INFINITEVLN: CONSTANT-MEMORY STREAMING VISION-AND-LANGUAGE NAVIGATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Current Vision-and-Language Navigation (VLN) models are constrained by growing context length and high computational cost in long-horizon scenarios, limiting their development on real-world robots that require continuous observation and low-latency decision-making. In this paper, we present **InfiniteVLN**, a streaming VLN framework that enables constant-memory navigation by maintaining a bounded, instruction-conditioned belief state. Specifically, we formalize VLN as a Partially Observable Semi-Markov Decision Process (POSMDP) and decouple inference process into two distinct steps: an *observation* step that continuously updates persistent visual memory, and a *decision* step that generates actions conditioned on the current memory state. To enable efficient streaming inference, we introduce **Asymmetric Context Caching**, which preserves globally invariant instructions as a persistent instruction sink while maintaining local spatiotemporal consistency within a bounded KV cache. To retain historical evidence beyond the local context, we further propose **Instruction-Gated Recurrent Memory**, which derives adaptive write gates from the relative attention mass of visual tokens to the instruction, and selectively consolidates task-relevant information into a fixed-capacity associative memory through Delta Rule updates. Together, these designs transform long-horizon VLN from full historical context modeling into instruction-conditioned evidence consolidation under a bounded memory state, enabling efficient and scalable streaming navigation. Extensive experiments on standard VLN benchmarks and challenging long-trajectory settings demonstrate that InfiniteVLN achieves superior performance, with substantially reduced memory consumption and inference latency.

1 INTRODUCTION

Vision-and-Language Navigation (VLN) (Cheng et al., 2025; Zhou et al., 2024; Wei et al., 2025; Xin et al., 2026b; Wei et al., 2026; Xin et al., 2026a) requires an embodied agent to perform long-horizon, goal-oriented navigation in partially observable environments by following natural-language instructions and continuously processing egocentric visual observations. Unlike traditional point-goal navigation or short-horizon visual control, VLN requires the agent to integrate historical observations, executed actions, and language constraints over time to infer unobserved environmental states and make subsequent decisions. VLN is therefore inherently a streaming perception-memory-decision process under partial observability. Recent advances in Vision-Language Models (VLMs) (Yang et al., 2024; Zhang et al., 2024c; Bai et al., 2025a; Ji et al., 2025) has enabled increasingly capable VLN agents by leveraging their multimodal understanding, semantic reasoning, and spatial knowledge to navigate unseen environments.

Despite these progresses, existing VLM-based VLN approaches face a fundamental tension between long-horizon history modeling and efficient, low-latency inference. Most methods (Wei et al., 2025; Zhang et al., 2024b; Wei et al., 2026) preserve navigation history by feeding past observations into the model as an ever-growing sequence of visual tokens, using either full-context concatenation or a continuously expanding KV cache. As the trajectory grows, this paradigm incurs progressively higher attention computation and memory consumption, making continuous onboard inference increasingly challenging for resource-constrained robots. Existing approaches (Wei et al., 2025; Zeng et al., 2025) mitigate these costs through fixed-rate frame sampling or visual token compression. However, such strategies could discard critical historical evidence or retain instruction-irrelevant

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

visual redundancy. Both make it difficult to maintain a compact yet globally consistent representation over extended trajectories. More importantly, they often overlook a fundamental **temporal asymmetry** in VLN: instructions remain static and globally valid throughout an episode, whereas visual observations evolve continuously and exhibit substantial spatiotemporal redundancy. This asymmetry suggests that long-horizon navigation should not require indiscriminately retaining the entire observation history, but rather selectively preserving the evidence necessary for decisions.

Motivated by this perspective, we propose **InfiniteVLN**, a streaming VLN framework that performs long-horizon navigation by maintaining a bounded, instruction-conditioned memory state rather than an ever-growing history. We formulate VLN as a Partially Observable Semi-Markov Decision Process (POSMDP) and maintain a finite-capacity representation that approximates the task-relevant belief state for decision making. Based on this formulation, InfiniteVLN decouples the conventional VLM inference into two distinct processes: an ***observation step***: which continuously incorporates incoming physical observations into persistent memory, and a ***decision step***, which queries the current memory state to generate navigation actions conditioned on the current memory state without modifying the persistent memory. This separation prevents transient reasoning tokens and decision-specific information from contaminating long-term memory for continuous perception and low-latency decision-making.

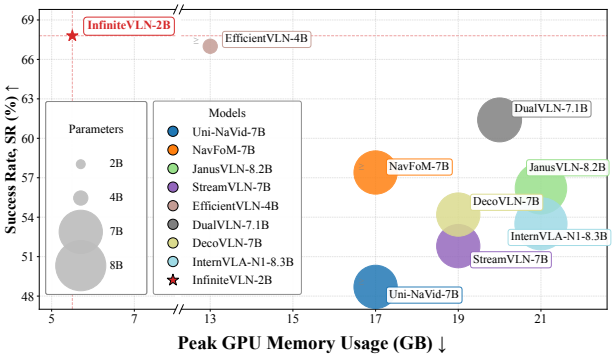

Figure 1: Compared to existing methods, InfiniteVLN has fewer parameters, lower GPU memory consumption, and better navigation accuracy.

To achieve constant-memory streaming inference, we introduce **Asymmetric Context Caching (ACC)**, which decouples the KV cache into a static instruction sink and a dynamic sliding local context by exploiting the temporal asymmetry between instructions and observations. While this design bounds cache growth and preserves local temporal coherence, the sliding window inevitably discards observations outside the current context. To overcome this limitation, we further propose **Instruction-Gated Recurrent Memory (IGRM)**, which maintains long-term navigation evidence in a bounded hidden state. Specifically, IGRM derives a recurrent update gate from the relative attention mass between visual tokens and the instruction sink, modulating each observation token's contribution to recurrent memory update. The selected visual evidence is recursively consolidated into a fixed-capacity associative memory matrix via the Delta Rule. This allows the agent to retain task-relevant history while filtering instruction-irrelevant observations, maintaining a compact belief state for long-horizon navigation. Together, ACC and IGRM enable InfiniteVLN to preserve both local spatiotemporal context and task-relevant historical evidence without allowing memory consumption to grow with the navigation horizon.

As illustrated in Figure 1, extensive experiments across both standard VLN benchmarks and challenging long-trajectory settings demonstrate that InfiniteVLN achieves superior navigation performance while substantially improving memory stability and computational efficiency. In summary, our main contributions are threefold:

- We introduce InfiniteVLN, a streaming VLN framework that formulates long-horizon navigation as bounded belief-state maintenance under a POSMDP and decouples persistent observation updates from transient decision generation.
- We propose Asymmetric Context Caching (ACC), which leverages temporal asymmetry between static instructions and dynamic observations to maintain a bounded KV cache while preserving local spatiotemporal consistency.
- We present Instruction-Gated Recurrent Memory (IGRM), derives an instruction-conditioned write gate from visual-to-instruction attention, and selectively consolidate task-relevant historical evidence into a fixed-capacity recurrent associative memory for long-horizon navigation.

2 Related Work

2.1 Vision-and-Language Navigation

With the rapid advancement of VLMs, recent research in VLN has transitioned from specialized navigation networks toward general-purpose embodied navigation frameworks built upon VLMs (Lin et al., 2024; Wang et al., 2024a). NaVid (Zhang et al., 2024b) leverages large-scale navigation data from continuous environments alongside web-scale multimodal data, enabling VLMs to encode historical observations into spatiotemporal contexts for action planning. StreamVLN (Wei et al., 2025) and JanusVLN (Zeng et al., 2025) address token inflation problem via sliding windows and slow-fast memory mechanisms, while DecoVLN (Xin et al., 2026b) reformulates long-term memory as an online optimization problem, selectively caching task-relevant frames in a memory bank for subsequent decision-making. DualVLN (Wei et al., 2026) introduces a fast-slow dual-system architecture that generates navigation trajectories at high frequencies, thereby surpassing conventional single-system architectures in both performance and efficiency. Furthermore, AgentVLN (Xin et al., 2026a) pioneers the VLM-as-brain paradigm by orchestrating encapsulated skills at the perception and decision layers, it realizes the first cross-platform edge deployment for VLN. Nevertheless, current VLN frameworks have yet to fundamentally resolve the computational efficiency bottleneck associated with long-horizon memory. During inference, these models are still compelled to ingest and process massive volumes of historical memory tokens.

2.2 Streaming Vision-Language Models

In recent years, VLMs (Zhang et al., 2024c; Wang et al., 2024a; Bai et al., 2025b;a; Qwen Team, 2026) have achieved remarkable progress in image understanding, short-video question answering, and long-video reasoning. However, most existing VLMs still adhere to an offline processing paradigm, which necessitates the ingestion of the entire video sequence all at once. The core challenge for streaming VLMs lies in continuously assimilating visual inputs under constrained computational and memory budgets, while maintaining an effective representation of both the current state and historical context at arbitrary intermediate time steps. Representative benchmarks in this direction include StreamingBench (Lin et al., 2026) and OVO-Bench (Li et al., 2025). LiveCC (Chen et al., 2025) leverages ASR timestamps to construct training sequences aligned with streaming perception, enabling real-time video commentary. StreamBridge (Wang et al., 2026) uses interleaved video-text sequences and diverse instruction formats to support multi-turn real-time understanding and proactive responses. Furthermore, Flash-VStream (Zhang et al., 2024a) and StreamingVLM (Xu et al., 2025b) introduce hierarchical memory modules to facilitate the construction of extended long contexts. Nevertheless, current streaming VLMs predominantly rely on external retrieval mechanisms or dual-layer memory architectures. These architectures introduce substantial GPU memory overhead and system complexity, making them difficult to deploy on embodied agents that require low-latency, real-time decision making.

3 InfiniteVLN

3.1 Streaming VLN with Bounded Memory

We formulate VLN as a Partially Observable Semi-Markov Decision Process (POSMDP), represented by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{T}, \mathcal{I} \rangle$. Here, $\mathcal{S}$, $\mathcal{A}$, and $\mathcal{O}$ denote the global state space, action space, and local observation space, respectively, while $\mathcal{I}$ represents the natural language instruction. $\mathcal{T}(s_{t+1}|s_t, a_t)$ specifies the probability of transitioning to state s_{t+1} given the current state s_t and action a_t. At each time step t, the agent receives only a local egocentric visual observations $o_t \in \mathcal{O}$, and therefore must infer the unobserved goal state from the historical context. This can be characterized by a belief state $b_t \approx P(s_t|o_{\leq t}, a_{<t}, \mathcal{I})$, which summarizes the accumulated evidence necessary for subsequent decision-making.

Existing VLM-based VLN methods typically rely on concatenated histories or ever-growing KV caches, resulting in increasing computational and memory overhead as the navigation horizon extends. To address this limitation, we reformulate VLN as a streaming perception–decision process that maintains a bounded, persistent memory state, as illustrated in Fig. 2. Specially, at each time

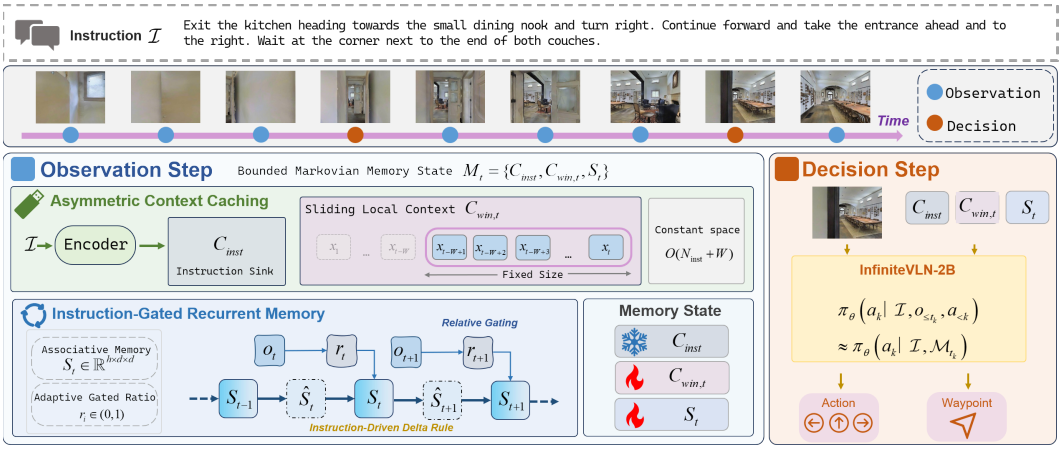

Figure 2: Overview of the InfiniteVLN framework. InfiniteVLN decouples inference process into an observation step and a decision step. Instruction is first encoded as a static instruction sink. During the observation step, the asymmetric context and instruction-gated recurrent memory are updated in real-time; during the decision step, action chunks or waypoints are generated. This approach significantly enhances inference efficiency while substantially reducing memory usage.

step t, we define a bounded Markovian memory state $\mathcal{M}_t$ as a fixed-capacity set:

$$\mathcal{M}_t = \{C_{\text{inst}}, C_{\text{win},t}, S_t\}, \tag{1}$$

where C_{inst} denotes the persistent instruction cache, $C_{\text{win},t}$ is the sliding local context, and S_t denotes the long-term recurrent memory states. Together, these components provide a bounded, instruction-conditioned representation of the accumulated navigation evidence, approximating the task-relevant belief state without retaining the complete observation history. Building upon this formulation, we decouple conventional VLM inference into two distinct processes. The **observation step** receives new physical observations and updates the memory state. Upon receiving an observation o_t, the model first encode it into visual features X_t, and then applies a state transition function f_{obs} to update the memory state:

$$\mathcal{M}_t = f_{\text{obs}}(\mathcal{M}_{t-1}, X_t). \tag{2}$$

The **decision step**, in contrast, generates navigation actions conditioned on the current memory state. Specifically, the navigation policy is approximated as

$$\pi_\theta\left(a_k \mid \mathcal{I}, o_{\le t_k}, a_{<k}\right) \approx \pi_\theta\left(a_k \mid \mathcal{I}, \mathcal{M}_{t_k}\right), \tag{3}$$

where π_θ denotes the navigation policy, t_k is the observation time corresponding to the k-th decision, and $o_{\le t_k}$ and $a_{<k}$ denotes the historical observations and previously executed actions, respectively. Importantly, the decision step operates on a temporary cache and leaves the persistent memory state unchanged, preventing transient decision-specific information from interfering with the accumulated visual evidence maintained within the long-term memory.

This asymmetric state-access mechanism fundamentally distinguishes InfiniteVLN from conventional autoregressive VLN inference. Instead of processing physical observations, reasoning tokens, and generated actions as a unified sequence, InfiniteVLN explicitly separates state acquisition from state querying. Observation events update the persistent memory state with incoming visual evidence, whereas decision events access this state in a read-only manner to generate navigation actions. By preventing decision-generated information from being recursively written into persistent memory, this design maintains an observation-grounded recurrent state that supports continuous perception and subsequent decision-making.

3.2 Asymmetric Context Caching

In streaming VLN, the memory state $\mathcal{M}_t$ must maintain a bounded approximation of the belief state b_t under the POSMDP formulation. However, the instruction and visual observations exhibit fundamentally different temporal characteristics. The instructions remains invariant throughout the entire

episode, providing global semantic constraints for interpreting observations and guiding subsequent decisions. In contrast, visual observations and executed actions evolve continuously over time and exhibit substantial spatiotemporal redundancy. Existing VLM-based approaches often retain instructions and historical observations symmetrically, retaining the full history in an ever-growing KV cache and thereby incurring substantial inference latency and memory overhead. To exploit this temporal asymmetry, we introduce Asymmetric Context Caching (ACC), which explicitly separates persistent task semantics from dynamic local observations by modeling them as a static instruction sink C_{inst} and a sliding local context $C_{\text{win},t}$, respectively.

Persistent Instruction Sink. Since the instruction $\mathcal{I}$ remains globally valid throughout the navigation episode, we encode it only once at the beginning and store its key-value representations in a persistent cache:

$$C_{\text{inst}} = \text{Encoding}(\mathcal{I}) = \{K_{\text{inst}}, V_{\text{inst}}\}. \tag{4}$$

At subsequent time steps, C_{inst} remains fixed and serves as a read-only semantic reference for incoming observations. Newly arriving visual tokens can attend to the instruction sink via cross-modal attention, while the sink itself remains fixed and is excluded from sequence updates. This design ensures that global task semantics remain accessible throughout navigation, independently of the evolving observation history.

Sliding Local Context. For the continuous observation stream $X_{1:t}$, we maintain a sliding local context window $C_{\text{win},t}$ with capacity W. At each observation step, the oldest cached entries are discarded as needed, while newly encoded observation and action information is appended to the retained context:

$$C_{\text{win},t} = \text{Concat}\big(C_{\text{win},t-1}[\text{stride}:], \text{Encoding}(x_t, a_{t-1})\big). \tag{5}$$

A direct sliding-window implementation can introduce positional inconsistency because cached keys are tied to their original positional phases after rotary positional encoding. To address this issue, ACC adopts a position-reconstructable caching strategy. The model stores raw key representations K_{raw} before applying rotary transformations and reconstructs their positional embeddings only when the cached context is retrieved. When the local window shifts, the retained observations are assigned window-relative temporal positions while preserving their intra-observation spatial coordinates. The corresponding rotary embeddings are then reconstructed on-the-fly without re-encoding historical observations. In this way, ACC maintains consistent relative spatiotemporal relations within the bounded local context while allowing the cache to be safely truncated and reused.

During attention-based retrieval, the interaction between the current query Q_t and the cached context is governed by:

$$\text{Attn}(Q_t, C_t) = \text{Softmax}\left(\frac{Q_t \cdot \mathcal{P}_{\text{gen}}(K_{\text{raw},t-W:t}, \Phi_t)^\top}{\sqrt{d}}\right) V_{t-W:t}, \tag{6}$$

where $\mathcal{P}_{\text{gen}}$ denotes a generalized positional encoding operator, and Φ_t denotes the dynamically reconstructed multimodal positional indices within the current local window. In our implementation, we adopt Qwen 3.5 (Qwen Team, 2026) as the foundation model, $\mathcal{P}_{\text{gen}}$ is instantiated as Partial-MRoPE. Here, Specifically, the temporal coordinates of retained observations are re-indexed relative to the current window, while their intra-observation spatial coordinates are preserved. This position-reconstructable cache decouples visual content from transient sequence positions, enabling bounded context shifting without sacrificing local spatiotemporal consistency.

This asymmetric architectural design significantly mitigates the computational and memory overhead inherent in streaming inference. ACC bounds the KV-cache memory footprint to $O(N_{\text{inst}}+W)$, independently of the navigation horizon, and with the computational complexity reduced to $O(W)$. This asymmetric caching design enables constant-memory streaming inference and bounded per-query computation, providing a foundation for efficient, low-latency navigation over extended temporal horizons.

3.3 Instruction-Gated Recurrent Memory

Although ACC preserves the spatiotemporal consistency of recent observations within a bounded local context, its sliding-window mechanism inevitably discards historical observations as the navigation horizon extends. However, observations outside the current window may contain critical

landmarks, spatial relations, or intermediate task evidence that remains essential for subsequent decisions. Consequently, relying solely on a bounded KV cache is insufficient to preserve task-relevant information over extended temporal horizons.

To address this limitation, we introduce Instruction-Gated Recurrent Memory (IGRM), which reformulates long-term memory maintenance as an *instruction-conditioned evidence consolidation* process rather than indiscriminate compression of the observation history. The central idea is to adaptively modulate the memory updates according to the instruction relevance of incoming visual information. Specifically, visual tokens that provide evidence aligned with the navigation instruction should receive greater emphasis in long-term memory updates, whereas redundant or instruction-irrelevant content should contribute less.

Instruction-Conditioned Gating. For the i-th visual token in the t-th observation, let $h_{t,i}$ denote its hidden representation. We estimate its instruction relevance from the attention distribution between the visual token and the persistent instruction sink. Let $\mathcal{I}$ denote the set of instruction tokens, and let $A^h_{(t,i),j}$ represent the attention weight from visual query $x_{t,i}$ to the j-th instruction token under the h-th attention head of a selected attention layer. We define the instruction attention mass as

$$m_{t,i} = \frac{1}{H} \sum_{h=1}^{H} \sum_{j \in \mathcal{I}} A^h_{(t,i),j}, \tag{7}$$

where H denotes the number of attention heads in the selected layer. To characterize the relative instruction relevance of each visual token within the current observation, IGRM computes an adaptive recurrent update gate:

$$r_{t,i} = \sigma\left(w^\top \mathrm{LN}(h_{t,i}) + \alpha \log \frac{m_{t,i} + \epsilon}{\bar{m}_t + \epsilon} + b \right), \tag{8}$$

where $\bar{m}_t = \frac{1}{N_t} \sum_{i=1}^{N_t} m_{t,i}$ is the mean instruction attention mass over the N_t visual tokens in the current observation. Here, $r_{t,i} \in (0, 1)$ denotes the adaptive update gate, $\sigma(\cdot)$ is the sigmoid function, w and b are learnable parameters, α controls the contribution of attention-derived relevance, and ϵ ensures numerical stability. By combining content evidence with instruction-conditioned attention, IGRM adaptively regulates the contribution of incoming visual tokens to long-term memory updates, assigning greater write strength to information that is more relevant to the instruction.

The instruction-conditioned gate is computed in a causal, layer-wise manner to preserve the sequential dependencies of recurrent memory updates. Specifically, let ℓ_a denote an ACC attention layer from which instruction-attention statistics are extracted. The resulting relevance gate $r^{(\ell_a)}_{t,i}$ modulates only the recurrent GDN layers until a new relevance estimate becomes available from the next ACC layer. Recurrent layers preceding the first available instruction gate follow their original GDN transitions.

Instruction-Gated Memory Update. IGRM maintains a fixed-capacity associative memory through Delta Rule-based recurrent updates. For each incoming visual token, the candidate memory state is computed as

$$\widehat{S}_{t,i} = \gamma_{t,i} S_{t,i-1} + \beta_{t,i} \left(v_{t,i} - \gamma_{t,i} S_{t,i-1} k_{t,i} \right) k_{t,i}^\top, \tag{9}$$

followed by an instruction-gated state update:

$$S_{t,i} = (1 - r_{t,i}) S_{t,i-1} + r_{t,i} \widehat{S}_{t,i}. \tag{10}$$

Here, $S_{t,i-1}$ denotes the associative memory state before processing the current token, while $k_{t,i}$ and $v_{t,i}$ are its projected key and value representations, respectively. $\beta_{t,i}$ is the step size of the Delta Rule update, and $\gamma_{t,i}$ governs forgetting in the underlying recurrent dynamics. The Delta Rule update corrects the existing memory according to the residual between the current value $v_{t,i}$ and the historical prediction $\gamma_{t,i} S_{t,i-1} k_{t,i}$. The instruction-conditioned gate $r_{t,i}$ further modulates the transition from the previous state to the candidate state, allowing instruction-relevant tokens to exert a stronger influence on the long-term memory while reducing the impact of redundant or irrelevant observations. Since the associative memory has fixed capacity and is updated recurrently, its storage complexity remains $O(1)$ with respect to the navigation horizon.

Table 1: Comparison results with SOTA methods on the Val-Unseen dataset for R2R-CE.

Method	Observation			R2R-CE				Mem.↓
	RGB	Pano.	Odo.	NE ↓	OS ↑	SR ↑	SPL ↑	
HPN+DN (Krantz et al., 2021)		✓	✓	6.31	40.0	36.0	34.0	–
VLN BERT (Hong et al., 2022)		✓	✓	5.74	53.0	44.0	39.0	–
CMA (Hong et al., 2022)		✓	✓	6.20	52.0	41.0	36.0	–
Reborn (An et al., 2022)		✓	✓	5.40	57.0	50.0	46.0	–
Ego^2-Map (Hong et al., 2023)		✓	✓	5.54	56.0	47.0	41.0	–
DreamWalker (Wang et al., 2023)		✓	✓	5.53	59.0	49.0	44.0	–
ETPNav (An et al., 2024)		✓	✓	4.71	65.0	57.0	49.0	–
Seq2Seq (Krantz et al., 2020)	✓			7.77	37.0	25.0	22.0	–
RGB-CMA (Krantz et al., 2020)	✓			9.55	10.0	5.0	4.0	–
AG-CMTP (Chen et al., 2021)		✓		7.90	39.0	23.0	19.0	–
R2R-CMTP (Chen et al., 2021)		✓		7.90	38.0	26.0	22.0	–
LAW (Raychaudhuri et al., 2021)	✓		✓	6.83	44.0	35.0	31.0	–
CM2 (Georgakis et al., 2022)	✓			7.02	41.0	34.0	27.0	–
WS-MGMap (Chen et al., 2022)	✓			6.28	47.0	38.0	34.0	–
ETPNav+FF (Wang et al., 2024b)	✓			5.95	55.8	44.9	30.4	–
AO-Planner (Chen et al., 2024)		✓		5.55	59.0	47.0	33.0	–
NaVid-7B (Zhang et al., 2024b)	✓			5.47	49.0	37.0	35.0	≥17G
Uni-NaVid-7B (Zhang et al., 2025b)	✓			5.58	53.3	47.0	42.7	≥17G
NaVILA-7B (Cheng et al., 2025)	✓			5.22	62.5	54.0	49.0	≥17G
NavFoM-7B (Zhang et al., 2025a)	✓			5.01	64.9	56.2	51.2	≥17G
JanusVLN-8.2B (Zeng et al., 2025)	✓			4.78	65.2	60.5	56.8	21G
EfficientVLN-4B (Zheng et al., 2025)	✓			4.18	**73.7**	64.2	55.9	≥13G
DualVLN-7.1B (Wei et al., 2026)	✓			4.05	70.7	64.3	58.5	20G
DecoVLN-7B (Xin et al., 2026b)	✓			5.01	63.5	56.3	50.5	19G
StreamVLN-7B (Wei et al., 2025)	✓			4.98	64.2	56.9	51.9	19G
InternVLA-N1-8.3B (Team, 2025)	✓			4.83	63.3	58.2	54.0	21G
DecoVLN-3B (Xin et al., 2026b)	✓			7.03	45.0	32.7	28.9	9G
InfiniteVLN-2B	✓			**4.02**	71.2	**65.1**	**59.3**	**5.5G**

Through IGRM mechanism, long-term memory updates are jointly determined by incoming visual content and its relevance to the navigation instruction. It selectively consolidates task-relevant landmarks, target regions, and spatial relations while reducing the accumulation of repetitive background information and transient visual variations. Consequently, S_t provides a compact approximation to the task-relevant belief state, supporting long-horizon navigation without memory growth over time.

4 Experiments

4.1 Experimental Setup

Following existing literature (Zhang et al., 2025b; Xin et al., 2026b; Wei et al., 2026), we evaluate InfiniteVLN on two standard continuous VLN benchmarks: R2R-CE (Anderson et al., 2018) and RxR-CE (Ku et al., 2020). Both benchmarks are built upon VLN-CE using the Habitat simulator and Matterport3D (Chang et al., 2017) indoor scenes, requiring agent to accomplish goal-oriented navigation in continuous environments guided by natural language instructions. In addition to these standard benchmarks, we systematically construct a Long-Trajectory VLN evaluation set to assess the model's stability and robustness under extended navigation horizon and long-context conditions.

4.2 Main Results

Table 1 and Table 2 report the results on the VLN-CE R2R and RxR benchmarks under the Val-Unseen setting. We compare InfiniteVLN with both multi-sensor waypoint prediction methods and recent VLM-based VLN approaches, and additionally report peak GPU memory usage for VLM-based models. Under the same benchmark setting, InfiniteVLN achieves competitive or superior navigation performance while using substantially fewer parameters and a much smaller memory footprint than existing VLM-based methods. Although architectural designs such as dual-

Table 2: Comparison with SOTA methods on RxR-CE Val-Unseen split.

Method	RGB	Pano.	Odo.	NE ↓	SR ↑	SPL ↑	nDTW ↑	Mem. ↓
VLN BERT (Hong et al., 2022)		✓	✓	8.98	27.0	22.6	46.7	–
CMA (Hong et al., 2022)		✓	✓	8.76	26.5	22.1	47.0	–
Reborn (An et al., 2022)		✓	✓	5.98	48.6	42.0	63.3	–
ETPNav (An et al., 2024)		✓	✓	5.64	54.7	44.8	61.9	–
Seq2Seq (Krantz et al., 2020)	✓			12.10	13.9	11.9	30.8	–
LAW (Raychaudhuri et al., 2021)	✓		✓	10.90	8.0	8.0	38.0	–
ETPNav+FF (Wang et al., 2024b)	✓		✓	8.79	25.5	18.1	–	–
AO-Planner (Chen et al., 2024)		✓		7.06	43.3	30.5	50.1	–
Uni-NaVid-7B (Zhang et al., 2025b)	✓			6.24	48.7	40.9	–	≥17G
NaVILA-7B (Cheng et al., 2025)	✓			6.77	49.3	44.0	58.8	≥17G
NavFoM-7B (Zhang et al., 2025a)	✓			5.51	57.4	49.4	60.2	≥17G
JanusVLN-8.2B (Zeng et al., 2025)	✓			6.06	56.2	47.5	62.1	21G
EfficientVLN-4B (Zheng et al., 2025)	✓			**3.88**	67.0	54.3	68.4	≥13G
DualVLN-7.1B (Wei et al., 2026)	✓			4.58	61.4	51.8	70.0	20G
DecoVLN-7B (Xin et al., 2026b)	✓			5.73	54.2	46.3	63.5	19G
StreamVLN-7B (Wei et al., 2025)	✓			6.22	52.9	46.0	61.9	19G
InternVLA-N1-8.3B (Team, 2025)	✓			5.91	53.5	46.1	65.3	21G
InfiniteVLN-2B	✓			4.32	**67.8**	**55.6**	**71.8**	**5.5G**

systems (Wei et al., 2026; Team, 2025) have substantially improved VLN performance, these methods predominantly rely on restricted historical sliding windows or external high-latency planners. In contrast, InfiniteVLN keeps peak memory usage bounded at approximately 5.5 GB, about one quarter of InternVLA (Team, 2025). This exceptional memory efficiency facilitates seamless deployment on resource-constrained physical robots and edge devices with limited computational and communication capabilities.

4.3 Long Horizon Navigation

Table 3: Performance on long-horizon navigation.

Method	NE↓	OS↑	SR↑	Mem.↓
DecoVLN (Xin et al., 2026b)	6.28	42.5	36.9	19G
InternVLA-N1 (Team, 2025)	5.29	53.6	45.3	21G
Ours	**4.93**	**55.8**	**50.1**	**5.5G**

Standard R2R-CE and RxR-CE episodes are relatively short, making it difficult to evaluate VLM-based VLN methods under long-context conditions. To rigorously evaluate the model's robustness in extended temporal scenarios, we construct a Long-Trajectory Navigation validation set based on the R2R and RxR benchmarks. *Please refer to the Appendix E for the details of construction protocol.* As illustrated in the Table 3, our algorithm significantly outperforms existing baselines while simultaneously requiring a markedly lower memory footprint.

4.4 Ablation Studies

We conduct comprehensive ablation studies to assess each architectural component on long-horizon dataset with the results summarized in the Table 4. Standard full-attention caching requires the model to retain the entire KV cache, resulting in rapidly increasing memory usage and $O(N^2)$ attention cost, which leads to high inference latency. ACC substantially reduces storage overhead, but its sliding-window truncation can lose global context and degrade performance. Integrating IGRM mitigates this forgetting by preserving instruction-relevant evidence in a bounded recurrent state, improving long-term belief maintenance. The further integration of the IGRM module effectively resolves this historical memory forgetting problem in long-trajectory navigation, successfully preserving critical evidence and ensuring robust long-term belief maintenance.

Sliding Local Context. We compare the proposed sliding local context with vanilla attention and standard sliding attention, as summarized in the Table 5. There are many visual aliases and repetitive structures in VLN environments, such as similar corridors and doors. The result indicates that sliding attention relies solely on local windows, making it prone to positional confusion in repetitive scenarios. Although ACC also employs a local window to process recent observations, its continuous retention of directive semantics enables the model to more reliably determine whether current local observations are relevant to the task objectives.

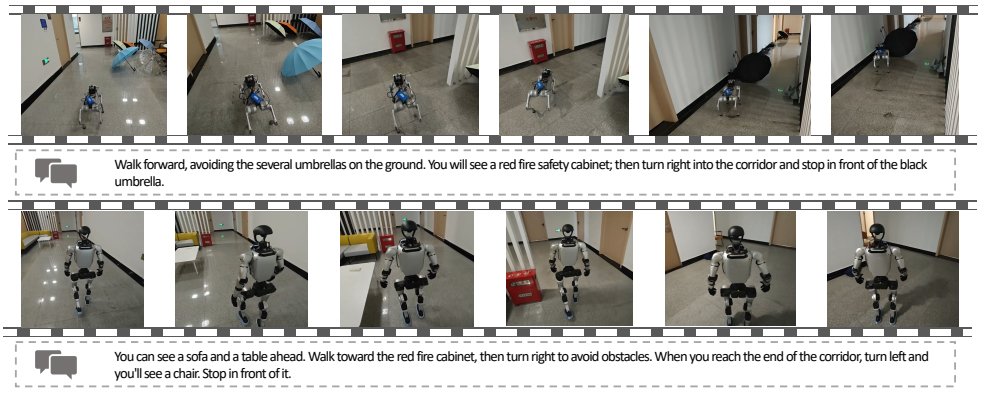

Figure 3: Real-world deployment of InfiniteVLN on Unitree Go2 and Unitree G1 robots.

Table 4: Ablation study of different modules.

ACC	IGRM	OS↑	SR↑	NE↓	Mem.↓
		50.9	43.8	5.32	10G
✓		47.2	41.6	5.65	5.5G
✓	✓	**55.8**	**50.1**	**4.93**	**5.5G**

Table 5: Comparison of attention modules.

Attention	NE↓	SR↑	Mem.↓
Vanilla	5.32	**43.8**	10G
Sliding	7.13	31.2	5.5G
ACC	**5.65**	41.6	**5.5G**

Table 6: Comparison of instruction gating strategies.

Gated Ratio (r)	NE↓	OS↑	SR↑
1	5.08	53.5	48.7
0.5	5.01	55.1	49.2
RAG	**4.93**	**55.8**	**50.1**

Instruction Gate. To validate the effectiveness of the relative-attention-mass gating (RAG) mechanism, Table 6 compares the navigation performance across different gating strategies. $r = 1$ means write all visual tokens into the long-term memory uniformly at maximum intensity. This strategy tends to fill the memory with irrelevant information, leading to memory saturation and pollution. When $r = 0.5$, it becomes impossible to distinguish important landmarks from background noise, resulting in insufficient recording of critical evidence and continuous accumulation of irrelevant information. Result demonstrates that our adaptive gating mechanism effectively measures the semantic salience of visual tokens and dynamically adjusts the writing intensity based on instruction relevance.

4.5 Real-World Deployment

In contrast to existing VLN methods that heavily rely on server-side inference, InfiniteVLN supports onboard deployment on low-power edge platforms such as the NVIDIA Jetson Orin. We deploy and evaluate the model on two robotic platforms: the Unitree G1 humanoid robot and the Unitree Go2 quadruped robot. Physical observations $\mathbf{O}$ are acquired using an Intel RealSense D455 camera and a Livox Mid-360 LiDAR. Real-world experiments show that InfiniteVLN can be transferred across different robot embodiments while maintaining efficient streaming inference. Specifically, the system sustains a streaming observation update rate of 2 FPS (observation step), and the decision step requires merely 1 second to generate the next action sequence.

5 Conclusion

This paper proposes InfiniteVLN, a constant-memory streaming framework for long-horizon navigation. We revisit VLN task as a partially observable decision-making problem and maintain a finite-capacity Markovian memory state that approximates the task-relevant belief state. InfiniteVLN decouples conventional single-step inference of VLMs into two distinct phases: observation step and decision step. Furthermore, we introduce Asymmetric Context Caching and Instruction-gated Recurrent Memory modules. Experiments across standard benchmarks, long-trajectory settings, and real-world robotic platforms show that InfiniteVLN achieves robust navigation with improved memory stability, lower memory usage, and reduced inference latency.

AI USE STATEMENT

In this work, we used generative AI tools solely for spelling checks and the correction of grammatical errors. We did not use generative AI tools for any other aspect of the research. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We have made efforts to ensure the reproducibility of this work. The main paper provides detailed descriptions of the proposed method, experimental setup, and evaluation protocols. Additional implementation details hyperparameter configurations are provided in the appendix. Together, these materials provide the information necessary to reproduce the main experimental results reported in this work.

REFERENCES

Dong An, Zun Wang, Yangguang Li, Yi Wang, Yicong Hong, Yan Huang, Liang Wang, and Jing Shao. 1st place solutions for rxr-habitat vision-and-language navigation competition. 2022.

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

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025a.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025b.

Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. In *International Conference on 3D Vision (3DV)*, pp. 667–676. IEEE, 2017.

Jiaqi Chen, Bingqian Lin, Xinmin Liu, Xiaodan Liang, and Kwan-Yee K Wong. Affordances-oriented planning using foundation models for continuous vision-language navigation. *arXiv preprint*, 2024.

Joya Chen, Ziyun Zeng, Yiqi Lin, Wei Li, Zejun Ma, and Mike Zheng Shou. Livecc: Learning video llm with streaming speech transcription at scale. In *CVPR*, 2025.

Kevin Chen, Junshen K Chen, Jo Chuang, Marynel Vázquez, and Silvio Savarese. Topological planning with transformers for vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

Peihao Chen, Dongyu Ji, Kunyang Lin, Runhao Zeng, Thomas H Li, Mingkui Tan, and Chuang Gan. Weakly-supervised multi-granularity map learning for vision-and-language navigation. In *Advances in Neural Information Processing Systems*, volume 35, pp. 27443–27456, 2022.

An-Chieh Cheng, Yandong Ji, Zhaojing Yang, Zaitian Gongye, Xueyan Zou, Jan Kautz, Erdem Bıyık, Hongxu Yin, Sifei Liu, and Xiaolong Wang. Navila: Legged robot vision-language-action model for navigation. *Robotics: Science and Systems*, 2025.

Georgios Georgakis, Karl Schmeckpeper, Karan Wanchoo, Soham Dan, Eleni Miltsakaki, Dan Roth, and Kostas Daniilidis. Cross-modal map learning for vision and language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Yicong Hong, Zun Wang, Qi Wu, and Stephen Gould. Bridging the gap between learning in discrete and continuous environments for vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Yicong Hong, Yang Zhou, Ruiyi Zhang, Franck Dernoncourt, Trung Bui, Stephen Gould, and Hao Tan. Learning navigational visual representations with semantic map supervision. 2023.

Yuheng Ji, Huajie Tan, Jiayu Shi, Xiaoshuai Hao, Yuan Zhang, Hengyuan Zhang, Pengwei Wang, Mengdi Zhao, Yao Mu, Pengju An, et al. Robobrain: A unified brain model for robotic manipulation from abstract to concrete. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1724–1734. IEEE, 2025.

Jacob Krantz, Erik Wijmans, Arjun Majundar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision and language navigation in continuous environments. 2020.

Jacob Krantz, Aaron Gokaslan, Dhruv Batra, Stefan Lee, and Oleksandr Maksymets. Waypoint models for instruction-guided navigation in continuous environments. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

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*, pp. 4392–4412, 2020.

Yifei Li, Junbo Niu, Ziyang Miao, Chunjiang Ge, Yuanhang Zhou, Qihao He, Xiaoyi Dong, Haodong Duan, Shuangrui Ding, Rui Qian, Pan Zhang, Yuhang Zang, Yuhang Cao, Conghui He, and Jiaqi Wang. Ovo-bench: How far is your video-llms from real-world online video understanding?, 2025. URL `https://arxiv.org/abs/2501.05510`.

Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. Video-llava: Learning united visual representation by alignment before projection. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 5971–5984, 2024.

Junming Lin, Zheng Fang, Chi Chen, Haoxuan Cheng, Zihao Wan, Fuwen Luo, Ziyue Wang, Peng Li, Yang Liu, and Maosong Sun. Streamingbench: Assessing the gap for mllms to achieve streaming video understanding. In *ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 12147–12151. IEEE, 2026.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL `https://qwen.ai/blog?id=qwen3.5`.

Sonia Raychaudhuri, Saim Wani, Shivansh Patel, Unnat Jain, and Angel X Chang. Language-aligned waypoint (law) supervision for vision-and-language navigation in continuous environments. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 4018–4028, 2021.

InternNav Team. InternVLA-N1: An open dual-system navigation foundation model with learned latent plans, 2025.

Haibo Wang, Bo Feng, Zhengfeng Lai, Mingze Xu, Shiyu Li, Weifeng Ge, Afshin Dehghan, Meng Cao, and Ping Huang. Streambridge: Turning your offline video large language model into a proactive streaming assistant. *Advances in Neural Information Processing Systems*, 38:132332–132359, 2026.

Hanqing Wang, Wei Liang, Luc Van Gool, and Wenguan Wang. Dreamwalker: Mental planning for continuous vision-language navigation. 2023.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024a.

Zihan Wang, Xiangyang Li, Jiahao Yang, Yeqi Liu, and Shuqiang Jiang. Sim-to-real transfer via 3d feature fields for vision-and-language navigation. In *Conference on Robot Learning*, pp. 2982–2995, 2024b.

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*, 2025.

Meng Wei, Chenyang Wan, Jiaqi Peng, Xiqian Yu, Yuqiang Yang, Delin Feng, Wenzhe Cai, Chenming Zhu, Tai Wang, Jiangmiao Pang, and Xihui Liu. Ground slow, move fast: A dual-system foundation model for generalizable vision-and-language navigation. In *International Conference on Learning Representations*, 2026.

Zihao Xin, Wentong Li, Yixuan Jiang, Ziyuan Huang, Bin Wang, Piji Li, Jianke Zhu, Jie Qin, and Shengjun Huang. Agentvln: Towards agentic vision-and-language navigation. *arXiv preprint arXiv:2603.17670*, 2026a.

Zihao Xin, Wentong Li, Yixuan Jiang, Bin Wang, Runmin Cong, Jie Qin, and Shengjun Huang. Decovln: Decoupling observation, reasoning, and correction for vision-and-language navigation. *arXiv preprint arXiv:2603.13133*, 2026b.

Jin Xu, Zhifang Guo, Hangrui Hu, Yunfei Chu, Xiong Wang, Jinzheng He, Yuxuan Wang, Xian Shi, Ting He, Xinfa Zhu, et al. Qwen3-omni technical report. *arXiv preprint arXiv:2509.17765*, 2025a.

Ruyi Xu, Guangxuan Xiao, Yukang Chen, Liuning He, Kelly Peng, Yao Lu, and Song Han. Streamingvlm: Real-time understanding for infinite video streams. *arXiv preprint arXiv:2510.09608*, 2025b.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving mamba2 with delta rule. In *International Conference on Learning Representations*, volume 2025, pp. 29687–29707, 2025.

Shuang Zeng, Dekang Qi, Xinyuan Chang, Feng Xiong, Shichao Xie, Xiaolong Wu, Shiyi Liang, Mu Xu, and Xing Wei. Janusvln: Decoupling semantics and spatiality with dual implicit memory for vision-language navigation. *arXiv preprint arXiv:2509.22548*, 2025.

Haoji Zhang, Yiqin Wang, Yansong Tang, Yong Liu, Jiashi Feng, Jifeng Dai, and Xiaojie Jin. Flash-vstream: Memory-based real-time understanding for long video streams. *arXiv preprint arXiv:2406.08085*, 2024a.

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. *Robotics: Science and Systems*, 2024b.

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.

Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. Video instruction tuning with synthetic data. *arXiv preprint arXiv:2410.02713*, 2024c.

Duo Zheng, Shijia Huang, Yanyang Li, and Liwei Wang. Efficient-vln: A training-efficient vision-language navigation model. *arXiv preprint arXiv:2512.10310*, 2025.

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, 2024.

APPENDIX

A EFFICIENT INFINITEVLN

Existing VLM-based VLN methods Team (2025); Wei et al. (2025); Xin et al. (2026b) typically concatenate the complete history of observations, language instructions, and action sequences into a long context input, relying on full attention and a complete KV cache to maintain historical information. Although this paradigm retains ample context, its computational and storage overhead scales continuously with the trajectory length as shown in Fig. 4. This inevitably leads to compounding inference latency, making it highly inefficient and fundamentally ill-equipped to support long-horizon real-time navigation. As illustrated in Fig. 5, upon the completion of the agent's movement, the entire historical context must be fed into the VLM all at once to predict the next action. The agent is forced to wait in place until the VLM generates the subsequent action before resuming execution. During the robot's physical movement, the VLM cannot process observations in a streaming fashion; consequently, the GPU remains idle until the robot stops, at which point the historical observations are concatenated into a complete context for input. This alternating "stop-and-wait" execution pattern between the GPU and the robot's physical execution unit severely degrades the continuity of the agent's navigation and the computational efficiency of the hardware.

In contrast, InfiniteVLN approaches the problem from the fundamental nature of streaming navigation, reformulating VLN as a state transition process that continuously updates a belief state. Its core objective is not merely to compress the input length, but to shift the reasoning paradigm of the large model from repetitive inference over long contexts to the recursive updating of a bounded state. Specifically, InfiniteVLN decouples the traditional single-step VLM inference into an observation step and a decision step. The former receives continuous visual observations at a high frequency, exclusively updating the persistent memory state; the latter performs temporary inference based on the current state when an action decision is required, crucially refraining from writing the generated tokens back into the long-term memory. Through this architectural design, the model is able to approximate the belief state in the POSMDP using a fixed-capacity state $\mathcal{M}_t$, thereby maintaining robust historical perception capabilities throughout long-horizon navigation.

The core innovation of InfiniteVLN lies in the fact that it does not simply adopt sliding window attention to reduce computational cost. Instead, it proposes a hierarchical perceptual memory architecture tailored for streaming VLN. Within this framework, Asymmetric Context Caching (ACC) and Instruction-Gated Recurrent Memory (IGRM) assume the respective responsibilities of fine-grained local perception and global long-term perception. Together, they constitute a navigation state system that is both highly efficient and capable of long-horizon understanding.

ACC is primarily responsible for maintaining a high-resolution local context by explicitly differentiating the temporal properties of natural language instructions and visual observations. The instruction sink ensures that global task semantics remain continuously accessible. While standard sliding window attention truncates early tokens as the window shifts, ACC preserves the instruction as a read-only cache over the long term. This enables the model to leverage the complete instruction to constrain the interpretation of current observations at every time step, which is instrumental in preventing instruction drift during extended navigation. Concurrently, the sliding local context retains high-resolution visual evidence within the most recent window, enabling the model to make fine-grained assessments of the current local environment. Ultimately, ACC transforms context caching from uniform historical truncation into persistent task semantic grounding coupled with local observation perception, elegantly resolving the dual challenges of precise local perception and low-latency inference in long-horizon streaming navigation.

B ADDITIONAL DETAILS OF INSTRUCTION-GATED RECURRENT MEMORY

Gated Delta Networks (GDNs) Yang et al. (2025) represent a class of linear recurrent architectures tailored for long-sequence modeling. Their core objective is to maintain a rapidly accessible associative memory within a fixed-capacity recurrent state, thereby circumventing the continuous, unbounded growth of the Key-Value (KV) cache with sequence length inherent in standard Transformers.

$$\bar{S}_{t-1} = \gamma_t S_{t-1}, \tag{11}$$

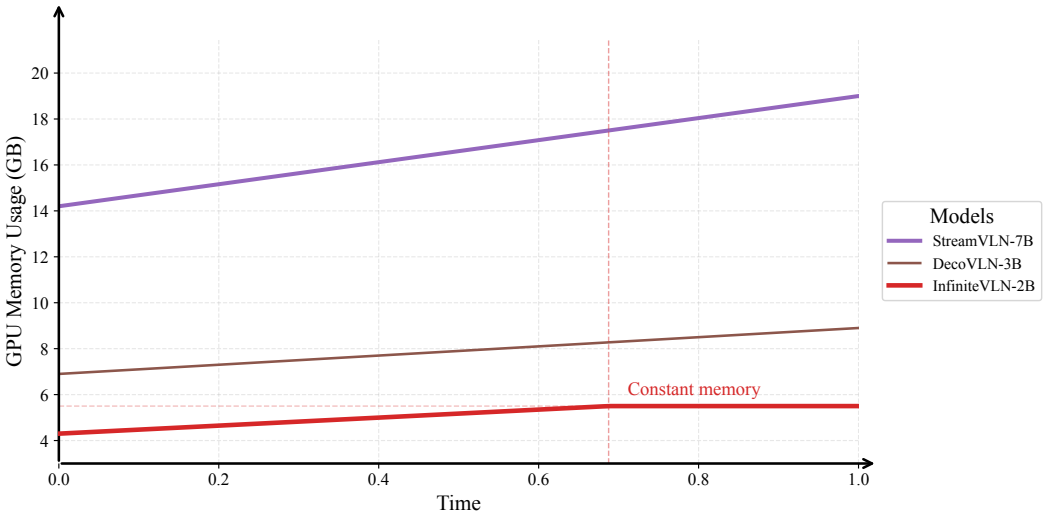

Figure 4: Memory usage over time during long-horizon navigation. Existing VLN methods, such as StreamVLN Wei et al. (2025), operate at 7B parameter scale and intrinsically rely on full attention alongside a complete KV cache to retain historical information. Consequently, their computational and storage overheads scale monotonically with the trajectory length. In contrast, InfiniteVLN is built upon a highly efficient 2B foundation model. By leveraging the proposed ACC and IGRM mechanisms to manage historical states, our framework achieves a strictly constant memory footprint during long-horizon navigation tasks. This remarkable memory efficiency facilitates its seamless deployment on resource-constrained edge devices.

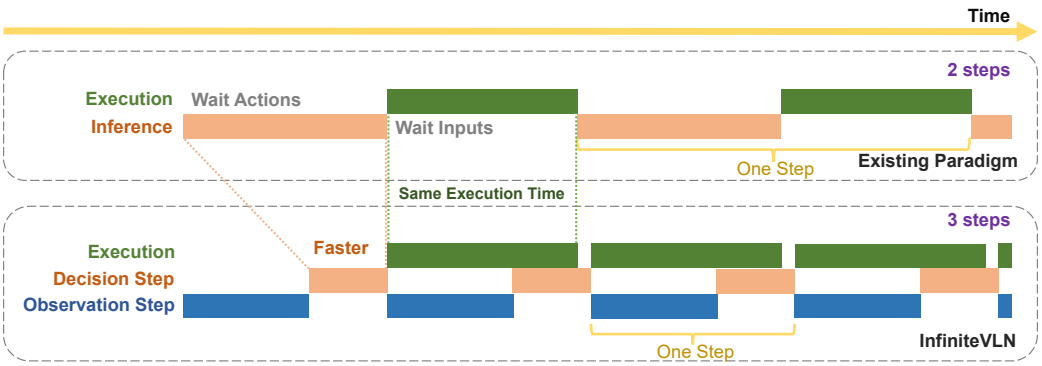

Figure 5: Efficiency comparison between InfiniteVLN and existing VLN paradigms. Existing VLN methods suffer from a rigid "stop-and-wait" paradigm where the robot must halt physical movement to wait for the model to process batched sensory data. In contrast, InfiniteVLN supports streaming perception and continuous inference. Our framework enables the GPU to seamlessly ingest real-time sensory streams and update the persistent memory state without stalling the hardware. This continuous integration completely eliminates idle waiting periods between perception and actuation. Consequently, assuming constant kinematic velocity, InfiniteVLN maximizes system throughput and effectively doubles the action execution rate per unit of time compared to conventional paradigms.

$$\hat{v}_t = \bar{S}_{t-1} k_t, \tag{12}$$

$$\Delta_t = v_t - \hat{v}_t, \tag{13}$$

$$S_t = \bar{S}_{t-1} + \beta_t \Delta_t k_t^\top. \tag{14}$$

Here, $\hat{v}_t = \bar{S}_{t-1} k_t$ denotes the current memory state's prediction of the value associated with the query key k_t; $\Delta_t = v_t - \hat{v}_t$ represents the prediction error; and $\beta_t \Delta_t k_t^\top$ signifies a low-rank correction applied to the associative matrix along the direction of the current key. However, the update

mechanism of standard GDNs fundamentally performs task-agnostic sequence compression. It automatically dictates the intensities of writing and forgetting primarily based on the content representation h_t of the current token, without explicitly distinguishing whether the token is relevant to a specific long-term task objective.

In the context of VLN, a substantial proportion of visual tokens in the navigation history originate from uninformative regions such as corridors, background walls, or irrelevant objects. If standard GDNs were employed to recursively update the long-term state, the model would indiscriminately write massive amounts of instruction-irrelevant local visual redundancy into S_t. This would inevitably lead to the saturation of the finite memory capacity with background noise, severely degrading the long-term retrievability of critical landmarks and target-relevant objects.

To alleviate the aforementioned limitations, we propose the IGRM module, which re-engineers the standard GDN from a generic sequence compression module into a specialized, instruction-conditioned evidence consolidation mechanism tailored for VLN. The core intuition is that, during navigation tasks, the long-term memory should not passively archive all historical visual inputs; rather, it should selectively and continuously integrate trajectory evidence that strictly aligns with the natural language instruction. Specifically, we formulate the bounded memory state of streaming VLN as follows:

$$\mathcal{M}_t = \{C_{\text{inst}}, C_{\text{win},t}, S_t\}, \tag{15}$$

where C_{inst} denotes the episode-level invariant instruction sink, $C_{\text{win},t}$ is the local sliding window, S_t represents the long-term associative memory matrix. Unlike standard GDNs that recursively model the complete, unsegmented token sequence directly, IGRM is explicitly embedded within the observation step. Consequently, it updates the long-term state exclusively upon receiving veridical physical observations:

$$\mathcal{M}_t = f_{\text{obs}}(\mathcal{M}_{t-1}, X_t). \tag{16}$$

IGRM seamlessly injects global instruction constraints into the navigation task, compelling the writing intensity to explicitly depend on the semantic correlation between the visual tokens and the instruction. Let $m_{t,i}$ denote the total attention mass received by the i-th observation token at time step t from the instruction sink C_{inst}, and let $\bar{m}_t$ represent the average attention mass of all tokens within the current frame. Formally, IGRM defines a relative attention mass gating mechanism as:

$$r_{t,i} = \sigma\left(w^\top \operatorname{LN}(h_{t,i}) + \alpha \log \frac{m_{t,i} + \epsilon}{\bar{m}_t + \epsilon}\right). \tag{17}$$

This gating scalar comprises two principal components. The first term, $w^\top \operatorname{LN}(h_{t,i})$, evaluates the intrinsic semantic richness of the token itself. The second term, $\log \frac{m_{t,i}+\epsilon}{\bar{m}_t+\epsilon}$, quantifies the token's instruction relevance relative to the baseline average of the current frame. When a visual token is highly correlated with the instruction, the attention mass it accumulates from C_{inst} typically exceeds the intra-frame average, thereby activating the write gate; conversely, for irrelevant tokens, the gate is effectively closed. This design entirely circumvents the need for an independent cross-modal alignment network, introducing zero additional inference overhead. Concurrently, by rigorously quantifying both token saliency and instruction relevance, this dynamic gating mechanism perfectly aligns with the stringent requirements of maintaining a goal-oriented belief state in VLN tasks.

Furthermore, IGRM forms a highly complementary synergy with the ACC module. While $C_{\text{win},t}$ retains the most recent W steps of observations to facilitate fine-grained local spatiotemporal reasoning, S_t actively absorbs navigation evidence possessing long-term value just as it is about to be truncated by the sliding window. Together, these two components architecturally correspond to the fundamental requirements of local consistency and global memory in VLN, respectively. Crucially, before any visual evidence slides out of the boundary of $C_{\text{win},t}$, it undergoes selective compression and is robustly injected into S_t via the IGRM gating mechanism.

C Implementation Details

We adopt Qwen3.5-2B Qwen Team (2026) as our foundation model, and the training pipeline is systematically divided into two stages: VLN Supervised Fine-Tuning (SFT) and Streaming SFT.



Figure 6: Visualization of outdoor search and long-horizon navigation.

During the first stage, we replace all standard attention modules with sliding window attention modules featuring a window size of $W = 8192$. The model acquires fundamental vision-language navigation capabilities by training on the AgentVLN-Instruction dataset Xin et al. (2026a). We optimize the network using AdamW with a global batch size of 128. A cosine annealing schedule is employed for learning rate decay, with the peak learning rate set to 2×10^{-5} and a warmup ratio of 0.03 to ensure optimization stability during the initial training phase. Throughout this stage, both the visual encoder and the cross-modal projection layer are strictly frozen to preserve their pre-trained representations.

During the Streaming SFT, complete navigation trajectories are unrolled into continuous temporal sequences comprising observation events and decision events, and we employ Low-Rank Adaptation (LoRA) for parameter-efficient fine-tuning. Each observation event exclusively processes the incoming visual observation to update the bounded memory state $\mathcal{M}_t$, whereas each decision event strictly reads from the current state to generate expert action supervision without modifying the persistent memory. We partition each trajectory into multiple contiguous segments. Each segment serves as an independent streaming training sample, within which the full computational graph is retained to execute backpropagation through time. Finally, to mitigate input distribution shift, we systematically interleave the standard SFT data from the first stage into the second-stage training mixture.

D Outdoor Deployment Experiments

We deployed the model on a real-world robotic platform and evaluated its long-horizon vision-language navigation and object search capabilities in outdoor environments. Compared to the short-trajectory indoor benchmarks, outdoor environments typically exhibit larger spatial scales, longer execution times, and higher visual redundancy. As shown in the Fig. 6, we separately tested the capabilities of outdoor object search and long-horizon navigation. Unlike standard VLN, which provides complete route descriptions, object search only specifies the target object or target category to be found. During extended navigation, a large volume of consecutive frames often contains highly repetitive background information such as roads and vegetation and so on. If full historical attention is employed, these redundant observations continuously occupy the context window and increase the computational burden. InfiniteVLN retains high-resolution temporal information only within a limited local window and compresses earlier history into a long-term state. This allows the agent to continuously execute long-distance navigation tasks within constrained GPU memory and computational resources, without the need to periodically flush historical context or reinitialize the model state.

E Evaluation Protocol

E.1 Construction of Long-Trajectory Navigation Validation Set

The construction protocol is defined as follows: First, from the original validation sets, we select two independent trajectories within the same scene—where the endpoint of the first trajectory is spatially proximate to the starting point of the second—and concatenate them to synthesize a contiguous, extended navigation path. Subsequently, we compute the agent's orientation at each waypoint; if the required heading variation exceeds 30 degrees, explicit turning actions are injected into the ground-truth action sequence to ensure physical realizability. Finally, we leverage the Qwen3 Xu et al. (2025a) to semantically fuse and rewrite the two original instructional segments, generating a singular, coherent long-horizon natural language instruction.

E.2 Evaluation Metrics

To comprehensively evaluate the agent's navigation performance, we adopt five widely used standard metrics: Success Rate (SR), Success weighted by Path Length (SPL), Navigation Error (NE), Oracle Success Rate (OS), and normalized Dynamic Time Warping (nDTW).

F Limitations

InfiniteVLN effectively achieves bounded compression of historical information via $C_{\text{win},t}$, S_t, significantly reducing both storage and computational overhead. However, certain fine-grained historical evidence may inadvertently be attenuated during the compression process, particularly in extremely long-horizon navigation tasks or scenarios necessitating complex backtracking. Furthermore, real-world robotic navigation intrinsically encompasses challenges related to low-level perception, precise localization, and physical kinematic constraints. Although InfiniteVLN provides a highly efficient state representation for high-level semantic decision-making, it fundamentally requires seamless integration with local planners, metric mapping modules, and continuous control policies to constitute a comprehensive, deployable system in the physical world. In the future, we intend to explore an updatable instruction sink tailored for dynamic human-robot interaction. In real-world applications, users may frequently modify objectives, provide supplementary instructions, or issue corrective feedback during an ongoing navigation episode. Future work will investigate extending the static C_{inst} into a dynamic task cache, enabling it to assimilate real-time linguistic feedback while concurrently preserving the overarching consistency of the historical task context.