

TDSR-VLA: Transition-aware Denoising Sequence Representations for Vision-Language-Action

Dongwoo Kim^{1*}, Keunho Song^{1*}, Seungmin Lee^{1*}, Hwanhee Ju¹, Eun Sang Cha^{1†}, and Daekyum Kim^{1,2†}

¹ Department of Mechanical Engineering, Korea University, Seoul, Republic of Korea

² School of Smart Mobility, Korea University, Seoul, Republic of Korea

kimdongwoo@korea.ac.kr, bill1869@korea.ac.kr, albert31115@gmail.com, jhh0415@korea.ac.kr,
jessecha@korea.ac.kr, daekyum@korea.ac.kr

*Equal contribution. †Corresponding authors.

Abstract. Vision-language-action (VLA) policies often become brittle under deployment distribution shift when only limited target-domain demonstrations are available. Subgoal-image planning addresses this issue by providing an explicit visual target. However, the target image specifies what future scene to reach, not how the robot should transition toward it. We propose TDSR-VLA, a VLA framework that reuses conditioning states from a diffusion-based Vision Planner to guide action generation. The Vision Planner predicts a subgoal image and provides two representations: a Sequence Representation (SR) grounded in the current observation and an Updated Sequence Representation (USR) refined during denoising steps. A flow-matching Action Expert generates continuous action chunks by injecting these representations through layer-wise prefix key-value conditioning. On real-world tasks, TDSR-VLA achieves an average OpenArm score of 0.83, outperforming GR00T N1.5 (0.74), π_0 (0.71), and SmolVLA (0.58), with the largest gains on object-cooperative tasks (0.88 vs. 0.70). On LIBERO, TDSR-VLA reaches 82.0% on Goal and 56.0% on Long without robot-demonstration pretraining, outperforming OpenVLA on both suites and π_0 on Long. Ablation studies show that planner-side conditioning states provide action-relevant transition cues, improving long-horizon manipulation under limited training data.

Keywords: Vision-Language-Action Models · Robot-Learning · Robotics

1 Introduction

Reliable real-world robot behavior remains a central challenge in robotics, requiring robots to understand semantic goals, perceive complex environments, and execute appropriate actions. Recent vision-language-action (VLA) models build on pretrained vision-language models (VLMs) and further train them for robot

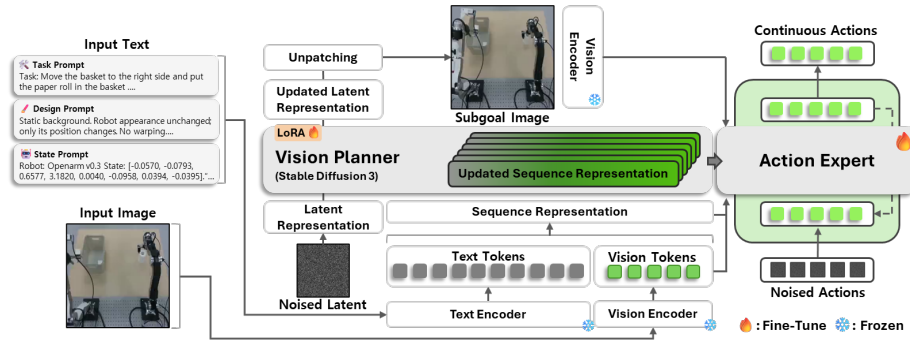


Fig. 1: TDSR-VLA Architecture. The Vision Planner generates a subgoal image and action-useful planner-side transition cues (SR, USR) from the current observation and language instruction. The Action Expert then produces a continuous action chunk via conditional flow matching [34], conditioned on SR/USR and SigLIP-encoded visual tokens from multiple current views (e.g., top-down and wrist cameras) as well as the generated subgoal image, enabling current-grounded yet subgoal-guided control.

control with diverse multi-robot demonstrations [16, 19, 27]. This approach uses the semantic knowledge of pretrained VLMs for language grounding and action-relevant representation learning. However, VLA policies still face challenges in long-horizon execution, cross-embodiment transfer and zero-shot generalization to unseen environments or task variations [27, 38, 40].

Among these challenges, long-horizon execution has motivated planning-to-control interfaces [2, 5, 25, 32, 41, 42] that condition the policy on intermediate planning signals rather than relying only on direct observation-to-action prediction. A widely used planning-to-control interface for manipulation is subgoal-image planning [2, 5, 25, 32, 41, 42]. In this approach, a model predicts a subgoal image that depicts a future state, and the policy uses this image as a visual target for action generation [5, 25, 41]. Diffusion-based visual planners are well suited to this model, because their multi-step generative process can synthesize high-fidelity subgoal images from language and visual context. However, the subgoal image mainly specifies what future state to reach and provides limited information about how to move toward that state. Recent methods supplement subgoal images with additional transition signals for policy conditioning, such as flow-based motion tokens [42], video features [32], and generated demonstrations [2]. Yet, these signals are typically produced as separate motion artifacts or features. They do not directly reuse the internal states of the subgoal generator. This leaves underexplored whether the denoising-time planner states already produced inside a diffusion visual planner can serve as action-conditioning representations. We therefore reuse these states to provide action-useful planner-side transition cues beyond the final subgoal image without training an additional flow, video, or features.

TDSR-VLA implements this idea with a decoupled Vision Planner and Action Expert. A diffusion-transformer Vision Planner [13, 28] predicts a subgoal image at a fixed horizon, conditioned on the current observation and instruction [5, 25]. In addition to the subgoal image, the planner provides action-useful planner-side transition cues that trace how it refines the current scene context toward the predicted subgoal [5, 13, 25, 28]. We cache two versions of these cues, an early Sequence Representation (SR) and a final Updated Sequence Representation (USR). SR anchors the grounded current context, while USR reflects instruction and subgoal-aligned updates. The Action Expert is a transformer policy trained to generate continuous action chunks via conditional flow matching [4, 21, 34, 35]. Rather than conditioning only on the final subgoal appearance, TDSR-VLA injects SR/USR into Action Expert layers via layer-wise prefix key-value conditioning [20, 35]. This lets lower layers stabilize current grounding while upper layers align actions with the planned transition. To enable stable learning from a small dataset, we fine-tune the Vision Planner with LoRA [18], precompute its subgoals and SR/USR outputs, and train the Action Expert with frozen Vision Planner on these cached planning signals.

We show that TDSR-VLA improves VLA execution through evaluations on both the LIBERO benchmark and real-world bimanual tasks [12, 22]. On LIBERO, TDSR-VLA achieves the success rate of 82.0% on Goal and 56.0% on Long without robot-demonstration pretraining, improving over OpenVLA [19] on both suites and showing a trade-off with π_0 [4]. On real-world bimanual manipulation, TDSR-VLA achieves the best overall score of 0.83, outperforming the strongest baseline, GR00T N1.5 [14, 26] (0.74). The largest gains appear on object-cooperative tasks requiring coordinated bimanual transitions. Together with USR ablations and frozen-feature probes, these results support that denoising-time planner representations serve as transition-aware action-conditioning signals for small-data, long-horizon, transition-sensitive VLA execution.

2 Related Work

2.1 Vision-Language-Action Models (VLA)

Vision-Language-Action (VLA) models map visual observations and language instructions to robot actions, typically by adapting large pretrained vision-language backbones (e.g., CLIP [29]/OpenCLIP [8] and T5 [30]-family encoders) with robot demonstrations and then fine-tuning for deployment [4, 8, 11, 16, 19, 29, 30, 43]. Recent open and scalable VLAs emphasize multi-task, multi-robot control and parameter-efficient adaptation, making fast fine-tuning a first-class capability for real-world use [16, 19, 37]. In parallel, world-model-style VLAs augment action learning with future visual prediction to better capture environment dynamics and support planning, but still face a gap between pretraining and deployment-time control [7, 42].

2.2 Action Experts and Continuous Action Generation

Many VLA pipelines improve continuous control by using an action expert that maps multimodal representations into action chunks, often trained with diffusion-style objectives that can model multi-modal behaviors in contact-rich manipulation [9,16,23]. Flow matching offers an alternative continuous-generation paradigm by learning conditional vector fields with simulation-free objectives, and supports efficient variants such as conditional flow matching and rectified flow [21,24,34]. However, even strong continuous generators must still infer what to do next and how to recover from limited supervision; under distribution shift and long horizons, missing transition cues remain a key failure mode [41,42].

2.3 Image Generators and Visual Subgoals for Planning

Image/video generative models can be reused for planning by predicting future observations or synthesizing reachable goal/subgoal images that a controller can track [5, 25, 32, 41]. Subgoal-image and visual-CoT [31]-style approaches make what to reach explicit and can improve instruction grounding by inserting intermediate visual targets before acting [25, 41]. Complementarily, web-scale video generation can provide motion priors, enabling policies to condition on generated demonstrations to reduce robot-data requirements in novel scenarios [2, 32, 36]. Still, a single subgoal image mainly specifies appearance and underconstrains stepwise transition and correction strategy.

3 Methodology

This section describes TDSR-VLA, a two-module Vision-Language-Action framework designed for small-data long-horizon manipulation under distribution shift. As shown in Fig. 1, TDSR-VLA explicitly decouples (i) visual planning, which benefits from large-scale pretrained visual priors, from (ii) action execution, which must remain learnable and stable with only a small number of robot demonstrations. Concretely, we combine a diffusion-based Vision Planner that produces a future-oriented visual plan with a flow-matching Action Expert that generates continuous action chunks conditioned on both the subgoal and action-useful planner-side transition cues. This design directly addresses the key bottleneck highlighted in the introduction: in small-data settings, end-to-end policies must learn “what to do next” and “how to do it” with the same parameters, causing brittle adaptation and error accumulation.

3.1 TDSR-VLA Overview

Problem setup and notation At environment time step t , TDSR-VLA receives a language instruction (task text) ℓ , a low-dimensional robot state $q_t \in \mathbb{R}^{d_q}$ (e.g., on LIBERO, $d_q = 16$ on real-world tasks), and RGB observations I_t^1 (main

view) and $I_t^2, I_t^3, \dots, I_t^n$ depending on the camera setup or robot embodiment. The current observation is formulated as

$$o_t = [I_t^1, I_t^2, \dots, I_t^n, q_t]. \quad (1)$$

The model outputs a continuous action chunk

$$A_{t:t+H-1} = \{a_t, a_{t+1}, \dots, a_{t+H-1}\}, \quad H = 50, \quad (2)$$

where each action $a_{t+i} \in \mathbb{R}^{d_a}$ (e.g., $d_a = 7$ on LIBERO). Internally, actions are padded to a fixed dimension $d_{\max} = 32$ (denoted $\text{pad}(\cdot)$) for a uniform interface.

Decoupled generative factorization TDSR-VLA models a joint conditional distribution over an action chunk and planning signals:

$$P_\theta(A_{t:t+H-1}, S_{t+\Delta}, R_t \mid o_t, \ell). \quad (3)$$

$S_{t+\Delta}$ is a subgoal image representing a reachable future scene at a fixed horizon Δ , and $R_t = (SR_t, USR_t)$ denotes action-useful planner-side transition cues extracted from the Vision Planner’s diffusion transformer [28] during subgoal generation. We explicitly factorize the model as

$$\begin{aligned} & P_\theta(A_{t:t+H-1}, S_{t+\Delta}, R_t \mid o_t, \ell) \\ &= P_\psi(S_{t+\Delta}, R_t \mid I_t^1, q_t, \ell) P_\varphi(A_{t:t+H-1} \mid S_{t+\Delta}, R_t, o_t, \ell). \end{aligned} \quad (4)$$

The first term on the right-hand side, $P_\psi(S_{t+\Delta}, R_t \mid I_t^1, q_t, \ell)$, represents the visual planning policy parameterized by the Vision Planner, which infers the subgoal image and action-useful planner-side transition cues (sequence representation and updated sequence representation). The second term, $P_\varphi(A_{t:t+H-1} \mid S_{t+\Delta}, R_t, o_t, \ell)$, denotes the low-level control policy parameterized by the Action Expert, which generates continuous action chunks conditioned on the explicit visual plan provided by the Vision Planner.

A subgoal image provides a concrete target appearance: what the next reachable state should look like. However, in long-horizon execution with limited recovery coverage, appearance alone can underconstrain how to move and how to correct small errors online. We therefore also condition the Action Expert on $R_t = (SR_t, USR_t)$, the planner-state pair derived from the planner’s internal denoising transformer states. Intuitively, R_t provides planner-side, action-useful transition cues beyond the final subgoal appearance.

3.2 TDSR-VLA Specification

Vision Planner We implement the Vision Planner as a modified Stable Diffusion 3 (SD3) [13] pipeline with an MM-DiT [13] denoising transformer backbone. The planner maps (I_t^1, q_t, ℓ) to a subgoal image $S_{t+\Delta}$ and produces $R_t = (SR_t, USR_t)$ as action-useful planner-side transition cues.

The planner uses three text-conditioning routes consistent with SD3 [13]-style conditioning: prompts for CLIP-L [29], CLIP-G [8], and T5 [30]. We construct

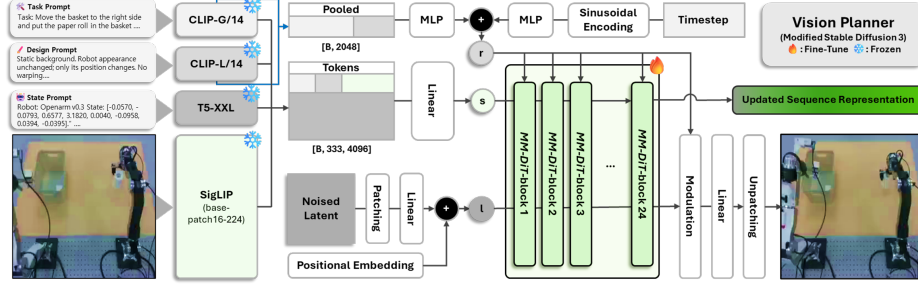


Fig. 2: Vision Planner Architecture. The Vision Planner is an MM-DiT [13] diffusion transformer [28] conditioned on a language prompt and SigLIP-encoded current observations. It generates a subgoal image and exposes denoising-time hidden sequences (SR and USR) that are forwarded to the Action Expert as action-useful planner-side transition cues.

these prompts from the task text ℓ , the current state vector q_t serialized in decimal fixed-point format (6 digits), and a fixed template that indicates the prediction horizon Δ (default offset: 10 frames). As shown in Fig. 2, to ground planning in the physical scene, we additionally encode the current observation I_t^1 using a SigLIP vision encoder to obtain visual tokens. These visual tokens are integrated into the diffusion transformer [28]’s conditioning stream, *i.e.*, the planner receives a unified multimodal conditioning sequence. This explicit visual grounding mitigates the mismatch between text-only planning and real camera observations under deployment shifts.

For Vision Planner training, we supervise subgoal generation using the ground-truth future observation at horizon Δ . Let $S_{t+\Delta}^{\text{gt}}$ denote the target subgoal image and let

$$x_0^S = E_{\text{VAE}}(S_{t+\Delta}^{\text{gt}}) \quad (5)$$

be its latent representation. Following the rectified-flow formulation used in SD3 [13], we sample $\epsilon \sim \mathcal{N}(0, I)$ and a flow time τ , and construct a straight interpolation between the subgoal latent and noise:

$$x_\tau^S = (1 - \tau)x_0^S + \tau\epsilon, \quad u_\tau^S = \epsilon - x_0^S. \quad (6)$$

The Vision Planner predicts a velocity field conditioned on the current image, robot state, and instruction:

$$v_\psi = f_\psi(x_\tau^S, \tau \mid I_t^1, q_t, \ell). \quad (7)$$

We train the Vision Planner with the rectified-flow objective

$$\mathcal{L}_{\text{RF}} = \mathbb{E}_{\tau, x_0^S, \epsilon} \left[\|v_\psi - u_\tau^S\|_2^2 \right]. \quad (8)$$

During Vision Planner denoising, we extract two conditioning-stream hidden sequences from the MM-DiT [13]. The Vision Planner encodes multimodal conditions (instruction ℓ , current view(s) I_t , and state q_t) into conditioning tokens,

projects them to the MM-DiT [13] hidden dimension, and jointly processes them with noised latent tokens throughout denoising. While SD3 [13] decodes the final denoised latent into an RGB subgoal image $S_{t+\Delta}$, the conditioning-side hidden states produced during this joint attention are typically discarded; we explicitly reuse them as planning-aware features for the Action Expert.

1) **sequence representation**: an early-step conditioning hidden sequence that anchors execution to the grounded current view I_t^1 , state information q_t , and text instruction ℓ ;

2) **updated sequence representation**: the final-step conditioning hidden sequence after denoising, *i.e.*, conditioning tokens iteratively updated via joint attention with the noised latent across MM-DiT [13] blocks.

We write

$$R_t = (SR_t, USR_t), \quad SR_t \in \mathbb{R}^{L_p \times d_p^{\text{SD3}}}, \quad USR_t \in \mathbb{R}^{L_p \times d_p^{\text{SD3}}}, \quad (9)$$

with $d_p^{\text{SD3}} = 1536$ and token length $L_p = 333$, and inject them into the Action Expert layer-wise, where SR_t provides grounded current-context cues and USR_t provides plan-aligned transition cues.

Action Expert The Action Expert is a decoder-only transformer [35]. Unlike autoregressive token policies, we generate continuous actions via conditional flow matching [34], which is well-suited for smooth high-dimensional control and supports efficient sampling with a small number of ODE solver steps [4].

We represent each continuous action vector $a_t \in \mathbb{R}^{d_a}$ in a fixed-width format by zero-padding it to $d_{\text{max}} = 32$ [4]. The padded action is then mapped to the transformer hidden space through a learned linear projection, producing a latent action token sequence of length H . After flow-based sampling in the latent space, we map tokens back to the action space with an output projection and keep only the first d_a dimensions.

We embed the continuous flow time τ using sinusoidal encoding [35] followed by an MLP. The resulting embedding conditions the transformer via Adaptive RMSNorm (AdaRMS) [3, 10], enabling time-aware velocity prediction.

In the phase-specific prefix rule, lower layers use SR_t to preserve current-state grounding, whereas upper layers use USR_t as prefix keys and $\text{SigLIP}(S_{t+\Delta})$ as prefix values to guide subgoal-aligned trajectory generation.

As shown in Fig. 3, at each layer i , queries are produced from the current action suffix tokens, while keys/values are formed by concatenating a visual prefix (built from Vision Planner outputs and SigLIP [39] tokens) with suffix keys/values computed from the same action tokens. We reduce the planner states SR_t and USR_t into key tokens, and reduce SigLIP token sequences from current views and the generated subgoal $S_{t+\Delta}$ into value tokens. We then define a layer-wise prefix rule:

$$\text{Layer}_{\text{current}}^{0:11} : K_i^{\text{pre}} = \text{MLP}(SR_t), V_i^{\text{pre}} = \text{MLP}(\text{SigLIP}(I_t^{1:n})), \quad (10)$$

$$\text{Layer}_{\text{subgoal}}^{12:17} : K_i^{\text{pre}} = \text{MLP}(USR_t), V_i^{\text{pre}} = \text{MLP}(\text{SigLIP}(S_{t+\Delta})). \quad (11)$$

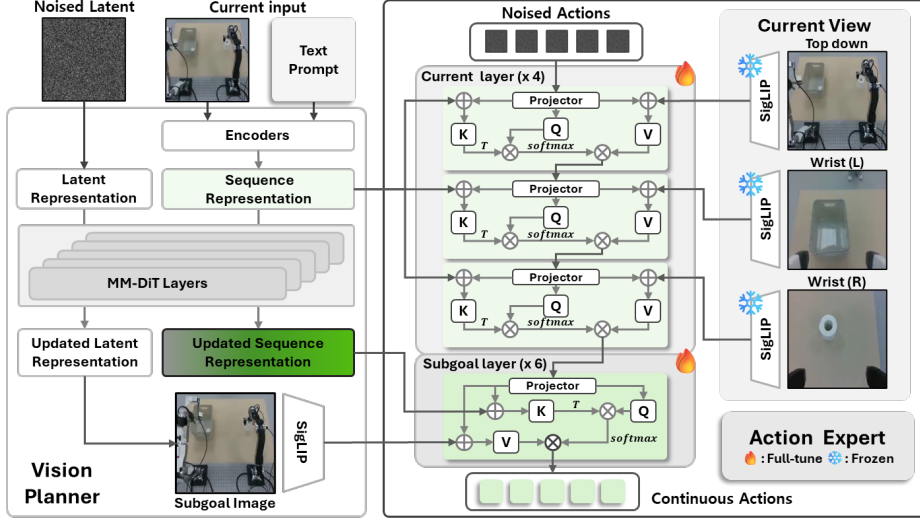


Fig. 3: Action Expert Architecture. The Action Expert is a Gemma [15]-based transformer that generates continuous action chunks via conditional flow matching [34]. It performs prefix KV conditioning by injecting SR/USR as keys and SigLIP [39] tokens from multiple current views and the subgoal image as values, using lower layers for current grounding and higher layers for subgoal guidance.

Within each layer, Action Expert also produces suffix keys/values $K_i^{\text{suf}}, V_i^{\text{suf}}$ from action tokens and concatenates them with the prefix along the sequence dimension:

$$K_i^{\text{all}} = \text{concat}(K_i^{\text{pre}}, K_i^{\text{suf}}), \quad V_i^{\text{all}} = \text{concat}(V_i^{\text{pre}}, V_i^{\text{suf}}). \quad (12)$$

We perform self-attention over a prefix-suffix sequence: the visual prefix provides planner/vision evidence, and the action suffix contains latent action tokens to be updated. Within the suffix, action tokens attend bidirectionally across the entire H -step chunk, allowing the model to coordinate mutually dependent action steps jointly rather than committing left-to-right. As a result, the Action Expert produces smoother and more globally consistent action chunks under high-dimensional visual guidance from the prefix. For each training sample, we draw $z \sim \mathcal{N}(0, I)$ and $\tau \sim \text{Beta}(\alpha = 1.5, \beta = 1.0)$, followed by $\tau \leftarrow 0.999\tau + 0.001$ to avoid degenerate endpoints. We construct noisy actions:

$$x_\tau = (1 - \tau)x_0 + \tau z, \quad u_\tau = z - x_0. \quad (13)$$

The Action Expert predicts a velocity field conditioned on time and prefix-KV:

$$v_\varphi = f_\varphi(x_\tau, \tau \mid \{K_i^{\text{pre}}, V_i^{\text{pre}}\}_{i=0}^{17}), \quad (14)$$

and we minimize

$$\mathcal{L}_{\text{CFM}} = \mathbb{E}_{\tau, x_0, z} [\|v_\varphi - u_\tau\|_2^2]. \quad (15)$$

Algorithm 1 TDSR-VLA test-time closed-loop control

Require: Vision Planner P_ψ , Action Expert f_φ , instruction ℓ , initial observation o_0
Require: Hyperparameters: subgoal offset Δ , chunk length H , ODE steps K , replanning horizon $m \leq H$

- 1: $t \leftarrow 0$; $o_t \leftarrow o_0$
- 2: **while** not done **do**
- 3: $(I_t^{1:n}, q_t) \leftarrow \text{Parse}(o_t)$
- 4: $(S_{t+\Delta}, SR_t, USR_t) \leftarrow P_\psi(I_t^1, q_t, \ell)$
- 5: $(\text{prefixKV}, \text{prefixMask}) \leftarrow \text{BuildPrefixKV}(SR_t, USR_t, I_t^{1:n}, S_{t+\Delta})$
 Keys: SR_t for layers 0–11, USR_t for layers 12–17
 Values: $\text{SigLIP}(I_t^{1:n})$ for layers 0–11, $\text{SigLIP}(S_{t+\Delta})$ for layers 12–17
- 6: $x^{(0)} \sim \mathcal{N}(0, I)$; $\Delta\tau \leftarrow -1/K$
- 7: **for** $k = 0$ to $K - 1$ **do**
- 8: $\tau_k \leftarrow 1 - k/K$
- 9: $v_k \leftarrow f_\varphi(x^{(k)}, \tau_k \mid \text{prefixKV}, \text{prefixMask})$
- 10: $x^{(k+1)} \leftarrow x^{(k)} + \Delta\tau v_k$
- 11: **end for**
- 12: $A_{t:t+H-1} \leftarrow \text{Proj}_{\text{out}}(x^{(K)}[:, : d_a])$
- 13: Execute actions $a_t, \dots, a_{t+m-1}$
- 14: Observe o_{t+m} ; $o_t \leftarrow o_{t+m}$; $t \leftarrow t + m$
- 15: **end while**

This objective encourages the model to learn a direct transport from noise to the action distribution while conditioning on both observation- and plan-derived signals, improving stability in small-data long-horizon training.

3.3 Training Recipe

Training for Vision Planner We train Vision Planner [13] to predict a subgoal image at horizon Δ from (I_t^1, q_t, ℓ) . We use LoRA [18] on MM-DiT [13] while freezing the VAE and text/vision encoders. The planner is trained with the rectified-flow objective $\mathcal{L}_{\text{RF}}$ defined in Eq. (8). This stage transfers large-scale visual priors into a task- and robot-aware subgoal generator.

Precomputing before Training Action Expert Diffusion-based planning is computationally expensive if executed inside every Action Expert training step. To make training practical and reproducible, we run the fine-tuned Vision Planner over the training dataset and cache generated subgoal images $S_{t+\Delta}$ and the two Vision-Planner hidden sequences (SR_t, USR_t) . Concretely, we store $(S_{t+\Delta}, SR_t, USR_t)$. During Action Expert training, a precomputed-only mode loads these cached signals from disk, bypassing repeated diffusion inference. Missing cache entries cause a runtime error, so complete cache coverage is required for training/evaluation. This strategy has a second, conceptual benefit: by freezing the Vision Planner and using its cached outputs, we keep planning signals stable while training the Action Expert, directly preventing the planning–control coupling highlighted as a core failure mode in small-data adaptation.

Training for Action Expert We train the Action Expert with the conditional flow-matching [34] loss $\mathcal{L}_{\text{CFM}}$. Trainable parameters include the Action Expert weights, action input/output projections, the time-embedding MLP (AdaRMS conditioning), and the Vision Planner/SigLIP [39]-MLP reducers. We normalize low-dimensional states and actions using quantile normalization to improve training stability and reduce sensitivity to scale changes across tasks.

Inference Mechanism At test time, TDSR-VLA runs in a closed-loop receding-horizon manner.

1) **Planning step.** Given (I_t^1, q_t, ℓ) , the Vision Planner generates $S_{t+\Delta}$ and extracts (SR_t, USR_t) .

2) **Prefix construction.** We compute SigLIP [39] tokens for observations and the generated subgoal, reduce all planner/vision features to 256-dimensional tokens, and build layer-wise prefix-KV $\{K_i, V_i\}$.

3) **Action sampling by ODE integration.** We sample an initial latent action $x^{(0)} \sim \mathcal{N}(0, I)$ and integrate the learned ODE with Euler steps (default $K = 10$) over a time grid $\{\tau_k\}$ from 1 to 0:

$$x^{(k+1)} = x^{(k)} + \Delta\tau f_\varphi\left(x^{(k)}, \tau_k \mid \{K_i^{\text{pre}}, V_i^{\text{pre}}\}\right), \quad (16)$$

where $\Delta\tau$ is the step size (negative when integrating from 1 to 0). After K steps, we map $x^{(K)}$ back to actions using Proj_{out} , slice the first d_a dimensions, and execute the action chunk. We outline the overall control loop in Algorithm 1.

4 Experiments

We evaluate TDSR-VLA in two settings: single-arm manipulation in simulation and bimanual manipulation in the real world. Our study addresses three questions: (1) how TDSR-VLA compares with strong VLA baselines [4, 14, 26, 33], (2) how performance depends on the depth of layer-wise subgoal injection, and (3) how the Updated Sequence Representation (USR) affects control quality. We therefore use two evaluation tracks and two ablations: (i) LIBERO [22] benchmark evaluation for standardized simulation generalization, and (ii) Open-Arm [12] real-world evaluation under matched data budgets, training schedules, and rollout protocols; plus ablations on subgoal-layer depth and USR usage.

4.1 LIBERO Benchmark Evaluation

LIBERO Evaluation Setup. We evaluate on LIBERO [22] Goal and Long (10 tasks each, 50 demonstrations per task), following the standard protocol of 500 episodes per suite and reporting suite-wise average success rates. TDSR-VLA uses the optimized hyperparameters (subgoal-injection depth and planner denoising steps). We compare against π_0 [4] under the same pretraining setting without robot-demonstration pretraining, following the success rate reported by SmolVLA [33].

Table 1: LIBERO Benchmark Results.

Models	Pre-training	LIBERO Goal	LIBERO Long
OpenVLA	Yes	79.2	53.7
π_0	No	89.0	48.0
TDSR-VLA (Ours)	No	82.0	56.0

Main Results. Table 1 reports the main LIBERO results, where success rates are computed over 50 evaluation trials per task. Relative to OpenVLA (with pre-training), TDSR-VLA improves both Goal (82 vs. 79.2) and Long (56 vs. 53.7). Compared with π_0 [4] (without pre-training), TDSR-VLA is lower on Goal (82 vs. 89) but higher on Long (56 vs. 48). This trend suggests that TDSR-VLA can effectively handle long-horizon tasks even without robot-data pretraining. As further discussed in Section 4.2, this long-horizon capability is also reflected in bimanual real-robot experiments.

4.2 Real-World Evaluation

Real-World Evaluation Setup. We evaluate TDSR-VLA and three baselines [4, 14, 26, 33] on six real-world bimanual tasks using two 7-DOF torque-controlled OpenArm v0.3 arms [12] with 1-DOF parallel-jaw grippers of OpenArm v1.0 shown in Fig. 4. A Pinocchio [6]-based gravity-compensation controller runs at 100 Hz during both collection and evaluation.

We record synchronized joint states, gripper states, and RGB images from three RealSense D435 cameras (one overhead and two wrist-mounted) at 30 Hz. The dataset contains 100 demonstrations per task. All methods are fine-tuned for 10 epochs and evaluated with 20 trials per task; all methods run at 30 Hz. To quantify trial-level uncertainty, we compute nonparametric 95% bootstrap confidence intervals and paired bootstrap differences over matched task-trial indices. TDSR-VLA achieves the highest mean OpenArm score of 0.825 with 95% confidence interval of [0.767, 0.879], with a positive paired gain over GR00T N1.5 of +0.083 with a 95% confidence interval of [0.004, 0.163].

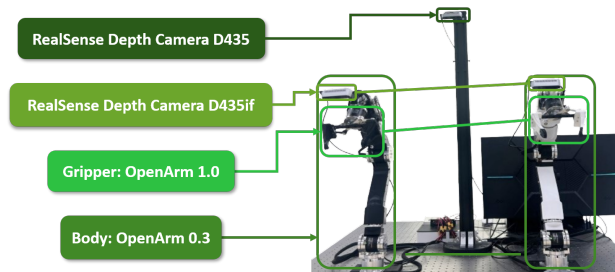


Fig. 4: Real-world OpenArm evaluation setup. Two 7-DOF OpenArm v0.3 manipulators with parallel-jaw grippers are used with one overhead and two wrist-mounted RGB cameras for bimanual task evaluation.

Table 2: Task descriptions and per-arm scoring metrics.

Task	Description	Score metrics
Task a (Sequential Tasks)	Move the basket to the right side and put the paper roll into the basket (Paper Roll into Basket)	0.5 (Left Arm): Move the basket to the right. 0.5 (Right Arm): Grab the paper roll and put it into the basket.
Task b (Sequential Tasks)	Put the umbrellas into the basket (Umbrella Placing)	0.5 (Left Arm): Put the left umbrella into the basket. 0.5 (Right Arm): Put the right umbrella into the basket.
Task c (Object-Cooperative Tasks)	Move the Rubik’s Cube on the right end to the left end using both arms (Bimanual Cube Transfer)	0.5 (Left Arm): Move the Rubik’s Cube from the center to the left end. 0.5 (Right Arm): Move the Rubik’s Cube from the right end to the center.
Task d (Sequential Tasks)	Open the first floor of the drawer and put the steel cup into the opened floor (Steel Cup into Drawer)	0.5 (Left Arm): Open and close the drawer. 0.5 (Right Arm): Put the steel cup into the opened floor.
Task e (Sequential Tasks)	Put the brown cup into the basket and put the green cup on the plate (Dual Cup Sorting)	0.5 (Left Arm): Put the brown cup into the basket. 0.5 (Right Arm): Put the green cup on the plate.
Task f (Object-Cooperative Tasks)	Pass the orange wine glass to the left arm to put the orange wine glass into the basket (Glass Handover)	0.5 (Left Arm): Grab the orange wine glass from the right arm and put it into the basket. 0.5 (Right Arm): Grab the orange wine glass and pass it to the left arm.

We compare with SmolVLA [33], π_0 [4], and GR00T N1.5 [14, 26], all fine-tuned from their public robot-demonstration-pretrained checkpoints [4, 14, 26, 33]. For TDSR-VLA, we fine-tune the SD3 [13]-based Vision Planner, cache subgoal and planner-sequence outputs on the training set, freeze the planner, and train the Action Expert for 10 epochs without robot-policy pretraining. We use a per-arm scoring rule in Table 2.

Main Results. Fig. 5 reports average trial scores across six tasks. TDSR-VLA achieves the best overall score (0.83), outperforming GR00T N1.5 [14, 26] (0.74), π_0 [4] (0.71), and SmolVLA [33] (0.58). On sequential tasks, TDSR-VLA obtains 0.80, modestly above π_0 [4] and GR00T N1.5 [14, 26] (both about 0.76). On object-cooperative tasks, the gain is larger: 0.88 for TDSR-VLA versus GR00T N1.5 [14, 26] (0.70) and π_0 [4] (0.63), *i.e.*, a +0.18 gain over the strongest baseline. These results indicate that explicit visual subgoals and planner-side transition-cue conditioning are particularly effective for tightly coordinated bimanual control.

4.3 Ablation Studies

We conduct ablations on two factors: the degree of subgoal-image injection in the Action Expert and the use of the Updated Sequence Representation (USR) in subgoal layers.

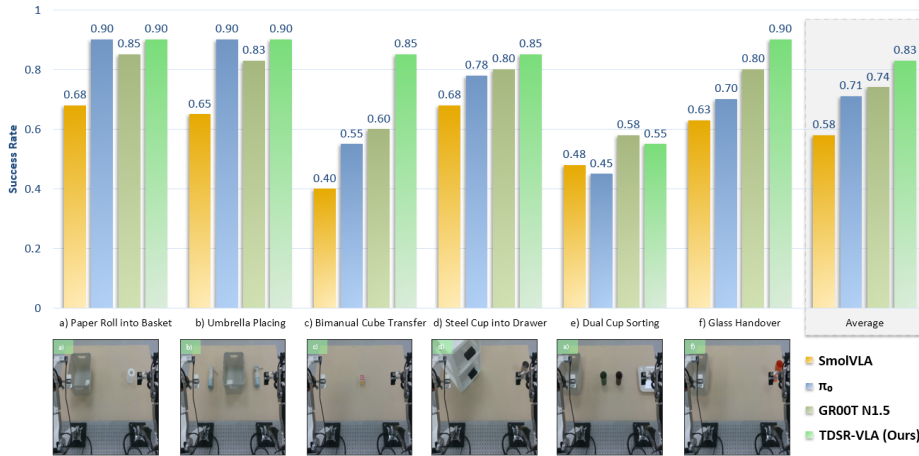


Fig. 5: Main results on OpenArm bimanual tasks. This figure compares the average per-trial OpenArm [12] score across six real-world bimanual tasks (sequential versus object-cooperative) under the per-arm scoring protocol, where TDSR-VLA achieves the best overall score of 0.83, outperforming GR00T N1.5 [14, 26] (0.74), π_0 [4] (0.71), and SmolVLA [33] (0.58).

Performance Evaluation by Subgoal-Injection Depth. We evaluate OpenArm scores while varying the number of subgoal layers in the Action Expert, with results shown in Fig. 6(a). Using three subgoal layers yields performance close to the six-layer setting. In contrast, performance drops when the number of subgoal layers increases to nine, and using zero subgoal layers is also slightly worse than three layers. These findings indicate that the Action Expert performs best when maintaining an appropriate ratio between lower layers that ground actions in current states and upper layers that guide trajectory generation by attending to USR-based transition cues and SigLIP [39]-encoded subgoal-image tokens.

Performance Evaluation with and without USR. We ablate USR in subgoal layers to quantify its contribution to bimanual real-world performance under an otherwise fixed TDSR-VLA configuration. We fix the Action Expert to the best-performing six-layer subgoal-injection setting and compare two variants that differ only in subgoal-layer keys: (i) using USR as keys in subgoal layers and (ii) using the initial Sequence Representation (SR) as keys in subgoal layers, identical to current layers. As shown in Fig. 6(b), enabling USR improves the average OpenArm score from 0.70 (without USR) to 0.83 (with USR) under the same evaluation protocol. This gain indicates that subgoal layers benefit from plan-aligned transition cues, strengthening corrective behavior after minor deviations and mitigating compounding errors. Accordingly, we inject USR only into subgoal layers while earlier current layers retain SR, yielding better phase-specific conditioning. Compared with π_0 [4]-style fixed prefix-KV condi-

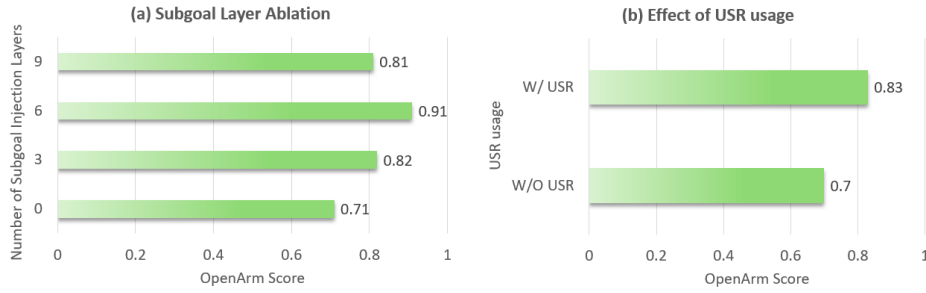


Fig. 6: Ablation results on subgoal image injection and Updated Sequence Representation (USR) in the Action Expert. (a) This figure shows the OpenArm score as the number of subgoal injection layers varies: using three layers achieves comparable performance to the six-layer setting, while nine layers degrade performance; removing subgoal layers is slightly worse than using three. (b) With six subgoal layers fixed, using USR as keys in subgoal layers improves the average OpenArm score from 0.70 to 0.83 compared to using the initial Sequence Representation (SR).

tioning, this design may incur modest overhead but provides stronger goal-phase alignment for bimanual coordination and handover timing.

To analyze why USR improves performance, we conduct frozen-feature ridge probing [1, 17] on unseen LIBERO trajectories. As shown in Fig. 7, USR outperforms State, Subgoal Image, and Shuffled USR features on future end-effector displacement (ΔEE_{10}), future action statistics (Mean Action₁₀), and progress prediction (Progress Bin), reaching $R^2 = 0.48$ for ΔEE_{10} , $R^2 = 0.40$ for Mean Action₁₀, and 0.74 Macro-F1 for Progress Bin classification. Since the Subgoal Image probe uses the same SigLIP pathway as the Action Expert and Shuffled USR removes timestep alignment, these results suggest that USR improves policy performance by providing planner-side, action-useful transition cues.

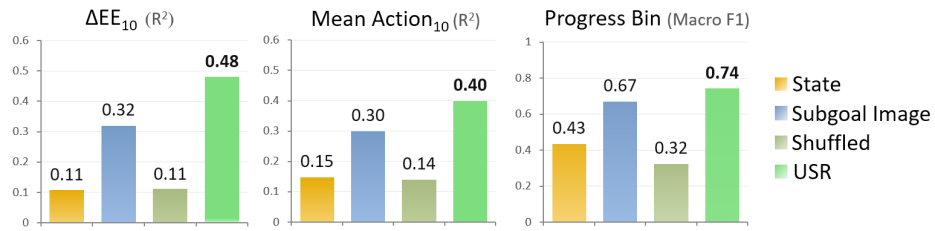


Fig. 7: Frozen-feature probing of USR. USR outperforms State, Subgoal Image, and Shuffled USR on unseen LIBERO future-motion, future-action, and progress probes, indicating timestep-aligned transition cues beyond final-subgoal appearance.

Table 3: Chunk- and task-level execution analysis. (a) Per-chunk latency of synchronous TDSR-VLA, dominated by Vision Planner inference. (b) Mean rollout duration across real-world tasks.

(a) Chunk-level latency			(b) Mean rollout duration	
Metric	Value	Notes	Model	Time (s)
Vision Planner latency	3014.8 ms	Planner forward time/chunk	SmolVLA	32.60
Action Expert latency	438.6 ms	Action Expert time/chunk	π_0	30.78
Total latency	3453.4 ms	End-to-end time/chunk	GR00T N1.5	46.42
Estimated idle time	1.79 s	Sync. replanning at 30 Hz	TDSR-VLA (Ours)	43.20

5 Conclusion, Limitations and Future Work

In this work, we introduce TDSR-VLA, a novel Vision-Language-Action framework that decouples visual planning from low-level control to address the data inefficiency and functional entanglement of traditional end-to-end models. By utilizing a Stable Diffusion 3 [13]-based Vision Planner to generate explicit subgoal images alongside action-useful planner-side transition cues, our model effectively guides a flow-matching Action Expert through a hierarchical conditioning structure. Comprehensive experiments across diverse real-world bimanual manipulation tasks show that TDSR-VLA substantially surpasses existing VLA baselines, with especially strong gains in scenarios that demand tight object-level coordination. Ultimately, by demonstrating that explicit visual reasoning can be efficiently transferred to low-level control, TDSR-VLA establishes a robust foundation for scalable, long-horizon robot manipulation in data-scarce environments.

While TDSR-VLA demonstrates strong overall performance, its sequential planning-then-acting architecture introduces latency as a primary systems limitation. As shown in Table 3(a), per-chunk latency is dominated by Vision Planner inference (3014.8 ms out of 3453.4 ms total, $\approx 87\%$), and naive synchronous replanning can induce an estimated 1.79 s idle interval before the next chunk becomes available. This limitation is most visible in tasks requiring fine-grained, continuous motor execution, such as Dual Cup Sorting, where TDSR-VLA scores 0.55, slightly below GR00T N1.5 [14, 26] (0.58), which maps VLM embeddings directly to actions through its Diffusion Transformer [28] without an intermediate image-generation step. Table 3(b) further shows that TDSR-VLA has a task-level rollout duration comparable to GR00T N1.5 [14, 26]; together with Table 3(a), this indicates that the runtime cost of TDSR-VLA is primarily driven by repeated per-chunk Vision Planner inference. Addressing this latency through asynchronous subgoal generation that overlaps planning with execution is therefore an important direction for future work.

References

1. Alain, G., Bengio, Y.: Understanding intermediate layers using linear classifier probes. arXiv preprint arXiv:1610.01644 (2016). <https://doi.org/10.48550/>

- arXiv.1610.01644
2. Bharadhwaj, H., Dwibedi, D., Gupta, A., Tulsiani, S., Doersch, C., Xiao, T., Shah, D., Xia, F., Sadigh, D., Kirmani, S.: Gen2Act: Human video generation in novel scenarios enables generalizable robot manipulation. In: Conf. Robot Learn. (CoRL). Proceedings of Machine Learning Research, vol. 305, pp. 3936–3951. PMLR (2025), <https://proceedings.mlr.press/v305/bharadhwaj25a.html>, accessed 27 June 2026
 3. Black, K., Brown, N., Darpinian, J., Dhabalia, K., Driess, D., Esmail, A., Equi, M.R., Finn, C., Fusai, N., Galliker, M.Y., Ghosh, D., Groom, L., Hausman, K., Ichter, B., Jakubczak, S., Jones, T., Ke, L., LeBlanc, D., Levine, S., Li-Bell, A., Mothukuri, M., Nair, S., Pertsch, K., Ren, A.Z., Shi, L.X., Smith, L., Springenberg, J.T., Stachowicz, K., Tanner, J., Vuong, Q., Walke, H., Walling, A., Wang, H., Yu, L., Zhilinsky, U.: $\pi_{0.5}$: A vision-language-action model with open-world generalization. In: Conf. Robot Learn. (CoRL). Proceedings of Machine Learning Research, vol. 305, pp. 17–40. PMLR (2025), <https://proceedings.mlr.press/v305/black25a.html>, accessed 27 June 2026
 4. Black, K., Brown, N., Driess, D., Esmail, A., Equi, M.R., Finn, C., Fusai, N., Groom, L., Hausman, K., Ichter, B., Jakubczak, S., Jones, T., Ke, L., Levine, S., Li-Bell, A., Mothukuri, M., Nair, S., Pertsch, K., Shi, L.X., Smith, L., Tanner, J., Vuong, Q., Walling, A., Wang, H., Zhilinsky, U.: π_0 : A vision-language-action flow model for general robot control. In: Proc. Robot.: Sci. Syst. (RSS). Los Angeles, CA, USA (Jun 2025). <https://doi.org/10.15607/RSS.2025.XXI.010>
 5. Black, K., Nakamoto, M., Atreya, P., Walke, H., Finn, C., Kumar, A., Levine, S.: Zero-shot robotic manipulation with pre-trained image-editing diffusion models. In: ICLR (2024). <https://doi.org/10.48550/arXiv.2310.10639>
 6. Carpentier, J., Saurel, G., Buondonno, G., Mirabel, J., Lamiriaux, F., Stasse, O., Mansard, N.: The Pinocchio C++ library – a fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives. In: IEEE/SICE International Symposium on System Integration (SII) (2019). <https://doi.org/10.1109/SII.2019.8700380>
 7. Cen, J., Yu, C., Yuan, H., Jiang, Y., Huang, S., Guo, J., Li, X., Song, Y., Luo, H., Wang, F., Zhao, D., Chen, H.: WorldVLA: Towards autoregressive action world model. arXiv preprint arXiv:2506.21539 (2025). <https://doi.org/10.48550/arXiv.2506.21539>
 8. Cherti, M., Beaumont, R., Wightman, R., Wortsman, M., Ilharco, G., Gordon, C., Schuhmann, C., Schmidt, L., Jitsev, J.: Reproducible scaling laws for contrastive language-image learning. In: CVPR. pp. 2818–2829 (2023). <https://doi.org/10.1109/CVPR52729.2023.00276>, https://openaccess.thecvf.com/content/CVPR2023/html/Cherti_Reproducible_Scaling_Laws_for_Contrastive_Language-Image_Learning_CVPR_2023_paper.html, accessed 27 June 2026
 9. Chi, C., Feng, S., Du, Y., Xu, Z., Cousineau, E., Burchfiel, B.C., Song, S.: Diffusion Policy: Visuomotor policy learning via action diffusion. In: Proc. Robot.: Sci. Syst. (RSS). Daegu, Republic of Korea (Jul 2023). <https://doi.org/10.15607/RSS.2023.XIX.026>, <https://roboticsproceedings.org/rss19/p026.html>, accessed 27 June 2026
 10. Crowson, K., Baumann, S.A., Birch, A., Abraham, T.M., Kaplan, D.Z., Shippole, E.: Scalable high-resolution pixel-space image synthesis with hourglass diffusion transformers. In: ICML (2024), <https://openreview.net/forum?id=WRIn2HmtBS>, accessed 27 June 2026

11. Driess, D., Xia, F., Sajjadi, M.S.M., Lynch, C., Chowdhery, A., Ichter, B., Wahid, A., Tompson, J., Vuong, Q., Yu, T., Huang, W., Chebotar, Y., Sermanet, P., Duckworth, D., Levine, S., Vanhoucke, V., Hausman, K., Toussaint, M., Greff, K., Zeng, A., Mordatch, I., Florence, P.: PaLM-E: An embodied multimodal language model. In: ICML. Proceedings of Machine Learning Research, vol. 202, pp. 8469–8488. PMLR (2023). <https://doi.org/10.48550/arXiv.2303.03378>, <https://proceedings.mlr.press/v202/driess23a.html>, accessed 27 June 2026
12. Enactic: OpenArm: an open-source humanoid arm for physical AI research and deployment in contact-rich environments. <https://github.com/enactic/openarm> (2025), GitHub repository, accessed 27 June 2026
13. Esser, P., Kulal, S., Blattmann, A., Entezari, R., Müller, J., Saini, H., Levi, Y., Lorenz, D., Sauer, A., Boesel, F., Podell, D., Dockhorn, T., English, Z., Rombach, R.: Scaling rectified flow transformers for high-resolution image synthesis. In: ICML. Proceedings of Machine Learning Research, vol. 235, pp. 12606–12633. PMLR (2024), <https://proceedings.mlr.press/v235/esser24a.html>, accessed 27 June 2026
14. Fan, J., Ma, Y., Zhu, S., Yu, J., Agrawal, A., Guo, J., Ma, W.C., Huang, H., Lu, Y., Fang, K., Yao, B., Li, X., Li, S., Wu, H., Chen, K., Bhardwaj, N., Coumans, E., Rosman, G., Choraria, R., Matasov, R., Zhu, Y., Zhang, K., Lee, K.H., Sung, Y.L., Wang, A., Kerr, J., Sargent, K., Michalewski, H., Joo, H., Sreenath, K., Levine, S.: GR00T N1: An open foundation model for generalist humanoid robots. arXiv preprint arXiv:2503.14734 (2025). <https://doi.org/10.48550/arXiv.2503.14734>, <https://arxiv.org/abs/2503.14734>, accessed 27 June 2026
15. Gemma Team: Gemma: Open models based on Gemini research and technology. arXiv preprint arXiv:2403.08295 (2024). <https://doi.org/10.48550/arXiv.2403.08295>
16. Ghosh, D., Walke, H.R., Pertsch, K., Black, K., Mees, O., Dasari, S., Hejna, J., Kreiman, T., Xu, C., Luo, J., Tan, Y.L., Chen, L.Y., Vuong, Q., Xiao, T., Sanketi, P.R., Sadigh, D., Finn, C., Levine, S.: Octo: An open-source generalist robot policy. In: Proc. Robot.: Sci. Syst. (RSS). Delft, Netherlands (Jul 2024). <https://doi.org/10.15607/RSS.2024.XX.090>, <https://www.roboticsproceedings.org/rss20/p090.html>, accessed 27 June 2026
17. Hewitt, J., Liang, P.: Designing and interpreting probes with control tasks. In: EMNLP-IJCNLP. pp. 2733–2743 (2019). <https://doi.org/10.18653/v1/D19-1275>
18. Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: LoRA: Low-rank adaptation of large language models. In: ICLR (2022), <https://openreview.net/forum?id=nZeVKeeFYf9>, accessed 27 June 2026
19. Kim, M.J., Pertsch, K., Karamcheti, S., Xiao, T., Balakrishna, A., Nair, S., Rafailov, R., Foster, E.P., Sanketi, P.R., Vuong, Q., Kollar, T., Burchfiel, B., Tedrake, R., Sadigh, D., Levine, S., Liang, P., Finn, C.: OpenVLA: An open-source vision-language-action model. In: Conf. Robot Learn. (CoRL). Proceedings of Machine Learning Research, vol. 270, pp. 2679–2713. PMLR (2025), <https://proceedings.mlr.press/v270/kim25c.html>, accessed 27 June 2026
20. Li, X.L., Liang, P.: Prefix-tuning: Optimizing continuous prompts for generation. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). pp. 4582–4597 (2021), <https://aclanthology.org/2021.acl-long.353/>, accessed 27 June 2026

21. Lipman, Y., Chen, R.T.Q., Ben-Hamu, H., Nickel, M., Le, M.: Flow matching for generative modeling. In: ICLR (2023), <https://openreview.net/forum?id=PqvMRDCJT9t>, accessed 27 June 2026
22. Liu, B., Zhu, Y., Gao, C., Feng, Y., Liu, Q., Zhu, Y., Stone, P.: LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In: NeurIPS (2023), <https://neurips.cc/virtual/2023/poster/73415>, accessed 27 June 2026
23. Liu, S., Wu, L., Li, B., Tan, H., Chen, H., Wang, Z., Xu, K., Su, H., Zhu, J.: RDT-1B: A diffusion foundation model for bimanual manipulation. In: ICLR (2025), <https://openreview.net/forum?id=yAzN4tz7oI>, accessed 27 June 2026
24. Liu, X., Gong, C., Liu, Q.: Flow straight and fast: Learning to generate and transfer data with rectified flow. In: ICLR (2023), <https://openreview.net/forum?id=XVjTT1nw5z>, accessed 27 June 2026
25. Ni, F., Hao, J., Wu, S., Kou, L., Liu, J., Zheng, Y., Wang, B., Zhuang, Y.: Generate subgoal images before act: Unlocking the chain-of-thought reasoning in diffusion model for robot manipulation with multimodal prompts. In: CVPR. pp. 13991–14000 (2024). <https://doi.org/10.1109/CVPR52733.2024.01327>, https://openaccess.thecvf.com/content/CVPR2024/html/Ni_Generate_Subgoal_Images_before_Act_Unlocking_the_Chain-of-Thought_Reasoning_in_CVPR_2024_paper.html, accessed 27 June 2026
26. NVIDIA Research: GR00T N1.5: An improved open foundation model for generalist humanoid robots. https://research.nvidia.com/labs/gear/gr00t-n1_5/ (2025), official project page, accessed 27 June 2026
27. Open X-Embodiment Collaboration: Open X-Embodiment: Robotic learning datasets and RT-X models. In: IEEE International Conference on Robotics and Automation (ICRA). pp. 6892–6903 (2024). <https://doi.org/10.1109/ICRA57147.2024.10611477>
28. Peebles, W., Xie, S.: Scalable diffusion models with transformers. In: ICCV. pp. 4195–4205 (2023). <https://doi.org/10.1109/ICCV51070.2023.00387>
29. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning transferable visual models from natural language supervision. In: ICML. Proceedings of Machine Learning Research, vol. 139, pp. 8748–8763. PMLR (2021). <https://doi.org/10.48550/arXiv.2103.00020>, <https://proceedings.mlr.press/v139/radford21a.html>, accessed 27 June 2026
30. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J.: Exploring the limits of transfer learning with a unified text-to-text transformer. JMLR **21**(140), 1–67 (2020). <https://doi.org/10.48550/arXiv.1910.10683>, <https://www.jmlr.org/papers/v21/20-074.html>, accessed 27 June 2026
31. Shao, H., Qian, S., Xiao, H., Song, G., Zong, Z., Wang, L., Liu, Y., Li, H.: Visual cot: Unleashing chain-of-thought reasoning in multi-modal language models. arXiv preprint arXiv:2403.16999 (2024). <https://doi.org/10.48550/arXiv.2403.16999>
32. Shen, Y., Wei, F., Du, Z., Liang, Y., Lu, Y., Yang, J., Zheng, N., Guo, B.: VideoVLA: Video generators can be generalizable robot manipulators. arXiv preprint arXiv:2512.06963 (2025). <https://doi.org/10.48550/arXiv.2512.06963>
33. Shukor, M., Aubakirova, D., Capuano, F., Kooijmans, P., Palma, S., Zouitine, A., Aractingi, M., Pascal, C., Russi, M., Marafioti, A., Alibert, S., Cord, M., Wolf, T., Cadene, R.: SmolVLA: A vision-language-action model for affordable and efficient robotics. arXiv preprint arXiv:2506.01844 (2025). <https://doi.org/10.48550/arXiv.2506.01844>

34. Tong, A., Malkin, N., Huguet, G., Zhang, Y., Rector-Brooks, J., Fatras, K., Wolf, G., Bengio, Y.: Improving and generalizing flow-based generative models with minibatch optimal transport. TMLR (2024), <https://openreview.net/forum?id=CD9Snc73AW>, accessed 27 June 2026
35. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: NeurIPS. vol. 30, pp. 5998–6008 (2017). <https://doi.org/10.48550/arXiv.1706.03762>, https://proceedings.neurips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html, accessed 27 June 2026
36. Wu, H., Jing, Y., Cheang, C., Chen, G., Xu, J., Li, X., Liu, M., Li, H., Kong, T.: Unleashing large-scale video generative pre-training for visual robot manipulation. In: ICLR (2024), <https://openreview.net/forum?id=NxoFmGgWC9>, accessed 27 June 2026
37. Yadav, Y., Zhou, Z., Wagenmaker, A., Pertsch, K., Levine, S.: Robust finetuning of vision-language-action robot policies via parameter merging. arXiv preprint arXiv:2512.08333 (2025). <https://doi.org/10.48550/arXiv.2512.08333>
38. Yang, J.H., Sadigh, D., Finn, C.: Polybot: Training one policy across robots while embracing variability. In: Conference on Robot Learning (CoRL). Proceedings of Machine Learning Research, vol. 229, pp. 2955–2974. PMLR (2023), <https://proceedings.mlr.press/v229/yang23c.html>, accessed 27 June 2026
39. Zhai, X., Mustafa, B., Kolesnikov, A., Beyer, L.: Sigmoid loss for language image pre-training. In: ICCV. pp. 11975–11986 (2023). <https://doi.org/10.1109/ICCV51070.2023.01100>
40. Zhang, Y., Wang, C., Lu, O., Zhao, Y., Ge, Y., Sun, Z., Li, X., Zhang, C., Bai, C., Li, X.: Align-then-steer: Adapting the vision-language action models through unified latent guidance. In: ICLR (2026), <https://openreview.net/forum?id=T3i7Ifeatk>, poster, accessed 27 June 2026
41. Zhao, Q., Lu, Y., Kim, M.J., Fu, Z., Zhang, Z., Wu, Y., Li, Z., Ma, Q., Han, S., Finn, C., Handa, A., Lin, T.Y., Wetzstein, G., Liu, M.Y., Xiang, D.: CoT-VLA: Visual chain-of-thought reasoning for vision-language-action models. In: CVPR. pp. 1702–1713 (2025), https://openaccess.thecvf.com/content/CVPR2025/html/Zhao_CoT-VLA_Visual_Chain-of-Thought_Reasoning_for_Vision-Language-Action_Models_CVPR_2025_paper.html, accessed 27 June 2026
42. Zhong, Z., Yan, H., Li, J., Liu, X., Gong, X., Zhang, T., Song, W., Chen, J., Zheng, X., Wang, H., Li, H.: FlowVLA: Visual chain of thought-based motion reasoning for vision-language-action models. arXiv preprint arXiv:2508.18269 (2025). <https://doi.org/10.48550/arXiv.2508.18269>
43. Zitkovich, B., Haroun, A., Ravichandar, A., Vazquez, D., Ibarz, E., Hutter, F., Sukhatme, G.S., Hoffman, G., Chiang, H.T.L., Walke, H.R., Gur, I., Wu, J., David, J., Joshi, J., Irpan, A., Ichter, B., Hsu, J., Herzog, A., Hausman, K., Gopalakrishnan, K., Fu, C., Florence, P., Finn, C., Dubey, K.A., Driess, D., Ding, T., Choromanski, K.M., Chen, X., Chebotar, Y., Carbajal, J., Brown, N., Brohan, A., Arenas, M.G., Han, K.: RT-2: Vision-language-action models transfer web knowledge to robotic control. In: Conf. Robot Learn. (CoRL). Proceedings of Machine Learning Research, vol. 229, pp. 2165–2183. PMLR (2023), <https://proceedings.mlr.press/v229/zitkovich23a.html>, accessed 27 June 2026