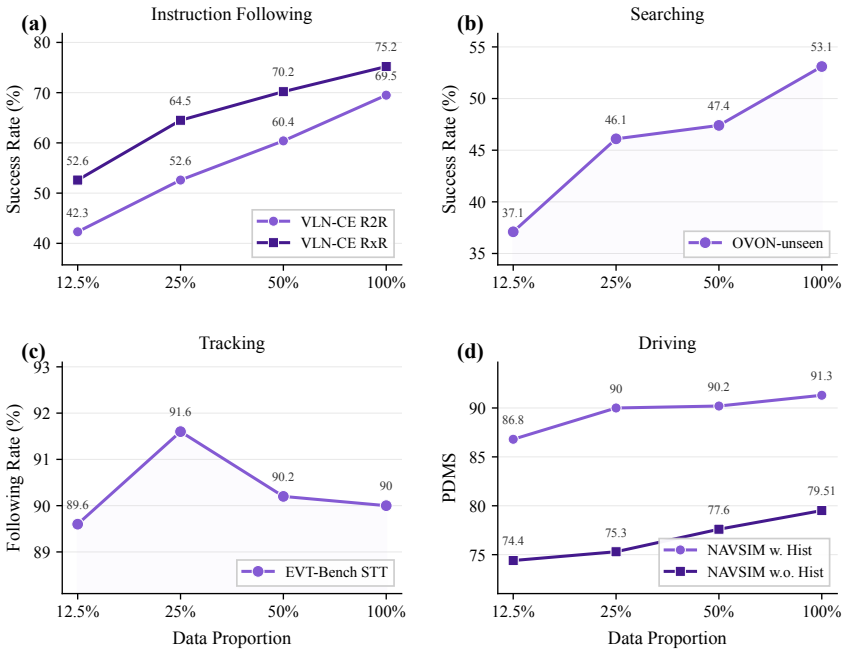


Figure 14: **Data scaling behavior of Qwen-RobotNav.** Performance on representative navigation benchmarks as a function of the training data fraction. Increasing the amount of navigation training data yields clear gains on most tasks, with especially strong improvements on long-horizon tasks such as VLN-CE RxR, while short-horizon tracking saturates earlier and exhibits mild non-monotonicity.

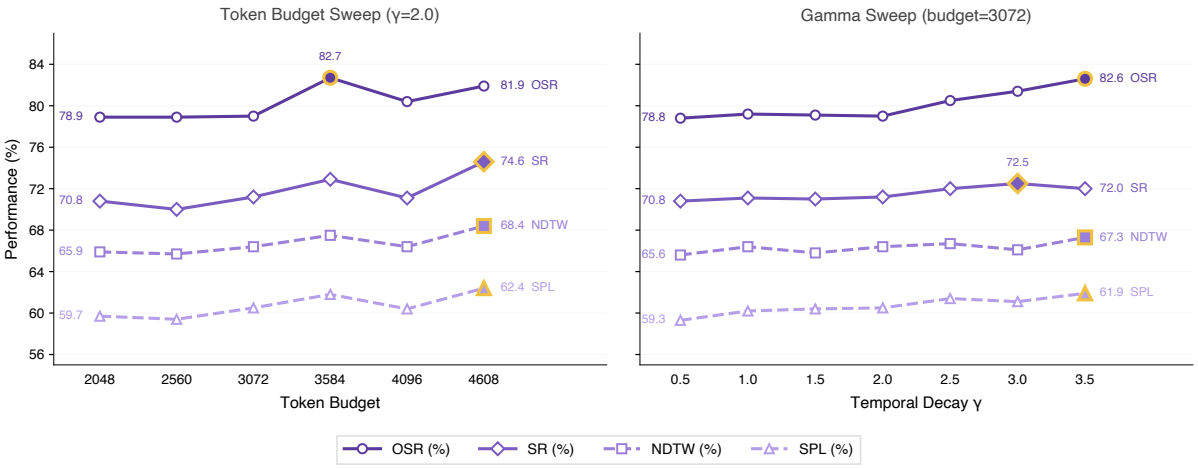


Figure 15: **Ablation on token budget B and temporal decay γ.** We evaluate Qwen-RobotNav-4B on 500 VLN-CE R2R Val-Unseen episodes under varying configurations. *Left:* Sweeping the token budget from 2048 to 4608 at fixed γ=2.0. *Right:* Sweeping the temporal decay from 0.5 to 3.5 at fixed B=3072.

5.5 Ablation Study

Effect of training data scale. Figure 14 shows how Qwen-RobotNav performance scales with the fraction of navigation training data used during training. Across representative benchmarks, increasing the data fraction from 12.5% to 100% leads to clear overall gains, most notably on instruction-following and driving tasks. The gains are particularly pronounced on long-horizon instruction-following tasks such as VLN-CE RxR, suggesting that broader trajectory coverage improves grounding between linguistic instructions and extended visual histories. On shorter-horizon tasks such as target tracking (EVT-Bench), performance improves rapidly with moderate data and then saturates with mild fluctuations, indicating that these tasks are less sensitive to additional training data once basic tracking behavior has been learned. Importantly, the full-data model remains competitive while preserving the stronger gains obtained on long-horizon and cross-embodiment tasks.

Effect of token budget and temporal decay. Figure 15 examines the two primary control parameters of the task-adaptive observation encoding: the total visual token budget B and the temporal decay factor γ. We evaluate Qwen-RobotNav-4B on 500 VLN-CE R2R Val-Unseen episodes, reporting SR, SPL, nDTW, and OSR. In the token budget sweep (left panel, γ=2.0), increasing B substantially improves performance over the low-budget setting, though the benefit is not strictly monotonic once the budget becomes large. SR improves from 70.8% at B=2048 to 74.6% at B=4608, while OSR rises from 78.9% and peaks at 82.7% when B=3584 before slightly decreasing at the largest budget. This pattern indicates that retaining more visual tokens generally provides richer spatial context and improves goal-reaching ability, but excessive or poorly allocated visual context can yield diminishing returns. In the gamma sweep (right panel, B=3072), OSR shows a clear overall improvement from 78.8% (γ=0.5) to 82.6% (γ=3.5), while SR peaks at 72.5% (γ=3.0) before slightly declining. Larger γ concentrates more tokens on recent frames, enhancing the model's ability to resolve the current scene at the expense of early-history context. For instruction-following tasks such as VLN-CE, where the agent must react to its immediate surroundings while still respecting earlier landmarks, this recency bias is beneficial but exhibits a trade-off: oracle success continues to improve at high γ, whereas strict success and path-quality metrics saturate or fluctuate slightly.

5.6 Real-World Deployment Results

To further validate the practical applicability of Qwen-RobotNav, we deploy it on a quadruped robot in a previously unseen, real-world exhibition hall. This environment is highly out-of-distribution, as the hall encompasses various types of environments such as living rooms, medical rooms, and open corridors, presenting a rich mixture of heterogeneous scenes and objects that are absent from the training distribution.

As shown in Figure 16, we evaluate a vision-and-language navigation task using *pure language instructions* without any goal images or coordinates. The robot is commanded to navigate from a living room area to a medical room located 21.78 m away. Throughout the trajectory, the model leverages different visual landmarks, such as furniture, doorways, and signage, to ground language instructions into spatial decisions, successfully traversing multiple visually distinct zones to reach the designated target area.

We further demonstrate fine-grained language-based control by issuing a *reverse language command* that instructs the robot to walk backward. Upon receiving this command, the model switches its locomotion mode and accurately executes the reverse trajectory, retracing the entire route until it returns to a position closely matching its initial starting pose. This result highlights three key capabilities: (1) Qwen-RobotNav can faithfully interpret and execute diverse motion primitives, including non-standard behaviors such as backward locomotion, purely from natural language; (2) the model effectively utilizes visual landmarks encountered during forward navigation to maintain spatial awareness along the return path; and (3) when directed to a previously visited location, the model can accurately return to that exact position, demonstrating precise spatial grounding in a cluttered, previously unseen environment.

We also evaluate Qwen-RobotNav in an indoor apartment setting, as shown in Figure 17. The key finding is that the model's behavior can be precisely controlled through natural language commands. For instance, when instructed to stop at the nightstand on the left side of the bed, the robot navigates into the bedroom and halts at the exact specified side; when instructed to turn around before exiting to the living room, the robot faithfully completes the detour rather than taking a direct path. In the four representative examples we showcase, the robot consistently translates fine-grained verbal directives into accurate navigation behaviors across multiple rooms, confirming that Qwen-RobotNav provides reliable and precise language-based control in complex indoor environments. We further evaluate the agentic navigation interface in a real indoor environment with a mobile robot. As shown in Figure 18, the user gives an open-ended instruction: the robot should check whether a green umbrella was left at Cotti Coffee and also report salient observations along the way. Unlike a short route-following command, this task requires the system to infer the destination from language, maintain task progress over many decision turns, use visual landmarks for localization, and update its plan as new evidence is collected. The selected turns in Figure 18 summarize the high-level agent loop rather than every low-level control step. The agent first parses the user request and observes the initial scene to establish its starting position. It then updates its memory, identifies useful landmarks, and decides to follow the corridor toward the likely store area. During navigation, the agent records intermediate cues such as the Alibaba logo, uses them to refine its localization, and continues searching for the exact Cotti Coffee location. After reaching the store area, the robot inspects the scene and observes a green umbrella, allowing the agent to produce the final response that the umbrella appears to still be there.

This example illustrates the intended division of labor in the agentic Qwen-RobotNav system. The upper-level agent decomposes the open-ended user request into successive sub-goals and maintains an evidence notebook across turns, while Qwen-RobotNav executes grounded navigation segments

Figure 16: **Real-world VLN deployment in an unseen exhibition hall.** The robot dog navigates 21.78 m from a living room to a medical room following pure language instructions, leveraging different visual landmarks along the route. Upon receiving a reverse language command, the robot precisely walks backward to its original starting position.

and returns trajectory evidence for subsequent planning. The real-world episode shows that the same interface can support long-horizon navigation, landmark-based reasoning, memory update, and final evidence-grounded response generation outside simulated benchmark settings.

Figure 17: **Indoor deployment with verbal commands.** The robot executes navigation tasks in an apartment setting using step-by-step verbal instructions, traversing between the bedroom, living room, and bathroom while responding to fine-grained spatial directives.

6 Conclusion

We have presented Qwen-RobotNav, a unified navigation model built on Qwen3-VL that reframes the central challenge of multi-task navigation as a *context modeling* problem rather than an architecture

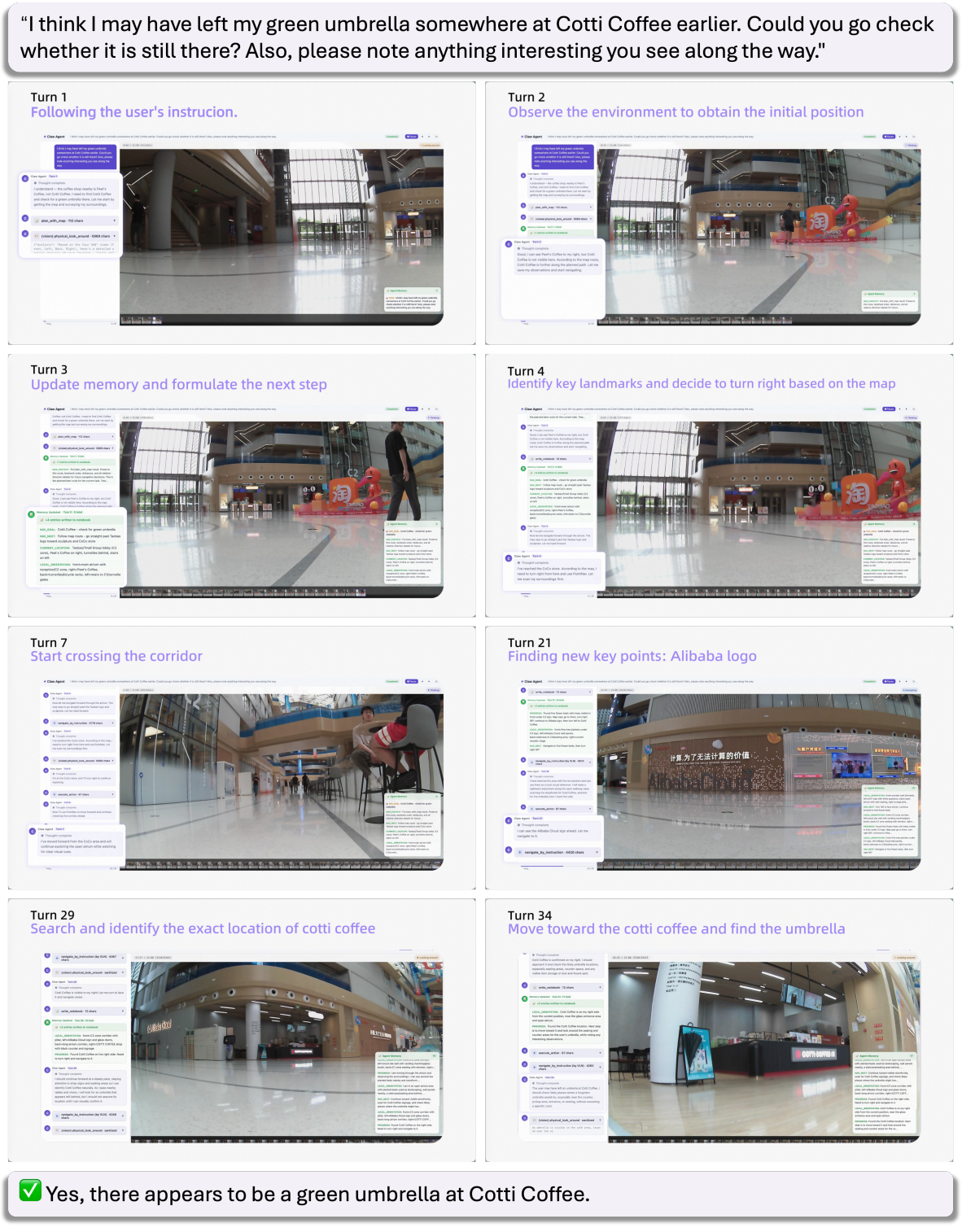


Figure 18: **Real-world long-horizon navigation with agentic Qwen-RobotNav.** On a real robot, the agent answers an open-ended request by decomposing the task into sub-goals, following landmarks to Cotti Coffee, and verifying the green umbrella from visual evidence. Selected turns show the loop of upper-level planning, Qwen-RobotNav execution, memory updates, and final response generation.

or task-head problem. Our key insight is that diverse navigation tasks, from instruction following to target tracking to autonomous driving, share the same perception and planning backbone but demand fundamentally different strategies for consuming the observation stream. Rather than committing to a single fixed strategy, Qwen-RobotNav exposes a parameterised observation encoding interface that allows an outer agent or human operator to dynamically select the appropriate context modeling

strategy for each sub-task at inference time, with training-time randomisation ensuring robustness to any configuration without retraining.

This design principle yields two broader implications. First, it transforms a navigation model from a fixed policy into a *reconfigurable navigation primitive*: the same model seamlessly switches between global-history mode for exploration and recency-focused mode for precise local manoeuvring, enabling complex long-horizon behaviours to emerge from the composition of simple, well-defined working modes. Second, natural-language viewpoint and temporal identification demonstrate that architectural simplicity can be a strength: by communicating structure through ordinary vocabulary tokens rather than learned positional embeddings, the model preserves its open-world language grounding at zero additional cost.

Extensive experiments across navigation, tracking, autonomous driving, and embodied question answering validate these principles, and zero-shot transfer to real-world robots confirms that the unified design generalises beyond simulation. We believe the task-adaptive observation encoding introduced here, treating observation context as a first-class, externally controllable degree of freedom, offers a promising direction for building navigation foundation models that are both broadly capable and practically deployable within agentic navigation systems.

7 Contributions and Acknowledgments

Core Contributors: Jiazhao Zhang[*†], Gengze Zhou[*†], Hang Yin[*], Yiyang Huang[*], Zixing Lei[*], Qihang Peng[*], Chenxu Lv[‡], Haoqi Yuan, Jie Zhang, Xiaoyue Chen, An Yang, Fei Huang, Junyang Lin, Dayiheng Liu, Jingren Zhou, Chenfei Wu[‡], Xiong-Hui Chen[‡]

Contributors: Zhuoyuan Yu, Jingyang Fan, Zhixuan Liang, Pei Lin, Ye Wang, Anzhe Chen, Shuai Bai, Lulu Hu, Xuancheng Ren

Acknowledgments: We acknowledge the National Pilot Base for Embodied Intelligence Applications for providing the real-robot experimental environment and equipment.

References

AMAP CV Lab. ABot-N0: Technical report on the vla foundation model for versatile embodied navigation. *arXiv preprint*, 2025.

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*, 2024.

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *CVPR*, 2018.

Dhruv Batra, Aaron Gokaslan, Aniruddha Kembhavi, Oleksandr Maksymets, Roozbeh Mottaghi, Manolis Savva, Alexander Toshev, and Erik Wijmans. ObjectNav revisited: On evaluation of embodied agents navigating to objects. In *arXiv preprint arXiv:2006.13171*, 2020.

Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11621–11631, 2020.

Wenzhe Cai, Delin Feng, Yu Liu, Jiangmiao Pang, Jiaqi Peng, Chenyang Wan, Hanqing Wang, Liuyi Wang, Tai Wang, Meng Wei, Yuqiang Yang, Xiqian Yu, and Chenming Zhu. Internvla-n1: An open dual-system vision-language navigation foundation model with learned latent plans. Technical report, Shanghai AI Laboratory, 2025. URL https://internrobotics.github.io/internvla-n1.github.io/.

Yihan Cao, Jiazhao Zhang, Zhinan Yu, Shuzhen Liu, Zheng Qin, Qin Zou, Bo Du, and Kai Xu. Cognav: Cognitive process modeling for object goal navigation with llms. *arXiv preprint arXiv:2412.10439*, 2024.

[*]Equal contribution.
[†]Project lead.
[‡]Corresponding author.

Angel X. Chang, Angela Dai, Thomas A. Funkhouser, Maciej Halber, Matthias Nießner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from RGB-D data in indoor environments. In *2017 International Conference on 3D Vision, 3DV 2017, Qingdao, China, October 10-12, 2017*, pp. 667–676. IEEE Computer Society, 2017. doi: 10.1109/3DV.2017.00081. URL https://doi.org/10.1109/3DV.2017.00081.

Yitong Chen, Lingchen Meng, Wujian Peng, Zuxuan Wu, and Yu-Gang Jiang. Comp: Continual multimodal pre-training for vision foundation models. *arXiv preprint arXiv:2503.18931*, 2025a.

Yuntao Chen, Yuqi Wang, and Zhaoxiang Zhang. Drivinggpt: Unifying driving world modeling and planning with multi-modal autoregressive transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 26890–26900, 2025b.

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

Kashyap Chitta, Aditya Prakash, Bernhard Jaeger, Zehao Yu, Katrin Renz, and Andreas Geiger. Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. *IEEE transactions on pattern analysis and machine intelligence*, 45(11):12878–12895, 2022.

OpenScene Contributors. Openscene: The largest up-to-date 3d occupancy prediction benchmark in autonomous driving. https://github.com/OpenDriveLab/OpenScene, 2023.

Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Embodied question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–10, 2018.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *CVPR*, 2017.

Monika Gupta, Ritu Tiwari, and G Laxmidhar Raja. A novel policy for multi-robot navigation using image-based visual servoing. In *Proceedings of the International Conference on Informatics and Analytics*, 2016.

Weijing Hu, Jun Wang, Teng Hu, Jiteng Chen, Siwen Xue, Yufeng Yue, Haoran Xie, Weixun Zhang, Huchuan Lu, Zongqing Lu, Haibin He, and Bolei Wang. OmniNav: A unified framework for prospective exploration and visual-language navigation. *arXiv preprint arXiv:2510.06436*, 2025.

Weijing Hu, Jun Wang, Teng Hu, Jiteng Chen, Siwen Xue, Yufeng Yue, Yanyun Wu, Haibin He, Bolei Wang, Huchuan Lu, and Zongqing Lu. AstraNav-World: World model for foresight control and consistency. *arXiv preprint arXiv:2603.23745*, 2026.

Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, et al. Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 17853–17862, 2023.

Bo Jiang, Shaoyu Chen, Hao Gao, Bencheng Liao, Qian Zhang, Wenyu Liu, and Xinggang Wang. Vadv2: End-to-end vectorized autonomous driving via probabilistic planning. In *The Fourteenth International Conference on Learning Representations*, 2024.

Kaixuan Jiang, Yang Liu, Weixing Chen, Jingzhou Luo, Ziliang Chen, Ling Pan, Guanbin Li, and Liang Lin. Beyond the destination: A novel benchmark for exploration-aware embodied question answering. In *IEEE/CVF International Conference on Computer Vision, ICCV 2025, Honolulu, HI, USA, October 19-25, 2025*, pp. 9091–9101. IEEE, 2025.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*, 2017.

Kushal Kafle, Brian Price, Scott Cohen, and Christopher Kanan. Dvqa: Understanding data visualizations via question answering. In *CVPR*, 2018.

Samira Ebrahimi Kahou, Vincent Michalski, Adam Atkinson, Ákos Kádár, Adam Trischler, and Yoshua Bengio. Figureqa: An annotated figure dataset for visual reasoning. *arXiv preprint arXiv:1710.07300*, 2018.

Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara L. Berg. Referitgame: Referring to objects in photographs of natural scenes. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pp. 787–798. ACL, 2014. doi: 10.3115/V1/D14-1086. URL https://doi.org/10.3115/v1/d14-1086.

Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *European Conference on Computer Vision (ECCV)*, 2020.

Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4392–4412, 2020.

Yuxuan Kuang, Hai Lin, and Meng Jiang. Openfmnav: Towards open-set zero-shot object navigation via vision-language foundation models. *arXiv preprint arXiv:2402.10670*, 2024.

Chengyang Li, Shuai Wang, Kejiang Ye, Weijie Yuan, Boyu Zhou, Yik-Chung Wu, Cheng-Zhong Xu, and Huseyin Arslan. Memory centric power allocation for multi-agent embodied question answering. *CoRR*, abs/2604.17810, 2026a.

Kailin Li, Zhenxin Li, Shiyi Lan, Yuan Xie, Zhizhong Zhang, Jiayi Liu, Zuxuan Wu, Zhiding Yu, and Jose M Alvarez. Hydra-mdp++: Advancing end-to-end driving via expert-guided hydra-distillation. *arXiv preprint arXiv:2503.12820*, 2025a.

Pengxiang Li, Yinan Zheng, Yue Wang, Huimin Wang, Hang Zhao, Jingjing Liu, Xianyuan Zhan, Kun Zhan, and XianPeng Lang. Discrete diffusion for reflective vision-language-action models in autonomous driving. In *The Fourteenth International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=XJxXSMLDoZ.

Yingyan Li, Lue Fan, Jiawei He, Yuqi Wang, Yuntao Chen, Zhaoxiang Zhang, and Tieniu Tan. Enhancing end-to-end autonomous driving with latent world model. *arXiv preprint arXiv:2406.08481*, 2024.

Yingyan Li, Yuqi Wang, Yang Liu, Jiawei He, Lue Fan, and Zhaoxiang Zhang. End-to-end driving with online trajectory evaluation via bev world model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 27137–27146, 2025b.

Yongkang Li, Kaixin Xiong, Xiangyu Guo, Fang Li, Sixu Yan, Gangwei Xu, Lijun Zhou, Long Chen, Haiyang Sun, Bing Wang, et al. Recogdrive: A reinforced cognitive framework for end-to-end autonomous driving. *arXiv preprint arXiv:2506.08052*, 2025c.

Zhenxin Li, Shihao Wang, Shiyi Lan, Zhiding Yu, Zuxuan Wu, and Jose M Alvarez. Hydra-next: Robust closed-loop driving with open-loop training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 27305–27314, 2025d.

Zhixuan Liang, Yuxiao Chen, Yurong You, Peter Karkus, Wenhao Ding, Boyi Li, Alexander Popov, Yan Wang, Maximilian Igl, Yiming Li, et al. Planning-aligned token compression for long-context autonomous driving. *arXiv preprint arXiv:2606.07464*, 2026.

Sihao Lin, Zerui Li, Xunyi Zhao, Gengze Zhou, Liuyi Wang, Rong Wei, Rui Tang, Juncheng Li, Hanqing Wang, Jiangmiao Pang, Anton van den Hengel, Jiajun Liu, and Qi Wu. Vlnverse: A benchmark for vision-language navigation with versatile, embodied, realistic simulation and evaluation. *CoRR*, abs/2512.19021, 2025. doi: 10.48550/ARXIV.2512.19021. URL https://doi.org/10.48550/arXiv.2512.19021.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

Jiahang Liu, Yunpeng Qi, Jiazhao Zhang, Minghan Li, Shaoan Wang, Kui Wu, Hanjing Ye, Hong Zhang, Zhibo Chen, Fangwei Zhong, et al. Trackvla++: Unleashing reasoning and memory capabilities in vla models for embodied visual tracking. *arXiv preprint arXiv:2510.07134*, 2025.

Yuxing Long, Wenzhe Cai, Hongcheng Wang, Guanqi Zhan, and Hao Dong. Instructnav: Zero-shot system for generic instruction navigation in unexplored environment. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Conference on Robot Learning, 6-9 November 2024, Munich, Germany*, Proceedings of Machine Learning Research, pp. 2049–2060. PMLR, 2024.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Arjun Majumdar, Anurag Ajay, Xiaohan Zhang, Pranav Putta, Sriram Yenamandra, Mikael Henaff, Sneha Silwal, Paul Mcvay, Oleksandr Maksymets, Sergio Arnaud, Karmesh Yadav, Qiyang Li, Ben Newman, Mohit Sharma, Vincent Berges, Shiqi Zhang, Pulkit Agrawal, Yonatan Bisk, Dhruv Batra, Mrinal Kalakrishnan, Franziska Meier, Chris Paxton, Alexander Sax, and Aravind Rajeswaran. Openeqa: Embodied question answering in the era of foundation models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16488–16498, 2024.

Lingchen Meng, Jianwei Yang, Rui Tian, Xiyang Dai, Zuxuan Wu, Jianfeng Gao, and Yu-Gang Jiang. Deepstack: Deeply stacking visual tokens is surprisingly simple and effective for lmms. In *Advances in Neural Information Processing Systems*, volume 37, pp. 23464–23487, 2024.

Dujun Nie, Xianda Guo, Yiqun Duan, Ruijun Zhang, and Long Chen. Wmnav: Integrating vision-language models into world models for object goal navigation. *arXiv preprint arXiv:2503.02247*, 2025.

Qihang Peng, Henry Zheng, and Gao Huang. Proxytransformation: Preshaping point cloud manifold with proxy attention for 3d visual grounding. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 24582–24592, 2025.

Qihang Peng, Xuesong Chen, Chenye Yang, Shaoshuai Shi, and Hongsheng Li. Colavla: Leveraging cognitive latent reasoning for hierarchical parallel trajectory planning in autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17809–17819, June 2026.

Yuankai Qi, Qi Wu, Peter Anderson, Xin Wang, William Yang Wang, Chunhua Shen, and Anton van den Hengel. Reverie: Remote embodied visual referring expression in real indoor environments. In *CVPR*, 2020.

Santhosh Kumar Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alexander Clegg, John M. Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X. Chang, Manolis Savva, Yili Zhao, and Dhruv Batra. Habitat-matterport 3d dataset (HM3D): 1000 large-scale 3d environments for embodied AI. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/34173cb38f07f89ddbebc2ac9128303f-Abstract-round2.html.

Allen Z. Ren, Jaden Clark, Anushri Dixit, Masha Itkina, Anirudha Majumdar, and Dorsa Sadigh. Explore until confident: Efficient exploration for embodied question answering. In Dana Kulic, Gentiane Venture, Kostas E. Bekris, and Enrique Coronado (eds.), *Robotics: Science and Systems XX, Delft, The Netherlands, July 15-19, 2024*, 2024.

Manolis Savva, Jitendra Malik, Devi Parikh, Dhruv Batra, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, and Vladlen Koltun. Habitat: A platform for embodied AI research. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pp. 9338–9346. IEEE, 2019. doi: 10.1109/ICCV.2019.00943. URL https://doi.org/10.1109/ICCV.2019.00943.

Saumya Saxena, Blake Buchanan, Chris Paxton, Bingqing Chen, Narunas Vaskevicius, Luigi Palmieri, Jonathan Francis, and Oliver Kroemer. Grapheqa: Using 3d semantic scene graphs for real-time embodied question answering. *CoRR*, abs/2412.14480, 2024.

Shuai Shao, Zeming Li, Tianyuan Zhang, Chao Peng, Gang Yu, Xiangyu Zhang, Jing Li, and Jian Sun. Objects365: A large-scale, high-quality dataset for object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 8430–8439, 2019.

Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2446–2454, 2020.

Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. Vision-and-dialog navigation. In *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 394–406. PMLR, 2020.

Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, et al. Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*, 2025.

Junyang Wang, Shuxun Zou, Qianli Liu, Jiangming Zou, Hui Li, and Jianxin Liu. GN0: Unified end-to-end training paradigm for vision-and-language navigation. *arXiv preprint arXiv:2606.03233*, 2026.

Liuyi Wang, Xinyuan Xia, Hui Zhao, Hanqing Wang, Tai Wang, Yilun Chen, Chengju Liu, Qijun Chen, and Jiangmiao Pang. Rethinking the embodied gap in vision-and-language navigation: A holistic study of physical and visual disparities. *CoRR*, abs/2507.13019, 2025a. doi: 10.48550/ARXIV.2507.13019. URL https://doi.org/10.48550/arXiv.2507.13019.

Shaoan Wang, Jiazhao Zhang, Minghan Li, Jiahang Liu, Anqi Li, Kui Wu, Fangwei Zhong, Junzhi Yu, Zhizheng Zhang, and He Wang. TrackVLA: Embodied visual tracking in the wild. *arXiv preprint arXiv:2505.23189*, 2025b.

Zihan Wang, Xiangyang Li, Jiahao Yang, Yeqi Liu, Junjie Hu, Ming Jiang, and Shuqiang Jiang. Lookahead exploration with neural radiance representation for continuous vision-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13753–13762, 2024.

Zun Wang, Jialu Li, Yicong Hong, Songze Li, Kunchang Li, Shoubin Yu, Yi Wang, Yu Qiao, Yali Wang, Mohit Bansal, and Limin Wang. Bootstrapping language-guided navigation learning with self-refining data flywheel. In *International Conference on Learning Representations*, volume 2025, pp. 23542–23568, 2025c.

Meng Wei, Chenyang Wan, Jiaqi Peng, Xiqian Yu, Yuqiang Yang, Delin Feng, Wenzhe Cai, Chenming Zhu, Tai Wang, Jiangmiao Pang, et al. Ground slow, move fast: A dual-system foundation model for generalizable vision-and-language navigation. *arXiv preprint arXiv:2512.08186*, 2025a.

Meng Wei, Chenyang Wan, Xiqian Yu, Tai Wang, Yuqiang Yang, Xiaohan Mao, Chenming Zhu, Wenzhe Cai, Hanqing Wang, Yilun Chen, et al. Streamvln: Streaming vision-and-language navigation via slowfast context modeling. *arXiv preprint arXiv:2507.05240*, 2025b.

Xinshuo Weng, Boris Ivanovic, Yan Wang, Yue Wang, and Marco Pavone. Para-drive: Parallelized architecture for real-time autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15449–15458, 2024.

Erik Wijmans, Abhishek Kadian, Ari Morcos, Stefan Lee, Irfan Essa, Devi Parikh, Manolis Savva, and Dhruv Batra. DD-PPO: Learning near-perfect pointgoal navigators from 2.5 billion frames. In *International Conference on Learning Representations (ICLR)*, 2020.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Shengming Yin, Shuai Bai, Xiao Xu, Yilei Chen, Yuxiang Chen, Zecheng Tang, Zekai Zhang, Zhengyi Wang, An Yang, Bowen Yu, Chen Cheng, Dayiheng Liu, Deqing Li, Hang Zhang, Hao Meng, Hu Wei, Jingyuan Ni, Kai Chen, Kuan Cao, Liang Peng, Lin Qu, Minggang Wu, Peng Wang, Shuting Yu, Tingkun Wen, Wensen Feng, Xiaoxiao Xu, Yi Wang, Yichang Zhang, Yongqiang Zhu, Yujia Wu, Yuxuan Cai, and Zenan Liu. Qwen-image technical report. *CoRR*, abs/2508.02324, 2025. doi: 10.48550/ARXIV.2508.02324. URL https://doi.org/10.48550/arXiv.2508.02324.

Zebin Xing, Xingyu Zhang, Yang Hu, Bo Jiang, Tong He, Qian Zhang, Xiaoxiao Long, and Wei Yin. Goalflow: Goal-driven flow matching for multimodal trajectories generation in end-to-end autonomous driving. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 1602–1611, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, et al. Qwen3 technical report, 2025a.

Yuncong Yang, Han Yang, Jiachen Zhou, Peihao Chen, Hongxin Zhang, Yilun Du, and Chuang Gan. 3d-mem: 3d scene memory for embodied exploration and reasoning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2025, Nashville, TN, USA, June 11-15, 2025*, pp. 17294–17303. Computer Vision Foundation / IEEE, 2025b.

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. *arXiv preprint arXiv:2410.08189*, 2024.

Hang Yin, Haoyu Wei, Xiuwei Xu, Wenxuan Guo, Jie Zhou, and Jiwen Lu. Gc-vln: Instruction as graph constraints for training-free vision-and-language navigation. *arXiv preprint arXiv:2509.10454*, 2025a.

Hang Yin, Xiuwei Xu, Linqing Zhao, Ziwei Wang, Jie Zhou, and Jiwen Lu. Unigoal: Towards universal zero-shot goal-oriented navigation. *arXiv preprint arXiv:2503.10630*, 2025b.

Hang Yin, Jiazhao Zhang, Yinan Liang, Jiahang Liu, Minghan Li, and He Wang. Alldaynav: Lifelong navigation via real-world reinforcement learning. *arXiv preprint arXiv:2606.10927*, 2026.

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, 2024a.

Naoki Yokoyama, Ram Ramrakhya, Abhishek Das, Dhruv Batra, and Sehoon Ha. HM3D-OVON: A dataset and benchmark for open-vocabulary object goal navigation. *arXiv preprint arXiv:2409.14296*, 2024b.

Chengran Yuan, Zhanqi Zhang, Jiawei Sun, Shuo Sun, Zefan Huang, Christina Dao Wen Lee, Dongen Li, Yuhang Han, Anthony Wong, Keng Peng Tee, et al. Drama: An efficient end-to-end motion planner for autonomous driving with mamba. *arXiv preprint arXiv:2408.03601*, 2024.

Kuo-Hao Zeng, Zichen Zhang, Kiana Ehsani, Rose Hendrix, Jordi Salvador, Alvaro Herrasti, Ross Girshick, Aniruddha Kembhavi, and Luca Weihs. Poliformer: Scaling on-policy rl with transformers results in masterful navigators. *arXiv preprint arXiv:2406.20083*, 2024.

Haochen Zhang, Nirav Savaliya, Faizan Siddiqui, and Enna Sachdeva. FAST-EQA: efficient embodied question answering with global and local region relevancy. In *IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2026, Tucson, AZ, USA, March 6-10, 2026*, pp. 1664–1673. IEEE, 2026.

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

Jiazhao Zhang, Anqi Li, Yunpeng Qi, Minghan Li, Jiahang Liu, Shaoan Wang, Haoran Liu, Gengze Zhou, Yuze Wu, Xingxing Li, et al. Embodied navigation foundation model. *arXiv preprint arXiv:2509.12129*, 2025a.

Jiazhao Zhang, Kunyu Wang, Shaoan Wang, Minghan Li, Haoran Liu, Songlin Wei, Zhongyuan Wang, Zhizheng Zhang, and He Wang. Uni-NaVid: A video-based vision-language-action model for unifying embodied navigation tasks. *Robotics: Science and Systems*, 2025b.

Lingfeng Zhang, Qiang Zhang, Hao Wang, Erjia Xiao, Zixuan Jiang, Honglei Chen, and Renjing Xu. Trihelper: Zero-shot object navigation with dynamic assistance, 2024b. URL https://arxiv.org/abs/2403.15223.

Tongjie Y Zhang and Ching Y. Suen. A fast parallel algorithm for thinning digital patterns. *Communications of the ACM*, 27(3):236–239, 1984.

Xunyi Zhao, Gengze Zhou, and Qi Wu. Vln-mme: Diagnosing mllms as language-guided visual navigation agents. *arXiv preprint arXiv:2512.24851*, 2025.

Henry Zheng, Hao Shi, Yong Xien Chng, Rui Huang, Zanlin Ni, Tianyi Tan, Qihang Peng, Yepeng Weng, Zhongchao Shi, and Gao Huang. Denseg: Alleviating vision-language feature sparsity in multi-view 3d visual grounding. In *Autonomous Grand Challenge CVPR 2024 Workshop*, volume 2, pp. 6, 2024.

Henry Zheng, Hao Shi, Qihang Peng, Yong Xien Chng, Rui Huang, Yepeng Weng, zhongchao shi, and Gao Huang. Densegrounding: Improving dense language-vision semantics for ego-centric 3d visual grounding. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=iGafR0hSln.

Fangwei Zhong, Kui Wu, Hai Ci, Churan Wang, and Hao Chen. Empowering embodied visual tracking with visual foundation models and offline rl. In *European Conference on Computer Vision*, pp. 139–155. Springer, 2024.

Yufeng Zhong, Chengjian Feng, Feng Yan, Fanfan Liu, Liming Zheng, and Lin Ma. Robotrom-nav: A unified framework for embodied navigation integrating perception, planning, and prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6416–6425, 2025.

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.

Gengze Zhou, Yicong Hong, Zun Wang, Chongyang Zhao, Mohit Bansal, and Qi Wu. Same: Learning generic language-guided visual navigation with state-adaptive mixture of experts. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7794–7807, 2025a.

Zewei Zhou, Tianhui Cai, Seth Z Zhao, Yun Zhang, Zhiyu Huang, Bolei Zhou, and Jiaqi Ma. Autovla: A vision-language-action model for end-to-end autonomous driving with adaptive reasoning and reinforcement fine-tuning. *arXiv preprint arXiv:2506.13757*, 2025b.

Fengda Zhu, Xiwen Liang, Yi Zhu, Qizhi Yu, Xiaojun Chang, and Xiaodan Liang. Soon: Scenario oriented object navigation with graph-based exploration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12689–12699, June 2021.

Ziyu Zhu, Xilin Wang, Yixuan Li, Zhuofan Zhang, Xiaojian Ma, Yixin Chen, Baoxiong Jia, Wei Liang, Qian Yu, Zhidong Deng, et al. Move to understand a 3d scene: Bridging visual grounding and exploration for efficient and versatile embodied navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8120–8132, 2025.