TRACK-MM: DUAL-MEMORY EMBODIED VISUAL TRACKING WITH LATENT MOTION PRIORS

Anonymous authors
Paper under double-blind review

ABSTRACT

Embodied visual tracking (EVT) is an essential capability for embodied agents, which needs persistent target tracking in dynamic, partially observable environments. The core challenge lies in maintaining target identity under distraction with limited computing and memory resources. To address the challenges, we propose Track-MM, a lightweight dual-memory framework for robust embodied visual tracking. Track-MM builds target-aware latent motion priors by leveraging pre-trained video foundation model representations via a lightweight target-aware motion adapter. The framework separately maintains appearance and motion memories with independent update gates to adapt to dynamic observations while retaining target identity. Benefiting from the lightweight design, Track-MM runs on resource-limited platforms with reduced parameters than mainstream methods. Evaluations on EVT-Bench demonstrate that our method strikes a favorable balance between tracking performance and computational efficiency. Real-world experiments further demonstrate promising zero-shot sim-toreal transfer, where the method maintains reliable tracking under varied lighting conditions on NVIDIA Orin platforms with practical deployment efficiency.

1 INTRODUCTION

Embodied visual tracking (Zhong et al., 2019; Zhang et al., 2024; Wang et al., 2025b; Wu et al., 2025; Zhong et al., 2024; Liu et al., 2025; Zhong et al., 2023) requires an agent to continuously track a target while producing safe and low-latency actions from egocentric observations in dynamic, partially observable environments. In practice, occlusion, target disappearance, and visually similar distractors frequently disrupt observation reliability, making stable target following difficult.

Early embodied tracking methods often formulate EVT as a reinforcement learning problem (Zhong et al., 2019; He et al., 2025; Huang et al., 2025; Zhong et al., 2023), learning control policies directly from interaction. However, these approaches rely on substantial task-specific training and remain limited in robustness under occlusion, distractors, and temporary target disappearance.

More recently, vision-language-action (VLA) paradigms have significantly advanced EVT in open-vocabulary and unstructured environments (Zhang et al., 2024; Wang et al., 2025b; Liu et al., 2025; Cheng et al., 2024; Yu et al., 2026), encoding observations and language instructions into a shared representation space with the reasoning and generalization capabilities of large foundation models (Bai et al., 2025; Achiam et al., 2023; Comanici et al., 2025; Grattafiori et al., 2024). This paradigm enables more flexible target specification and adaptability across diverse scenarios.

However, under partial observability, existing methods often fuse appearance and motion cues into a single history representation despite their distinct roles and reliability. Appearance preserves target identity, while motion provides short-term continuity under noisy or missing observations. Such homogeneous fusion can cause memory contamination, identity switches, and unstable recovery, motivating separate memory modeling of appearance and motion.

We propose Track-MM, a lightweight framework with separate appearance and motion memories for target identity and temporal continuity. A Target-Aware Motion Adapter (TMA) conditions pretrained video features on instruction-grounded target representations. Independent reliability-aware gates update the two memories, while a compact policy backbone and waypoint prediction head enable deployment on resource-constrained platforms.

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.

Experiments on EVT-Benchmark (Wang et al., 2025b) demonstrate that Track-MM achieves a well-balanced performance across success, tracking, and collision metrics, remaining competitive with methods several times its size. To further evaluate real-world deployment, we implement the system on a Unitree Go2-W quadruped robot powered by an NVIDIA Jetson Orin NX 8GB edge computing platform. Real-world experiments conducted under diverse lighting conditions demonstrate that Track-MM enables stable embodied tracking and effective zero-shot sim-to-real generalization under limited onboard computational resources.

Our contributions are summarized as follows:

- We propose Track-MM, a lightweight dual-memory framework that represents target identity and motion continuity with separate appearance and motion memories under partial observability.
- We introduce a target-aware latent motion prior and a dual-gated memory update mechanism. The former adapts pretrained video representations into target-specific temporal evidence, while the latter independently regulates appearance and motion memory updates under ambiguous observations.
- Our framework is efficient and lightweight, achieving well-balanced performance on EVT-Benchmark with only 1.2B parameters and demonstrating effective zero-shot sim-to-real transfer on resource-constrained platforms.

2 Related Work

2.1 Embodied Visual Tracking

A line of recent studies (Zhong et al., 2019; Chen et al., 2022; Wang et al., 2024a) combines vision-language foundation models (Comanici et al., 2025; Wang et al., 2024b) with reinforcement learning for embodied visual tracking. Representative methods such as AD-VAT (Zhong et al., 2019) and RSPT (Zhong et al., 2023) use structured visual features or hierarchical reinforcement learning to learn policies from environment interaction, but limited generalization and weak feature-policy alignment reduce their effectiveness in dynamic and partially observable environments. Another line of work studies the VLA paradigm (Zhang et al., 2024; Wang et al., 2025b; Liu et al., 2025; Yu et al., 2026), which maps multimodal inputs directly to action sequences using pre-trained foundation models. Although this formulation supports flexible embodied reasoning, large model capacities and memory footprints can hinder deployment on resource-constrained platforms. Track-MM instead builds on VLA with a lightweight memory mechanism for efficient onboard deployment.

2.2 Vision-Language-Action Models

Vision-language-action (VLA) architectures unify perception, language grounding, and decision-making within a single model (Cheng et al., 2024; Black et al., 2024; Kim et al., 2024). Uni-NaVid (Zhang et al., 2024) unifies multiple navigation tasks in a video-based formulation, while TrackVLA (Wang et al., 2025b) and TrackVLA++ (Liu et al., 2025) extend this paradigm to language-conditioned target following. Despite this integration, most VLA models lack an explicit representation of physical scene dynamics. Temporal evolution is implicitly embedded in the backbone, without a dedicated mechanism for target motion and its interaction with agent actions. Track-MM integrates latent motion priors to improve reasoning over target dynamics and action-conditioned temporal evolution in embodied visual tracking.

2.3 Video Foundation Models in Embodied Tasks

Video foundation models (Assran et al., 2025; Tong et al., 2022; Zhang et al., 2025a) learn transferable spatiotemporal representations from large-scale video data and increasingly serve as temporal priors for embodied perception and control. WorldVLN (Zhao et al., 2026) and Fast-WAM (Yuan et al., 2026) use autoregressive video backbones (Liu et al., 2026; Wan et al., 2025) as world models for navigation, with the latter removing future video generation at inference to reduce latency. However, generic video features often entangle target motion with ego-motion, background changes, and distractor dynamics, while their computational cost limits robotic deployment. Track-MM addresses

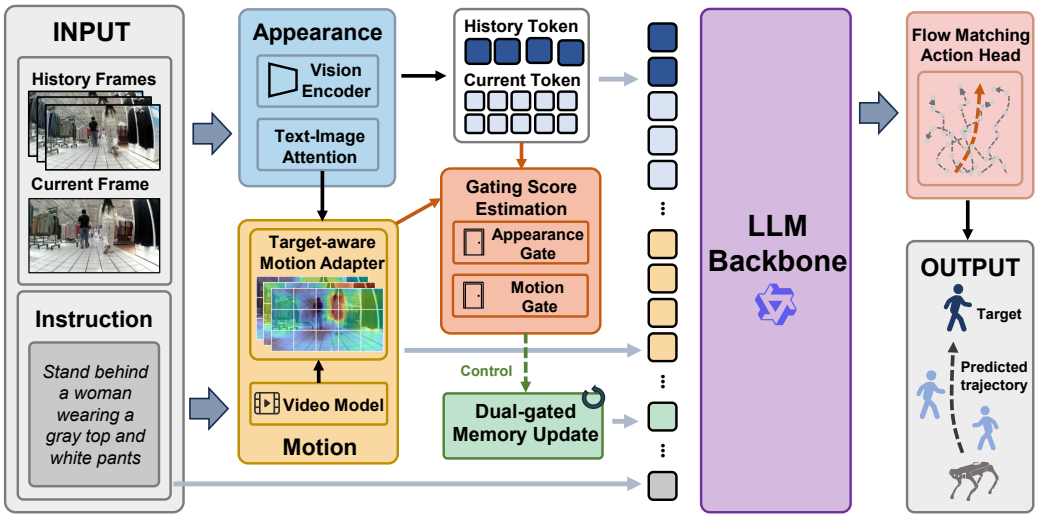


Figure 1: Overview of Track-MM. Given an observation sequence and a language instruction, the proposed framework jointly models appearance and motion representations to predict continuous trajectories for embodied visual tracking.

these issues with a lightweight Target-Aware Motion Adapter that converts frozen V-JEPA tokens into target-conditioned motion cues for memory updates.

3 Method

3.1 Task Formulation

We formulate embodied visual tracking as language-conditioned waypoint generation. Given an instruction I describing the target, the agent receives the current observation o_t at each timestep t and has access to the history $O_{t-1} = \{o_1, o_2, \ldots, o_{t-1}\}$, where each $o_t \in \mathbb{R}^{H\times W\times 3}$ is an egocentric RGB image. The goal is to predict waypoints $W_t = \{w_1, \ldots, w_8\}$, where $w_i = (x_i, y_i, z_i) \in \mathbb{R}^3$, as a near-horizon plan for stable target following. A tracking episode is considered successful if the agent avoids collisions, stops within a distance of 1–3 meters from the target, and maintains a frontal view facing the target.

3.2 Architecture Overview

Figure 1 summarizes Track-MM. Appearance tokens and target-conditioned video tokens update separate memories through branch-specific gates. A compact policy backbone processes the updated memories, current tokens, and instruction, and a flow-matching head predicts waypoints.

3.3 Appearance Feature Encoding

Given an observation o_t, DINOv3 (Siméoni et al., 2025) and SigLIP (Tschannen et al., 2025) extract appearance features, which are concatenated and projected into shared visual tokens $F_t \in \mathbb{R}^{N_t\times C}$. Following the observation encoding setting of TrackVLA (Wang et al., 2025b), we maintain a 31-frame history, compressing each historical frame into four coarse tokens to form $F_{\text{history}} \in \mathbb{R}^{124\times C}$, while the current frame uses denser tokens $F_{\text{current}} \in \mathbb{R}^{64\times C}$. A lightweight text–image attention head then conditions these visual tokens on the language instruction I and outputs F_{ti} for memory updating and target-aware motion adaptation.

3.4 TARGET-AWARE LATENT MOTION PRIOR

The pretrained video encoder provides latent tokens that capture rich but generic scene dynamics. To make them target-aware, TMA injects instruction-conditioned target information from the text–image branch into the latent video representation. Given the projected video tokens $F_{\mathrm{vm}}^{p} \in \mathbb{R}^{M\times C}$ and aligned text–image tokens $F_{\mathrm{ti}}^{p} \in \mathbb{R}^{M\times C}$, TMA first refines the target-importance map $a_0 \in [0,1]^M$ from the text–image attention response into $a = \sigma\left(\mathrm{logit}(a_0) + \mathcal{P}_{\mathrm{imp}}(F_{\mathrm{ti}}^{p})\right)$, where $\sigma(\cdot)$ denotes the sigmoid function and $\mathcal{P}_{\mathrm{imp}}(\cdot)$ is a lightweight projection network. The text–image tokens are then softly enhanced as $\tilde{F}_{\mathrm{ti}} = F_{\mathrm{ti}}^{p} \odot (1 + \gamma_{\mathrm{imp}} a)$, where a is broadcast along the channel dimension and γ_{imp} is a fixed scaling hyperparameter set to 0.5.

After target enhancement, TMA performs bidirectional target–video adaptation. Since F_{vm}^{p} and $\tilde{F}_{\mathrm{ti}}$ are aligned to the same token layout, both adaptation paths produce features with size $M \times C$. The first path uses video tokens as queries to retrieve target-conditioned information from the enhanced text–image tokens:

$$F_{\mathrm{v2t}} = \mathrm{Attn}\left(F_{\mathrm{vm}}^{p}, \tilde{F}_{\mathrm{ti}}, \tilde{F}_{\mathrm{ti}}\right) \in \mathbb{R}^{M\times C}. \tag{1}$$

The second path uses the enhanced text–image tokens as queries to retrieve motion information from the latent video tokens:

$$F_{\mathrm{t2v}} = \mathrm{Attn}\left(\tilde{F}_{\mathrm{ti}}, F_{\mathrm{vm}}^{p}, F_{\mathrm{vm}}^{p}\right) \in \mathbb{R}^{M\times C}. \tag{2}$$

The two adapted features are then fused as $F_{\mathrm{fuse}} = \omega \odot F_{\mathrm{v2t}} + (1-\omega) \odot F_{\mathrm{t2v}}$, where the token-wise weight ω is predicted by a lightweight projection network from F_{v2t}, F_{t2v}, and F_{vm}^{p}. The fused representation, together with the original video tokens, is projected into a correction term and added back with a residual weight $\eta = 0.1$, followed by layer normalization:

$$F_{\mathrm{mot}} = \mathrm{LN}\left(F_{\mathrm{fuse}} + \eta\, \mathcal{P}_{\mathrm{out}}\left([F_{\mathrm{vm}}^{p}; F_{\mathrm{fuse}}]\right)\right) \in \mathbb{R}^{M\times C}. \tag{3}$$

To encourage F_{mot} to preserve spatial target information, we attach a lightweight mask prediction head $\mathcal{P}_{\mathrm{mask}}$. The motion tokens are decoded into a low-resolution target mask:

$$\hat{S}_t = \sigma\left(\mathcal{P}_{\mathrm{mask}}\left(F_{\mathrm{mot}}\right)\right) \in [0,1]^{H_m\times W_m}. \tag{4}$$

Given the training mask $S_t \in \{0,1\}^{H\times W}$, we resize it to $S_t^m \in \{0,1\}^{H_m\times W_m}$ and optimize

$$\mathcal{L}_{\mathrm{mask}} = \mathrm{BCE}\left(\hat{S}_t, S_t^m\right) + \lambda_{\mathrm{dice}}\mathcal{L}_{\mathrm{Dice}}\left(\hat{S}_t, S_t^m\right). \tag{5}$$

3.5 DUAL-GATED MEMORY UPDATE

Appearance memory should be updated conservatively under occlusion or distractors to prevent identity drift, whereas motion memory must adapt to target dynamics while suppressing ego-motion and irrelevant motion. Track-MM therefore uses branch-specific gates to update each memory according to its own evidence.

3.5.1 GATING SCORE ESTIMATION

Let $b \in \{\mathrm{app}, \mathrm{mot}\}$ index the appearance and motion branches, whose current evidence is defined as $F_{\mathrm{app}}^{t} = F_{\mathrm{current}}$ and $F_{\mathrm{mot}}^{t} = F_{\mathrm{mot}}$, respectively. The update gate is estimated from three complementary factors: target observability, memory–evidence concentration, and response continuity.

Target observability. We estimate target observability from the mask predicted by TMA. Given $\hat{S}_t \in [0,1]^{H_m\times W_m}$, a lightweight visibility projection maps the predicted mask to a scalar observability score:

$$v^t = \mathcal{P}_{\mathrm{vis}}\left(\hat{S}_t\right), \qquad v^t \in [0,1], \tag{6}$$

where $\mathcal{P}_{\mathrm{vis}}(\cdot)$ is a shallow convolutional projection with pooling and a sigmoid activation, summarizing whether the target is spatially present and visible in the current observation.

Memory–evidence concentration. For each branch, the current evidence is compared with the previous memory through a branch-specific cross-attention interaction $A_b^t = \mathcal{C}_b(F_b^t, M_b^{t-1})$, where $\mathcal{C}_b(\cdot)$ denotes the branch-specific cross-attention. A branch-specific projection then produces the current response distribution $w_b^t = \mathrm{softmax}(\mathcal{P}_b(A_b^t))$, where $\mathcal{P}_b(\cdot)$ denotes the projection network for branch b. Since w_b^t is computed after current evidence interacts with previous memory, it measures how localized the current memory–evidence correspondence is. We compute its normalized concentration using entropy:

$$E_b^t = -\sum_{i=1}^{N_b} w_{b,i}^t \log\left(w_{b,i}^t + \epsilon\right), \tag{7}$$

$$s_b^t = 1 - \frac{E_b^t}{\log N_b}, \qquad s_b^t \in [0,1]. \tag{8}$$

Response continuity. To reduce abrupt writes caused by response jumps, we measure whether the response location changes smoothly over time. Let $\mathbf{c}_{b,i}$ denote the normalized spatial coordinate of the i-th token in branch b. For video tokens, tokens sharing the same spatial location use the corresponding spatial coordinate. The response centroid and its displacement are computed as

$$\mathbf{p}_b^t = \sum_{i=1}^{N_b} w_{b,i}^t \mathbf{c}_{b,i}, \qquad d_b^t = \left\|\mathbf{p}_b^t - \mathbf{p}_b^{t-1}\right\|_2. \tag{9}$$

We convert this displacement into a response-continuity score:

$$q_b^t = \exp\left(-\left(\frac{d_b^t}{d_{\max,b}}\right)^2\right), \qquad q_b^t \in [0,1], \tag{10}$$

where $d_{\max,b}$ is set as the 95th percentile of target-centroid displacement in the training set for branch b. This term suppresses sudden response jumps that are unlikely under normal tracking while allowing feasible target displacement.

The final branch-specific update coefficient is

$$g_b^t = v^t \cdot s_b^t \cdot q_b^t. \tag{11}$$

Thus, memory writing is encouraged only when the target is observable, the memory–evidence match is localized, and the response location remains temporally continuous.

3.5.2 Appearance–Motion Memory Update

The appearance and motion memories are updated using their respective branch-specific gates g_{app}^t and g_{mot}^t. For each branch, we first interpolate between the previous memory M_b^{t-1} and the current evidence:

$$\widetilde{F}_b^t = \left(1 - g_b^t\right) M_b^{t-1} + g_b^t F_b^t. \tag{12}$$

The gated features are subsequently processed by the convolution-based memory encoder $\mathcal{G}$ to produce the updated memory tokens M_{app}^t and M_{mot}^t. The two memory tokens are concatenated with the current multimodal tokens and provided to the policy backbone for waypoint prediction.

3.6 Action Generation

We use Qwen3–0.6B (Bai et al., 2025) to encode appearance, motion, memory, and instruction features into a global action embedding:

$$z = \mathrm{LLM}(F_{\text{history}}, F_{\text{current}}, F_{\text{mot}}, M_{\text{app}}, M_{\text{mot}}, F_{\text{text}}). \tag{13}$$

A conditional flow matching head (Tong et al., 2023) generates waypoint sequences conditioned on z. With the interpolation $\mathbf{W}_\tau = (1-\tau)\epsilon + \tau\mathbf{W}$, where $\epsilon \sim \mathcal{N}(0, I)$ and $\tau \sim \mathcal{U}(0,1)$, the head predicts the target velocity $\mathbf{u}_\tau = \mathbf{W} - \epsilon$ via

$$\mathcal{L}_{\text{fm}} = \mathbb{E}_{\tau,\epsilon}\left[\left\|\mathcal{F}_{\text{fm}}(\mathbf{W}_\tau, \tau, z) - \mathbf{u}_\tau\right\|_2^2\right]. \tag{14}$$

The overall objective is $\mathcal{L} = \mathcal{L}_{\text{fm}} + \lambda_{\text{mask}}\mathcal{L}_{\text{mask}}$.

Table 1: Quantitative comparison on EVT-Bench.

Method	Size	STT			DT			AT		
		SR ↑	TR ↑	CR ↓	SR ↑	TR ↑	CR ↓	SR ↑	TR ↑	CR ↓
IBVS† (Gupta et al., 2016)	–	42.9	56.2	3.75	10.6	28.4	6.14	15.2	39.5	4.90
PoliFormer (Zeng et al., 2024)	–	4.67	15.5	40.1	2.62	13.2	44.5	3.04	15.4	41.5
EVT (Zhong et al., 2024)	–	24.4	39.1	42.5	3.23	11.2	47.9	17.4	21.1	45.6
EVT‡ (Zhong et al., 2024)	–	32.5	49.9	40.5	15.7	35.7	53.3	18.3	21.0	44.9
Uni-NaVid (Zhang et al., 2024)	7B	25.7	39.5	41.9	11.3	27.4	43.5	8.26	28.6	43.7
TrackVLA (Wang et al., 2025b)	7B	85.1	78.6	1.65	57.6	63.2	5.80	50.2	63.7	17.1
NavFoM (Zhang et al., 2025b)	7B	85.0	80.5	—	61.4	68.2	—	—	—	—
TrackVLA++ (Liu et al., 2025)	7B	86.0	81.0	2.10	66.5	68.8	4.71	51.2	63.4	15.9
VLingNav (Wang et al., 2026)	7B	88.4	81.2	2.07	67.6	73.5	5.51	—	—	—
ReferTrack (Ye et al., 2026)	4B	**89.4**	**92.5**	1.6	**73.3**	**81.8**	7.6	**74.1**	**85.7**	7.7
Track-MM (Ours)	1.2B	89.3	86.7	**1.19**	68.9	69.2	**2.22**	66.8	71.4	**2.64**

4 EXPERIMENTS

4.1 DATA PREPARATION

We construct the training data based on the TrackVLA dataset (Wang et al., 2025b). Trajectories are generated in the Habitat simulator across diverse indoor environments, including HM3D (Ramakrishnan et al., 2021) and MP3D (Chang et al., 2017). To obtain the target masks used for supervising TMA, we employ the Habitat-based target-mask sensor provided by OmTrackVLA (Lee et al., 2025). For trajectory generation, the embodied agent follows an oracle policy (Anderson et al., 2018) derived from global path planning. Safety and smoothing constraints are applied during execution to avoid abrupt motion and collisions. After filtering unstable or invalid trajectories, the training set contains 12358 sequences.

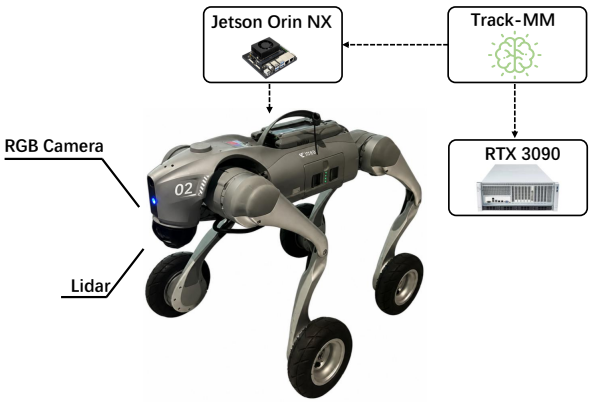


Figure 2: Real-world platform.

4.2 EXPERIMENT SETUPS

4.2.1 BENCHMARK AND METRICS

We evaluate our method on EVT-Bench, which contains three task settings with increasing difficulty. **Single-Target Tracking (STT)** evaluates following a clearly specified target without significant distractions. **Distracted Tracking (DT)** introduces multiple people, requiring the agent to track the correct target among distractors. **Ambiguity Tracking (AT)** further includes individuals with highly similar appearances and tests identity consistency. We adopt three metrics: **success rate**

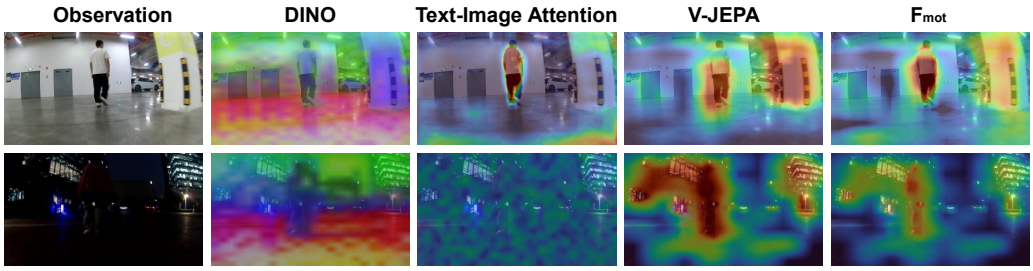


Figure 3: Qualitative motivation for separately modeling appearance and motion cues and for the proposed target-aware motion adapter.

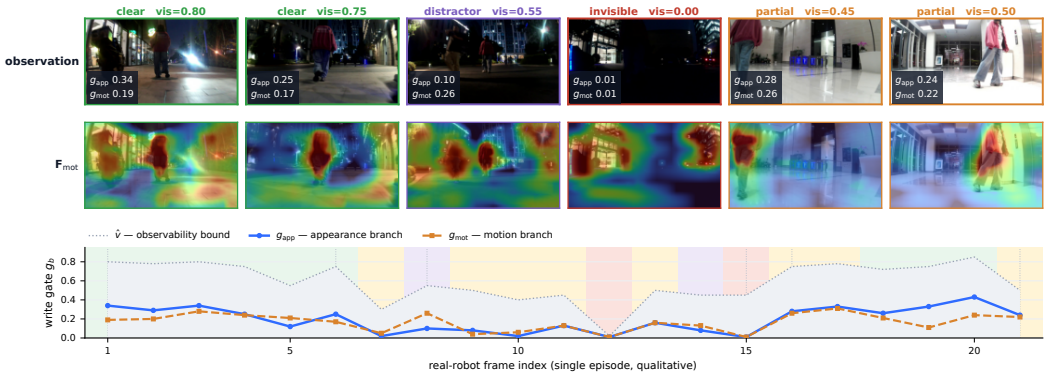


Figure 4: Gate behavior during a real-robot episode. The top row shows representative observations with visibility annotations, the middle row visualizes F_{mot}. The bottom row shows the gates under different visibility conditions.

(SR), **tracking rate (TR)**, and **collision rate (CR)**. SR measures successful stopping within 1–3 meters while facing the target, TR measures the proportion of time steps in which the target remains within the field of view and 1–3 meters, and CR measures navigation collisions.

4.2.2 Implementation Details

Track-MM is implemented based on OmTrackVLA and trained on an NVIDIA RTX 5000 Ada GPU for 10 epochs. Input frames are resized to 384×384, and trainable modules are optimized with AdamW using a learning rate of 2×10^{-4} and batch size 16. We use 21.6M DINOv3-ViT-S/16 pretrained on LVD-1689M and 0.2B Siglip-base-patch16-512 as vision encoders, initialize the text–image cross-attention head from Grounding DINO Tiny, instantiate the latent video prior with V-JEPA 2.1 (Mur-Labadia et al., 2026), and use Qwen3-0.6B as the policy backbone. These pretrained encoders and the LLM backbone are frozen. We set the loss weight $\lambda_{\text{mask}} = 0.3$, $\lambda_{\text{dice}} = 0.5$.

4.2.3 Real-world Deployment

Our real-world platform is a Unitree Go2-W quadruped robot equipped with an NVIDIA Jetson Orin NX 8GB edge computer and a wide-angle RGB camera with a 120° field of view, as shown in Figure 2. Track-MM is deployed through ROS2 and supports fully onboard inference as well as edge-cloud collaborative inference, where compute-intensive inference is offloaded to an NVIDIA RTX 3090 workstation while the robot handles sensing, communication, and low-level control.

4.2.4 Result Comparison

Table 1 compares our method with representative approaches on EVT-Bench across three task settings. The symbol † indicates methods that rely on explicit detection models, while ‡ denotes ap-

proaches that adopt SoM (Yang et al., 2023) combined with GPT-4o (Hurst et al., 2024) as the backbone. Bold and underline indicate the best and second-best results, respectively. Track-MM attains the second-best SR and TR on STT with only 1.2B parameters, within 0.1% of the best SR at roughly one-third the model size. On DT and AT, Track-MM achieves SR within 4.4% and 7.3% of the best results, respectively, while all methods exceeding its SR rely on 4B–7B backbones. Overall, Track-MM achieves a well-balanced performance across the three settings, with the best collision rate across all tasks.

Table 2: Zero-shot real-world comparison between OmTrackVLA and Track-MM.

Scenario	**Trials**	**OmTrackVLA**		**Track-MM**	
		Succ.	**SR**	**Succ.**	**SR**
Shopping mall	12	8	66.7	11	91.7
Park	8	6	75.0	7	87.5
Office building	10	8	80.0	8	80.0
Outdoor night	12	7	58.3	11	91.7
Overall	42	29	69.0	37	88.1

4.3 Feature Analysis

Figure 3 compares PCA visualizations of DINO appearance features with normalized response heatmaps. In the first example, appearance attention identifies the target, while raw V-JEPA features also respond to irrelevant scene variations. Under low illumination, appearance responses weaken, whereas temporal features retain target information. TMA produces more compact, target-aligned responses in both cases. These complementary reliability patterns motivate separate appearance and motion memories.

Figure 4 visualizes representative observations, motion features, and gate trajectories from a real-robot episode, where 21 observations are sampled at 20-frame intervals. The appearance gate g_{app} correlates strongly with the annotated target visibility ($r = 0.690$, $p < 0.001$). Its mean decreases from 0.290 under clear visibility to 0.158 under partial occlusion and 0.010 when the target is invisible. The two gates diverge most clearly in the presence of distractors: g_{app} decreases to 0.090, whereas g_{mot} remains at 0.195, close to its clear-view mean. This behavior suppresses unreliable appearance updates while retaining motion cues for temporal continuity, which cannot be represented by a single shared gate.

Table 3: Component ablations on the DT and AT settings of EVT-Bench.

Variant	**DT**			**AT**		
	SR ↑	**TR** ↑	**CR** ↓	**SR** ↑	**TR** ↑	**CR** ↓
w/o TMA	56.4	59.7	7.78	62.3	68.1	3.74
w/o TMA & V-JEPA	51.0	58.2	7.91	58.0	58.9	8.28
w/o Mask Prediction	60.9	63.3	4.07	63.3	62.6	3.36
w/o Memory	61.8	59.3	5.59	50.7	53.0	5.85
Unified Memory + Single Gate	63.1	65.3	3.24	58.3	60.4	4.32
Unified Memory + Dual Gate	65.6	67.9	2.99	60.1	65.0	3.06
Dual Memory w/o Gate	61.7	65.6	4.03	57.0	61.7	5.46
Dual Memory + Single Gate	66.4	67.0	2.85	63.5	67.6	2.83
Ours	**68.9**	**69.2**	**2.22**	**66.8**	**71.4**	**2.64**

4.4 Real-World Experiment

The simulation-trained policy is deployed directly on the platform introduced in Section 4.2.3, without real-world fine-tuning or online adaptation. With batch size 1 and FP16 precision, it runs fully

onboard at 6 FPS with an average end-to-end latency of 161 ms, and an optional edge-cloud mode using an RTX 3090 achieves 19 FPS and 55 ms latency including communication overhead.

We compare Track-MM with OmTrackVLA under the same protocol in four scenarios covering abrupt motion, temporary target disappearance, distractors, crowds, and low-light conditions. As shown in Table 2, Track-MM succeeds in 37 of 42 trials (88.1%), compared with 29 of 42 trials (69.0%) for OmTrackVLA. The largest absolute SR gains occur in the outdoor-night and shopping-mall scenarios, reaching 33.4% and 25.0% , respectively. These results indicate promising zero-shot sim-to-real transfer across diverse environments.

Table 4: Ablation of motion representations and temporal window length on EVT-Bench STT. K is the number of observations processed by the motion branch.

Motion Representation	**SR** ↑	**TR** ↑	**CR** ↓
Optical Flow	66.7	70.5	7.81
VideoMAE2	86.1	85.9	4.06
InternVideo-Next	79.3	79.7	3.53
V-JEPA 2	85.5	82.4	1.62
V-JEPA 2.1, $K = 4$	**89.3**	86.7	**1.19**
V-JEPA 2.1, $K = 1$	77.2	80.0	2.29
V-JEPA 2.1, $K = 8$	84.4	81.6	1.83
V-JEPA 2.1, $K = 16$	87.0	**88.9**	1.36
V-JEPA 2.1, $K = 32$	82.2	83.0	2.11

4.5 Ablation Study

Table 3 reports component ablations on DT and AT. Removing the entire motion branch (w/o TMA & V-JEPA) causes substantial degradation, showing the importance of motion cues for robust tracking. Retaining V-JEPA but removing TMA also degrades performance, indicating that generic video features require target-aware adaptation. Removing mask prediction, including its auxiliary supervision, leads to a consistent drop, suggesting that target-mask prediction provides useful observability and localization cues. The lower part of Table 3 examines the memory and gating designs. Removing memory causes a particularly large degradation on AT, highlighting the importance of historical information under appearance ambiguity. Unified Memory stores appearance and motion features in one state, using either a single shared gate or two branch-specific gates. Dual Memory maintains separate states, but removing gating or sharing one gate still underperforms Track-MM. These results show that separate appearance and motion memories and different update gates are both necessary, especially under ambiguity and distractor interference.

We compare optical flow (Farnebäck, 2003) with pretrained video representations, including VideoMAE2 (Wang et al., 2023), InternVideo-Next (Wang et al., 2025a), V-JEPA 2 (Assran et al., 2025), and V-JEPA 2.1 (Mur-Labadia et al., 2026). As shown in Table 4, video representations generally outperform optical flow, indicating the limitations of low-level motion cues. V-JEPA 2.1 performs best overall with $K = 4$, whereas the degradation at $K = 1$ confirms the importance of temporal context. Longer windows yield no consistent gains, and $K = 32$ further degrades performance, likely due to stale or irrelevant observations. We therefore adopt V-JEPA 2.1 with $K = 4$ by default.

5 Conclusion

We presented Track-MM, a lightweight dual-memory framework for embodied visual tracking. By combining a target-aware latent motion prior with updated appearance and motion memories separately, Track-MM preserves target identity while maintaining short-term temporal continuity under partial observability. Experiments on EVT-Bench show that Track-MM achieves strong performance across standard tracking metrics, and real-world deployment on Jetson Orin NX demonstrates its practical efficiency and zero-shot transfer potential.

AI USE STATEMENT

In this work, we used GPT-6 Astra, a generative AI tool, to provide feedback on the research methodology of the proposed method, assist with manuscript drafting, and review the code. We have not used generative AI tools for any other tasks with required disclosure. All AI-assisted feedback and text were manually reviewed and revised by the authors before adoption. All experimental results reported in this paper were obtained from actual experiments conducted by the authors, and none of the numbers, tables, or figures were generated by AI. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

The real-world experiments in this work involve a robot tracking human targets in public and semi-public environments. All tracked individuals were informed and provided consent prior to the experiments. The recorded egocentric observations are used solely for this research and will not be released in any form that identifies individuals.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

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 conference on computer vision and pattern recognition*, pp. 3674–3683, 2018.

Mahmoud Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.

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*, 2025.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0:: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

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*, pp. 667–676. IEEE, 2017.

Jinyu Chen, Chen Gao, Erli Meng, Qiong Zhang, and Si Liu. Reinforced structured state-evolution for vision-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15450–15459, 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. *arXiv preprint arXiv:2412.04453*, 2024.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Gunnar Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Scandinavian conference on Image analysis*, pp. 363–370. Springer, 2003.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Mohit Gupta, Laxmidhar Behera, Venkatesh K Subramanian, and Mo M Jamshidi. A novel approach to robotic heading control using visual servoing. *IEEE Systems Journal*, 12(3):2876–2884, 2016.

Honglin He, Yukai Ma, Wayne Wu, and Bolei Zhou. From seeing to experiencing: Scaling navigation foundation models with reinforcement learning. *arXiv preprint arXiv:2507.22028*, 2025.

Ting Huang, Dongjian Li, Rui Yang, Zeyu Zhang, Zida Yang, and Hao Tang. Mobilevla-r1: Reinforcing vision-language-action for mobile robots. *arXiv preprint arXiv:2511.17889*, 2025.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Kyusong Lee, Heting Ying, and Tiancheng Zhao. OmTrackVLA: Open-source visual language action model for visual navigation and following. `https://github.com/om-ai-lab/OmTrackVLA`, 2025. GitHub repository.

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.

Jinlai Liu, Jian Han, Bin Yan, Hui Wu, Fengda Zhu, Xing Wang, Yi Jiang, Bingyue Peng, and Zehuan Yuan. Infinitystar: Unified spacetime autoregressive modeling for visual generation. *Advances in Neural Information Processing Systems*, 38:170054–170072, 2026.

Lorenzo Mur-Labadia, Matthew Muckley, Amir Bar, Mahmoud Assran, Koustuv Sinha, Michael Rabbat, Yann LeCun, Nicolas Ballas, and Adrien Bardes. V-jepa 2.1: Unlocking dense features in video self-supervised learning. *arXiv preprint arXiv:2603.14482*, 2026.

Santhosh K Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alex Clegg, John Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, et al. Habitat-matterport 3d dataset (hm3d): 1000 large-scale 3d environments for embodied ai. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021.

Oriane Siméoni, Huy V Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, et al. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025.

Alexander Tong, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Kilian Fatras, Guy Wolf, and Yoshua Bengio. Conditional flow matching: Simulation-free dynamic optimal transport. *arXiv preprint arXiv:2302.00482*, 2(3), 2023.

Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *Advances in neural information processing systems*, 35:10078–10093, 2022.

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.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Chenting Wang, Yuhan Zhu, Yicheng Xu, Jiange Yang, Lang Lin, Ziang Yan, Yali Wang, Yi Wang, and Limin Wang. Internvideo-next: Towards general video foundation models without video-text supervision. *arXiv preprint arXiv:2512.01342*, 2025a.

Jiawei Wang, Teng Wang, Wenzhe Cai, Lele Xu, and Changyin Sun. Boosting efficient reinforcement learning for vision-and-language navigation with open-sourced llm. *IEEE Robotics and Automation Letters*, 10(1):612–619, 2024a.

Limin Wang, Bingkun Huang, Zhiyu Zhao, Zhan Tong, Yinan He, Yi Wang, Yali Wang, and Yu Qiao. Videomae v2: Scaling video masked autoencoders with dual masking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 14549–14560, 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*, 2024b.

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.

Shaoan Wang, Yuanfei Luo, Xingyu Chen, Aocheng Luo, Dongyue Li, Chang Liu, Sheng Chen, Yangang Zhang, and Junzhi Yu. Vlingnav: Embodied navigation with adaptive reasoning and visual-assisted linguistic memory. *arXiv preprint arXiv:2601.08665*, 2026.

Kui Wu, Shuhang Xu, Hao Chen, Churan Wang, Zhoujun Li, Yizhou Wang, and Fangwei Zhong. Vlm can be a good assistant: Enhancing embodied visual tracking with self-improving vision-language models. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 13154–13161. IEEE, 2025.

Jianwei Yang, Hao Zhang, Feng Li, Xueyan Zou, Chunyuan Li, and Jianfeng Gao. Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v. In *arXiv preprint arXiv:2310.11441*, 2023.

Hanjing Ye, Tianle Zeng, Jiazhao Zhang, Shaoan Wang, Zibo Zhang, Weisi Situ, Yuchen Zhou, Yonggen Ling, and Hong Zhang. Refertrack: Referring then tracking for embodied visual tracking. *arXiv preprint arXiv:2607.20061*, 2026.

Zhuoyuan Yu, Yuxing Long, Zihan Yang, Chengyan Zeng, Hongwei Fan, Jiyao Zhang, and Hao Dong. Correctnav: Self-correction flywheel empowers vision-language-action navigation model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 18737–18745, 2026.

Tianyuan Yuan, Zibin Dong, Yicheng Liu, and Hang Zhao. Fast-wam: Do world action models need test-time future imagination? *arXiv preprint arXiv:2603.16666*, 2026.

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

Boqiang Zhang, Kehan Li, Zesen Cheng, Zhiqiang Hu, Yuqian Yuan, Guanzheng Chen, Sicong Leng, Yuming Jiang, Hang Zhang, Xin Li, et al. Videollama 3: Frontier multimodal foundation models for image and video understanding. *arXiv preprint arXiv:2501.13106*, 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. *arXiv preprint arXiv:2412.06224*, 2024.

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*, 2025b.

Baining Zhao, Jiacheng Xu, Weicheng Feng, Xin Zhang, Zhaolu Wang, Haoyang Wang, Shilong Ji, Ziyou Wang, Jianjie Fang, Zhiheng Zheng, et al. Worldvln: Autoregressive world action model for aerial vision-language navigation. *arXiv preprint arXiv:2605.15964*, 2026.

Fangwei Zhong, Peng Sun, Wenhan Luo, Tingyun Yan, and Yizhou Wang. Ad-vat: An asymmetric dueling mechanism for learning visual active tracking. In *International Conference on Learning Representations*, 2019.

Fangwei Zhong, Xiao Bi, Yudi Zhang, Wei Zhang, and Yizhou Wang. Rspt: reconstruct surroundings and predict trajectory for generalizable active object tracking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 3705–3714, 2023.

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.

A Appendix

A.1 Dataset

We use the official TrackVLA dataset collected in the Habitat environments, covering three task settings: Single-Target Tracking (STT), Distracted Tracking (DT), and Ambiguity Tracking (AT). To ensure trajectory quality, we exclude trajectories of collisions and lost targets, as well as with a tracking ratio (TR) below 0.5. After filtering, the resulting training set contains 12,358 trajectories, comprising 4,149 STT trajectories, 4,206 DT trajectories, and 4,003 AT trajectories. All trainable components are optimized on this filtered training set, while hyperparameters are selected using the validation split.

A.2 Additional Qualitative Results

We provide additional qualitative results in simulated and real-world environments, together with intermediate feature visualizations that illustrate the effect of the Target-Aware Motion Adapter (TMA).

Tracking results in simulation and the real world. Figure 5 presents representative tracking sequences under diverse target appearances and environmental conditions. The examples cover different clothing styles, illumination changes, background interference, visually similar distractors, and partial occlusion. Track-MM consistently follows the instruction-specified target across these conditions, illustrating its robustness to both appearance variation and environmental disturbances. In particular, the leftmost example in the bottom row contains severe foreground interference and large-area target occlusion. Track-MM nevertheless maintains the target identity and continues tracking after the target becomes visible again.

Effect of the Target-Aware Motion Adapter. Figure 6 compares the generic V-JEPA 2.1 representation with the target-aware motion representation produced by TMA. Although the pretrained video encoder captures rich scene dynamics, its responses may also cover background regions and task-irrelevant objects. By incorporating instruction-conditioned text–image evidence, TMA produces a more target-focused representation in F_{mot}, with stronger responses around the person specified by the language instruction. The correspondence between F_{mot} and the text–image attention response is consistently observed across targets with different appearances. These visualizations qualitatively support the effectiveness of TMA in converting generic video features into instruction-conditioned motion representations.



Figure 5: Additional tracking results in simulated and real-world environments.

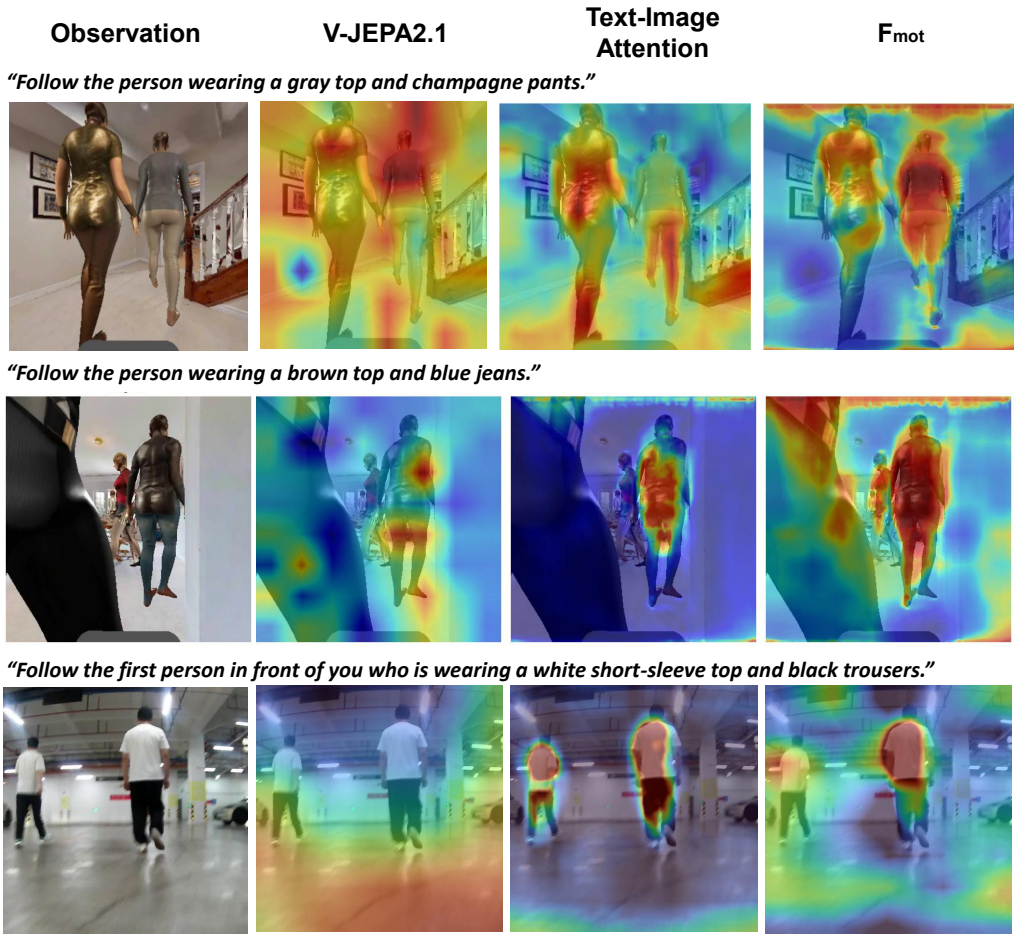


Figure 6: Qualitative visualization of the Target-Aware Motion Adapter.

Additional real-world results. Figure 7 shows representative real-world tracking sequences under three challenging conditions. In the dynamic occlusion example, a moving obstacle temporarily blocks the target, but Track-MM maintains the target identity and continues tracking after the target becomes visible again. Under an abrupt change in target motion, the policy adapts its tracking behavior as the person begins to turn. In the target-recovery sequence, the person temporarily leaves

the camera view and is successfully reacquired after re-entering the scene. Together, these examples illustrate the practical behavior of the appearance and motion memories under short-term observation interruptions and motion changes.

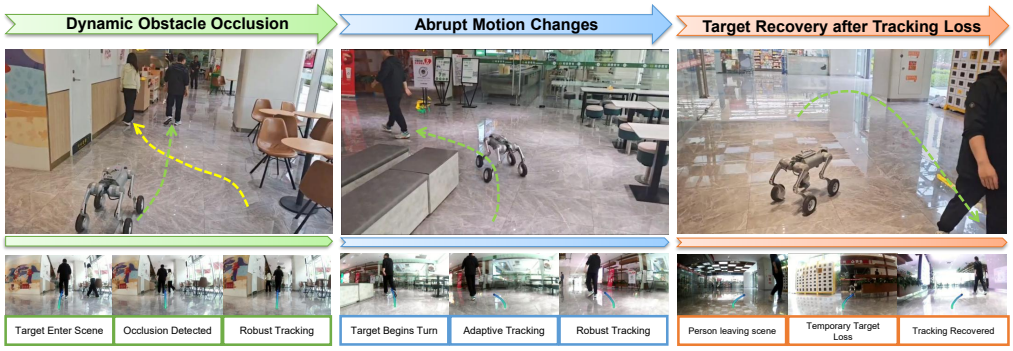


Figure 7: Representative real-world tracking sequences under dynamic obstacle occlusion, abrupt target-motion changes, and temporary target loss.

A.3 Architecture Details

A.3.1 Input Streams

At timestep t, the policy receives the current RGB observation o_t, the language instruction I, and a finite observation history. The appearance branch uses the current frame and compressed history tokens. The motion branch uses a temporal window of $K = 4$ observations, specifically $\{o_{t-3}, o_{t-2}, o_{t-1}, o_t\}$ (i.e., the current frame plus the three most recent past frames). The mask predicted from F_{mot} is used only for the current timestep since o_t is included in the motion window.

A.3.2 Representation Shapes

Table 5 summarizes the shapes of the intermediate token representations, memory states, and predicted target mask used in Track-MM.

Symbol	Shape	Definition
F_{history}	124×1024	31 historical frames compressed to 4 tokens per frame.
F_{current}	64×1024	Current-frame appearance tokens.
F_{vm}	512×1024	V-JEPA 2.1 video tokens.
F^p_{vm}	64×1024	Projected V-JEPA 2.1 video tokens.
F^p_{ti}	64×1024	Projected instruction-conditioned text–image tokens.
F_{mot}	64×1024	Target-aware motion tokens after TMA.
M^t_{app}	32×1024	Appearance memory at timestep t.
M^t_{mot}	32×1024	Motion memory at timestep t.
$\hat{S}_t$	96×96	Target mask.

Table 5: Shapes of the intermediate representations, memory states, and predicted target mask in Track-MM.

A.3.3 Trainable and Frozen Modules

As summarized in Table 6, Track-MM retains the pretrained visual, video, language, and text–image attention modules as frozen backbones. Only the lightweight TMA, dual-gated memory update module, and conditional flow-matching action head are optimized during training, amounting to approximately 63M trainable parameters out of roughly 1.2B total parameters. This parameter-

efficient design substantially reduces the training cost while preserving the general visual, temporal, and language representations learned by the pretrained models.

Module	Parameters	Status
DINOv3-ViT-S/16	21.6M	Frozen
SigLIP-base-patch16-512	200M	Frozen
V-JEPA 2.1 (ViT-L/16)	300M	Frozen
Qwen3-0.6B	600M	Frozen
Text–image attention from Grounding DINO Tiny	18M	Frozen
Target-Aware Motion Adapter and mask head	$\sim$20M	Trainable
Dual-gated memory update module	$\sim$37M	Trainable
Conditional flow-matching action head	$\sim$6M	Trainable
Total	$\sim$1.2B	$\sim$63M trainable

Table 6: Approximate parameter counts and training status of the principal modules.

A.3.4 Target-Aware Motion Adapter

The initial target-importance map a_0 is derived from the spatial attention response of the first cross-attention layer in the frozen text–image attention module. Specifically, the attention response associated with the target instruction is normalized across the M image-token positions using a softmax operation, yielding $a_0 \in [0,1]^M$. Before applying $\mathrm{logit}(\cdot)$, we clamp a_0 to $[\delta, 1-\delta]$, where $\delta = 10^{-6}$, to ensure numerical stability.

Spatiotemporal token alignment in TMA. The video encoder initially produces a spatiotemporal representation $F_{\mathrm{vm}} \in \mathbb{R}^{512\times1024}$. Before bidirectional cross-attention, $\mathcal{P}_{vm}$ performs token compression and channel projection, mapping the spatiotemporal video representation to $F^p_{\mathrm{vm}} \in \mathbb{R}^{64\times1024}$. The text–image representation is aligned to the same token and channel dimensions, yielding $F^p_{\mathrm{ti}} \in \mathbb{R}^{64\times1024}$. Consequently, the bidirectional cross-attention in Eqs. 1–2 operates directly between the aligned video and text–image representations and produces $F_{\mathrm{v2t}}, F_{\mathrm{t2v}} \in \mathbb{R}^{64\times1024}$. The resulting target-aware motion representation retains this aligned layout, with $F_{\mathrm{mot}} \in \mathbb{R}^{64\times1024}$.

Fusion of the bidirectional features. Because the two cross-attention outputs share the same token layout and channel dimension, they can be fused directly without additional spatial interpolation, giving $F_{\mathrm{fuse}} \in \mathbb{R}^{64\times1024}$ as defined in Section 3.4. The token-wise fusion weight is computed as

$$\omega = \sigma\left(\mathcal{P}_\omega\left([F_{\mathrm{v2t}}; F_{\mathrm{t2v}}; F^p_{\mathrm{vm}}]\right)\right), \tag{15}$$

where $[\cdot;\cdot;\cdot]$ denotes concatenation along the channel dimension. The concatenated representation therefore has shape 64×3072. Here, $\mathcal{P}_\omega$ is a two-layer MLP with a hidden dimension of 512 and a ReLU activation. Its output is passed through a sigmoid function to obtain $\omega \in [0,1]^{64\times1}$, which is broadcast along the channel dimension during the element-wise weighted fusion.

A.3.5 Dual-Gated Memory Update

Branch-specific memory gating. The appearance and motion memories are controlled by separate update gates that combine a shared observability prior v^t with branch-specific reliability estimates (Eq. 11). The shared term suppresses unreliable updates when the target is poorly observed, while the concentration score s^t_b and response-continuity score q^t_b are computed independently for each branch. Consequently, the two memories can follow different update dynamics according to the reliability of their respective appearance and motion evidence.

Design rationale for shared observability. We adopt a shared v^t to couple both branches to mask prediction confidence, preventing unreliable updates when the target is poorly localized. An alternative design could use branch-independent visibility factors (e.g., allowing the motion branch to update during visual occlusions if optical flow remains confident). However, we observed that a shared v^t provides more robust behavior in practice by preventing the motion branch from "hallucinating" motion when the target location is uncertain. When $v^t \to 0$ (target occluded or tracking

Gate configuration	DT SR (%)	AT SR (%)
$v \cdot s \cdot q$ (Ours)	**68.9**	**66.8**
$0.8\, s \cdot q$ (fixed $v = 0.8$)	65.2	62.4
$v \cdot q$ (fixed $s = 1$)	66.7	64.1
$v \cdot s$ (fixed $q = 1$)	67.3	65.0
$g = 0.2$ (constant)	64.8	61.7

Table 7: Component-level ablation of the three-factor memory gate. The full gate combines observability (v), response concentration (s), and temporal continuity (q).

lost), both gates freeze ($g^t_{\text{app}}, g^t_{\text{mot}} \to 0$), and the system coasts on the last known memory state. This design prioritizes **reliability over responsiveness** during occlusions.

First-frame update. To avoid the bootstrap problem where zero-initialized memories produce uniform attention responses (leading to maximal entropy and $s_b = 0$), we apply a warm-start strategy for the first timestep: the concentration gate factor s^1_b is clamped to a minimum value of 0.5 for the initial update, ensuring that the first observation can write to memory. After $t = 1$, the standard entropy-based s^t_b computation applies without modification.

Implementation details of memory updating and gating. Equation 12 of the main paper omits token alignment for notational simplicity. In the implementation, the previous memory state $M^{t-1}_b \in \mathbb{R}^{32\times1024}$ is first projected into the 64-token evidence space:

$$\overline{M}^{t-1}_b = \mathcal{P}^{\text{mem}}_b\left(M^{t-1}_b\right). \tag{16}$$

Here, the branch-specific projection $\mathcal{P}^{\text{mem}}_b$ maps the previous memory from $\mathbb{R}^{32\times1024}$ to $\mathbb{R}^{64\times1024}$.

The aligned memory is then fused with the current evidence $F^t_b \in \mathbb{R}^{64\times1024}$ using the branch-specific gate:

$$\widetilde{F}^t_b = \left(1 - g^t_b\right)\overline{M}^{t-1}_b + g^t_b F^t_b. \tag{17}$$

The fused representation is subsequently encoded into the updated memory state:

$$M^t_b = \mathcal{G}\left(\widetilde{F}^t_b\right). \tag{18}$$

The shared memory encoder $\mathcal{G}$ maps the fused 64-token representation back to $M^t_b \in \mathbb{R}^{32\times1024}$ and is applied separately to the appearance and motion branches. Both $\mathcal{P}^{\text{mem}}_b$ and $\mathcal{G}$ are included in the trainable dual-gated memory update module reported in Table 6.

The response-concentration and continuity terms follow the definitions in the main paper. We use $\epsilon = 10^{-8}$ for entropy computation and set $d_{\max,b}$ to the 95th percentile of the branch-specific response-centroid displacements observed on the training set. The resulting thresholds typically range from 0.15 to 0.25 in normalized image coordinates. Memory states are initialized to zero at the beginning of each episode. At the first timestep, we set $q^1_b = 1$ and retain p^1_b as the response-centroid reference for the subsequent update.

A.4 Ablation Definitions

The three-factor gate design (Eq. 11: $g = v \cdot s \cdot q$) combines visibility, spatial concentration, and temporal continuity. Table 7 isolates each factor's contribution.

Table 8 summarizes the operational definitions of the ablation variants. We group them by the component being evaluated: the motion pathway, mask prediction, and memory architecture. Unless otherwise specified, all other model components and training settings remain unchanged.

A.5 Real-World Deployment Protocol

A.5.1 Platform and Runtime

The real-world platform is a Unitree Go2-W robot equipped with an NVIDIA Jetson Orin NX 8GB edge computer and a wide-angle RGB camera with a 120-degree field of view. The real-world plat-

Variant	Operational definition
Motion pathway	
w/o TMA	Removes the Target-Aware Motion Adapter and directly feeds F_{vm} into the subsequent motion-processing pathway.
w/o TMA & V-JEPA	Removes the entire motion branch by setting F_{mot} to zero. Observability estimation and memory updating rely only on appearance evidence.
Mask prediction	
w/o Mask Prediction	Removes the mask prediction head $\mathcal{P}_{\mathrm{mask}}$ and the mask loss $\mathcal{L}_{\mathrm{mask}}$. The observability score is instead estimated by $\mathcal{P}_{\mathrm{vis}}$ from pooled motion tokens.
Memory architecture	
w/o Memory	Removes the entire memory mechanism, including recurrent memory states, memory updating, and memory tokens provided to the language-model policy backbone.
Unified Memory + Single Gate	Concatenates the appearance and motion features into a unified representation, which is stored in a single memory and updated using one shared gate.
Unified Memory + Dual Gate	Maintains a single unified memory but computes separate update scores from the appearance and motion features before combining their gated evidence for memory updating.
Dual Memory w/o Gate	Maintains separate appearance and motion memories and retains temporal attention between the previous memory and current evidence, but performs the updates without gate-score control.
Dual Memory + Single Gate	Maintains separate appearance and motion memories but updates both using the same gate. The shared gate is computed from the concatenation of the appearance and motion features.
Ours	Maintains separate appearance and motion memories, each updated by its own branch-specific gate. The two gates share the observability prior v^t but use branch-specific concentration and temporal-continuity scores.

Table 8: Operational definitions of the motion-pathway, mask-prediction, and memory-architecture ablation variants.

form is deployed through ROS2. Unless otherwise stated, the policy is transferred from simulation without real-world fine-tuning or online adaptation.

Fully onboard inference uses batch size 1 and FP16 precision. The reported runtime is 6 FPS with 161 ms average end-to-end latency. The optional edge-cloud mode offloads model inference to an RTX 3090 workstation and reports 55 ms latency including communication overhead; the throughput (19 FPS) is measured via pipelined batching and differs from sequential latency.

A.5.2 Trial Protocol and Evaluation Rigor

The real-world evaluation follows a controlled comparison protocol designed to assess zero-shot sim-to-real transfer under matched conditions. We conduct 42 trials across 4 representative environments (shopping mall, park, office building, outdoor night) that share key characteristics with EVT-Bench simulation tasks: presence of human targets, indoor/outdoor variation, dynamic obstacles, and varying illumination.

Protocol design.

- **Target consistency**: Each trial uses the same designated human target for both Track-MM and OmTrackVLA. Targets are selected to represent typical tracking scenarios (pedestrians, shoppers, office workers).
- **Scenario matching**: For each environment, we define a canonical route and maintain consistent starting positions. While exact replication across trials is impossible in uncontrolled real-world settings (e.g., dynamic pedestrian flow, lighting changes), we enforce route consistency within each scenario and ensure both methods encounter comparable environmental conditions.
- **Success criteria**: Success requires the robot to maintain 1–3m distance from the target, frontal view (target visible in frame), and collision-free navigation. These criteria are assessed frame-by-frame from recorded video and verified by two independent annotators. Trials are marked as failures if the robot loses the target for >5 consecutive seconds or collides with obstacles.
- **Baseline comparison**: Both methods use the same robot platform (Unitree Go2-W), camera configuration, and ROS2 control stack. OmTrackVLA is evaluated using the publicly released checkpoint (v0.1.2-stable) without modification. Trials for each method are con-

Method	Scenario	Failure type	Description
Track-MM	Shopping mall	Distractor switch	The tracker confused the target with a visually similar pedestrian.
Track-MM	Outdoor night	Low light	The target was lost under poor illumination.
Track-MM	Park	Target leaves FOV	Rapid lateral motion caused the target to leave the camera field of view.
Track-MM	Office	Collision	The robot collided with furniture while navigating through a constrained area.
Track-MM	Outdoor night	System delay	Increased prediction latency delayed the tracking response on the edge platform.
OmTrackVLA	Shopping mall	Distractor switch	The tracker switched to visually similar pedestrians in crowded scenes.
OmTrackVLA	Outdoor night	Low light	The target was lost under poor illumination.
OmTrackVLA	Park	Target leaves FOV	Slow response to rapid lateral motion prevented timely target re-acquisition.
OmTrackVLA	Office	Collision	The robot collided with obstacles while navigating narrow corridors.

Table 9: Representative failure modes observed during the real-world evaluation of Track-MM and OmTrackVLA.

ducted sequentially within the same time window (same day, similar lighting) to minimize environmental variation.

- **Duration and scope**: Individual trial duration ranges from 30 seconds to 3 minutes depending on target motion. Total deployment time is approximately 8 hours including setup and transitions.

Statistical interpretation. The observed success rates (Track-MM: 37/42 = 88.1%, OmTrackVLA: 29/42 = 69.0%) yield a Fisher exact test $p = 0.061$. While this does not reach the conventional $p < 0.05$ threshold, the marginal significance combined with the consistency of the advantage across diverse scenarios (3 out of 4 environments show $>$10 pp gains) provides evidence for improved real-world robustness. We acknowledge that a larger sample size would be required to establish statistical significance at stricter thresholds.

Limitations of real-world evaluation. Despite protocol controls, several sources of uncontrolled variation remain: (1) dynamic pedestrian flow and obstacle placement vary across trials; (2) exact lighting conditions cannot be replicated for sequential method comparisons; (3) human target motion is inherently stochastic and cannot be scripted. These factors contribute to within-method variance and limit the power of statistical tests. Future work should consider paired-trial designs where both methods are evaluated simultaneously on identical episodes (e.g., via video replay or scripted robot targets).

A.5.3 Failure Taxonomy

Table 9 summarizes the representative failure modes observed during the real-world trials. Both methods remain vulnerable to visual ambiguity, poor illumination, rapid target motion, and navigation failures. Track-MM additionally exhibited an occasional system-level delay on the edge platform. Overall, distractor confusion and low-light conditions remain important challenges for robust embodied visual tracking.

A.6 Efficiency and Latency Breakdown

Figure 8 presents a Pareto-style comparison between model size and DT success rate. Track-MM uses a Qwen3-0.6B policy backbone and has 1.2B parameters in total, placing it in the upper-left region with the best performance–efficiency trade-off. In contrast, several strong baselines rely on 7B-scale backbones, such as VLingNav, NavFoM and TrackVLA++. Despite its smaller model size,

Track-MM attains a DT success rate competitive with all 7B-scale baselines, suggesting that structured dual-memory modeling and the lightweight target-aware motion adapter are more effective than simply scaling the policy backbone.

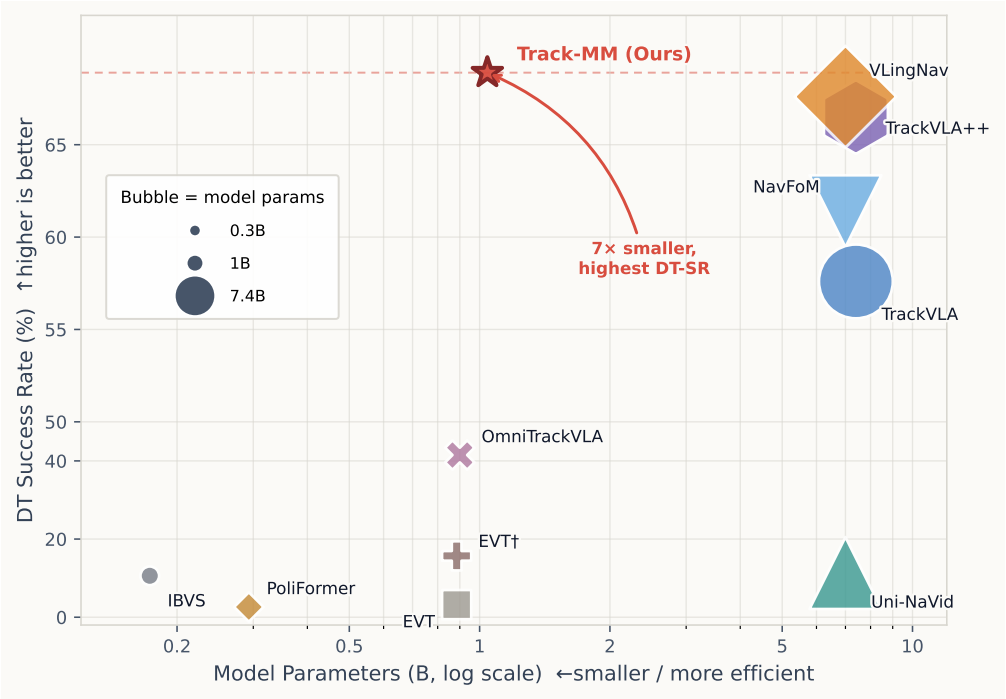


Figure 8: Efficiency–capability Pareto frontier on EVT-Bench DT.

We profile the runtime of each component with a batch size of one using FP16 inference. Table 10 reports the average per-frame latency on the Jetson Orin NX and RTX 3090. The complete Track-MM pipeline achieves an end-to-end latency of 161 ms on the Jetson platform, corresponding to approximately 6.2 FPS, with a measured peak memory usage of 3.1 GB. The flow-matching action head uses five Euler integration steps, and the video encoder processes a temporal window of $K = 4$ frames.

Component	Jetson (ms)	RTX 3090 (ms)
Image preprocessing	5	2
DINOv3 + SigLIP	42	11
V-JEPA 2.1	35	9
TMA + prediction heads	18	5
Qwen3-0.6B	48	22
Flow-matching action head	11	6
ROS2 communication/control	2	—
End-to-end	**161**	**55**

Table 10: Per-frame latency breakdown with batch size 1 and FP16 inference.

Table 11 summarizes the model scale and reported onboard deployment efficiency of representative embodied visual tracking methods. Track-MM contains approximately 1.2B parameters, of which only about 63M (5.2%) are trainable, and achieves 6 FPS with fully onboard inference on a Jetson Orin NX with 8 GB of memory. TrackVLA++ reports an inference speed of 4.8 FPS on an NVIDIA RTX 4090 in its original paper. However, owing to its substantially larger model size and memory requirements, we do not evaluate TrackVLA++ using fully onboard inference on the 8 GB Jetson platform. Its reported desktop-GPU throughput is therefore mentioned only for reference and is not

Method	Total params	Trainable params	Onboard FPS
OmTrackVLA	$\sim$1.0B†	$\sim$13M†	N/R
TrackVLA++	$\sim$7B	$\sim$0.5B†	N/R
VLingNav	$\sim$7B	N/R	N/R
Track-MM	$\sim$1.2B	$\sim$63M	6*

Table 11: Model scale and reported onboard inference speed. Values marked with † are estimated from public implementations or model configurations. $*$ denotes fully onboard inference measured on a Jetson Orin NX 8 GB using batch size 1 and FP16 inference. N/R denotes not reported.

Item	Configuration
Data and input configuration	
Simulator	Habitat with HM3D and MP3D scenes
Training trajectories	12,358
Input resolution	384×384
Appearance history	31 historical frames
Motion window	$K = 4$ observations
Waypoint horizon	8 waypoints, each in $\mathbb{R}^3$
Optimization	
Optimizer	AdamW
Learning rate	2×10^{-4}
Batch size	16
Training epochs	10
Training hardware	NVIDIA RTX 5000 Ada
Weight decay	0.01
Gradient clipping	1.0
Learning-rate schedule	Cosine decay with 500-step warm-up
Loss weights	$\lambda_{\text{mask}} = 0.3$, $\lambda_{\text{dice}} = 0.5$
Model and inference configuration	
Frozen modules	DINOv3, SigLIP, V-JEPA 2.1, Qwen3-0.6B, and Grounding-DINO text–image attention
Flow-matching	Euler method with 5 inference steps
Random seeds	
Data/environment seed	42, fixed across training runs
PyTorch initialization	3 independent seeds; see Table 13
Evaluation seeds	42, 45, 66, 88, and 100 per checkpoint

Table 12: Dataset, optimization, model, inference, and random-seed settings used in the experiments.

included in the table. Since the compared methods do not report onboard throughput under matched hardware and runtime settings, the table should not be interpreted as a controlled speed comparison.

A.7 Training Details and Reproducibility

Table 12 summarizes the dataset configuration, optimization settings, and inference protocol used in our experiments. We train all learnable components for 10 epochs using AdamW, while keeping the pretrained visual, video, language, and text–image attention modules frozen. Unless otherwise specified, the same configuration is used across all task settings and ablation experiments.

A.7.1 Multi-Seed Training and Evaluation

We evaluate the robustness of Track-MM to both training randomness and evaluation conditions. We train three independent checkpoints while fixing the data-sampling and training-environment seed to 42. The three runs use different PyTorch seeds, thereby varying model initialization and other stochastic optimization operations while keeping the training data distribution fixed.

Each checkpoint is evaluated on DT and AT using five environment and episode-sampling seeds: 42, 45, 66, 88, and 100. This yields five evaluation results per checkpoint and 15 results per task. For each checkpoint, we first average SR over the five evaluation seeds. In Table 13, SR is reported as the mean and standard deviation of the three checkpoint-level averages. The evaluation standard deviation is computed within each checkpoint across the five evaluation seeds and then averaged over the three checkpoints.

Task	SR (%, mean $\pm$ train std.)	Eval std. (pp)
DT	68.7 ± 1.2	0.5
AT	66.5 ± 1.4	0.6

Table 13: Multi-seed performance on DT and AT.

A.8 Baseline Implementation Details

We summarize the model configurations, evaluation settings, and result sources of the principal baselines considered in the main paper. Reported quantities are taken from the corresponding publications when available, whereas values reconstructed from public implementations are explicitly marked as estimates. Unless otherwise stated, we use publicly released checkpoints without retraining.

A.8.1 OmTrackVLA

OmTrackVLA is an open-source vision–language–action model for embodied visual tracking released by Lee et al. (2025) through its public repository.[1] Because no peer-reviewed paper was available at the time of our evaluation, its configuration is reconstructed from the released code and checkpoint.

Model configuration. The released model uses DINOv3 and SigLIP visual encoders together with a Qwen3-0.6B planner backbone to predict waypoint trajectories from RGB observations and language instructions. Based on the public configuration, we estimate approximately 1.0B total parameters and 13M trainable planner-side parameters. The estimate includes the visual projector, temporal-view embeddings, action token, and waypoint head; the pretrained visual encoders and planner backbone are treated as frozen.

Evaluation. We evaluate the publicly released checkpoint without retraining, using the same observation resolution, action space, and evaluation protocol as Track-MM. On the Jetson Orin NX, we measure an approximate end-to-end latency of 180 ms under our deployment configuration. Because implementation details and runtime optimizations differ between methods, this value is provided only as a reference rather than as a component-wise controlled comparison.

Comparison with Track-MM. OmTrackVLA and Track-MM have comparable total parameter scales ($\sim$1.0B and $\sim$1.2B, respectively), but Track-MM allocates more trainable capacity to target-aware motion adaptation and explicit dual memory ($\sim$63M versus an estimated $\sim$13M). Under this protocol, Track-MM improves DT and AT SR over OmTrackVLA by 27.4 and 6.8 percentage points, respectively.

A.8.2 TrackVLA++

TrackVLA++ extends TrackVLA with additional motion-modeling and memory components. Its published configuration uses an approximately 7B-scale backbone. We estimate approximately 0.5B trainable parameters when including its additional motion modules and parameter-efficient backbone adaptation. This value should be interpreted as an approximate model-level estimate rather than an exact parameter count.

TrackVLA++ is trained on the EVT-Bench training data together with additional RoboVLM data and uses 20-step diffusion sampling for action generation. We evaluate its released checkpoint without retraining. The original paper reports 4.8 FPS on an NVIDIA RTX 4090 but does not provide fully onboard inference results on an edge platform comparable to our Jetson Orin NX. Owing to its substantially larger model scale and memory requirements, we do not evaluate TrackVLA++ using fully onboard inference on the 8 GB Jetson platform. Its published throughput is therefore treated as a reference rather than a directly matched efficiency comparison.

[1] https://github.com/om-ai-lab/OmTrackVLA

TrackVLA++ also reports that multi-view visual observations provide a substantial improvement over single-view input. This observation is relevant when interpreting methods that use additional camera views, since the number of visual viewpoints constitutes an important experimental variable independent of the policy architecture.

A.8.3 VLingNav

VLingNav is a concurrent language-conditioned navigation method using an approximately 7B-scale VLA backbone. Its publication does not provide a complete breakdown of frozen and trainable parameters. The published architecture also does not describe explicit dual-memory or latent motion-prior modules comparable to those in Track-MM.

Because matched onboard latency measurements are unavailable, we compare VLingNav and Track-MM primarily in terms of benchmark performance and reported model scale. We do not draw a direct conclusion regarding runtime efficiency across the two methods.

A.8.4 CoMaTrack

CoMaTrack reports competitive embodied-tracking performance but uses synchronized observations from four camera viewpoints, whereas Track-MM and the baselines in our main comparison use a single-view RGB stream. TrackVLA++ reports that multi-view input provides a substantial performance improvement, making the number of camera views an important confounding variable in a direct comparison. Furthermore, the CoMaTrack implementation and pretrained checkpoint were not publicly available at the time of submission, preventing us from reevaluating it under our single-view protocol. We therefore discuss CoMaTrack as relevant concurrent work but exclude its reported results from the quantitative comparison. A controlled evaluation under matched camera inputs will be considered once its implementation becomes available.

A.8.5 Evaluation Protocol Consistency

All baselines are evaluated on the same EVT-Bench test split using identical success criteria:

- Success: Agent maintains 1–3m distance from target, frontal view, no collision
- Episode length: 500 timesteps maximum
- Environment seed: 42 for main results; 5 evaluation seeds $\times$ 3 training runs for Table 13
- Baselines are evaluated with the same 5 evaluation seeds on their single released checkpoint, since we did not retrain them

Baseline checkpoints are obtained from official releases (OmTrackVLA, TrackVLA++) or reimplemented following published papers (VLingNav). We did not retrain any baseline models; all comparisons use publicly available or author-provided checkpoints evaluated under our protocol.

A.9 Limitations

Despite its competitive performance, Track-MM has several limitations:

- **Onboard inference speed.** Track-MM achieves approximately 6 FPS using fully onboard inference on a Jetson Orin NX. While this throughput is sufficient for the tracking scenarios considered in this work, it may be inadequate for rapidly moving targets, abrupt camera motion, or safety-critical maneuvers that require a higher control frequency. Future work will explore model compression, feature caching, and more efficient temporal encoders.
- **Scale of the real-world evaluation.** Our real-world experiments cover four representative environments with 42 trials per method. However, the evaluation remains limited in scale, and variables such as illumination, pedestrian behavior, target motion, and background dynamics cannot be perfectly controlled across sequential trials. Larger evaluations with more diverse environments, randomized execution orders, and better matched trial conditions are needed to establish broader real-world generalization.

- **Frozen pretrained backbones.** The pretrained visual and video backbones remain frozen throughout training to reduce computational cost and preserve their general representations. We do not investigate whether full fine-tuning or parameter-efficient adaptation of these backbones could provide stronger task-specific appearance and motion features. Exploring lightweight adaptation strategies may further improve tracking under domain shifts and challenging visual conditions.
- **Shared observability prior.** The appearance and motion gates share the observability prior v^t. This conservative design suppresses unreliable memory updates when the target is poorly observed. However, it may also freeze the motion memory during occlusion even when the latent motion representation retains useful temporal evidence. Future work will investigate branch-specific observability estimation and more adaptive memory updates during temporary target absence.