DECOWAM: Decoupled Whole-Body World-Action Model for Legged Mobile Manipulation

Siyuan Ma[*,1], Boshi Zhang[*,1], Yutian Zhang[*,2], Qinglian Wu[3],
Jiaqi Zhai[4], Dong Wei[†,4], Qiaojun Yu[†,2]
[1]Tsinghua University, Beijing, China
[2]Shanghai Artificial Intelligence Laboratory, Shanghai, China
[3]Harbin Institute of Technology, Harbin, China
[4]Hangzhou Yunshenchu Technology Co., Ltd. (DEEP Robotics), Hangzhou, China
[*]Equal contribution. [†]Corresponding authors.

***Abstract*—Mobile manipulation requires a robot to predict how locomotion and arm motion jointly alter future observations and control. Existing world–action models, developed largely for fixed-base platforms, do not explicitly distinguish camera ego-motion from base and arm actions. Here we introduce DECOWAM, a whole-body world–action model that separates these factors through dedicated conditional interfaces. DECOWAM freezes an adapted FastWAM backbone and trains residual adapters, an action-equivalent future bottleneck distilled from privileged observations, adversarially separated base and arm latents, and base-velocity conditioning for video prediction. We further introduce ARMDOG, a real-robot dataset that synchronizes video, whole-body state and action, and language. On a fixed replay protocol, DECOWAM improved both future-video and action prediction over FastWAM, reducing action MSE by 21.7% with 25.95M trainable adaptation parameters. Across 79 closed-loop trials per method, it achieved the highest observed whole-body coordination and base-displacement robustness among the compared systems, while task completion remained comparable to the strongest baseline. These results show that embodiment-aware factorization can support parameter-efficient joint visual prediction and whole-body control under moving viewpoints.**

I. Introduction

A robot that can both *move through* and *act on* the physical world is the long-standing target of embodied AI research. Recent vision–language–action (VLA) and world–action models—Aloha [1], RT-1/2 [2], [26], OpenVLA [3], PaLM-E [27], SayCan [28], π_0 [4], RDT-2 [5], and Motus—have produced impressive results on stationary bimanual platforms, where the camera geometry is fixed and the action space is restricted to arm trajectories. Legged mobile manipulators change the modeling problem: the camera is carried by a moving base, and the policy must coordinate high-rate arm motion with lower-rate base velocity commands.

a) Problem formulation.: Given an instruction ℓ, a current RGB observation x_0, and robot state s, a whole-body world–action model predicts a future action chunk $\hat{\mathbf{a}}_{1:K}$ and a future video clip $\hat{x}_{1:T}$. In our setting, $\mathbf{a}_{1:K} \in \mathbb{R}^{K\times 14}$ contains arm joints, gripper state, base velocity, and loader-compatible padding. The difficulty is structural. First, the camera coordinate system changes with the legged base, so apparent image motion mixes scene dynamics, arm motion, and ego-motion. Second, the action vector is multi-factor: arm/gripper channels and base-velocity channels have different semantics and different control time scales. Third, base velocity has a dual role: it is a target to be predicted by the action expert and an observation that explains camera motion for the video expert.

b) Why is legged mobile manipulation harder?: These properties make legged-arm modeling different from fixed-base manipulation:

- **Dynamic viewpoint.** The on-board camera moves with the base. Hand–eye geometry varies continuously, and image streams contain a mixture of ego-motion and scene motion. A mobile-manipulation world model needs a route for representing camera motion rather than treating all pixel displacement as scene dynamics.
- **Multi-rate action coupling.** Arm joint trajectories require high-rate control ($\sim$15–30 Hz), while base velocity commands are typically issued at a lower rate ($\sim$3–5 Hz). Concatenating both into a single uniformly sampled action chunk asks one representation to cover navigation-scale velocity and manipulation-scale joint corrections at the same time.
- **Hierarchical intent.** Real tasks interleave *where to go* with *how to act*. A single monolithic latent struggles to represent navigation-scale decisions and manipulation-scale corrections simultaneously.

c) Our approach.: The central idea of this paper is a decoupled modeling paradigm for mobile manipulation: a world–action model should represent *where the base moves*, *how the arm acts*, and *how the camera ego-motion changes future pixels* as explicit factors. We implement this paradigm on top of FastWAM, a Wan-2.2-based world–action backbone with a paired ActionDiT branch for action chunks. The action-equivalent future bottleneck supplies a modest causal training signal from privileged future latents. Staged frozen adaptation turns the method into a parameter-efficient system by keeping

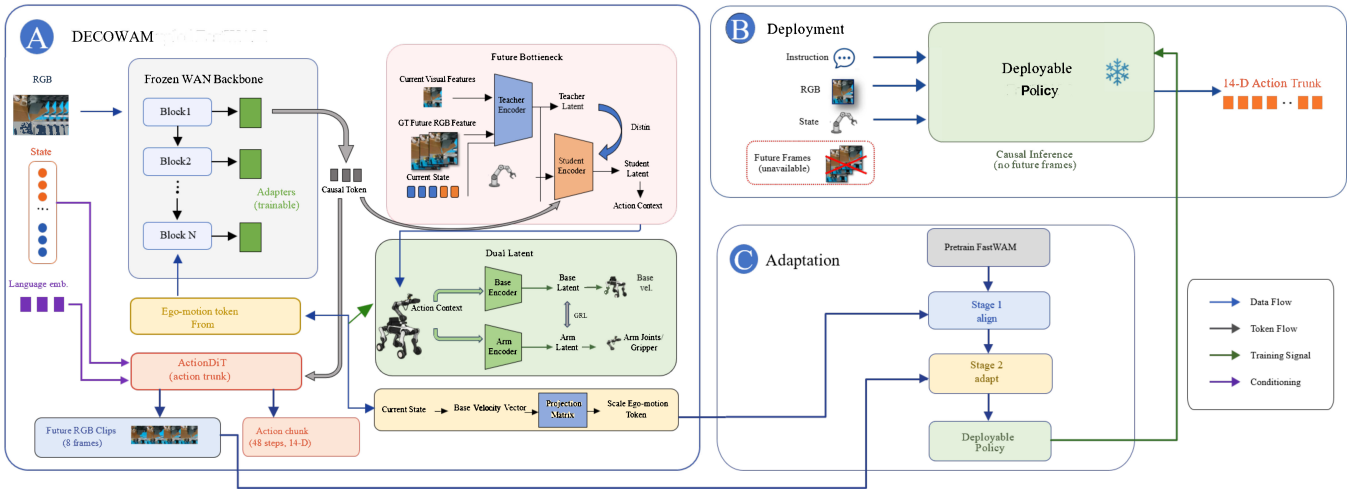

Fig. 1. DECOWAM architecture, training, and deployment. (A) A frozen WAN backbone is augmented with trainable residual adapters, a teacher–student future bottleneck, separated base/arm latents, and base-velocity ego-motion conditioning; ActionDiT produces future RGB clips and a 48-step, 14-D action chunk. (B) Deployment removes future-frame inputs and the privileged teacher path, yielding strictly causal 14-D action inference. (C) Staged training first aligns pretrained FastWAM to ARMDOG and then adapts the decoupled modules to obtain the deployable policy.

the base FastWAM prior fixed in the final stage and learning only residual robot-specific pathways. The base/arm dual latent with GRL is the main action-factorization mechanism, separating navigation-scale base commands from manipulation-scale arm commands. The base-velocity token is the explicit ego-motion interface for the video branch: it conditions future visual rollout on the current normalized base velocity rather than asking the video branch to infer camera motion only from pixels.

d) Dataset.: Method development is only half of the story. Legged mobile manipulation requires data in which visual change, base ego-motion, arm motion, and language intent are synchronized rather than recorded as separate logs. We therefore build ARMDOG, a real-robot data resource for a quadrupedal platform with a 6-DoF arm. Its contribution is the embodiment-complete model interface: each converted episode aligns a 15 Hz RGB video stream, a $T\times14$ whole-body state/action tensor with explicit base and arm channels, natural-language instruction text, and a precomputed language embedding. The current converted world–action snapshot contains 217 episodes from 27 task folders and 56,041 synchronized frames. This data organization is what makes it possible to train and evaluate base/arm factorization, ego-motion-aware video conditioning, and whole-body world–action prediction within one replay protocol.

e) Contributions.:

1) We formulate legged mobile manipulation as a decoupled world–action modeling problem in which base control, arm manipulation, and camera ego-motion enter through explicit, semantically aligned interfaces.
2) We realize this formulation in DECOWAM through frozen residual adaptation, causal distillation from privileged future latents, adversarial base/arm factorization, and base-velocity-conditioned video prediction.
3) We introduce the synchronized ARMDOG dataset and evaluate DECOWAM through controlled replay and real-robot experiments, demonstrating parameter-efficient prediction, improved whole-body coordination, and stronger perturbation tolerance.

II. Related Work

a) Vision–language–action models for fixed-base robots.: Modern VLA models—RT-1/2 [2], [26], OpenVLA [3], π_0 [4], Octo [7], RDT-2 [5], X-VLA [6], PaLM-E [27], SayCan [28], Gato [29], and RoboCat [30]—map language and pixels to robot actions through transformer policies pre-trained on large heterogeneous corpora. Task-conditioned manipulation policies such as BC-Z, PerAct, VIMA, and Diffusion Policy [25], [31]–[33] provide complementary evidence that language, visual tokens, and action trajectories can share a policy interface. These models provide strong action-policy references, and we include adapted $\pi_{0.5}$ and X-VLA runs in the replay experiments. Their standard action heads do not explicitly model future RGB rollout or separate base-induced camera motion from arm-induced scene change.

b) World models for robotic control.: World-model approaches [8]–[10], [34], [35] generate future observations and use them either as a learned simulator or as a structural prior on the policy. Video diffusion and interactive-video systems [11], [36], [37] further show that future visual prediction can serve as a powerful generative prior. Recent video-based world–action models such as UniSim [11], UVA [12], X-WAM [13], FastWAM, and Motus combine visual prediction with action modeling. Our work does not pursue general simulator scaling. It studies embodiment-specific adaptation: how to keep a pretrained video prior useful while separating ego-motion, base action, arm action, and future-equivalent control information on a legged-manipulation dataset.

c) Locomotion and mobile manipulation.: Legged locomotion has been studied extensively through reinforcement

learning [15], [16], with controllers that output base velocity commands but do not engage manipulation. Wheeled and home mobile-manipulation systems [17]–[19], [23], [38], [39] have produced cascaded policies that separate navigation and manipulation. Mobile ALOHA [40] demonstrates the value of whole-body teleoperation for bimanual mobile manipulation, but few works learn a unified policy, and fewer still ground that policy in a predictive video model. We are not aware of prior work that performs unified whole-body world-action modeling on a legged base with a manipulator.

d) Robot datasets.: Open-X Embodiment [20], DROID [21], BridgeData V2 [41], RoboNet [42], RLBench [43], CALVIN [44], Language Table [45], ManiSkill2 [46], LIBERO [47], RoboTwin [22], and Aloha-derived collections [1] provide broad coverage for fixed-base or wheeled manipulation, but they do not expose the combination needed by our model: a legged-base ego-motion stream, manipulator actions, visual observations, and language instructions in one synchronized training unit. ARMDOG fills this complementary regime by recording quadrupedal mobile manipulation and converting it into a video–state/action–language format consumed by FastWAM-style world–action models.

III. Background: FastWAM World–Action Backbone

We briefly review the FastWAM architecture, on which our method builds. FastWAM couples a Wan-2.2 video diffusion backbone [14] with an ActionDiT branch that predicts chunked actions using the same continuous-flow training interface. In the ARMDOG configuration, the model consumes the current RGB frame, a precomputed language context, and a 14-D proprioceptive state token, then jointly produces future RGB frames and a 48-step whole-body action chunk. The original FastWAM checkpoint evaluated here has 6725.44M total parameters and 6020.75M trainable parameters during full fine-tuning.

We retain this joint video/action factorization and add four mechanisms that decouple the adaptation problem in parameter space, future-information usage, action factors, and ego-motion conditioning.

IV. Method

A. Problem Formulation

Let ℓ denote a language instruction, x_0 the current RGB observation, and $s_0 \in \mathbb{R}^{14}$ the current whole-body state. We learn the conditional joint model

$$p_\theta(x_{1:T}, \mathbf{a}_{1:K} \mid x_0, s_0, \ell)\,, \tag{1}$$

where $x_{1:T}$ is a future video and $\mathbf{a}_{1:K} \in \mathbb{R}^{K\times 14}$ is an action chunk. We use $T{=}8$ and $K{=}48$. Each action has the semantic decomposition

$$\mathbf{a}_k = \left[\mathbf{a}_k^{\mathrm{arm}}, a_k^{\mathrm{grip}}, \mathbf{a}_k^{\mathrm{base}}, \mathbf{a}_k^{\mathrm{pad}}\right], \qquad \mathbf{a}_k^{\mathrm{base}} \in \mathbb{R}^3. \tag{2}$$

The arm and gripper occupy channels [0:7], base velocity occupies [7:10], and loader padding occupies [10:14]. Base velocity is both a control target and a source of camera ego-motion. DECOWAM therefore separates arm control, base control, and visual ego-motion instead of encoding them in one undifferentiated context.

Our backbone is FastWAM, which pairs an ActionDiT action expert with a WAN video expert. Both experts receive language and proprioceptive context. The action expert predicts flow over action chunks, while the video expert predicts flow over future visual latents. DECOWAM preserves this joint interface and changes how embodiment-specific information conditions each expert. Future observations supervise the model only during training, so deployment remains causal in (x_0, s_0, ℓ).

B. Staged Parameter-Efficient Adaptation

Training separates domain alignment from structural adaptation. In Stage 1, all FastWAM parameters Θ are adapted to ARMDOG for 50k steps:

$$\Theta^{(1)} = \arg\min_{\Theta} \mathbb{E}_{\mathcal{D}}\left[\mathcal{L}_{\mathrm{video}}(\Theta) + \mathcal{L}_{\mathrm{action}}(\Theta)\right]. \tag{3}$$

This stage aligns the video prior, action expert, and proprioceptive interface with the moving-camera observations and quadruped–arm action space.

Stage 2 freezes $\Theta^{(1)}$ and optimizes only

$$\Phi = \{\phi_{\mathrm{adp}}, \phi_q, \phi_{\mathrm{ba}}, \phi_{\mathrm{ego}}\}\,, \qquad \Phi^\star = \arg\min_{\Phi} \mathcal{L}\Big(\Theta^{(1)}, \Phi\Big)\,. \tag{4}$$

The four parameter groups represent residual adapters, an action-equivalent future bottleneck, base/arm factorization, and ego-motion conditioning. This restriction reduces the Stage-2 trainable footprint from 6020.75M to 25.95M parameters.

The frozen WAN backbone is adapted after each block through

$$h_l^+ = h_l + \alpha_l W_{\mathrm{up}}^{(l)} \sigma\Big(W_{\mathrm{down}}^{(l)} \mathrm{LN}(h_l)\Big)\,, \tag{5}$$

where $W_{\mathrm{down}}^{(l)}$ projects to a 128-D bottleneck, $W_{\mathrm{up}}^{(l)}$ restores the hidden dimension, and σ is SiLU. The residual branch learns a compact, robot-specific correction while preserving the pretrained video prior.

C. Decoupled Conditional Interfaces

a) Action-equivalent future bottleneck.: Future frames contain information about action-equivalent outcomes that is unavailable from the current frame alone. We distill this privileged information from a teacher into a causal student.

Let $e_0 = \psi_{\mathrm{vae}}(x_0)$ and $e_{1:T} = \psi_{\mathrm{vae}}(x_{1:T})$ be WAN-VAE latents. We summarize each latent tensor using

$$c = \rho(e_0), \qquad f = \rho(e_{1:T}), \qquad \rho(e) = [\mathrm{mean}(e), \mathrm{std}(e)]\,. \tag{6}$$

The teacher and student embeddings are

$$z_t = q_t([c, f, s_0]), \qquad z_s = q_s([c, s_0]), \qquad z_t, z_s \in \mathbb{R}^{d_q}. \tag{7}$$

Only z_s conditions the causal action expert:

$$\tilde{u}^a = u^a + \eta_q B_q z_s, \tag{8}$$

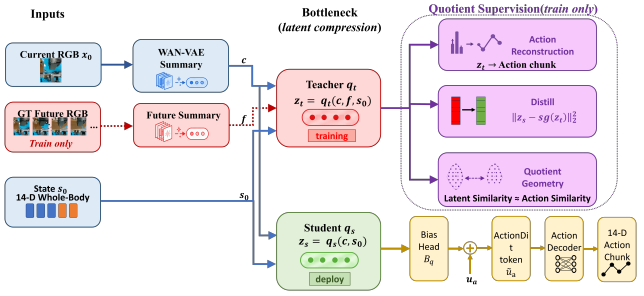

Fig. 2. Future-information bottleneck. A privileged teacher observes current and future visual summaries, whereas the causal student observes only the current summary and robot state. Only the student is retained during deployment.

where u^a denotes its context tokens and $\eta_q \in [0,1]$ controls the residual bias.

The bottleneck is trained with action reconstruction, teacher–student distillation, and geometry preservation:

$$\begin{aligned}\mathcal{L}^q_{\text{rec}} &= \|r_s(z_s) - \mathbf{a}_{1:K}\|_2^2 \\ &\quad + \|r_t(z_t) - \mathbf{a}_{1:K}\|_2^2, \\ \mathcal{L}_q &= \lambda^q_{\text{act}}\mathcal{L}^q_{\text{rec}} + \lambda^q_{\text{dist}}\,\|z_s - \text{sg}(z_t)\|_2^2 \\ &\quad + \lambda^q_{\text{geom}}\mathcal{L}_{\text{geom}},\end{aligned} \tag{9}$$

where sg stops gradients. For a batch of size B, the geometry term is

$$d^{ij}_z = \frac{\|z^i_t - z^j_t\|_2}{\tau_z}, \qquad d^{ij}_a = \frac{\|\mathbf{a}^i_{1:K} - \mathbf{a}^j_{1:K}\|_2}{\tau_a}, \tag{10}$$

$$\begin{aligned}\mathcal{L}_{\text{geom}} &= \big(B(B-1)\big)^{-1} \\ &\quad \times \sum_{i\neq j} \text{SL1}\big(d^{ij}_z, d^{ij}_a\big).\end{aligned} \tag{11}$$

with robust scales τ_z and τ_a obtained from batch medians. Thus, trajectories with similar normalized actions are encouraged to remain close in the privileged latent space. Distillation transfers this structure to the deployable student.

b) Base–arm factorization.: The action context contains navigation-scale and manipulation-scale information. We map its pooled representation into two 16-D factors:

$$z_{\text{base}} = b_\phi(u^a), \qquad z_{\text{arm}} = m_\phi(u^a), \qquad z_{\text{base}}, z_{\text{arm}} \in \mathbb{R}^{16}. \tag{12}$$

Their concatenation conditions the action expert through

$$\bar{u}^a = \tilde{u}^a + \eta_{\text{ba}} B_{\text{ba}}[z_{\text{base}}, z_{\text{arm}}]. \tag{13}$$

Let $\mathbf{a}^b_{1:K} = \mathbf{a}_{1:K,7:10}$ and $\mathbf{a}^m_{1:K} = \mathbf{a}_{1:K,0:7}$. Direct heads preserve the assigned factor, while gradient-reversal cross heads suppress information about the opposite factor:

$$\begin{aligned}\mathcal{L}_{\text{disent}} &= \big\|g_b(z_{\text{base}}) - \mathbf{a}^b_{1:K}\big\|_2^2 + \|g_m(z_{\text{arm}}) - \mathbf{a}^m_{1:K}\|_2^2 \\ &\quad + \big\|\tilde{g}_b(\text{GRL}(z_{\text{arm}})) - \mathbf{a}^b_{1:K}\big\|_2^2 \\ &\quad + \|\tilde{g}_m(\text{GRL}(z_{\text{base}})) - \mathbf{a}^m_{1:K}\|_2^2.\end{aligned} \tag{14}$$

The prediction heads minimize all reconstruction terms, whereas gradient reversal changes the sign of cross-task gradients entering the encoders. Each latent is therefore encouraged to retain its assigned control factor and discard the other.

c) Ego-motion-aware video conditioning.: For a body-mounted camera, apparent image motion combines scene dynamics, manipulator motion, and base-induced viewpoint change. We expose the last component using the normalized current base velocity

$$v_0 = \Pi_{\text{base}}(s_0) = (v_x, v_y, \omega_z) \in \mathbb{R}^3. \tag{15}$$

Each video token receives the same projected ego-motion condition:

$$\tilde{h}^v_i = h^v_i + \beta B_v v_0, \qquad i = 1, \ldots, N_v, \tag{16}$$

where B_v maps velocity into the WAN hidden dimension. This token does not impose geometric warping. It supplies an explicit explanatory variable for camera-frame motion. Base velocity consequently acts as an action target in Eq. (2) and a visual condition in Eq. (16).

D. Training Objective and Deployment

Both experts use conditional flow matching [48]. For a target $y \in \{\mathbf{a}_{1:K}, e_{1:T}\}$, noise $\epsilon \sim \mathcal{N}(0, I)$, and time $\tau \sim \mathcal{U}(0,1)$, define

$$y_\tau = (1-\tau)\epsilon + \tau y, \qquad v^\star(y_\tau, \tau) = y - \epsilon. \tag{17}$$

The corresponding objective is

$$\mathcal{L}_{\text{FM}}(F_\theta; y, c) = \mathbb{E}_{\tau,\epsilon}\left[\|F_\theta(y_\tau, \tau, c) - v^\star(y_\tau, \tau)\|_2^2\right]. \tag{18}$$

We instantiate this loss as $\mathcal{L}_{\text{action}}$ with context $\bar{u}^a$ and as $\mathcal{L}_{\text{video}}$ with context $\tilde{h}^v$. The complete Stage-2 objective is

$$\mathcal{L} = \lambda_v\mathcal{L}_{\text{video}} + \lambda_a\mathcal{L}_{\text{action}} + \gamma_q\lambda_q\mathcal{L}_q + \gamma_{\text{ba}}\lambda_{\text{ba}}\mathcal{L}_{\text{disent}}. \tag{19}$$

We set $\lambda_v = \lambda_a = 1.0$, $\lambda_q = 0.2$, and $\lambda_{\text{ba}} = 0.1$. The reported run sets γ_q, γ_{ba}, η_q, and η_{ba} to one throughout Stage 2.

At inference, the teacher q_t and all auxiliary prediction heads are removed. The model computes z_s, $(z_{\text{base}}, z_{\text{arm}})$, and v_0 from current inputs, then samples both flows using only (x_0, s_0, ℓ).

V. The ARMDOG Dataset

ARMDOG is a model-facing real-robot resource for legged mobile manipulation. It synchronizes moving-camera RGB-D, proprioception, IMU, base state, whole-body commands, and language instructions on a wheeled quadruped robot with 16 leg joints, a 6-DoF arm, and a 1-DoF gripper. The FastWAM-compatible conversion aligns raw HDF5 streams to 15 Hz and stores each episode as $e_i = (V_i, Q_i, \ell_i, \phi(\ell_i))$: RGB video, a structured $T_i \times 14$ state/action tensor, instruction text, and a precomputed language embedding. Training consumes the current frame, eight future frames at 384×320, the initial state, and a normalized 48-step action chunk.

Figure 3 summarizes the dataset at the task, corpus, and model-interface levels. In the full quality-filtered corpus, Bottle

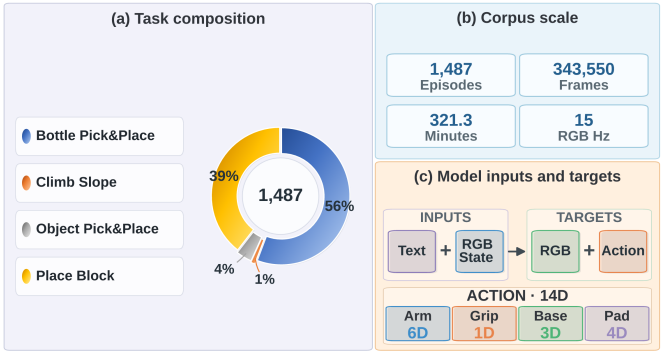

Fig. 3. Composition and model-facing structure of ARMDOG. (a) Episode distribution across four task families in the quality-filtered corpus: Bottle Pick&Place (56%), Place Block (39%), Object Pick&Place (4%), and Climb Slope (1%). (b) Corpus scale after timestamp alignment and quality filtering: 1,487 episodes, 343,550 RGB frames, 321.3 minutes, and a 15 Hz frame rate. (c) Model interface: text, current RGB, and robot state form the inputs, while future RGB and a 14-D action are prediction targets. The action vector separates 6-D arm, 1-D gripper, 3-D base-velocity, and 4-D loader-padding channels.

Pick&Place and Place Block account for 56% and 39% of episodes, respectively, while Object Pick&Place contributes 4% and Climb Slope contributes 1%. This mix emphasizes object-centric mobile manipulation while retaining a smaller locomotion-focused slice. Panel (c) makes the learning contract explicit: language text together with current RGB and state conditions future RGB and structured whole-body action prediction. The corpus-level counts in the figure are distinct from the model-specific training and evaluation subsets described below.

The 14-D tensor exposes the embodiment factors used by our method: indices [0:6] are arm targets, [6] is gripper opening, [7:10] is base velocity (v_x, v_y, ω_z), and [10:14] is loader padding. The June 5, 2026 audit starts from 795 raw HDF5 episodes, quarantines duplicates, aligns streams by timestamp interpolation, repairs short dropouts, and records per-episode integrity metadata. As reported in Fig. 3(b), the dataset after timestamp alignment and quality filtering contains 1487 episodes from 5 task folders and 343,550 frames, approximately 321.3 minutes at 15 Hz. These full-corpus statistics are distinct from the downstream subsets: frozen decoupled Stage-2 uses 214 episodes from 26 tasks after excluding Legacy val, while all replay results use the fixed Box-val slice with 23 episodes, eight tasks, and 4,323 frames. The planned release includes raw/cleaned HDF5 files, the world–action conversion, immutable split manifests, and a datasheet [24].

VI. Experiments

The experiments follow an evidence ladder from controlled prediction to physical deployment. We first compare DECOWAM with its FastWAM initialization, then test the structured paths through ablation. Broader reference comparisons establish its position among VLA and WAM systems. Closed-loop trials finally assess task progress, whole-body coordination, and robustness on the physical robot.

A. Replay Protocol and Compared Models

Open-loop replay uses a fixed 23-episode ***box_val__**** slice drawn from eight box-manipulation task folders. The slice covers ***box_move***, ***box_soft***, and ***box_stay*** variants. FastWAM-family models receive identical current-frame, state, and language inputs. Each model predicts eight future RGB frames at 384×320 resolution and a 48-step, 14-D action chunk. Unless stated otherwise, results are computed over the same 16 replay batches with shared normalization, inputs, and evaluator code.

a) Models.: The primary baseline is the Stage-1 FastWAM model trained on the converted ARMDOG snapshot. Checkpoints at 40k, 50k, and 80k steps expose sensitivity to the stopping point. DECOWAM starts from the 50k checkpoint and freezes all FastWAM parameters during Stage 2. The trainable path contains 128-D WAN adapters, a 64-D action-equivalent future bottleneck, 16-D base and arm latents, and base-velocity conditioning. Adapted action-only VLA systems and runnable WAM systems provide broader context.

b) Metrics.: Video quality is measured by frame MSE, PSNR, global SSIM [49], and LPIPS [50]. Action prediction is evaluated by MSE, MAE, and the mean Euclidean error of each normalized 14-D action vector. These measures jointly cover pixel fidelity, structural agreement, perceptual similarity, and control error over the complete arm–gripper–base interface used by the deployed system.

B. Main Results and Reference Comparisons

Table I summarizes the main FastWAM replay comparison.

TABLE I
Main replay diagnostics on the 23-episode ARMDOG slice.

Model	F-MSE↓	PSNR↑	A-MSE↓
Original FastWAM (40k)	1.616e-3	30.468	1.154e-4
Original FastWAM (50k)	1.032e-3	31.441	6.87e-5
Original FastWAM (80k)	2.136e-3	28.544	4.77e-4
DECOWAM(50k)	**8.77e-4**	**31.663**	**5.38e-5**

DECOWAM reduced frame MSE by 15.03% and action MSE by 21.71% relative to its 50k FastWAM initialization (Table I). PSNR increased by 0.222 dB, providing a consistent pixel-space result. The 50k checkpoint is also the strongest FastWAM checkpoint in this sweep, so Stage 2 improves both prediction branches from the best observed Stage-1 starting point. The following ablations identify how the structured paths contribute to this gain.

a) Internal module ablation.: Table II reports a separately trained ablation suite under the same 16-batch protocol. All variants share the Stage-1 initialization and residual adapters. Two variants remove the action-equivalent future bottleneck or ego-motion token. The adapter-only control removes all three structured paths while retaining the residual adaptation mechanism. Because this suite was trained separately, its full-model row is the appropriate within-suite reference.

TABLE II
Internal ablation of the frozen decoupled FastWAM modules on the same replay protocol.

Variant	F-MSE↓	PSNR↑	SSIM↑	A-MSE↓	A-MAE↓
Full frozen decoupled	**9.35e-4**	**31.378**	**9.9241e-1**	**8.09e-5**	**4.310e-3**
w/o quotient	1.02e-3	31.228	9.9170e-1	8.40e-5	4.316e-3
w/o base-vel.	1.01e-3	31.221	9.9178e-1	8.80e-5	4.342e-3
w/o decoupled	9.80e-4	31.105	9.9138e-1	9.60e-5	5.135e-3

Every removal degraded all reported metrics. Relative to the adapter-only control, the full model reduced frame MSE by 4.6%, action MSE by 15.7%, and action MAE by 16.1%. Removing either the future bottleneck or base-velocity conditioning worsened both output branches, demonstrating that the future-equivalent representation and explicit ego-motion signal improve the shared video–action model beyond residual adaptation alone.

b) Comparison with VLA and WAM references.: The action-only VLA comparison isolates action quality, while the RGB column makes the additional predictive capability of the WAM interface explicit.

TABLE III

Action replay comparison with VLA references on the same 23-episode ARMDOG slice.

Model	Family	RGB	A-MSE↓	A-MAE↓	A-L2↓
$\pi_{0.5}$	VLA	no	1.79e-4	3.60e-3	2.83e-2
X-VLA	VLA	no	**2.11e-5**	**1.82e-3**	**1.00e-2**
GR00T	VLA	no	4.18e-3	2.51e-2	1.83e-1
DECOWAM	WAM	yes, 8f	5.38e-5	4.01e-3	2.24e-2

Table III shows the strength of a dedicated action policy: X-VLA attains the lowest error on all three metrics. DECOWAM is nevertheless second in A-MSE and A-L2, improving them by 69.9% and 21.1% over $\pi_{0.5}$; its A-MAE is within 11.1% of $\pi_{0.5}$ and 84.0% lower than GR00T. Thus, although it is not the top action-only model, DECOWAM remains in the strong VLA range. Moreover, it simultaneously predicts a 48-step whole-body action and eight 384×320 future RGB frames, whereas the VLA references stop at the action chunk. This rollout explicitly forecasts object motion, contact geometry, and base-induced viewpoint change, adding future awareness while retaining competitive Action performance.

We next compare the joint interface with runnable video–action references.

TABLE IV

World–action model references on the same ARMDOG data snapshot.

Model	Train.	RGB	F-MSE↓	PSNR↑	SSIM↑	LPIPS↓	A-MSE↓	A-MAE↓	A-L2↓
DECOWAM	25.95M	8f 384p	**8.77e-4**	**31.66**	**9.93e-1**	**2.95e-2**	**5.38e-5**	**4.01e-3**	**2.24e-2**
FastWAM	6020.75M	8f 384p	1.03e-3	31.44	9.92e-1	3.03e-2	6.87e-5	4.08e-3	2.35e-2
Motus	5894.81M	8f 384p	5.19e-3	23.41	9.57e-1	1.01e-1	5.05e-4	9.97e-3	5.99e-2
Cosmos 2.5	–	8f 384p	4.28e-2	14.38	6.63e-1	2.71e-1	–	–	–
X-WAM	5037.75M	4f 160p	2.62e-3	26.57	9.81e-1	3.50e-2	6.31e-4	1.11e-2	2.48e-2
UVA	261.62M	4f 128p	1.79e-2	19.32	8.40e-1	2.00e-1	9.76e-4	1.31e-2	9.81e-2

Among WAM references, DECOWAM ranks first on every reported video and action metric in Table IV. Under the matched eight-frame 384×320 interface, it lowers FastWAM frame/action MSE by 15.03%/21.71%. Relative to Motus, frame/action MSE falls by 83.1%/89.3%; relative to the shorter, lower-resolution X-WAM output, the reductions are 66.6%/91.5%. DECOWAM therefore adds future prediction over action-only VLAs and, unlike the other WAMs, leads both output branches. Figure 4 visually supports this result: DECOWAM preserves workspace geometry and object layout while competing rollouts accumulate blur or viewpoint drift.

Fig. 4. Qualitative comparison with WAM references on one fixed ARMDOG replay sample. The panels show the ground-truth future RGB montage and predictions produced through each model's native interface. Red dashed lines mark camera-view boundaries. In this example, DECOWAM better preserves workspace geometry and object layout, while several references accumulate blur or viewpoint drift.

C. Real-Robot Deployment Experiments

Closed-loop evaluation uses the physical quadruped–arm platform with common observations, language inputs, low-level control, and safety constraints. Each method is tested in 79 trials. Table V(a) reports completion time and cumulative progress through approach, grasp, transport, placement, and task completion. Every stage uses all attempts as its denominator, revealing where failures accumulate.

Table V(b) evaluates base docking, whole-body coordination, base-displacement robustness, and autonomous recovery. Percentages are computed from counts out of 79 and rounded to one decimal place. Together, the two parts measure both end-to-end task progress and the whole-body capabilities required to sustain that progress on hardware.

DECOWAM completes 46 tasks (58.2%) with a mean completion time of 49 s. It records the highest approach and transport rates among the methods in Table V(a), and carries 96.4% of successful grasps into transport. The table therefore confirms that the future-video branch sustains strong closed-loop Action performance throughout the task. Among WAMs, DECOWAM is 16 s faster than FastWAM and 33 s faster than X-WAM, while achieving 58.2% success versus 57.0% and 15.2%. Its stage profile highlights mobile-base approach and post-grasp transport as particular strengths of decoupled whole-body modeling.

TABLE V(b)

Real-robot whole-body robustness over 79 trials per method. All entries are success rates (%); best values are bold.

Model	BD-SR↑	WBCM-SR↑	BDP-SR↑	AR-SR↑
GR00T	70.9	16.5	1.3	11.4
$\pi_{0.5}$	**87.3**	36.7	11.4	25.3
FastWAM	83.5	34.2	12.7	27.8
X-WAM	69.6	27.8	5.1	12.7
DECOWAM	**87.3**	**44.3**	**30.4**	**32.9**

In Table V(b), BD-SR, WBCM-SR, BDP-SR, and AR-SR denote docking, coordination, displacement robustness, and recovery. DECOWAM ties the highest docking rate and leads coordination, displacement robustness, and recovery. Against FastWAM, WBCM-SR/BDP-SR rise by 10.1/17.7 points;

TABLE V(a)

Real-robot task progress and completion efficiency over 79 trials per method. Stage completion rates are cumulative over all attempts; best values are bold.

Model	Mean Completion Time (s) ↓	Cumulative Approach Completion Rate (%) ↑	Cumulative Grasp Completion Rate (%) ↑	Cumulative Transport Completion Rate (%) ↑	Cumulative Placement Completion Rate (%) ↑	Task Success Rate (%) ↑
GR00T	57	73.4	13.9	11.4	8.9	8.9
$\pi_{0.5}$	50	89.9	62.0	53.2	49.4	49.4
FastWAM	65	91.1	63.3	59.5	57.0	57.0
X-WAM	82	77.2	26.6	19.0	15.2	15.2
DECOWAM	**49**	**92.4**	**69.6**	**67.1**	**58.2**	**58.2**

against X-WAM, by 16.5/25.3 points. Its clearest hardware advantage is therefore sustained base–arm coordination as viewpoint and contact geometry evolve.

Relative to FastWAM, DECOWAM improves approach, grasp, transport, and placement by 1.3, 6.3, 7.6, and 1.2 percentage points, respectively, while reducing mean completion time by 16 s. Relative to X-WAM, the corresponding gains are 15.2, 43.0, 48.1, and 43.0 points, together with a 33 s reduction in completion time. The largest margins appear in transport, whole-body coordination, and base-displacement robustness—precisely where locomotion changes the camera viewpoint while the arm preserves a manipulation constraint. DECOWAM therefore improves both final task completion and the continuity of the intermediate behavior that enables it.

These real-robot results isolate the contribution of world–action modeling. DECOWAM produces an explicit eight-frame prediction alongside its whole-body action chunk. This performance follows from three innovations: the future bottleneck transfers scene evolution to control, dual latents separate navigation from manipulation, and base-velocity conditioning exposes camera ego-motion to the video expert. Together they keep future-aware commands aligned with the changing physical scene, extending coupled WAMs with more coherent whole-body behavior.

Figure 5 visualizes this advantage. X-VLA and FastWAM fail to keep the base stationary during the tabletop reach. We attribute this behavior to extensive training on motion-heavy data, which biases the baseline policies toward continued movement and degrades their static-hold capability. DECOWAM instead preserves base pose, arm reach, and end-effector alignment to complete the interaction. Together with Table V(a), the example shows that future-aware world–action modeling complements raw Action accuracy by keeping commands coherent with scene evolution under whole-body motion.

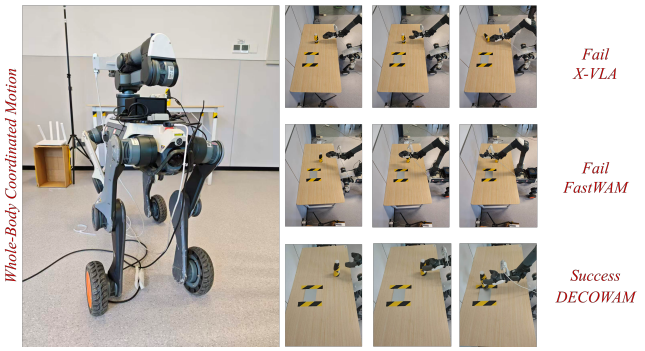

Fig. 5. Representative whole-body coordinated-motion trials on the quadruped–arm platform. The three frames in each row progress from left to right. X-VLA (top) and the original FastWAM (middle) exhibit unintended base motion when a static pose is required and fail to finish the interaction, whereas DECOWAM (bottom) succeeds. The example qualitatively complements the aggregate WBCM-SR results in Table V(b).

D. Parameter-Efficient Adaptation and Deployment

Table VI separates total model size, trainable adaptation parameters, and evaluator latency. This distinction is necessary because freezing parameters reduces optimization cost without shrinking the deployed network.

TABLE VI

Deployment diagnostics on the DECOWAM checkpoint.

Model	Total M	Train. M	ms↓	F-MSE↓	A-MSE↓
FastWAM	6725.44	6020.75	**1196.6**	1.032e-3	6.87e-5
DECOWAM	6751.38	**25.95**	1333.2	**8.77e-4**	**5.38e-5**

DECOWAM reduces the number of parameters updated during Stage 2 by approximately 232-fold, from 6020.75M to 25.95M. This concentrated adaptation path improves both frame and action MSE while adding only 11.4% evaluator latency. Combined with the replay and robot results, the deployment profile defines a distinct operating point: a causal video–action interface with explicit embodiment factors, future-frame prediction, and parameter-efficient robot specialization.

VII. Conclusion

We presented DECOWAM, a whole-body world–action model that explicitly separates base motion, arm manipulation, and camera ego-motion. On synchronized ARMDOG data, staged frozen adaptation and the future-equivalent bottleneck improve every reported FastWAM video/action diagnostic while reducing Stage-2 trainable parameters from 6.021B to 25.95M. DECOWAM remains close to strong VLA Action performance while predicting eight future RGB frames, and leads the real-robot approach, transport, coordination, and displacement-robustness rates over 79 trials per method. The combined results establish a joint model that couples competitive control with explicit prediction of scene evolution under whole-body motion.

REFERENCES

[1] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning fine-grained bimanual manipulation with low-cost hardware," in *Proc. RSS*, 2023.

[2] A. Brohan *et al.*, "RT-2: Vision–language–action models transfer web knowledge to robotic control," in *Proc. CoRL*, 2023.

[3] M. J. Kim *et al.*, "OpenVLA: An open-source vision–language–action model," *arXiv:2406.09246*, 2024.

[4] K. Black *et al.*, "π_0: A vision–language–action flow model for general robot control," *arXiv:2410.24164*, 2024.

[5] S. Liu *et al.*, "RDT-1B: A diffusion foundation model for bimanual manipulation," *arXiv:2410.07864*, 2024.

[6] X-VLA Team, "X-VLA: Soft-prompted transformer as a scalable cross-embodiment vision-language-action model," *arXiv:2510.10274*, 2025.

[7] Octo Model Team, "Octo: An open-source generalist robot policy," in *Proc. RSS*, 2024.

[8] D. Ha and J. Schmidhuber, "World models," in *Proc. NeurIPS*, 2018.

[9] D. Hafner *et al.*, "Mastering diverse domains through world models," *arXiv:2301.04104*, 2023.

[10] J. Bruce *et al.*, "Genie: Generative interactive environments," in *Proc. ICML*, 2024.

[11] S. Yang *et al.*, "Learning interactive real-world simulators," in *Proc. ICLR*, 2024.

[12] S. Li, Y. Gao, D. Sadigh, and S. Song, "Unified video action model," *arXiv:2503.00200*, 2025.

[13] X-WAM Team, "Unified 4D world action modeling from video priors with asynchronous denoising," *arXiv:2604.26694*, 2026.

[14] "Wan-2.2: An open foundation video model," technical report, 2025.

[15] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning," in *Proc. CoRL*, 2022.

[16] G. B. Margolis and P. Agrawal, "Walk these ways: Tuning robot control for generalization with multiplicity of behavior," in *Proc. CoRL*, 2023.

[17] R. Yokoyama, A. Clegg, E. Undersander, S. Ha, D. Batra, and A. Rai, "Adaptive skill coordination for robotic mobile manipulation," in *Proc. CoRL*, 2023.

[18] F. Xia, C. Li, R. Martín-Martín, O. Litany, A. Toshev, and S. Savarese, "ReLMoGen: Integrating motion generation in reinforcement learning for mobile manipulation," in *Proc. ICRA*, 2021.

[19] "Spot mobile manipulation reports," Boston Dynamics technical reports, 2023.

[20] Open X-Embodiment Collaboration, "Open X-Embodiment: Robotic learning datasets and RT-X models," in *Proc. ICRA*, 2024.

[21] A. Khazatsky *et al.*, "DROID: A large-scale in-the-wild robot manipulation dataset," in *Proc. RSS*, 2024.

[22] "RoboTwin 2.0: A bimanual simulation benchmark," technical report, 2025.

[23] "HomeRobot: Open vocabulary mobile manipulation," in *Proc. CoRL*, 2023.

[24] T. Gebru *et al.*, "Datasheets for datasets," *Commun. ACM*, vol. 64, no. 12, pp. 86–92, 2021.

[25] C. Chi *et al.*, "Diffusion policy: Visuomotor policy learning via action diffusion," in *Proc. RSS*, 2023.

[26] A. Brohan *et al.*, "RT-1: Robotics transformer for real-world control at scale," *arXiv:2212.06817*, 2022.

[27] D. Driess *et al.*, "PaLM-E: An embodied multimodal language model," in *Proc. ICML*, 2023.

[28] M. Ahn *et al.*, "Do as I can, not as I say: Grounding language in robotic affordances," in *Proc. CoRL*, 2022.

[29] S. Reed *et al.*, "A generalist agent," *Trans. Mach. Learn. Res.*, 2022.

[30] K. Bousmalis *et al.*, "RoboCat: A self-improving generalist agent for robotic manipulation," *arXiv:2306.11706*, 2023.

[31] E. Jang *et al.*, "BC-Z: Zero-shot task generalization with robotic imitation learning," in *Proc. CoRL*, 2021.

[32] M. Shridhar, L. Manuelli, and D. Fox, "Perceiver-actor: A multi-task transformer for robotic manipulation," in *Proc. CoRL*, 2022.

[33] Y. Jiang *et al.*, "VIMA: General robot manipulation with multimodal prompts," in *Proc. ICML*, 2023.

[34] D. Hafner, T. Lillicrap, M. Norouzi, and J. Ba, "Mastering Atari with discrete world models," in *Proc. ICLR*, 2021.

[35] N. Hansen *et al.*, "TD-MPC2: Scalable, robust world models for continuous control," in *Proc. ICLR*, 2024.

[36] J. Ho *et al.*, "Video diffusion models," in *Proc. NeurIPS*, 2022.

[37] T. Brooks *et al.*, "Video generation models as world simulators," OpenAI technical report, 2024.

[38] A. Szot *et al.*, "Habitat 2.0: Training home assistants to rearrange their habitat," in *Proc. NeurIPS*, 2021.

[39] S. Srivastava *et al.*, "BEHAVIOR: Benchmark for everyday household activities in virtual, interactive, and ecological environments," in *Proc. CoRL*, 2021.

[40] Z. Fu *et al.*, "Mobile ALOHA: Learning bimanual mobile manipulation with low-cost whole-body teleoperation," in *Proc. CoRL*, 2024.

[41] H. R. Walke *et al.*, "BridgeData V2: A dataset for robot learning at scale," in *Proc. CoRL*, 2023.

[42] S. Dasari *et al.*, "RoboNet: Large-scale multi-robot learning," in *Proc. CoRL*, 2019.

[43] S. James, Z. Ma, D. R. Arrojo, and A. J. Davison, "RLBench: The robot learning benchmark and learning environment," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 3019–3026, 2020.

[44] O. Mees, L. Hermann, E. Rosete-Beas, and W. Burgard, "CALVIN: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks," *IEEE Robot. Autom. Lett.*, vol. 7, no. 3, pp. 7327–7334, 2022.

[45] C. Lynch *et al.*, "Interactive language: Talking to robots in real time," *IEEE Robot. Autom. Lett.*, vol. 8, no. 12, pp. 7857–7864, 2023.

[46] J. Gu *et al.*, "ManiSkill2: A unified benchmark for generalizable manipulation skills," in *Proc. ICLR*, 2023.

[47] B. Liu *et al.*, "LIBERO: Benchmarking knowledge transfer for lifelong robot learning," in *Proc. NeurIPS*, 2023.

[48] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow matching for generative modeling," in *Proc. ICLR*, 2023.

[49] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, 2004.

[50] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proc. CVPR*, 2018.

[51] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs trained by a two time-scale update rule converge to a local Nash equilibrium," in *Proc. NeurIPS*, 2017.

[52] T. Unterthiner, S. van Steenkiste, K. Kurach, R. Marinier, M. Michalski, and S. Gelly, "Towards accurate generative models of video: A new metric and challenges," arXiv:1812.01717, 2018.

[53] N. Houlsby *et al.*, "Parameter-efficient transfer learning for NLP," in *Proc. ICML*, 2019.

[54] E. J. Hu *et al.*, "LoRA: Low-rank adaptation of large language models," in *Proc. ICLR*, 2022.

[55] Y. Ganin *et al.*, "Domain-adversarial training of neural networks," *J. Mach. Learn. Res.*, vol. 17, no. 59, pp. 1–35, 2016.

[56] I. Higgins *et al.*, "β-VAE: Learning basic visual concepts with a constrained variational framework," in *Proc. ICLR*, 2017.

[57] H. Kim and A. Mnih, "Disentangling by factorising," in *Proc. ICML*, 2018.