RETHINKING WORLD-ACTION MODEL FOR COMPOSITIONAL AND IN-CONTEXT ROBOTIC MANIPULATION

Anonymous authors
Paper under double-blind review

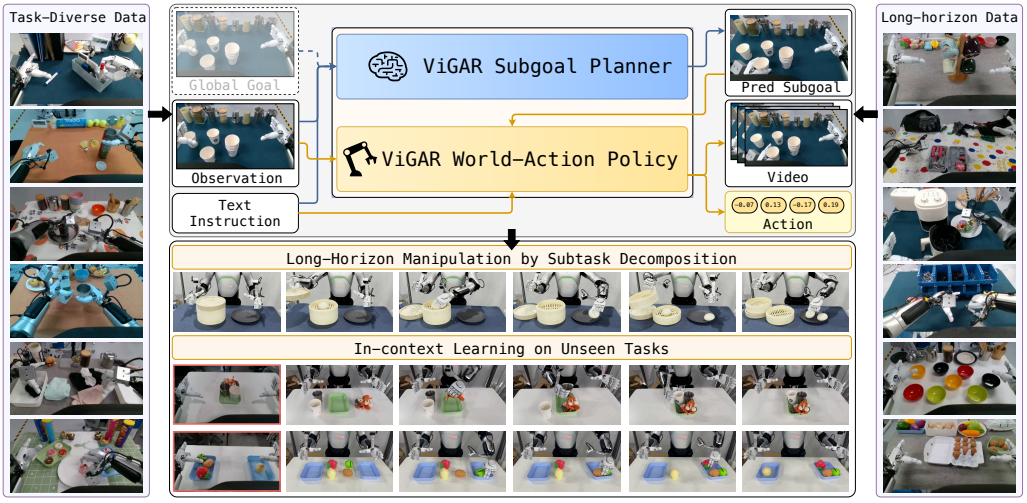


Figure 1: **Overview of ViGAR.** Our framework factorizes long-horizon compositional manipulation tasks into visual subgoal planning and subgoal execution, with both modules trained on task-diverse, long-horizon robot data. ViGAR achieves strong compositional manipulation performance in both simulated and real-world settings, and generalizes to unseen tasks through in-context steering.

ABSTRACT

Long-horizon compositional manipulation has become increasingly important for real-world robot deployment, where a single task involves multiple coordinated subtasks. Existing world-action models (WAMs) jointly predict short-horizon visual futures and actions, but typically lack explicit subtask-level reasoning. We propose **Visual Goal-conditioned Action Reasoning (ViGAR)**, a hierarchical framework that factorizes manipulation into a visual subgoal planner and a subgoal executor. Given the current observation and global instruction, the subgoal planner predicts a visual subgoal for the next subtask. The subgoal executor then jointly generates future visual trajectories and actions conditioned on the predicted subgoal. Both components share a pretrained world-model representation, enabling task-level planning and action generation to benefit from common physical knowledge. Moreover, our framework naturally supports in-context learning: using a global goal image as context can induce different subtask decompositions and behaviors without parameter updates. On the RoboTwin Clean2Random benchmark, ViGAR achieves 82.00% and 67.02% success rates under the *Clean* and *Random* settings, respectively, surpassing the strongest baseline by 12.81% in average success rate. Real-robot experiments on five compositional and two in-context learning tasks further confirm the effectiveness of ViGAR.

1 INTRODUCTION

Robots in production and service settings increasingly face compositional manipulation tasks that involve multiple coordinated subtasks. Such tasks require policies to maintain progress across subtask transitions while generalizing to unseen scenes, objects, and spatial layouts, rather than replaying

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.

a fixed sequence from training. Policies for real-world deployment therefore need both embodied generalization and mechanisms for organizing behavior over extended task horizons. World-Action Models (WAMs) offer an appealing paradigm for this goal. By jointly predicting future visual states and actions, WAMs learn physically grounded action dynamics from dense video supervision and have demonstrated strong generalization across tasks, environments, embodiments, and visual distribution shifts (Chen et al., 2025a; Li et al., 2026; Ye et al., 2026; Agarwal et al., 2026).

Despite these strengths, existing WAMs are not explicitly designed for long-horizon compositional manipulation. WAM formulations typically provide a useful local control signal by modeling what is likely to happen next, but do not explicitly organize behavior across a sequence of subtasks or expose a task-level interface through which the desired progression can be specified. Zhu et al. (2026) reveals that task progression requires more than supervision on short-term physical evolution, and Intelligence et al. (2025) further demonstrates the value of predicting semantic subtasks for open-world long-horizon manipulation and generalization. The core challenge is therefore to endow WAMs with explicit task-level decision making while preserving their strong world-centric prediction and physical generalization.

One line of work externalizes this decision making to a language planner that exploits VLMs to decompose a task into linguistic subtasks for a policy to execute. However, language subtask descriptions often underspecify precise spatial outcome without ambiguity, whereas visual subgoals impose stronger constraints on the desired scene configuration (Intelligence et al., 2026). Our key insight is that compositional manipulation can be factorized into *long-term future selection* and *short-term future realization*. World-action modelling for long-horizon tasks should first make explicit the target state the robot intends to realize, and then generate the actions that will bring it about.

To this end, we propose **Visual Goal-conditioned Action Reasoning (ViGAR)**, which treats subgoal images as explicit decision variable linking task-level reasoning to physical action generation. Given the current observation and the global task instruction, ViGAR first performs visual goal reasoning to predict the desired subtask goal image. Then, conditioned on current observation, global task instruction and predicted subgoal image, it jointly predicts future visual trajectories and corresponding robot actions leading to the subgoal. Both task-level decision and physical realization are learned in one shared generative world-model representation, so that the model can leverage the same physical knowledge for both stages.

Moreover, our framework naturally supports in-context learning: with the scene and instruction held fixed, changing only the specified global goal induces a different composition of behaviors, forcing the policy to reorganize its action sequence rather than default to the most likely training trajectory. We term this *in-context robotic manipulation*: the global goal image serves as context from which the model infers a subtask decomposition, so an out-of-distribution global goal elicits a task composition unseen in training, with no parameter update.

Experiments on both simulation benchmark and real-robot compositional tasks validate the effectiveness of ViGAR. On the RoboTwin Clean2Random benchmark (Chen et al., 2025b), ViGAR achieves success rates of 82.00% and 67.02% under the *Clean* and *Random* settings, respectively, surpassing the strongest baseline (Yang et al., 2026a) by 12.81 percentage points in average success rate. In real-robot experiments, ViGAR outperforms strong VLA and Cosmos3-Nano-Policy baselines on five compositional tasks, and exhibits in-context learning ability on unseen global goals, confirming the validity of our method. Our contributions are summarized as follows:

- We factorize compositional manipulation into *long-term future selection* and *short-term future realization*, and propose **ViGAR**, a hierarchical framework that predicts the next subgoal as an explicit decision variable and conditions joint video-action generation on it.
- We identify *in-context robotic manipulation* as a capability that emerges from explicit goal conditioning. With the scene and instruction held fixed, a new global goal image elicits a different subtask decomposition and a correspondingly reorganized action sequence, specifying out-of-distribution task compositions at deployment without any parameter update.
- ViGAR outperforms strong baselines on both simulation benchmark and real-robot compositional manipulation tasks, and exhibits strong in-context learning ability on unseen global goals, confirming the validity of our approach.

2 Related Works

VLAs for Compositional Robotic Manipulation. Vision-language-action (VLA) models map visual observations and language instructions to continuous action trajectories, and have demonstrated strong generalizability across objects, scenes, and tasks (Brohan et al., 2023; Black et al., 2024; Intelligence et al., 2025; Team, 2025; Zhai et al., 2026; Contributors, 2025; Fu et al., 2026; Qu et al., 2025). For compositional manipulation, however, a direct instruction-to-action mapping is often insufficient due to increased complexity, and recent systems introduce subtask information as an explicit planning interface, where a VLM or a pretrained world model synthesizes subtask information that conditions a VLA policy (Intelligence et al., 2025; Cai et al., 2026; Intelligence et al., 2026; Long et al., 2026).

World Models for Robotic Manipulation. World models learn the structure and evolution of physical environments by predicting future states, and world-action models (WAMs) extend this paradigm by jointly modeling future observations and actions. Recent studies have demonstrated that WAMs exhibit strong generalizability beyond the training domain, owing to their explicit modeling of geometric and temporal transitions (Li et al., 2026; Ye et al., 2026; Agarwal et al., 2026; Ma et al., 2026). However, WAMs typically focus on local action generation and lack a mechanism for reasoning about the plan a compositional task requires. Hu et al. (2026) and Liang et al. (2026) generate interleaved subtask text and subgoal images within a unified model, but these predictions either guide action generation implicitly through intermediate features or serve primarily as planning outputs, rather than acting as explicit subgoal conditions for the downstream policy.

In-Context Learning for Robotic Manipulation. In large language models, a novel task can be performed simply by specifying it in the context without any parameter update, termed *in-context learning (ICL)* (Brown et al., 2020; Wei, 2022; Wei et al., 2022). Carried into robotic manipulation, this moves the acquisition of a new task out of training-time imitation and into inference time, bypassing the need for costly data collection and model fine-tuning. Zhou et al. (2026) performs ICL by prompting a causal video-action model with human demonstration videos; Chen et al. (2026) acquires a new skill from a single human video by predicting the robot's own future observations along the demonstrated progression; SkildS1 (AI, 2026) and GEN-1.5 (Team, 2026) likewise condition on a demonstration video or a short sensorimotor trajectory. In each case the context conveys *how* a task is performed, and the policy re-executes that procedure under its own embodiment. We instead specify a task by its terminal world state: a global goal image constrains only *what* the scene should become, leaving the corresponding subtask decomposition to the model.

3 Preliminaries

3.1 VLA and WAM for Robotic Manipulation

Consider a robot task specified by a language instruction ℓ. At inference step t, a VLA policy π_θ^{VLA} with parameters θ predicts an H_a-step action chunk $\mathbf{A}_{t+1} := \mathbf{a}_{t+1:t+H_a} \in \mathbb{R}^{H_a \times d_a}$ conditioned on current multi-view observation $\mathbf{o}_t = \{\mathbf{o}_t^v\}_{v=1}^V$, proprioceptive state $\mathbf{s}_t \in \mathbb{R}^{d_s}$, and textual metadata μ specifying the embodiment, camera resolution, and control mode:

$$\mathbf{A}_{t+1} \sim \pi_\theta^{\text{VLA}}(\mathbf{A}_{t+1}|\mathbf{o}_t, \mathbf{s}_t, \ell, \mu),$$

where H_a denotes the action chunk size. A WAM policy π_ψ^{WAM} with parameters ψ is similarly conditioned on $\mathbf{o}_t, \mathbf{s}_t, \ell, \mu$, but instead parameterizes a joint conditional distribution over a short visual future $\mathbf{O}_{t+1} := \mathbf{o}_{t+1:t+H_v}$ and an action chunk $\mathbf{A}_{t+1}$:

$$(\mathbf{O}_{t+1}, \mathbf{A}_{t+1}) \sim \pi_\psi^{\text{WAM}}(\mathbf{O}_{t+1}, \mathbf{A}_{t+1}|\mathbf{o}_t, \mathbf{s}_t, \ell, \mu).$$

3.2 Problem Formulation for Compositional Robot Tasks

Let $T \in \mathcal{T}$ denote a compositional robot task specified by a global instruction $\ell \in \mathcal{L}$, where $\mathcal{T}$ and $\mathcal{L}$ denote the space of language-conditioned robot tasks and the space of free-form text instructions, respectively. To address the challenge of compositional manipulation, instead of modeling a policy $\pi(\cdot|\mathbf{o}_t, \mathbf{s}_t, \ell, \mu)$ that generates action based on the global instruction ℓ, we decompose T into n intermediate subtasks $T_i \in \mathcal{T}$, such that $T = (T_1, \cdots, T_n)$. For each subtask T_i at stage i, we specify a subtask instruction ℓ_i describing the subtask goal in text, and a subgoal image $\mathbf{g}_i \in \mathcal{I}$ specifying the visual target state the robot should reach, where $\mathcal{I}$ is the space of images. The objective is then to

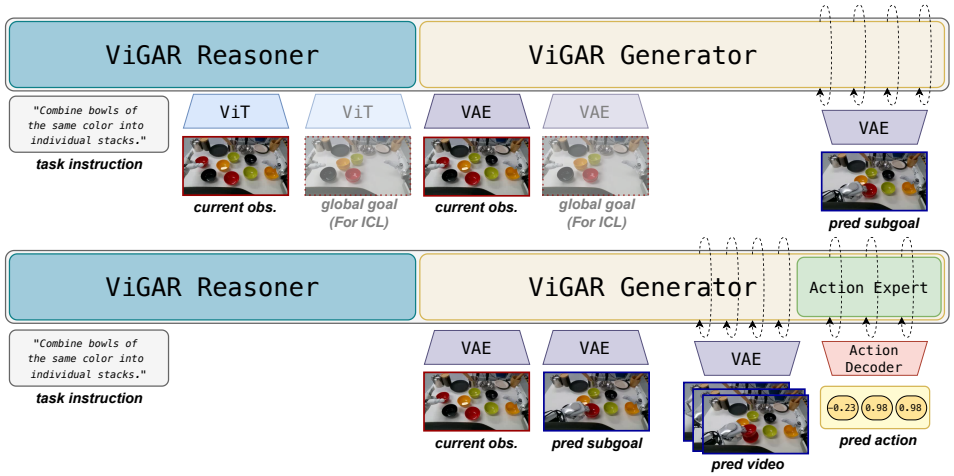


Figure 2: **Model architecture of ViGAR:** (1) the subgoal planner $\mathcal{G}_\theta$ generates the next subgoal image from the task instruction and current observation, optionally conditioned on a global goal image for in-context learning (ICL); (2) the policy π_ϕ^{WAM} takes in the predicted subgoal and jointly predicts the future frames and the action chunk.

train a robot policy π that leverages the subtask information $(\ell_i, \mathbf{g}_i)$ to predict the action trajectory for subtask T_i:

$$\mathbf{A}_{t+1}^{T_i} \sim \pi(\cdot|\mathbf{o}_t, \mathbf{s}_t, \ell_i, \mathbf{g}_i, \mu),\ i = 1, \cdots, n,$$

where t denotes the current timestep. The robot executes $\mathbf{A}_{t+1}^{T_i}$ until the current subtask is completed, at which point the subtask instruction and subgoal image are advanced to ℓ_{i+1} and $\mathbf{g}_{i+1}$, respectively. In the following part, we use 'subgoal' to term $(\ell_i, \mathbf{g}_i),\ i = 1, \cdots, n$.

3.3 In-Context Task Specification for Robotic Manipulation

In-context learning (ICL) performs a task specified at deployment rather than acquired during training. In addition to language instruction ℓ, let $\mathcal{C}$ denote an optional context supplied at inference time. A policy π with parameters ϕ performs in-context manipulation if a new task is realized by changing ℓ or $\mathcal{C}$ alone without the update of ϕ:

$$\mathbf{A}_{t+1} \sim \pi_\phi(\cdot|\mathbf{o}_t, \mathbf{s}_t, \ell, \mathcal{C}, \mu). \tag{1}$$

Recent work (Zhou et al., 2026; Chen et al., 2026; AI, 2026; Team, 2026) on robotic in-context learning often instantiates $\mathcal{C}$ as a demonstration video, such as a human operation video $\mathbf{V} := \mathbf{v}_{1:K}$, which prescribes the procedure the robot should reproduce.

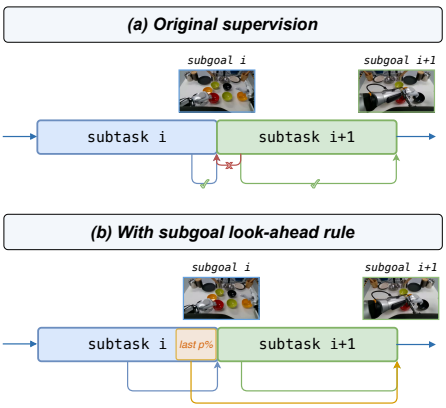
4 Method

In this section, we describe the two essential parts of ViGAR: a subgoal planner and a subgoal-guided world-action model. We divide each compositional manipulation task into multiple subtasks, where each subtask contains an atom action and a subgoal $(\ell_i, \mathbf{g}_i)$. Our subgoal planner generates a subgoal for each subtask, as illustrated in section 4.1. Then, our world-action policy predicts the action chunk from the current observation, proprioceptive state and generated subgoal, as detailed in section 4.2. Finally, we show in section 4.3 that by specifying a global goal as an inference-time context, the subgoal planner can generate a subtask decomposition consistent with the global goal, allowing a new task composition to be specified at deployment without any parameter updates.

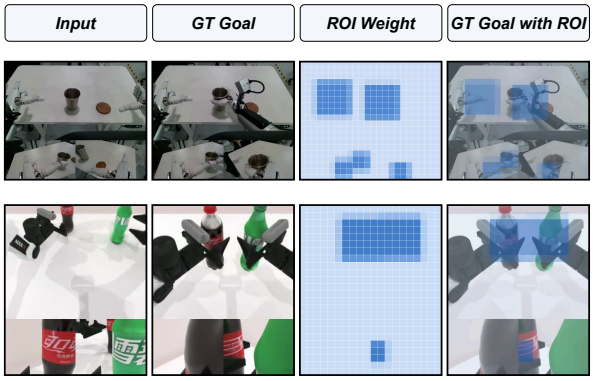
Both the subgoal planner and the downstream policy are initiated from the same pretrained world model (Agarwal et al., 2026). This backbone follows a dual-branch design, with a *reasoner* branch processing language and semantic visual tokens, and a *generator* branch generating image or video via flow matching. The two branches interact through a shared multimodal attention layer.

4.1 Subgoal Planner for Task-Level Reasoning

In subgoal planning, we fix $\ell_i = \ell$ for all i, rather than decomposing the global instruction into stage-specific subtask text, as the subgoal image $\mathbf{g}_i$ already provides precise grounding of the target



(a) **Subgoal look-ahead rule** (eq. (2)): redirecting the last $p\%$ of T_i to $\mathbf{g}_{i+1}$ makes the supervision consistent across the boundary.



(b) **End-effector ROI weighting** (eq. (3)): the planning loss is weighted toward the end-effector region-of-interest (ROI), so that supervision concentrates where manipulation actually happens rather than on static background.

Figure 3: Implementation design in subgoal supervision: (a) temporal boundary handling via look-ahead redirection, and (b) spatial weighting via end-effector region-of-interest masks.

state, and retaining ℓ serves as a consistent semantic anchor for the overall task. This also alleviates potential error accumulation that subtask text generation could introduce to the downstream policy.

Guided by this design, we cast subgoal planning as an image-editing problem: given a reference frame $\mathbf{o}_t$ within subtask T_i and the global instruction ℓ, the subgoal planner $\mathcal{G}_\theta$ predicts a subgoal image $\mathbf{g}_i$ useful for executing T_i and continuing the global task T.

Notably, as shown in fig. 3a, frame $\mathbf{o}_t$ at the end of T_i can be visually similar to early frames of T_{i+1}, while the two are assigned different targets, $\mathbf{g}_i$ and $\mathbf{g}_{i+1}$. Without correction, the planner tends to confuse them at inference time, regenerating $\mathbf{g}_i$ once the robot has entered T_{i+1} and driving the plan back to a subtask already completed. To keep supervision consistent for visually similar inputs, we redirect such $\mathbf{o}_t$ to target $\mathbf{g}_{i+1}$ by the following *subgoal look-ahead rule*:

$$\text{target}(\mathbf{o}_t) = \begin{cases} \mathbf{g}_{i+1}, & \mathbf{o}_t \in \text{last } p\% \text{ of } T_i,\ i < n \\ \mathbf{g}_i, & \text{otherwise.} \end{cases} \tag{2}$$

For notation simplicity, we still denote $\mathbf{g}_i$ as the target assigned to $\mathbf{o}_t$ after this subgoal look-ahead rule. Training pairs $(\mathbf{o}_t, \ell, \mathbf{g}_i)$ are constructed from subtask-segmented robot demonstrations, with each subtask corresponding to a demonstration segment ending at its target frame.

Formally, we train $\mathcal{G}_\theta$ with a flow-matching objective. Let $\mathbf{x}_0^g = \mathcal{E}_v(\mathbf{g}_i)$ denote the target latent and $\mathbf{z}_t = \mathcal{E}_v(\mathbf{o}_t)$ denote the reference latent. We construct a noisy latent $\mathbf{x}_\tau^g = (1-\tau)\mathbf{x}_0^g + \tau\boldsymbol{\epsilon}$, where $\tau \sim \mathcal{U}[0,1], \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$, and define the flow-matching target $\mathbf{v}_{\text{plan}} = \boldsymbol{\epsilon} - \mathbf{x}_0^g$.

Because manipulation-related changes are concentrated near the end effector, we construct an end-effector-centered mask to emphasize these regions during training. As shown in fig. 3b, we project the corresponding expert end-effector positions through the calibrated cameras and take the union of fixed-size windows around visible projections to form a binary mask $\mathbf{m}_i$, and further area-pooling it to $\tilde{\mathbf{m}}_i$ under latent resolution. For each target latent token u, we define its weight as

$$w_i(u) = \frac{1 + \lambda \tilde{m}_i(u)}{\frac{1}{|\Omega|}\sum_{u' \in \Omega}(1 + \lambda \tilde{m}_i(u'))}, \qquad \frac{1}{|\Omega|}\sum_{u \in \Omega} w_i(u) = 1. \tag{3}$$

The subgoal planner is then optimized with the following weighted flow-matching loss:

$$\mathcal{L}_{\text{Plan}} = \mathbb{E}_{(\mathbf{o}_t, \ell, \mathbf{g}_i), \tau, \boldsymbol{\epsilon}} \left[\left\| \sqrt{\mathbf{w}_i} \odot \left(\mathcal{G}_\theta\left(\mathbf{x}_\tau^g, \tau \mid \mathbf{z}_t, \ell\right) - \mathbf{v}_{\text{plan}} \right) \right\|_2^2 \right],$$

where $\mathbf{w}_i$ collects the weights $w_i(u)$. The predicted subgoal $(\ell, \hat{\mathbf{g}}_i)$ serves as a task-level reasoning proxy for the downstream action policy to complete T_i.

4.2 Subgoal-guided World-Action Model

The policy π_ϕ^{WAM} predicts a short visual future $\mathbf{O}_{t+1}$ and an action chunk $\mathbf{A}_{t+1}$ conditioned on the current observation $\mathbf{o}_t$, proprioceptive state $\mathbf{s}_t$, and the predicted subgoal $(\ell, \hat{\mathbf{g}}_\mathbf{i})$ of the active subtask

T_i. It is trained with a joint flow-matching objective over the visual future and the action chunk. Let $\mathbf{x}_0^v = \mathcal{E}_v(\mathbf{O}_{t+1})$ and $\mathbf{x}_0^a = \mathbf{A}_{t+1}$ denote the visual and action targets, and $\mathbf{z}_t = \mathcal{E}_v(\mathbf{o}_t)$ denote the reference latent. We construct noisy targets $\mathbf{x}_\tau^v = (1-\tau)\mathbf{x}_0^v + \tau\boldsymbol{\epsilon}^v$ and $\mathbf{x}_\tau^a = (1-\tau)\mathbf{x}_0^a + \tau\boldsymbol{\epsilon}^a$, with $\boldsymbol{\epsilon}^v, \boldsymbol{\epsilon}^a \sim \mathcal{N}(0, \mathbf{I})$, and define the flow-matching targets $\mathbf{v}_{\text{vid}} = \boldsymbol{\epsilon}^v - \mathbf{x}_0^v$ and $\mathbf{v}_{\text{act}} = \boldsymbol{\epsilon}^a - \mathbf{x}_0^a$. Conditioned on $\mathbf{z}_t, \mathbf{s}_t, \ell, \hat{\mathbf{g}}_i, \mu$, the policy π_ϕ^{WAM} jointly predicts $(\hat{\mathbf{v}}_\phi^v, \hat{\mathbf{v}}_\phi^a) = \pi_\phi^{\text{WAM}}(\mathbf{x}_\tau^v, \mathbf{x}_\tau^a, \tau \mid \mathbf{z}_t, \mathbf{s}_t, \ell, \hat{\mathbf{g}}_i, \mu)$ and is optimized by minimizing:

$$\mathcal{L}_{\text{WAM}} = \mathbb{E}_{(\mathbf{o}_t, \ell, \hat{\mathbf{g}}_i), \tau, \boldsymbol{\epsilon}^v, \boldsymbol{\epsilon}^a} \left[\|\hat{\mathbf{v}}_\phi^v - \mathbf{v}_{\text{vid}}\|_2^2 + \|\hat{\mathbf{v}}_\phi^a - \mathbf{v}_{\text{act}}\|_2^2 \right].$$

Notably, the predicted subgoal image $\hat{\mathbf{g}}_i$ can be injected into π_ϕ^{WAM} via a *semantic route*, where $\hat{\mathbf{g}}_i$ is tokenized by the vision encoder and consumed by the reasoner branch, or a *geometric route*, where $\hat{\mathbf{g}}_i$ is VAE-encoded into a clean latent and fed directly to the generator branch alongside the reference latent $\mathbf{z}_t$. We adopt the *geometric route* as it grounds the subgoal at the pixel level rather than through a coarser semantic embedding (see section 5.5 for elaboration). $\hat{\mathbf{g}}_i$ is therefore encoded into $\mathbf{z}_i^g = \mathcal{E}_v(\hat{\mathbf{g}}_i)$ with the same VAE encoder $\mathcal{E}_v$ used for $\mathbf{o}_t$, while the instruction ℓ continues to condition the policy through the reasoner branch. We also add $\mathbf{z}_i^g$ with a zero-initialized learnable embedding to distinguish it from other latent tokens in the generator branch.

4.3 In-Context Task Specification via a Global Goal

We instantiate the inference-time context $\mathcal{C}$ in eq. (1) as a global goal image $\mathbf{G} \in \mathcal{I}$, the terminal world state that the compositional task T should produce. As shown in fig. 2, the context enters the subgoal planner alone, which predicts $\hat{\mathbf{g}}_i \sim \mathcal{G}_\theta(\cdot \mid \mathbf{z}_t, \ell, \mathbf{G})$ at subtask T_i , where $\mathbf{z}_t = \mathcal{E}_v(\mathbf{o}_t)$ denote the reference latent. During training, $\mathbf{G}$ is the terminal frame of each demonstration trajectory, and the planning objective for $\mathcal{G}_\theta$ can be written as:

$$\mathcal{L}_{\text{Plan-ICL}} = \mathbb{E}_{(\mathbf{o}_t, \ell, \mathbf{g}_i, \mathbf{G}), \tau, \boldsymbol{\epsilon}} \left[\left\| \sqrt{\mathbf{w}_i} \odot \left(\mathcal{G}_\theta \left(\mathbf{x}_\tau^g, \tau \mid \mathbf{z}_t, \ell, \mathbf{G} \right) - \mathbf{v}_{\text{plan}} \right) \right\|_2^2 \right],$$

where $\mathbf{w}_i$ is the end-effector ROI weight defined in eq. (3). At inference, $\mathbf{G}$ is replaced by the user-specified global goal image. The downstream policy $\pi_\phi^{\text{WAM}}(\cdot|\mathbf{z}_t, \mathbf{s}_t, \ell, \hat{\mathbf{g}}_i, \mu)$ remains unchanged as it conditions only on $\hat{\mathbf{g}}_i$, through which the effect of $\mathbf{G}$ is already mediated. Therefore, under our framework, a global goal image unseen during training can induce a task decomposition $(T_1, \cdots, T_n)$ that was never demonstrated, with θ and ϕ left unchanged.

5 Experiments

5.1 Training configuration

Implementation. The subgoal planner $\mathcal{G}_\theta$ and subgoal-guided world-action model π_ϕ^{WAM} of ViGAR are initialized from the pretrained Cosmos3-Nano checkpoint (Agarwal et al., 2026), a 16B-A8B-parameter omnimodal world model that adopts dual-branch mixture-of-transformers architecture. The subgoal planner $\mathcal{G}_\theta$ is adapted from the pretrained backbone by transforming its image-to-video generation capability into image editing. The policy π_ϕ^{WAM} is post-trained following the standard recipe for adapting world models into robot policies (Ye et al., 2026; Agarwal et al., 2026). More implementation details are listed in section B.

5.2 Results on Simulation Benchmark

Simulation Benchmark. We evaluate the compositional manipulation ability of ViGAR on the RoboTwin Clean2Random benchmark (Chen et al., 2025b) using the Aloha AgileX dual-arm embodiment. We adopt a multi-task training setup where the policy is trained exclusively on 50 clean expert demonstrations per task (Yuan et al., 2026a), 2500 demonstrations in total. All models are evaluated on 100 rollouts per task under both Clean and Random conditions.

Baselines. For visual subgoal prediction quality, we benchmark against VISTA (Long et al., 2026) and RxBrain (Liang et al., 2026). For policy performance, we compare ViGAR against strong baselines spanning two categories. For VLA baselines, we evaluate against StarVLA (Community, 2026), Abot-M0 (Yang et al., 2026b), X-VLA (Zheng et al., 2026), and $\pi_{0.5}$ (Intelligence et al., 2025). For WAM baselines, we evaluate against FastWAM (Yuan et al., 2026b), LingbotVA (Li et al., 2026), and 4D-WAM (Yang et al., 2026a). We further include a controlled baseline, Cosmos3-Nano-RoboTwin, obtained by post-training Cosmos3-Nano on the same 2500 clean RoboTwin trajectories with the same recipe as π_ϕ^{WAM} but without subgoal guidance, thereby isolating the contribution of subgoal guidance from that of the pretrained backbone.

Table 1: Simulation manipulation performance and visual subgoal prediction quality of ViGAR. **Bold**: best, Underline: second best.

(a) Success rate (%) on the RoboTwin Clean2Random benchmark. Methods are grouped by policy paradigm.

Type	Model	Clean	Random	Avg.
VLA	StarVLA	46.50	3.20	24.90
	Abot-M0	57.40	30.40	43.90
	X-VLA	68.00	20.90	44.50
	$\pi_{0.5}$	70.70	46.00	58.40
WAM	FastWAM	77.80	1.90	39.90
	LingbotVA	80.70	34.60	57.70
	4D-WAM	81.50	41.80	61.70
	Cosmos3-Nano -RoboTwin	77.06	22.46	49.76
	ViGAR	**82.00**	**67.02**	**74.51**

(b) Performance of visual subgoal prediction on the RoboTwin Clean2Random and real-world robot benchmark.

Model	LPIPS (↓)	DINO-cos (↑)	ΔIoU (↑)	RetAcc (↑)
RoboTwin				
VISTA	0.38	0.54	0.22	0.82
RxBrain	0.73	0.19	0.18	0.79
ViGAR $\mathcal{G}_\theta$	**0.13**	**0.80**	**0.65**	**0.94**
Real Robot				
VISTA	0.38	0.61	0.24	0.31
RxBrain	0.60	0.29	0.18	0.22
ViGAR $\mathcal{G}_\theta$	**0.11**	**0.81**	**0.66**	**0.76**

Visual Subgoal Prediction Quality. Before evaluating manipulation success rate, we assess the quality of the visual subgoals generated by $\mathcal{G}_\theta$ on RoboTwin. We compare the generated subgoals with their ground-truth targets using LPIPS, DINO cosine similarity (DINO-cos), Change of IoU (ΔIoU), and retrieval correctness (RetAcc). Detailed definitions of these metrics are provided in section C.2. As shown in table 1b, ViGAR achieves the best performance across all metrics on both the RoboTwin benchmark and real-robot experiments. These results indicate that ViGAR generates visually faithful, semantically aligned, and subtask-correct intermediate targets.

Policy Performance. As shown in table 1a, ViGAR achieves the highest success rates among the compared methods under both the *Clean* (82.00%) and *Random* (67.02%) settings. The performance boost mostly comes from the *Random* setting, whereas *Clean* performance is largely saturated across WAMs. The benefit of explicit subgoal selection therefore materializes precisely when the scene layout can no longer be memorized from demonstrations.

Notably, Cosmos3-Nano-RoboTwin, which follows the same policy adaptation recipe as ViGAR but omits explicit subgoal guidance, reaches only 22.46% under *Random* setting. The performance gap suggests that ViGAR benefits from predicting visual subgoals and conditioning action generation on them, validating the effectiveness of our methods.

5.3 Compositional manipulation in Real-world Robot Deployment

Real-robot Experiment Setup. We conduct real-robot experiments on an AGIBOT A2 robot, using its two 7-DoF arms, two wrist cameras, and one front-facing camera. For this setting, ViGAR is first mid-trained on ∼900 hours of teleoperation data collected from the same embodiment, and then post-trained on task-specific real-robot demonstrations. We evaluate ViGAR on five long-horizon compositional manipulation tasks. Each task consists of multiple subtasks that must be completed sequentially. The tasks are as follows:

- *Pile Paper Cups*: pick up paper cups and stack them into one pile. This task contains three subtasks.
- *Stack Colored Bowls*: organize bowls of multiple colors into separate stacks, with bowls of the same color grouped together. This task contains three subtasks.
- *Stamp over Letter*: stamp a letter and hand it to a person. This task contains four subtasks.
- *Tidy-up Desktop*: collect the scattered trash and objects from the desktop and place them in a trash bin. This task contains four subtasks.
- *Take-out Steam Buns*: retrieve one steam bun from each tier of a two-tier bamboo steamer and place the buns on a plate. This task contains five subtasks.

A visualization of the task processes is provided in fig. 6. Each task is evaluated on 20 rollouts, and we report average task progress for each task to evaluate models' performances.

Baselines. We compare ViGAR with two baselines: the VLA policy $\pi_{0.5}$ and Cosmos3-Nano-Policy. Cosmos3-Nano-Policy is trained using the same real-robot data and policy adaptation recipe

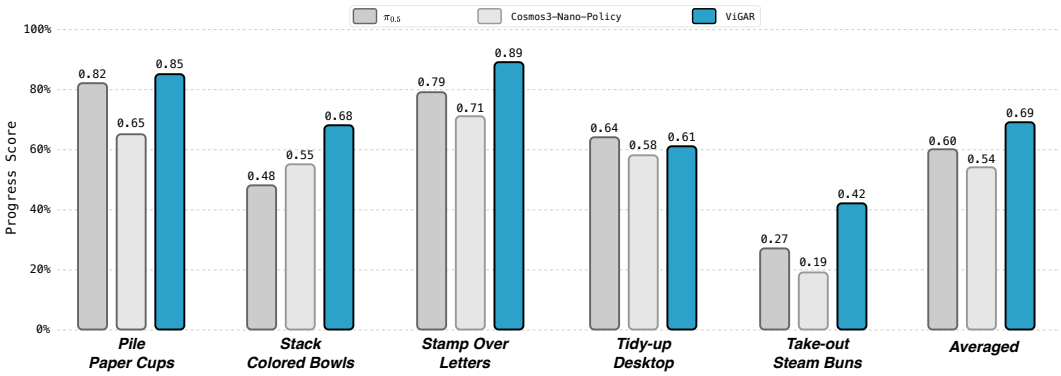


Figure 4: **Real-world deployment results.** We evaluate ViGAR on five long-horizon compositional manipulation tasks. Our method outperforms all baselines on the averaged task progress score.

as ViGAR, but without explicit subgoal guidance. This comparison allows us to examine the contribution of visual subgoal planning under the same real-robot training setting.

Visual Subgoal Prediction Quality. We similarly assess the visual subgoals generated from real-robot observations using the same metrics and baselines as in section 5.2. As shown in table 1b, ViGAR achieves the best performance across all metrics, consistent with its results on RoboTwin.

Policy Performance. As shown in table 4, ViGAR achieves the best overall performance across the five real-robot tasks, outperforming both baseline policies in task progress score. The results suggest that explicit visual subgoals provide useful task-level guidance for executing long-horizon manipulation sequences in the physical world. The comparison with Cosmos3-Nano-Policy further indicates that the improvement is associated with subgoal-guided action generation, beyond the effect of the pretrained world-model backbone.

5.4 Steering Real-world Robot Manipulation via In-Context Learning

To examine whether a global goal image can steer ViGAR toward task compositions observed or not observed during training, we have designed two tasks to test this capability:

- *Fruit Arrangement*: the robot is required to place a set of fruits into two trays, with the type and the number of fruits in each tray specified by a global goal image.
- *Desktop item storage:* the robot is required to place a set of desktop items into one storage box, with the type of items in the box specified by a global goal image.

The training demonstrations cover only a restricted set of placement patterns, while the test-time global goal image may provide an unseen pattern. We hold fixed the initial scene and language instruction, and provide a global goal image to specify the desired task composition. This visual context is provided to the subgoal planner $\mathcal{G}_\theta$, whose predictions guide the downstream policy π_ϕ^{WAM}.

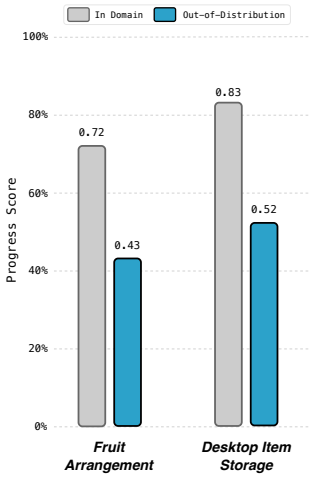
As shown in figs. 5a and 5b, ViGAR executes the placements specified by the global goal, composing tasks unseen in training with 77.5% (in-domain) and 47.5% (OOD) task progress score. This shows that visual context can guide subtask selection and composition, validating that learned manipulation skills can be composed into a new task through an inference-time specification of the desired terminal state.

5.5 Ablation Studies

Effectiveness of Implementation Design. We investigate the effectiveness of leveraging subgoal look-ahead (eq. (2)) rule in training $\mathcal{G}_\theta$ and π_ϕ^{WAM}, and end-effector ROI-weighted flow matching loss (eq. (3)) in training $\mathcal{G}_\theta$. As shown in table 2b, removing either component degrades performance under both *Clean* and *Random* settings, validating the effectiveness of our implementation design. Specifically, end-effector ROI weighting contributes most to the performance gain, as it concentrates supervision more on manipulation-related region the static background. Subgoal look-ahead also brings improvement by preventing regressing back to subtasks that have already been completed.



(a) **Qualitative results.** Successful executions of *Fruit Arrangement* and *Desktop Item Storage* performed by ViGAR. Left column (blue border): global goal image; right columns(red border): task progress.



(b) **Task progress scores** of ViGAR under in-domain and out-of-distribution global goals, each evaluated over 20 rollouts.

Figure 5: Results of two in-context learning tasks in real-world robot deployment: (a) qualitative execution sequences and (b) task progress scores on in-domain and out-of-distribution settings.

Table 2: Ablations studies on the implementaton of ViGAR. Success rates (%) are aggregated over 50 RoboTwin tasks; **Bold**: best.

(a) Ablation on the choice of goal injection routes using ground-truth goal images.

Goal injection	Clean	Random	Avg.
None	76.58	21.78	49.18
Reasoner	78.42	23.52	50.97
Generator	78.92	**45.42**	**62.17**
Generator + Reasoner	**82.16**	41.28	61.72

(b) Ablation on the effectiveness of subgoal look-ahead rule and ROI-reweighted training.

Subgoal supervision	Clean	Random	Avg.
ViGAR	**82.00**	**67.02**	**74.51**
w/o subgoal look-ahead	79.80	65.92	72.86
w/o EEF ROI weighting	79.28	65.64	72.46
w/o both	72.68	52.40	62.54

Goal Injection Route. To isolate the effect of the injection route, we use ground-truth goal images in this controlled ablation rather than predicted ones. Alongside a baseline without goal guidance, we compare three injection schemes: vision-encoder tokens provided to the reasoner (*semantic route*), VAE latents provided to the generator (*geometric route*), and both routes combined. For a fair comparison, all variants are trained for 30k steps on the same 2500 RoboTwin *Clean* trajectories.

As shown in table 2a, generator-only injection achieves the highest average success rate. Adding the reasoner route improves performance on *Clean* but reduces robustness under *Random* conditions. These results motivate the geometric route as the goal injection mechanism used in ViGAR.

6 Conclusion

In this work, we addressed the gap between short-horizon world-action prediction and task-level planning for compositional manipulation. We introduced ViGAR, which factorizes manipulation into visual subgoal planning and subgoal-conditioned joint video-action generation. By representing intermediate targets as images, ViGAR connects task-level decisions with the physical dynamics modeled by world-action models. Experiments on RoboTwin and five real-robot tasks support the effectiveness of this design. Furthermore, a global goal image provides an interface for inference-time steering, enabling ViGAR to compose learned manipulation behaviors into tasks absent from the training demonstrations without parameter updates. These findings suggest that explicit visual goals offer a useful connection between planning and action generation, providing a promising direction for compositional and steerable robotic manipulation.

AI USE STATEMENT

In this work, generative AI tools were used to assist with polishing of the manuscript, synthetic dataset generation, and method implementation. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Niket Agarwal, Arslan Ali, Jon Allen, Martin Antolini, Adeline Aubame, Alisson Azzolini, Junjie Bai, Maciej Bala, Yogesh Balaji, Josh Bapst, et al. Cosmos 3: Omnimodal world models for physical ai. *arXiv preprint arXiv:2606.02800*, 2026.

Skild AI. Introducing s1: In-context learning for robotics. August 2026. URL `https://skild.ai/blogs/s1`.

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.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Qingwen Bu, Jisong Cai, Li Chen, Xiuqi Cui, Yan Ding, Siyuan Feng, Shenyuan Gao, Xindong He, Xuan Hu, Xu Huang, et al. Agibot world colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. *arXiv preprint arXiv:2503.06669*, 2025.

Xiaowei Cai, Yunuo Cai, Bingao Chen, Jingxiao Chen, Zhi Chen, Siyuan Feng, Tengyu Hou, Jingshun Huang, Han Jiang, Runkun Ju, et al. τ_0-vla: a hierarchical robot foundation model with world-model-guided test-time computation. *arXiv preprint arXiv:2608.16885*, 2026.

Boyuan Chen, Tianyuan Zhang, Haoran Geng, Caiyi Zhang, Peihao Li, Kiwhan Song, William T Freeman, Jitendra Malik, Pieter Abbeel, Russ Tedrake, et al. Large video planner enables generalizable robot control. *arXiv preprint arXiv:2512.15840*, 2025a.

Guangyan Chen, Meiling Wang, Te Cui, Zichen Zhou, Qi Shao, Shalfun Li, Hang Su, Roy Gan, Hao Wang, Mengyin Fu, et al. Robots acquire manipulation skills in seconds from a single human video. *arXiv preprint arXiv:2607.20033*, 2026.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, et al. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025b.

StarVLA Community. Starvla: A lego-like codebase for vision-language-action model developing. *arXiv preprint arXiv:2604.05014*, 2026.

InternVLA-M1 Contributors. Internvla-m1: A spatially guided vision-language-action framework for generalist robot policy. *arXiv preprint arXiv:2510.13778*, 2025.

Yiyang Fu, Chubin Zhang, Shukai Gong, Yufan Deng, Kaiwei Sun, Qiyang Min, Qibin Hou, Yansong Tang, Jianan Wang, and Daquan Zhou. Stablevla: Towards robust vision-language-action models without extra data. *arXiv preprint arXiv:2605.18287*, 2026.

Yucheng Hu, Jianke Zhang, Yuanfei Luo, Yanjiang Guo, Xiaoyu Chen, Xinshu Sun, Kun Feng, Qingzhou Lu, Sheng Chen, Yangang Zhang, et al. Bagelvla: Enhancing long-horizon manipulation via interleaved vision-language-action generation. *arXiv preprint arXiv:2602.09849*, 2026.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Physical Intelligence, Bo Ai, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Greg Balke, Kevin Black, George Bokinsky, Shihao Cao, Thomas Charbonnier, et al. $\pi_{0.7}$: a steerable generalist robotic foundation model with emergent capabilities. *arXiv preprint arXiv:2604.15483*, 2026.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

Lin Li, Qihang Zhang, Yiming Luo, Shuai Yang, Ruilin Wang, Fei Han, Mingrui Yu, Zelin Gao, Nan Xue, Xing Zhu, et al. Causal world modeling for robot control. *arXiv preprint arXiv:2601.21998*, 2026.

Haotian Liang, Mingkang Chen, Yufei Huang, Yuchun Guo, Xiaomeng Zhu, Xiangli Shi, Kaixuan Wang, Yunxuan Mao, Weijie Zhou, Ling Chen, et al. Rxbrain: Embodied cognition foundation model with joint language-visual reasoning and imagination. *arXiv preprint arXiv:2607.14187*, 2026.

Feng Liu, Shiwei Zhang, Xiaofeng Wang, Yujie Wei, Haonan Qiu, Yuzhong Zhao, Yingya Zhang, Qixiang Ye, and Fang Wan. Timestep embedding tells: It's time to cache for video diffusion model, 2025. URL https://arxiv.org/abs/2411.19108.

Qian Long, Yueze Wang, Jiaxi Song, Junbo Zhang, Peiyan Li, Wenxuan Wang, Yuqi Wang, Haoyang Li, Shaoxuan Xie, Guocai Yao, et al. Scaling world model for hierarchical manipulation policies. *arXiv preprint arXiv:2602.10983*, 2026.

Juncheng Ma, Jianxin Bi, Yufan Deng, Xuanran Zhai, Kewei Zhang, Ye Huang, Bo Liang, Shukai Gong, Jiankai Tu, Xiaotian Tang, Jiaxin Li, Kaiqi Chen, Duomin Wang, Yuqi Wang, Bingyi Kang, Eric Huang, Zhiyang Dou, Zhen Dong, Enze Xie, Wojciech Matusik, Tat-Seng Chua, and Daquan Zhou. Humanscale: Egocentric human video can outperform real-robot data for embodied pretraining, 2026. URL https://arxiv.org/abs/2606.20521.

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903. IEEE, 2024.

Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Yan Ding, Zhigang Wang, JiaYuan Gu, Bin Zhao, Dong Wang, et al. Spatialvla: Exploring spatial representations for visual-language-action model. *arXiv preprint arXiv:2501.15830*, 2025.

Generalist Team. Gen-1.5: Embodied foundation models are one-shot learners. *Generalist AI Blog*, 2026. https://generalistai.com/blog/gen-1.5.

RDT Team. Rdt2: Enabling zero-shot cross-embodiment generalization by scaling up umi data, September 2025. URL https://github.com/thu-ml/RDT2.

Yang Tian, Yuyin Yang, Yiman Xie, Zetao Cai, Xu Shi, Ning Gao, Hangxu Liu, Xuekun Jiang, Zherui Qiu, Feng Yuan, et al. Interndata-a1: Pioneering high-fidelity synthetic data for pre-training generalist policy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 976–985, 2026.

J Wei. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Shihan Wu, Xuecheng Liu, Shaoxuan Xie, Pengwei Wang, Xinghang Li, Bowen Yang, Zhe Li, Kai Zhu, Hongyu Wu, Yiheng Liu, et al. Robocoin: An open-sourced bimanual robotic data collection for integrated manipulation. *arXiv preprint arXiv:2511.17441*, 2025.

Lishan Yang, Wenxuan Song, Xi Wang, Pingyue Sheng, Zheng Fang, Ziyang Zhou, Junjie He, Haodong Yan, Jiayi Chen, Nan Sun, et al. 4d-wam: Infusing spatiotemporal awareness into world action models through trajectory fields. *arXiv preprint arXiv:2608.08023*, 2026a.

Yandan Yang, Shuang Zeng, Tong Lin, Xinyuan Chang, Dekang Qi, Junjin Xiao, Haoyun Liu, Ronghan Chen, Yuzhi Chen, Dongjie Huo, et al. Abot-m0: Vla foundation model for robotic manipulation with action manifold learning. *arXiv preprint arXiv:2602.11236*, 2026b.

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, et al. World action models are zero-shot policies. *arXiv preprint arXiv:2602.15922*, 2026.

Haoqi Yuan, Zhixuan Liang, Anzhe Chen, Ye Wang, Haoyang Li, Pei Lin, Yiyang Huang, Zixing Lei, Tong Zhang, Jiazhao Zhang, et al. Qwen-robotmanip technical report: Alignment unlocks scale for robotic manipulation foundation models. *arXiv preprint arXiv:2606.17846*, 2026a.

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*, 2026b.

Xuanran Zhai, Zekai Huang, Longyan Wu, Qianyou Zhao, Qiaojun Yu, Jieji Ren, Ce Hao, and Harold Soh. SkillVLA: Tackling Combinatorial Diversity in Dual-Arm Manipulation via Skill Reuse. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2026. doi: 10.15607/RSS.2026.XXII.082.

Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, et al. X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. In *International Conference on Learning Representations*, volume 2026, pp. 60580–60606, 2026.

Jiaming Zhou, Qihang Zhang, Gangwei Xu, Cunxin Fan, Yujie Zhao, Ruilin Wang, Yiming Luo, Shuai Yang, Xing Zhu, Yujun Shen, et al. Zero-wam: In-context world-action modeling from human videos for open-ended task generalization. *arXiv preprint arXiv:2608.26103*, 2026.

Jian Zhu, Jianjun Zhang, Taiyi Su, Tianbin Liu, Zhangyuan Wang, Kai Xie, Zitai Huang, Chong Ma, Youzhang He, Tianjian Wang, et al. Dswam: A dual-system world action foundation model for fine-grained robot manipulation. *arXiv preprint arXiv:2607.04927*, 2026.

A Dataset

Simulation Data. For simulation experiments, each demonstration is divided into annotated stages, where an input observation is paired with the expert image at the end of its target stage (i.e. subgoal image), the corresponding instruction, proprioceptive state and action trajectories. We construct subtask annotations to the 2,500 clean expert rollouts provided by RoboTwin (Chen et al., 2025b) according to the compositional structure of each task. The following 19 tasks are treated as long-horizon and segmented into multiple subtasks:

beat block hammer, *blocks ranking rgb*, *blocks ranking size*, *dump bin bigbin*, *handover mic*, *hanging mug*, *move playingcard away*, *open laptop*, *place bread basket*, *place can basket*, *place cans plasticbox*, *place dual shoes*, *place phone stand*, *put bottles dustbin*, *put object cabinet*, *stack blocks three*, *stack blocks two*, *stack bowls three*, *stamp seal*.

For these tasks, the endpoint frame of each subtask is used as its ground-truth subgoal image. The remaining 31 tasks are identified as single-stage tasks with one subgoal image, namely the terminal frame of the trajectory. This preprocessing provides subtask-level supervision without changing the original expert rollouts.

Real-Robot Data. Our real-robot dataset comprises approximately 900 hours of teleoperated demonstrations collected using the Agibot A2 platform. It covers 752 tasks involving more than 200 distinct objects across over 10 diverse backgrounds. All episodes are annotated through a human-in-the-loop pipeline, in which each trajectory is manually segmented into subtasks corresponding to atomic robotic skills and aligned with the semantic structure of the overall task.

B Implementation Details

Implementation of Subgoal Planner $\mathcal{G}_\theta$. Cosmos3 has no native support for image editing, so we adapt its image-to-video capability as follows: the current observation $\mathbf{o}_t$ is encoded as a single clean conditioning latent frame $\mathbf{z}_t$, while the target subgoal image $\mathbf{g}_i$ is replicated four times before VAE encoding, yielding a single target latent frame. Multi-view observations and subgoal images are concatenated into a single frame prior to VAE encoding. During training of $\mathcal{G}_\theta$, we freeze the reasoner branch and update only the generator-side parameters.

We use two planner variants, termed *Base* and *ROI*, both initialized from Cosmos3-Nano and trained for a total of 100k steps on a large corpus of robot teleoperation data covering diverse bimanual manipulation tasks (O'Neill et al., 2024; Khazatsky et al., 2024; Bu et al., 2025; Wu et al., 2025; Tian et al., 2026). *Base* uses the standard flow-matching objective for image-editing training throughout, whereas *ROI* uses the standard objective for the first 30k steps, followed by 70k steps with end-effector region-of-interest weighting. The latter emphasizes the regions around the projected end-effectors in the target image, using a foreground-to-background weight ratio of $4{:}1$. Both variants use AdamW optimizer with a learning rate of 2×10^{-5} and a global batch size of 32. The learning rate is warmed up over 50 steps before held constant. The generator branch and its visual projections are trainable, while the reasoner branch and visual tokenizer remain frozen.

Implementation of World-Action Model π_ϕ^{WAM}. π_ϕ^{WAM} is adapted from the same Cosmos3-Nano backbone following the Cosmos3 robot-policy post-training recipe (Ye et al., 2026; Agarwal et al., 2026): an action encoder $\mathcal{E}_a$, an action decoder $\mathcal{D}_a$, and an action-modality embedding are introduced and zero-initialized for the target embodiment, along with the zero-initialized goal-role embedding used to tag subgoal latents (section 4.2). Multi-view observations and subgoal images follow the same concatenation scheme as in $\mathcal{G}_\theta$. We set the action chunk horizon $H_a = 48$ and video horizon $H_v = 48$ with a video downsample rate of 4, so that π_ϕ^{WAM} jointly predicts a chunk of 48 actions and 12 video frames for each control query.

To alleviate the train-test mismatch, we apply the trained $\mathcal{G}_\theta$ to the demonstrations used to train π_ϕ^{WAM} and obtain predicted subgoals $\hat{\mathbf{g}}_i$. Predictions that are semantically inconsistent or visually distorted are replaced by the corresponding ground-truth subgoals $\mathbf{g}_i$, and π_ϕ^{WAM} is trained on the resulting mixture of predicted and ground-truth subgoals.

During training, the reasoner branch remains frozen, while the generator branch, modality embedding, action projections, and goal-role embedding are trainable. Our policy is trained for 50k steps with a global batch size of 256. We use a base learning rate of 1×10^{-5}, with $10\times$ learning rates for the action projections and action embedding and $5\times$ for the goal embedding. All learning rates follow a linear schedule over a 100k-step cycle without warm-up. We also applies ColorJitter during policy training with brightness, contrast, and saturation strengths of $B = 0.3$, $C = 0.4$, and $S = 0.5$, respectively. Augmentation is disabled at inference.

Subtask transition of ViGAR. At deployment, ViGAR operates in a receding-horizon loop: at every control query, G_θ recomputes $\hat{\mathbf{g}}_i$ from the current observation (and an optional $\mathbf{G}$), after which π_ϕ^{WAM} predicts one action chunk. This procedure implicitly advances the subtask as the scene evolves, requiring no external task expert to detect the completion of a subtask.

Inference Infrastructure Optimization. For the policy π_ϕ^{WAM}, we apply a cascade of numerically lossless optimizations: (i) static compilation with CUDA Graph capture over the Transformer region eliminates kernel-launch overhead; (ii) fused QKV and SwiGLU projections reduce per-chunk GEMM dispatches; (iii) batched CFG packs the conditional and unconditional inputs into a single forward pass with independent attention boundaries, halving the number of network evaluations; (iv) exact condition caching reuses the goal-frame latent and text tokens by content hash when

Table 3: Training configurations and hyperparameters of ViGAR.

Setting	**Subgoal planner**	**Policy**
Pretrained backbone	Cosmos3-Nano	Cosmos3-Nano
Camera views	Head + two wrists	Head + two wrists
Image size	384×320	352×288
Optimizer	AdamW	FusedAdam
Adam (β_1, β_2)	$(0.9, 0.95)$	$(0.9, 0.99)$
Base LR	2×10^{-5}	1×10^{-5}
LR schedule	50-step warm-up, then constant	Linear decay
Global batch size	32	256
Weight decay	0	0.05
Gradient clipping	0.1	1.0
Precision	BF16	BF16

the subgoal has not changed. For the subgoal planner $\mathcal{G}_\theta$, the same optimization stack adopted for π_ϕ^{WAM} is applied. On top of this, we adopt TeaCache (Liu et al., 2025), which skips Transformer evaluations at diffusion steps whose residual change falls below a relative-ℓ_1 threshold. These optimizations accelerates policy inference and subgoal image generation by $2.5\times$ and $11.2\times$ on a single NVIDIA B20Z GPU, respectively. The inference optimization substantially improves the usability and instantaneity of ViGAR especially in real-world deployment.

C Additional Experiment Results

C.1 Visualization of Real-world robot task settings

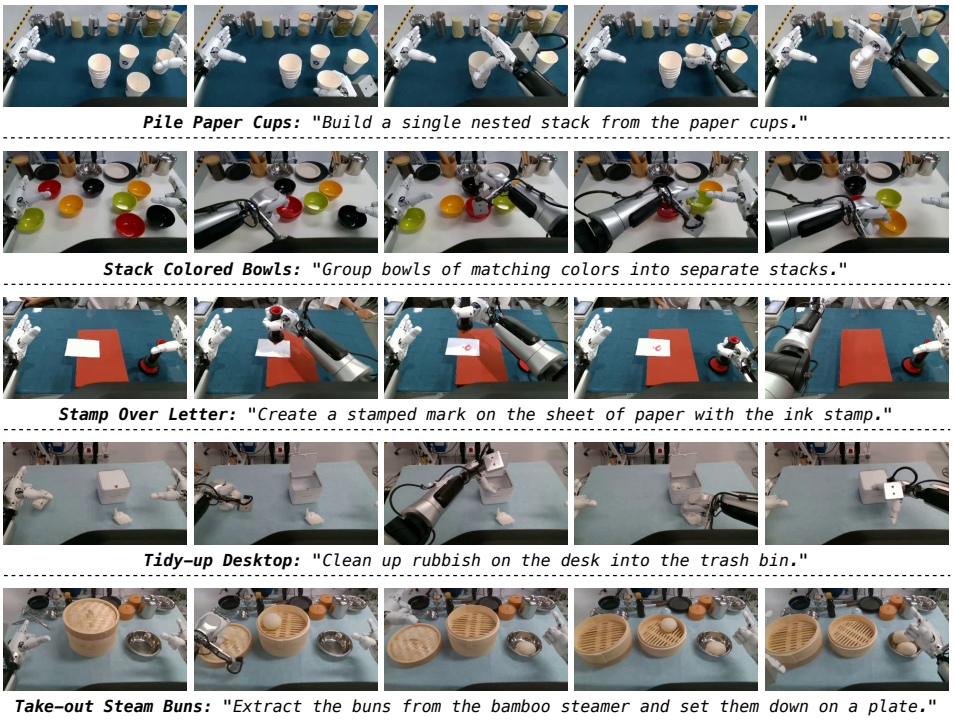


Figure 6: **Visualization of the five real-world compositional manipulation tasks.** The figure displays the task progress of *Pile Paper Cups, Stack Colored Bowls, Stamp over Letters, Tidy-up Desktop*, and *Take-out Steam Buns*.



Figure 7: **Visualization of the two real-world in-context learning tasks.** The figure displays the task progress of *Fruit Arrangement*, and *Desktop Item Storage*. Left Column (blue border): global goal image of the task; Right Columns (red border): task progress of in-context learning tasks.

C.2 Visual Subgoal Prediction Quality

We evaluate the subgoal planner independently of the downstream action policy. For each reference observation o_t, the planner predicts a subgoal image $\hat{g}_i$, which is compared with the corresponding ground-truth subgoal g_i. All methods use the same reference observations and evaluation targets on both RoboTwin and the real-robot benchmark.

We use the following metrics:

- **LPIPS** (↓): perceptual distance between the predicted subgoal $\hat{g}_i$ and the ground-truth subgoal g_i, computed using the AlexNet-based LPIPS metric. Lower values indicate better perceptual similarity.
- **DINO cosine similarity** (DINO-cos, ↑): cosine similarity between the DINOv2-base CLS features of $\hat{g}_i$ and g_i, measuring their semantic alignment. Higher values indicate stronger semantic similarity.
- **Change-IoU** (ΔIoU, ↑): IoU between the change masks of the predicted and ground-truth subgoals. The change masks are obtained by computing the grayscale difference between each target image and the input observation, followed by blurring and thresholding the difference at 25. Higher values indicate better alignment of the predicted and ground-truth spatial changes.
- **Retrieval accuracy** (RetAcc, ↑): DINOv2 features are used to retrieve the most similar image from the endpoint images of all subtasks in the evaluation set. RetAcc measures the fraction of predictions for which the retrieved endpoint corresponds to the ground-truth target subtask. Higher values indicate more accurate subtask-level prediction.

These metrics evaluate complementary aspects of visual subgoal prediction, including perceptual fidelity, semantic alignment, spatial-change consistency, and subtask-level correctness. The results are reported in table 1b.

C.3 Detailed Results on Real-robot Experiments

We quantify the performance of real-robot tasks by two metrics: task progress score and stage-wise success rate. Each long-horizon task is decomposed into K sequential stages, where a stage can only be attempted after all of its preceding stages have been completed. For each task and each method, we conduct $N = 20$ independent trials. In the i-th trial, we denote by $s_i \in \{0, 1, \ldots, K\}$ the number of stages that are successfully completed before the first failure; once a stage fails, the trial is terminated and all subsequent stages are regarded as failed.

Stage-wise success rate. The success rate of stage k is defined as the fraction of trials that successfully complete stage k (and therefore all stages before it):

$$\mathrm{SR}_k = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\left[s_i \geq k\right], \quad k = 1, \ldots, K, \tag{4}$$

where $\mathbf{1}[\cdot]$ is the indicator function. Due to the sequential nature of the stages, SR_k is non-increasing in k, and SR_K equals the full-task success rate.

Task progress score. This metric assigns partial credit according to the proportion of completed stages in each trial and averages it over all trials:

$$\mathrm{TPS} = \frac{1}{N} \sum_{i=1}^{N} \frac{s_i}{K} = \frac{1}{N} \sum_{i=1}^{N} \frac{\sum_{k=1}^{K} \mathbf{1}[s_i \geq k]}{K} = \frac{1}{K} \sum_{k=1}^{K} \mathrm{SR}_k, \tag{5}$$

The stage-wise success rate and task progress scores of all five long-horizon manipulation tasks and two in-context learning tasks are reported in table 4 and table 5.

We also provide the qualitative results of five real-world compositional manipulation tasks besides the task progress score illustrated in fig. 4. The visualization is shown in fig. 8.

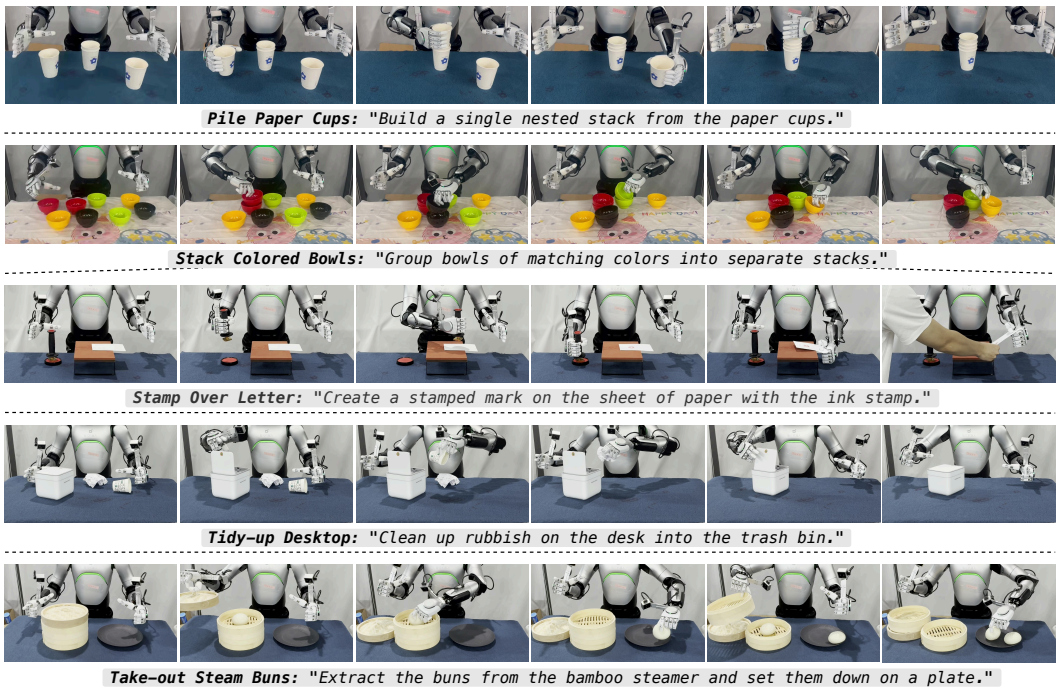


Figure 8: **Qualitative results of 5 real-world compositional manipulation tasks.** The figure displays the task progress of *Pile Paper Cups, Stack Colored Bowls, Stamp Over Letter, Tidy-up Desktop*, and *Take-out Steam Buns*.

Table 4: Stage-wise success rates and task progress scores of ViGAR on real-robot long-horizon manipulation tasks. Each task is evaluated over 20 rollouts. **Bold**: best task progress.

Task	Stage	$\pi_{0.5}$	Cosmos3-Nano-Policy	ViGAR
Pile Paper Cups	S1	0.95	0.85	0.90
	S2	0.85	0.65	0.85
	S3	0.65	0.45	0.80
	Progress	0.82	0.65	**0.85**
Stack Colored Bowls	S1	0.70	0.75	0.85
	S2	0.50	0.55	0.70
	S3	0.25	0.35	0.50
	Progress	0.48	0.55	**0.68**
Stamp over Letter	S1	1.00	0.95	1.00
	S2	0.80	0.85	0.95
	S3	0.70	0.60	0.80
	S4	0.65	0.45	0.80
	Progress	0.79	0.71	**0.89**
Tidy-up Desktop	S1	0.85	0.80	0.85
	S2	0.70	0.65	0.65
	S3	0.55	0.50	0.55
	S4	0.45	0.35	0.40
	Progress	**0.64**	0.58	0.61
Take-out Steam Buns	S1	0.60	0.50	0.65
	S2	0.35	0.25	0.55
	S3	0.25	0.15	0.45
	S4	0.15	0.05	0.30
	S5	0.00	0.00	0.15
	Progress	0.27	0.19	**0.42**
Average Progress		0.60	0.54	**0.69**

Table 5: Stage-wise success rates and task progress scores of ViGAR on in-context real-robot tasks. Each setting is evaluated over 20 rollouts. **Bold**: best task progress.

Task	Stage	In-domain	Out-of-Distribution
Fruit Arrangement	S1	0.95	0.65
	S2	0.70	0.35
	S3	0.50	0.30
	S4	–	0.10
	Progress	0.72	0.43
Desktop Item Storage	S1	0.90	0.65
	S2	0.75	0.30
	S3	–	0.10
	Progress	0.83	0.52