

A ADDITIONAL IMPLEMENTATION DETAILS

This section complements the implementation details in the main paper with the training protocol and memory configuration used for the default MEMVLN configuration.

Data Statistics. We summarize the training data statistics in Tab. 5. Each instruction-conditioned demonstration is treated as one episode. For simulation training, the number of navigation samples counts the step-wise action-prediction targets after action merging; the trajectory summarization data is derived from the same episodes. For real-world post-training, we collect 24 demonstrations using an Intel RealSense D455 RGB-D camera mounted on the G1 robot for training. The D455 RGB sensor has a nominal horizontal field of view of 90°. Its RGB frames are captured at 1280 × 720 and resized to 640 × 480 before model processing to match the simulation input resolution.

Table 5: **Training data statistics.** Navigation samples count step-wise action-prediction targets after action merging. Simulation sample counts are rounded to the nearest thousand.

Setting	Dataset	Training Data		RGB Input		
		Episodes	Nav. Samples	Resolution	HFOV	Camera
Simulation	R2R Train	10,819	~334K	640 × 480	90°	Habitat-Sim
	RxR Train	19,996	~1.02M	640 × 480	90°	Habitat-Sim
Real-world PT	G1 demonstrations	24	1,053	640 × 480	90°	Intel RealSense D455

Streaming Episode Sampler. Training a stateful memory policy requires temporally coherent data loading. Standard random sampling breaks an episode into unrelated steps, preventing the memory bank from being updated in the same chronological order encountered at inference time. We therefore employ a streaming episode sampler that assigns complete episodes to each device and feeds the steps within each episode sequentially. This protocol allows each device to maintain continuous memory streams while supporting distributed training and task mixing.

Default Memory Configuration. Following the memory module formulation in the main paper, the memory bank stores the visual feature f_t of each observation together with its timestep t and relative pose z_t ; at policy inference, z_t is derived from the executed action history. We denote the maximum number of retained frame entries as N_b . Each frame is spatially compressed to at most N_f patch tokens using ToMe (Bolya et al., 2023). A new observation is first appended to the bank. If this increases the bank size to $N_b + 1$, frame merging consolidates the two consecutive entries with the highest average patch-level cosine similarity, reducing the bank size by one and restoring the capacity constraint. The memory perceiver then uses N_m learnable queries to retrieve a fixed-size set of memory tokens m_t . Before each memory readout, the current visual feature is added to the bank; the queries are conditioned only on the current timestep and action-derived relative pose, while the instruction is introduced downstream by the LLM. Since each query produces one output token, N_m is also the number of memory tokens injected into the VLM. The default configuration uses $N_f = 100$, $N_b = 100$, and $N_m = 300$. Additional settings are summarized in Tab. 6.

Simulation Training. As described in the main paper, we adopt Qwen3-VL-4B (Bai et al., 2025) as the VLM backbone and StreamVGGT-1B (Zhuo et al., 2026) as the geometry encoder. The visual encoder and geometry encoder remain frozen. We adapt the LLM with LoRA (Hu et al., 2022), while training the memory module and geometry adapter from scratch. More details are summarized in Tab. 7.

Real-world Post-training. The real-world demonstrations record both the model prediction and the human-corrected executed action. We post-train the policy using the same token-level cross-entropy objective as in simulation, while assigning a larger sample weight to steps requiring correction:

$$\mathcal{L}_{PT} = \frac{\sum_t w_t \mathcal{L}_t}{\sum_t w_t}, \quad \mathcal{L}_t = -\log p(a_t^* | m_t, \mathcal{H}_t, \mathcal{T}). \quad (9)$$

We set w_t to 3.0 for corrected non-stop actions, 0.5 for actions accepted without correction, and 2.0 for stop actions. This correction-weighted behavior cloning objective focuses adaptation on states where the deployed policy requires human intervention.

Table 6: **Default memory module configuration.** Table 7: **MEMVLN training hyperparameters.**

Hyperparameter	Setting	Category	Setting
Local observation window k	16	Hardware	$8 \times \text{A100 80G GPUs}$
Max tokens per frame N_f	100	Precision	BF16
Memory bank capacity N_b	100	Training epochs	1
Memory token count N_m	300	Per-device batch size	1
Frame consolidation	Sim-based merging	Gradient accumulation steps	1
Spatial embedding	Fourier features	Effective batch size	$1 \times 8 \times 1 = 8$
Temporal embedding	Sinusoidal embedding	Optimizer	AdamW
ToMe layers	2	Base learning rate	2×10^{-5}
ToMe attention heads	16	Weight decay	0.05
Perceiver layers	2	Warmup steps	400
Perceiver attention heads	16	LoRA rank / alpha	128 / 256

B INFERENCE LATENCY AND COMPUTATIONAL COST

Evaluation Protocol. We measure system-level cost using fixed-input policy replay on one NVIDIA RTX 4090 with batch size 1 and the configured BF16/FlashAttention 2 inference setup. We compare the Base policy with the complete MEMVLN pipeline to measure the combined system overhead of StreamVGGT and episodic memory. The replay uses three frozen R2R episode traces comprising 102 fixed policy decisions. Each variant is evaluated in two fresh processes after the same 20-decision warm-up and state reset, yielding 204 timed observations per variant, including 168 observations in which the recent-context window contains 16 frames. Timing starts from RGB arrays in CPU memory and includes message construction, model preprocessing, host-to-device transfer, policy generation, decoding, and action parsing. It excludes sensor capture, Wi-Fi transfer, environment stepping, and the low-level robot controller. This fixed-input diagnostic characterizes policy cost and is not used to compute navigation metrics.

Table 8: **Fixed-replay inference latency and peak resource usage on one RTX 4090.** Latency statistics pool two fresh-process repeats of the same 102 fixed decisions; N denotes timed observations rather than independent episodes. The 16-frame recent-context subset contains 168 timed observations per variant. Policy rate is 1000/mean latency (ms). Resource rows report the maximum across valid repeats. Process GPU is the peak PID-scoped memory reported by `nvidia-smi`, and RSS is the peak host resident set size of the evaluation process.

Metric	Base	Full MEMVLN
Timed observations, N	204	204
Mean latency (ms)↓	503.2	691.0
P50 latency (ms)↓	535.2	704.3
P95 latency (ms)↓	550.0	779.8
Policy rate (Hz)↑	1.99	1.45
16-frame observations, N	168	168
16-frame mean latency (ms)↓	534.9	731.7
CUDA peak allocated (GiB)↓	10.725	17.104
CUDA peak reserved (GiB)↓	11.234	21.078
Process GPU peak (GiB)↓	11.69	21.54
Host RSS peak (GiB)↓	3.401	8.666

End-to-End Latency. As shown in Tab. 8, the complete pipeline increases mean policy latency from 503.2 to 691.0 ms (37.3%), corresponding to a sequential high-level policy rate of 1.45 Hz on the tested GPU. For the 16-frame recent-context subset, mean latency is 534.9 ms for Base and 731.7 ms for Full MEMVLN, a 36.8% increase.

Table 9: **Memory capacity ablations on R2R-CE and RxR-CE.**

(a) Memory bank capacity								
N_b	R2R Val-Unseen				RxR Val-Unseen			
	NE↓	OSR↑	SR↑	SPL↑	NE↓	SR↑	SPL↑	nDTW↑
0	5.87	57.4	50.9	47.2	5.72	54.9	49.0	64.5
100	5.19	61.2	55.0	50.5	5.59	55.4	48.4	64.6
200	6.24	58.7	50.2	45.7	6.15	54.3	48.0	61.5

(b) Memory-token count								
N_m	R2R Val-Unseen				RxR Val-Unseen			
	NE↓	OSR↑	SR↑	SPL↑	NE↓	SR↑	SPL↑	nDTW↑
0	5.87	57.4	50.9	47.2	5.72	54.9	49.0	64.5
300	5.19	61.2	55.0	50.5	5.59	55.4	48.4	64.6
600	6.75	53.9	45.1	40.8	6.64	50.4	44.0	59.3

Fixed Token Interface. For one R2R decision with a 16-frame recent-context window, Base receives 2,819 input tokens and Full MEMVLN receives 3,126. In both cases, 15 history frames are padded to 16 and grouped into eight temporal units; these units and the current image each contribute 300 visual tokens, for $(8 + 1) \times 300 = 2,700$ visual tokens. Full MEMVLN additionally contributes 300 fixed memory tokens and 7 associated prompt/special tokens, a fixed increase of 307 tokens under the evaluated prompt format. Other text tokens depend on the instruction. The bounded raw bank is not concatenated into the VLM context, so this memory-side input increment does not grow with route length.

Control Frequency. The measured mean latency in Tab. 8 corresponds to a sequential high-level policy rate of 1.45 Hz under the tested configuration. The real-robot client uses a separate low-level PID control thread to send commands through the Unitree SDK, nominally configured at approximately 30 Hz. The RTX 4090 benchmark measures high-level policy inference. We use *streaming* to denote causal online processing with bounded state.

Training Configuration. The main paper reports that the simulation model has approximately 1B trainable parameters and is trained for one epoch, taking approximately two days on 8×A100 80G GPUs. Additional training hyperparameters are summarized in Tab. 7.

C ADDITIONAL RESULTS

C.1 MEMORY CAPACITY

Table 9 reports the memory bank capacity and memory-token count ablations on both benchmarks. The default configuration uses $N_b = 100$, $N_m = 300$, and $N_f = 100$ patch tokens per stored frame. The zero-capacity setting retains the geometry encoder and trajectory-summary supervision.

C.2 LONG-HORIZON ANALYSIS PROTOCOL

We apply the same instruction-based grouping to all three model variants over 3,669 RxR-CE Val-Unseen episodes in Fig. 3. For instruction grouping complexity, we count the total occurrences of commas and periods in each instruction as a punctuation-based proxy. The four bins contain 1–3, 4–6, 7–10, and more than 10 punctuation marks, comprising 239, 819, 1,113, and 1,498 episodes, respectively.

Instruction-type subsets are selected by case-insensitive keyword matching with word boundaries. The Back subset matches *turn around*, *back*, or *behind*, yielding 1,272 episodes. The Stop subset matches *stop*, *wait*, *end*, *hallway*, *hall*, or *corridor*, yielding 2,072 episodes. The Pass subset matches *pass*, *past*, *through*, or *cross*, yielding 1,410 episodes. An instruction enters a subset when it contains at least one corresponding keyword or phrase. These subsets are constructed independently and can overlap; All denotes the complete set of 3,669 episodes.

Table 10: **Real-world evaluation instructions.** We use the instruction IDs in subsequent results.

ID	Instruction
I1	Now you are facing a brown curtain. Turn right and walk pass an armchair towards the fire extinguisher. Turn left and walk toward the glass door in front of you to the right, that’s your destination.
I2	Now you are facing the reception. On your left you should see some brown sofas. Walk past the sofas to the handwriting on the wall. Stop at the door on your right.
I3	Now you are in a corridor. Keep moving forward. Once you pass the bookshelf on the right side, turn right and go to the sink in the corner.
I4	Go straight towards the glass windows and past the sofas on your right. After you reach the windows, turn right. Keep moving forward and stop at the second door on your right.
I5	Now you are in a corridor. On your left, there are some white cabinets. Go past them and then turn left. You should see a red fire extinguisher on the ground. That box is your destination.
I6	Now you are in a lab. On your right there are some wooden stakes. Move forward and pass the chairs in front of you. You should see a robot sitting on a chair. Walk past the robot and stop at the bucket next to the lamp.

Table 11: **Per-instruction real-world results.** Each cell reports SR / CR (%) over five repeated trials for the corresponding instruction and method. Bold marks the best nonzero SR and lowest CR in each row.

ID	Base	+Mem	MemVLN	MemVLN-PT
I1	20 / 40	40 / 60	80 / 20	60 / 80
I2	0 / 60	0 / 60	20 / 60	20 / 40
I3	40 / 60	80 / 60	40 / 40	80 / 60
I4	0 / 100	0 / 100	0 / 60	0 / 60
I5	20 / 20	0 / 40	20 / 40	60 / 40
I6	20 / 60	60 / 80	80 / 60	60 / 0

C.3 REAL-WORLD RESULTS

Evaluation Protocol. We provide a detailed breakdown of the real-world G1 evaluation summarized in Sec. 4.4. All 30 instruction-conditioned episodes were collected in the same physical scene and split at the episode level: 24 demonstrations are used for post-training and six distinct episodes are held out for evaluation. We evaluate Base, +Mem, MemVLN, and MemVLN-PT, repeating each held-out instruction five times for 30 trials per method. In addition to success rate (SR), we report collision rate (CR), the fraction of trials involving physical contact, contact-induced blockage, or contact-induced termination; CR measures contact safety independently of task completion.

Deployment Setup. During real-world deployment, policy inference runs on a host workstation equipped with a single NVIDIA RTX 4090 GPU. RGB observations captured by the robot-mounted Intel RealSense D455 are transmitted to the host over Wi-Fi; after the host completes model inference, the predicted action is sent back to the G1 robot for execution.

Overall Performance. As shown in Fig. 4, Base succeeds in 5 of 30 runs, corresponding to 16.7% SR. Adding the memory module increases success to 9 of 30 runs (30.0% SR), and the full MemVLN model further improves success to 12 of 30 runs (40.0% SR). After real-world post-training with 24 demonstrations, MemVLN-PT succeeds in 14 of 30 runs (46.7% SR), giving two additional successful runs over MemVLN. CR remains unchanged at 46.7% for MemVLN and MemVLN-PT.

Instruction-level Analysis. As shown in Tab. 11, MemVLN-PT improves SR over MemVLN on I3 and I5, leaves it unchanged on I2 and I4, and decreases it on I1 and I6. Post-training lowers CR on I2 and I6 and increases it on I1 and I3. All four methods obtain 0% SR on I4, which requires selecting the second door on the right after passing an intermediate door.

Remaining Challenges. Collision remains a challenge: MemVLN and MemVLN-PT both have a CR of 46.7%. Post-training improves aggregate SR but leaves aggregate CR unchanged. Future work will explore contact-aware control and richer supervision for fine-grained spatial grounding.

C.4 MORE QUALITATIVE RESULTS

Simulation Results. Figure 6 and 7 show R2R-CE and RxR-CE navigation rollouts, including selected observations and actions along multi-step routes.



Figure 6: Qualitative results of MEMVLN on R2R-CE.

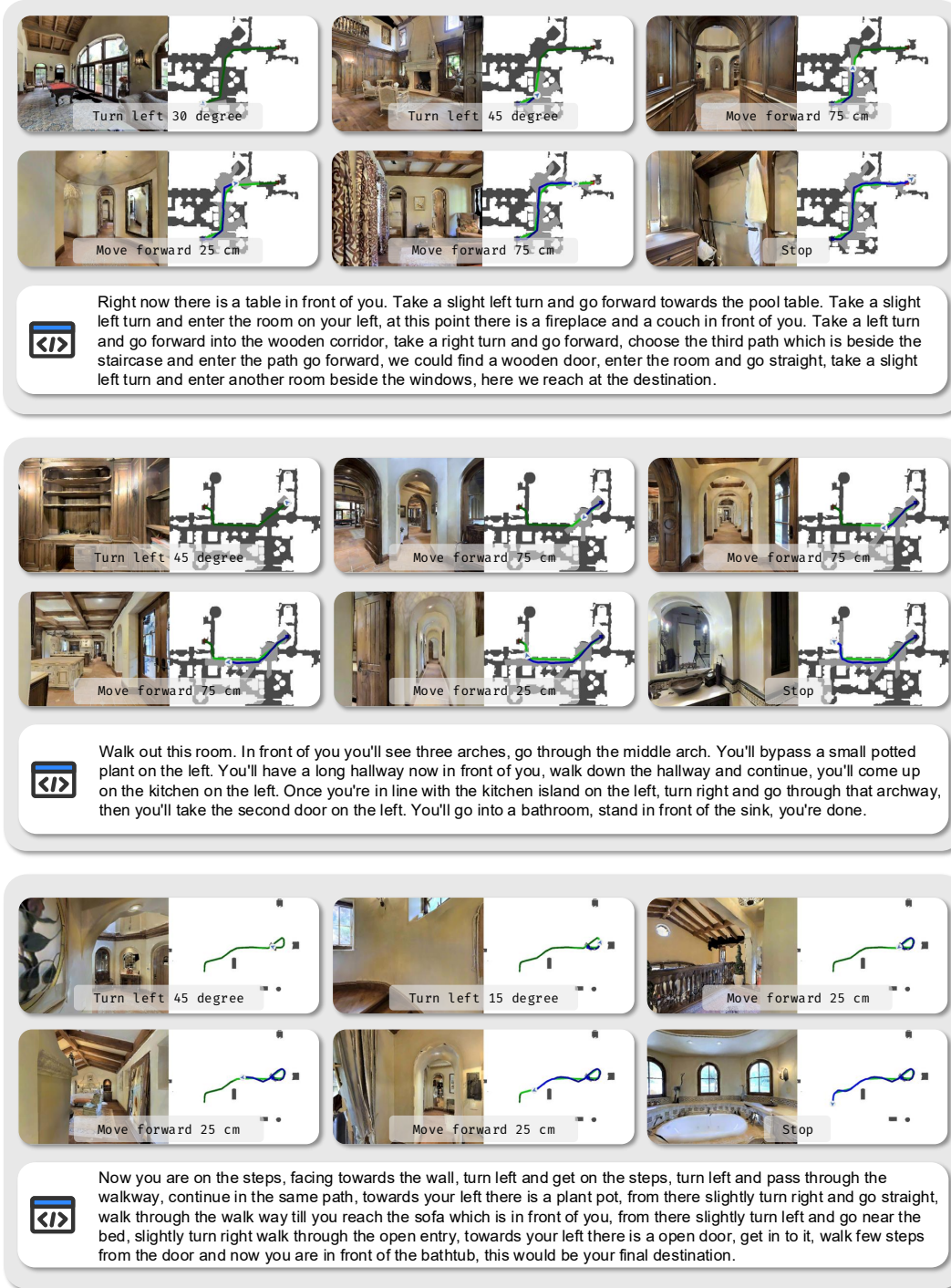


Figure 7: Qualitative results of MEMVLN on RxR-CE.

Real-world Results. Figure 8 shows two additional G1 navigation rollouts with their instructions, selected observations, and actions.

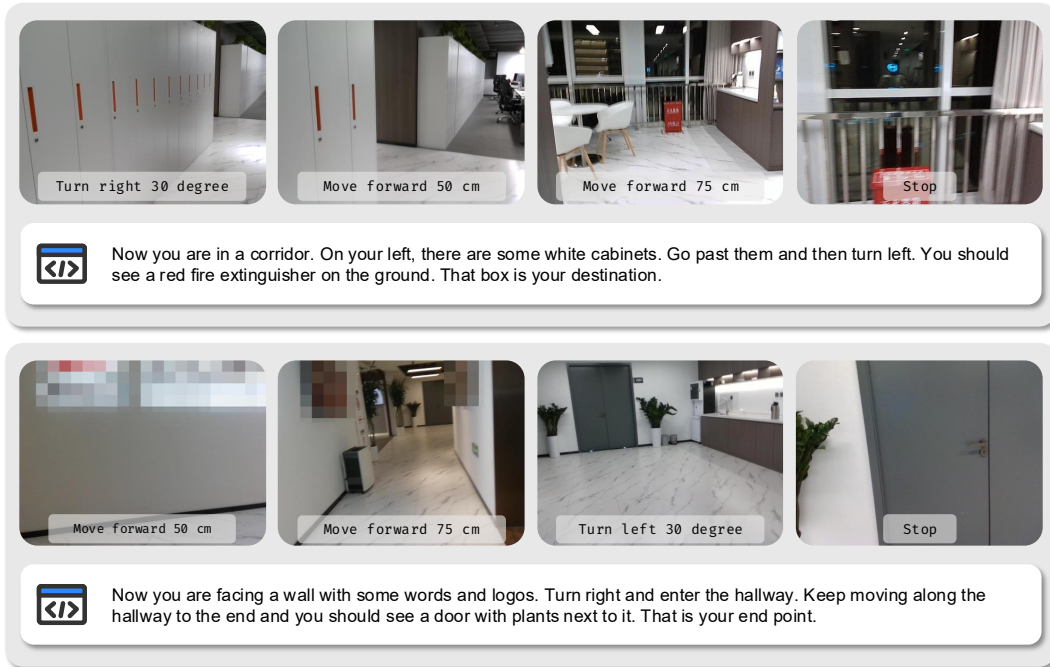


Figure 8: **Qualitative results of MEMVLN in the real world.** Selected observations and actions from two G1 navigation rollouts, together with their navigation instructions.