Vision-Language-Action Model for Robotic Manipulation of Textile Objects

Jose Edgar Hernandez Cancino Estrada

Supervisor: prof. dr. Danijel Skočaj
Co-supervisor: dr. Domen Tabernik

July 2026

University of Ljubljana
Faculty of Computer and Information Science

FRI

Vision-Language-Action Model for Robotic Manipulation of Textile Objects

Jose Edgar Hernandez Cancino Estrada

Erasmus Mundus Joint Master in Artificial Intelligence (EMAI)

Faculty of Computer and Information Science

Supervisor: prof. dr. Danijel Skočaj
Cosupervisor: dr. Domen Tabernik

Vision-language-action (VLA) models are end-to-end robot control models that inherit rich perceptual representations from pretrained vision-language foundation models. However, it remains unclear whether these representations are fully exploited during action generation. This work investigates whether task-relevant perceptual information can be recovered from the internal multimodal features of X-VLA and reused to improve robotic control, particularly for the spatial reasoning required to accurately grasp textile objects.

We introduce a lightweight visual-thought decoder trained to reconstruct expert-derived perceptual targets from the internal representations of the X-VLA backbone. The recovered information is incorporated in two ways: implicitly as an auxiliary training objective that encourages the backbone to preserve task-relevant structure, and explicitly as guidance tokens that are reintroduced into the action-generation process.

The proposed methods are evaluated on cube pickup, cloth pickup and placement, and cloth folding tasks. We compare the original X-VLA model with variants that either regularize its latent representations to preserve task-relevant information or make this information explicitly available during action generation. The results show that explicitly providing task-relevant information can improve policy performance in ambiguous or challenging situations. Across the evaluated tasks, implicit guidance through latent-space regularization produces mixed results, whereas explicit guidance yields improvements on some cloth manipulation tasks. These findings suggest that VLA backbones contain recoverable task-relevant perceptual structure, but that its usefulness depends on how this information is incorporated into the action-generation process and on the characteristics of the manipulation task.

VLA | ViT | Flow Matching | Robotics | Visual-Language-Action | multi-modal perception | robot manipulation

Vizualno-jezikovno-akcijski model za robotsko manipulacijo s tekstilnimi objekti

Vizualno-jezikovno-akcijski (VLA) modeli so enoviti modeli za vodenje robotov, ki bogate zaznavne predstavitve prevzemajo iz predhodno naučenih vizualno-jezikovnih temeljnih modelov. Vendar še vedno ni jasno, ali se te predstavitve pri generiranju akcij v celoti izkoristijo. V tem delu raziskujemo, ali je mogoče iz notranjih večmodalnih značilk modela X-VLA obnoviti zaznavne informacije, pomembne za izvedbo naloge, ter jih ponovno uporabiti za izboljšanje vodenja robota, zlasti pri prostorskem sklepanju, ki je potrebno za natančno prijemanje tekstilnih objektov.

V tem delu predstavimo lahek dekodirnik vizualnega sklepanja, naučen za rekonstrukcijo ekspertno določenih zaznavnih ciljev iz notranjih predstavitev osnovnega modela X-VLA. Obnovljene informacije vključimo na dva načina: implicitno kot pomožni učni signal, ki osnovni model spodbuja k ohranjanju strukture, pomembne za izvedbo naloge, ter eksplicitno v obliki usmerjevalnih žetonov, ki jih ponovno uvedemo v proces generiranja akcij. Predlagane metode ovrednotimo na nalogah pobiranja kock, pobiranja in odlaganja blaga ter prepogibanja blaga. Izvirni model X-VLA primerjamo z različicami, ki bodisi regularizirajo njegove latentne predstavitve, da bi ohranile informacije, pomembne za izvedbo naloge, bodisi omogočijo, da so te informacije med generiranjem akcij eksplicitno dostopne. Rezultati kažejo, da lahko eksplicitno podajanje informacij, pomembnih za izvedbo naloge, izboljša delovanje modela v dvoumnih ali zahtevnih primerih. Pri ovrednotenih nalogah implicitno usmerjanje z regularizacijo latentnega prostora daje mešane rezultate, medtem ko eksplicitno usmerjanje prinaša izboljšave pri nekaterih nalogah manipulacije s tekstilom. Te ugotovitve kažejo, da osnovni modeli VLA vsebujejo zaznavno strukturo, pomembno za izvedbo naloge, ki jo je mogoče obnoviti, vendar je njena uporabnost odvisna od načina vključevanja teh informacij v proces generiranja akcij in od značilnosti manipulacijske naloge.

VLA | ViT | ujemanje toka | robotika | vizualno-jezikovno-akcijski modeli| večmodalno zaznavanje | robotska manipulacija

Vision-Language-Action (VLA) models are a class of end-to-end multimodal policies for robotic control. They jointly process visual observations, natural-language instructions, and proprioceptive robot state in order to predict actions for completing a commanded task. VLA models combine two fundamental components: a perceptual vision-language backbone, responsible for scene understanding and instruction interpretation, and an action-generation mechanism, responsible for mapping multimodal representations to executable robot commands. In the model used in this thesis, X-VLA [1], this action-generation mechanism is based on continuous action prediction via iterative refinement using flow matching.

VLA models are attractive for general robotic control because they provide a unified framework for perception, task understanding, and action generation. Language conditioning allows a single policy to represent multiple tasks, while vision-language pretraining can provide semantic priors that improve contextual understanding and transfer across objects, environments, and task formulations.

However, adapting pretrained multimodal representations to robotic action generation remains challenging. Robotic control requires temporally consistent and physically executable outputs, and robot datasets are typically much smaller and heterogeneous than web-scale image-language corpora. This makes it difficult to preserve useful pretrained representations while adapting the model to specific embodiments, tasks, and environments.

Key challenges include poor cross-embodiment transfer for action prediction, the gap between semantic understanding and low-level control, degradation of useful pretrained representation priors during finetuning, and sensitivity to environmental variation. This thesis focuses on the representation and grounding aspects of this problem: how pretrained visual-language representations can be preserved, aligned, and exploited during task adaptation to provide more grounded action conditioning.

Thesis Objective We investigate how structured perceptual information, derived from pretrained expert vision models, can be incorporated into vision-language-action (VLA) policies through learned visual guidance. Our central hypothesis is that the pretrained multimodal backbone of a VLA already encodes task-relevant information and perceptual structure that are not fully exploited during action generation. Recovering this structure and, potentially, reintroducing it as an explicit guidance signal may improve policy

performance by making the internal representations more task-aligned and the generated actions more spatially precise.

We train lightweight decoders that reconstruct the outputs of frozen expert vision models from the latent representations of a pretrained VLA (X-VLA) and compare two ways of exploiting the recovered structure: *implicit* guidance, where the decoding objective serves purely as auxiliary supervision that shapes the policy's representations, and *explicit* guidance, where the decoded information is additionally injected back into the action-generation module as a conditioning signal. We place *particular emphasis on robotic cloth manipulation*, a setting where deformable geometry, self-occlusion, and the need to localize semantically meaningful grasp points (such as cloth corners) make precise perceptual grounding especially demanding. All approaches are evaluated on a low-cost SO-ARM101 manipulator, on real-world cloth corner-to-box and cloth-folding tasks, as well as semantic cube pick-and-place, learned from small demonstration datasets.

Our contributions are: (i) lightweight decoders that transform a pretrained VLA's latent tokens into expert-aligned perceptual representations without additional annotation effort; and (ii) implicit and explicit guidance methods that exploit the decoders' outputs, as auxiliary supervision and as direct conditioning of the action-generation module, respectively, and evaluated on real hardware, centered on cloth manipulation. Our experiments show that guidance improves target identification beyond the effect of additional training alone, and that the benefit of explicit reintroduction depends strongly on the chosen injection mechanism.

1. Related Work on Visual Reasoning and Structured Guidance

As vision-language-action (VLA) models extend vision-language models by connecting visual and linguistic understanding with robot action generation, while they benefit from powerful pretrained representations, the spatial and perceptual information used for control often remains implicit within the backbone. Indeed, analyses of VLM spatial competence show that while these models encode substantial scene structure, spatial reasoning remains a weak point unless it is explicitly surfaced [2, 3].

Recent work therefore explores structured intermediate representations, including visual reasoning tokens, auxiliary spatial supervision, task-specific decoders, and affordance-based guidance. These methods aim to make task-relevant information more accessible, either by shaping internal representations or by directly conditioning action generation. This broader class of approaches motivates our investigation of implicit and explicit guidance strategies that leverage external or internally derived task-relevant information to improve VLA control.

Vision Foundation Features for Robot Learning Self-supervised vision transformers such as DINO [4] and DINOv2 [5] produce general-purpose features usable across diverse tasks without requiring specialized fine-tuning for encoding relevant semantic information. Their utility for control has motivated work that transfers such features into robot policies: Theia [6] distills vision foundation models, including DINOv2, into a compact robot-learning backbone, showing that preserving spatial feature structure improves downstream policy learning. These approaches inject expert structure *before or during* policy pretraining by shaping the visual encoder. In contrast, we ask whether comparable expert-aligned structure can be recovered from the frozen encoding pipeline of an already-trained VLA, and re-used as a guidance signal.

Chain-of-Visual-Thought Chain-of-Visual-Thought (CoVT) [7] introduces a small set of continuous visual-thought tokens trained to reconstruct dense supervision signals, such as depth and segmentation maps, leveraging external vision experts. This is achieved either by direct expert-feature alignment or by producing tokens that can query expert latent spaces for visual reconstruction. This mechanism allows the model to reason in a compact visual latent space, providing access to an intermediate perceptual structure that improves visually grounded reasoning.

SG-VLA and Structured Auxiliary Supervision SG-VLA [8] improves a VLA's backbone representations through dense auxiliary supervision by attaching task-specific decoders to shared visual-language features, and supervising them with structured spatial targets such as segmentation masks, object poses, and robot-state quantities. This auxiliary co-training yields a substantial improvement (from 0.60 to 0.73 average success on ManiSkill-HAB [9]), but only under a multi-stage progressive training schedule: naively co-training randomly initialized decoders with the backbone was found to *degrade* performance. In the paper, the approach relies on simulator-provided annotations and is validated only in simulation. It however suggests a clear research direction: pretrained vision-language backbones can be encouraged to expose richer internal perceptual structure when trained to produce explicit intermediate representations, and this aligned representation can improve policy control.

AffordanceVLA and Affordance-Conditioned Action Generation AffordanceVLA [10] introduces an intermediate affordance-generation module that predicts task-oriented tokens describing which object to manipulate, where to interact, and how to execute the interaction geometrically. These affordance tokens are passed to the action expert as manipulation priors, making the predicted structure action-coupled rather than only auxiliary supervision. A related approach conditions policies on other explicit intermediates: RT-Trajectory [11] conditions on coarse 2D trajectory sketches, enabling generalization to tasks that are not easily specified by simple language conditioning alone. These approaches operate at the *output* level, predicting a human-readable artifact that the policy then consumes; our explicit guidance variants are the *feature-level* analogue, conditioning action generation directly on recovered backbone representations.

Our work follows the same general motivation of using structured intermediate information to improve VLA control, but separates two roles of such structure. First, as in auxiliary-supervision approaches such as SG-VLA, we study whether training decoders to recover task-relevant perceptual signals from a VLA visual-language backbone regularizes the policy representation and improves action execution. Second, we investigate whether the recovered guidance tokens can be reintroduced into the action-generation pathway as explicit conditioning signals. How the new conditioning stream enters a network is highly consequential: feature-wise modulation and attention-based conditioning are known to behave differently from naive input concatenation [12], and recent flow-matching VLAs such as π_0 [13] condition the action expert through attention rather than straight input concatenation.

In this work we compare two such injection strategies within a flow-matching action generation module: *direct input concatenation of the guidance tokens*, and a *modulated scheme* in which guidance is presented to selected layers through attention. This allows us to compare structured guidance both as a representation-level training signal and as a direct input to the action generator.

2. Background on action generation via Flow Matching

2.1. General formulation and objective. A vision-language-action (VLA) model can be viewed as a conditional policy

$$\pi_\theta : (o_t, q, s_t) \mapsto A_t, \qquad A_t = [a_t, \dots, a_{t+H-1}] \in \mathbb{R}^{H \times D_a}, \tag{1}$$

where o_t is the visual observation, q is the language instruction, s_t is the proprioceptive robot state, A_t is an action chunk over a horizon of H steps, and θ denotes the learnable parameters of the policy. Each action $a_t \in \mathbb{R}^{D_a}$ may encode joint positions, end-effector pose, gripper state, or a combination of these.

VLA models can instantiate this conditional policy using different action-generation objectives. Autoregressive approaches discretize continuous actions into tokens and predict them sequentially, as in RT-1 [14], RT-2 [15], and OpenVLA [16]. Diffusion-based policies instead generate continuous action trajectories through iterative denoising, as in Diffusion Policy [17].

More recently, flow-matching policies learn a continuous vector field that transports noise directly toward the demonstrated action distribution, as in π_0 [13] and X-VLA [1]. Because this thesis is based on the *X-VLA-0.9B* [1] model, the remainder of this section focuses on flow matching for action generation.

Flow Matching for Action Generation. In flow-matching VLA policies, action generation is formulated as learning a conditional vector field v_θ that transports an initial noisy action chunk into an executable action trajectory. Starting from $A_t^0 \sim \mathcal{N}(0, I)$, the optimal-transport interpolation between noise and the ground-truth action chunk A_t is

$$A_t^\lambda = (1-\lambda)A_t^0 + \lambda A_t, \qquad \lambda \in [0,1]. \tag{2}$$

The flow matching objective is thus:

$$\mathcal{L}(\theta) = \mathbb{E}_{\lambda \sim \mathcal{U}(0,1),\, (o_t,q,s_t,A_t)\sim\mathcal{D},\, A_t^0 \sim \mathcal{N}(0,I)} \left[\left\| v_\theta(A_t^\lambda \mid o_t, q, s_t) - (A_t - A_t^0) \right\|^2 \right], \tag{3}$$

minimized over a dataset $\mathcal{D}$ of recorded task demonstrations, each providing the visual observations, language instruction, proprioceptive state, and ground-truth action chunk that define the regression targets. Intuitively, the model learns to predict, from an observation at a given timestep t, the chunk of future actions to be executed.

At inference time, an action chunk is generated by integrating the learned vector field from noise to data:

$$A_t^{\lambda+\Delta\lambda} = A_t^\lambda + v_\theta(A_t^\lambda \mid o_t, q, s_t)\Delta\lambda, \qquad \Delta\lambda = \frac{1}{K}, \tag{4}$$

where K is the number of Euler integration steps from $\lambda = 0$ to $\lambda = 1$. A_t^1 is used as the predicted action chunk.

This action-generation formulation is used in recent VLA architectures such as π_0 [13] and X-VLA [1]. In practice, the objective is often instantiated through a structured action loss. The convention used in X-VLA, which we follow in this thesis, represents action sequences as sequences of end-effector poses: each action encodes the 3D position and the orientation of the end effector in a fixed reference frame, together with the gripper state, expressed as a binary open/closed status. Importantly, to make orientation prediction a well-posed continuous regression target, orientations are expressed in the continuous 6D rotation representation [18]. Unlike Euler angles or quaternions, this representation is continuous over SO(3), avoiding the discontinuities that would obstruct direct regression of rotation targets. The total action loss is

$$\mathcal{L}_{\text{act}} = \mathcal{L}_{\text{pos}} + \mathcal{L}_{\text{rot6D}} + \mathcal{L}_{\text{grip}}, \tag{5}$$

where $\mathcal{L}_{\text{pos}}$ and $\mathcal{L}_{\text{rot6D}}$ are scaled mean-squared errors over the position and 6D rotation channels, and $\mathcal{L}_{\text{grip}}$ is a binary cross-entropy loss over the gripper channel; in our setup the gripper is encoded as a binary open/closed state.

2.2. The X-VLA Architecture. X-VLA is built around a Florence-style vision-language backbone [19] and a flow-matching action-generation module. In addition to the pretrained backbone, the architecture includes trainable components that condition the action module: domain-specific soft prompts and learned projection layers, which enable parameter-efficient adaptation for diverse tasks while preserving the pretrained multimodal representations.

Multimodal Contextualization. X-VLA receives visual observations, a language instruction, and the proprioceptive robot state. Throughout this section, all quantities refer to a single timestep. Let

$$I = \{I_1, I_2, \ldots, I_V\} \tag{6}$$

denote the V available camera views, where I_1 is the primary view and the remaining views are auxiliary (the X-VLA implementation supports up to $V = 3$ simultaneous views). All views are encoded by $\mathcal{E}^{\text{vis}}$, the shared DaViT-based visual tower [20] inherited from the Florence-style backbone [19]:

$$E_v = \mathcal{E}^{\text{vis}}(I_v) \in \mathbb{R}^{T_{\text{img}} \times D_{\text{VLM}}}, \qquad v = 1, \ldots, V. \tag{7}$$

Only the primary-view tokens E_1 are fused with language inside the VLM encoder. The instruction q is tokenized and embedded by the text encoder $\mathcal{E}^{\text{txt}}$, yielding the sequence

$$W = \mathcal{E}^{\text{txt}}(q) \in \mathbb{R}^{N_{\text{lang}} \times D_{\text{VLM}}}. \tag{8}$$

The multimodal encoder input is the concatenation

$$R^{(0)} = [E_1;\, W] \in \mathbb{R}^{(T_{\text{img}} + N_{\text{lang}}) \times D_{\text{VLM}}}, \tag{9}$$

where superscripts in parentheses index encoder layers. The input is processed by the L-layer Florence-style encoder stack $\Phi_\theta^{(\ell)}$,

$$R^{(\ell+1)} = \Phi_\theta^{(\ell)}\left(R^{(\ell)}\right), \qquad \ell = 0, \ldots, L-1. \tag{10}$$

The resulting contextualized multimodal representation is the final layer's output,

$$Z = R^{(L)} = [z_1, z_2, \ldots, z_N] \in \mathbb{R}^{N \times D_{\text{VLM}}}, \qquad N = T_{\text{img}} + N_{\text{lang}}, \tag{11}$$

and we write $Z = \Phi_\theta^{\text{VLM}}(I_1, q)$ to denote the full VLM encoding pipeline, from raw image and instruction to contextualized tokens.

Auxiliary-view handling. The visual tokens of the auxiliary views are not included in Z_t. Instead, they are preserved separately and later supplied to the action-generation module. For a given timestep t, the token sequences of all non-primary views (E_2, and E_3 when a third informative view is available) are concatenated along the token dimension into a single auxiliary sequence

$$U_t = [E_2;\, \ldots;\, E_V] \in \mathbb{R}^{N_{\text{aux}} \times D_{\text{VLM}}}, \qquad N_{\text{aux}} = (V-1)\, T_{\text{img}}. \tag{12}$$

In our setup, $V = 2$, so $U_t = E_2$. The implementation additionally receives an all-zero placeholder image in the remaining camera slot. This placeholder is passed through the visual encoder but contains no scene information for which is ignored in the equations.

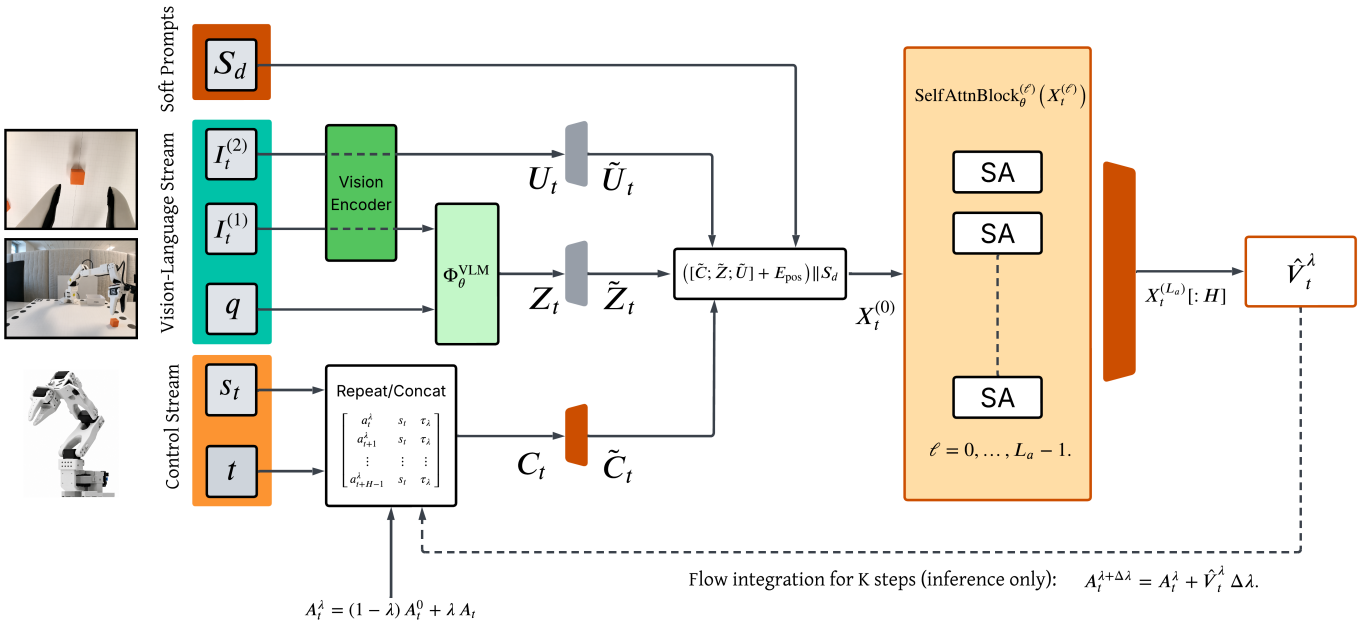


Figure 1. X-VLA action transformer. Three input streams are assembled into a single token sequence and fused by L_a bidirectional self-attention (SA) blocks. The main view $I^{(1)}$ and instruction q are encoded by the VLM $\Phi_\theta^{\mathrm{VLM}}$ into Z, while the auxiliary view $I^{(2)}$ passes only through the vision encoder (U). The figure shows the two-camera configuration (one main, one auxiliary view) used in our experimental setup. The control tokens $\tilde{C}$ combine the noisy action chunk $A^\lambda = (1-\lambda)A^0 + \lambda A$ with the repeated proprioceptive state s and flow-time embedding τ_λ. Learned positional embeddings E_{pos} are added to the control, vision–language, and auxiliary tokens only. The soft-prompt tokens S_d are appended afterwards and carry no positional information. After the final block, only the H control-token positions are decoded into the velocity prediction $\hat{V}^\lambda$, which at inference time is integrated for K steps (dashed loop) to transport Gaussian noise to an action chunk. Orange components (S_d, P_d^{ctrl}, P_d^{out}) are domain-specific in the original X-VLA formulation, i.e. instantiated per embodiment, while gray projections are shared. Since we train the full model for a single target setup, we treat them all projections as ordinary trainable parameters of the policy rather than a separate adaptation mechanism. Subscripts t in the figure denote the environment timestep.

Action Transformer. The X-VLA action module is a stack of L_a standard self-attention Transformer blocks that predicts the flow-matching target for a noisy action chunk, conditioned on the multimodal context. All conditioning is performed purely through the input sequence: there is no cross-attention involved. The complete architecture and information flow are illustrated in Fig. 1. Throughout, $\lambda \in [0, 1]$ denotes the flow-matching time of the denoising process, not the environment timestep. All quantities refer to a single control step t, whose index we omit.

The module receives four kinds of inputs. The contextualized vision–language tokens Z (main view and instruction, Eq. 11) and the auxiliary-view tokens U (Eq. 12) are each mapped into the hidden dimension D_h by a shared linear projection. The noisy action chunk $A^\lambda \in \mathbb{R}^{H \times D_a}$ is fused with the proprioceptive state s and a sinusoidal embedding τ_λ of the flow time, both repeated along the action horizon, and projected by a *domain-specific* linear layer:

$$\tilde{C} = P_d^{\mathrm{ctrl}}\big([\,A^\lambda;\ \mathrm{repeat}(s, H);\ \mathrm{repeat}(\tau_\lambda, H)\,]\big) \in \mathbb{R}^{H \times D_h}. \tag{13}$$

Learned positional embeddings are added to the concatenation of control, vision–language, and auxiliary-view tokens; the M_S domain-specific soft-prompt tokens S_d are appended afterwards and thus carry no positional information. The input to the first block is

$$X^{(0)} = \mathrm{concat}\Big(\big[\tilde{C}; \tilde{Z}; \tilde{U}\big] + E_{\mathrm{pos}},\ S_d\Big). \tag{14}$$

After L_a blocks of bidirectional self-attention over the full sequence, only the hidden states at the H control-token positions are decoded by a domain-specific output projection:

$$\hat{V}^\lambda = P_d^{\mathrm{out}}\big(X^{(L_a)}[:H]\big) \in \mathbb{R}^{H \times D_a}. \tag{15}$$

Thus, the action transformer implements the conditional velocity predictor

$$\hat{V}^\lambda = v_\theta\left(A^\lambda \mid Z,\, U,\, s,\, \lambda,\, S_d\right). \tag{16}$$

3. Visual-Thought Guidance

The vision-language backbone of a VLA model can be viewed as a general source of multimodal representations that encode task-relevant visual and linguistic information. Lightweight trainable modules can be introduced to recover and reorganize this information into a more structured form. The recovered structure can then serve two complementary purposes: it can then provide auxiliary supervision that regularizes the

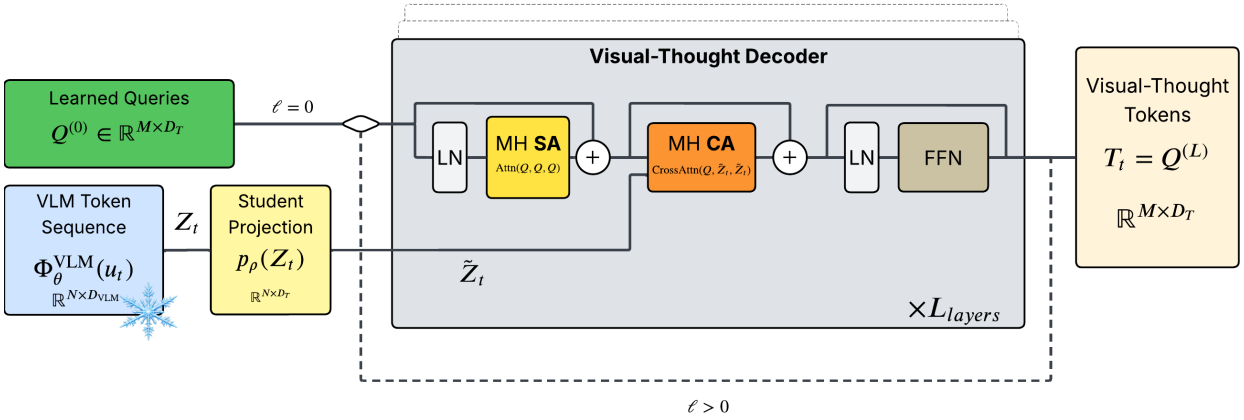


Figure 2. Visual-thought decoder architecture. The VLM token sequence Z_t is projected into the decoder dimension. The query tokens pass through stacked decoder blocks with query self-attention, residual cross-attention, and an FFN refinement. The final output T_t is the visual-thought token sequence. The VLM encoder $\Phi_\theta^{\mathrm{VLM}}$ can remain fixed during training, or be affected by the guidance loss gradients, depending on the objective and training stage.

backbone and encourages the preservation of task-relevant information, or it can be reintroduced into the action-generation pathway as an explicit guidance signal. This formulation is not tied to X-VLA and can, in principle, be applied to any VLA architecture that provides access to suitable intermediate multimodal features. In this work, we explore this idea using the X-VLA multimodal representation defined in Eq. 11, which is constructed from the primary camera view $I_t^{(1)}$ and the language instruction ℓ.

3.1. The Visual-Thought Decoder. We introduce a lightweight trainable module built on top of the VLM encoder that recovers and reorganizes selected task-relevant information from its vision-language features into a compact sequence of latent tokens, referred to as the *visual-thought token sequence*. Depending on the training formulation, these tokens are used either to regularize the learned representation through auxiliary supervision or as an explicit source of guidance for action generation (see Figure 2).

Formally, let f_ψ denote the visual-thought decoder with trainable parameters ψ. It maps the encoded representation into a learned token sequence

$$T_t = f_\psi(Z_t) = [\tau_{t,1}, \tau_{t,2}, \ldots, \tau_{t,M}] \in \mathbb{R}^{M \times D_T}, \tag{17}$$

where M is the number of visual-thought tokens and D_T is their hidden dimension.

First, the VLM tokens are mapped into the decoder hidden dimension D_T using a trainable residual two-layer MLP projection:

$$\tilde{Z}_t = p_\rho(Z_t), \qquad \tilde{Z}_t \in \mathbb{R}^{N \times D_T}. \tag{18}$$

The decoder maintains a learned bank of M visual-thought query tokens:

$$Q^{(0)} = \left[q_1^{(0)}, \ldots, q_M^{(0)}\right] \in \mathbb{R}^{M \times D_T}, \tag{19}$$

which initialize an L-layer query-refinement decoding process. At each decoder layer:

1. The visual-thought queries first interact through pre-normalized self-attention:

$$\bar{Q}^{(\ell)} = Q^{(\ell-1)} + \mathrm{SelfAttn}\left(\mathrm{LN}(Q^{(\ell-1)})\right). \tag{20}$$

2. The updated queries then attend over the projected VLM token sequence and incorporate the resulting cross-modal information through a residual connection:

$$R^{(\ell)} = \bar{Q}^{(\ell)} + \mathrm{CrossAttn}\left(\bar{Q}^{(\ell)}, \tilde{Z}_t, \tilde{Z}_t\right). \tag{21}$$

3. Finally, the queries are refined through a pre-normalized Transformer-style feed-forward block:

$$Q^{(\ell)} = R^{(\ell)} + \mathrm{FFN}\left(\mathrm{LN}(R^{(\ell)})\right). \tag{22}$$

After L stacked decoder layers, we obtain the final visual-thought token sequence (defined in Eq. 17).

The number of visual-thought tokens M depends on the downstream task and will be discussed further in later sections.

3.2. Expert Information from Visual-Thought Tokens. The visual-thought sequence T_t is trained to expose structured expert information through a task-specific decoder head. Given T_t, the head predicts an expert-derived output

$$\hat{Y}_t = g_\eta(T_t), \tag{23}$$

where g_η is a trainable module and $\hat{Y}_t$ denotes the recovered guidance signal.

The predicted output is supervised against an expert target $Y_t^\star$, which may correspond to dense spatial maps, patch-level feature targets, or other grounded perceptual structures. The generic guidance objective is

$$\mathcal{L}_{\text{guide}} = \mathcal{D}\left(\hat{Y}_t, Y_t^\star\right), \tag{24}$$

where $\mathcal{D}(\cdot,\cdot)$ is the discrepancy measure associated with the supervision type.

When multiple guidance targets are used, the objective becomes

$$\mathcal{L}_{\text{guide}} = \sum_{k=1}^{K} \lambda_k \mathcal{D}_k\left(\hat{Y}_t^{(k)}, Y_t^{(k)\star}\right), \tag{25}$$

where each term corresponds to one expert signal, $\mathcal{D}_k$ is its associated loss, and λ_k controls its relative weight.

3.3. Training Objectives for Visual-Thought Decoder Training.

Representative Feature Alignment Following SG-VLA [8], the objective is to encourage the VLA backbone to preserve semantically meaningful information by training the visual-thought tokens to reproduce structured patch-level semantic representations from a strong external vision expert, for which we use DINOv2 ViT-B/14 [5]. Given image I_t, let

$$E_t = [e_{t,1}, \ldots, e_{t,N_{\text{DINO}}}] \in \mathbb{R}^{N_{\text{DINO}} \times D_{\text{DINO}}}$$

denote the DINOv2 patch-token sequence, after discarding the global classification token. For this objective, the visual-thought output space is matched to the teacher token space:

$$M = N_{\text{DINO}}, \qquad D_T = D_{\text{DINO}}.$$

No additional prediction head is used; the visual-thought decoder output is compared directly to the teacher sequence:

$$\hat{E}_t = T_t = f_\psi(Z_t) \in \mathbb{R}^{N_{\text{DINO}} \times D_{\text{DINO}}}.$$

The feature-alignment loss is

$$\mathcal{L}_{\text{feat}} = \frac{1}{N_{\text{DINO}} D_{\text{DINO}}} \sum_{i=1}^{N_{\text{DINO}}} \sum_{j=1}^{D_{\text{DINO}}} \left(\hat{e}_{t,i,j} - e_{t,i,j}\right)^2, \tag{26}$$

i.e., the mean squared error between the predicted and teacher features, averaged over all N_{DINO} patch tokens and D_{DINO} feature channels.

This objective encourages the visual-thought sequence to retain patch-level visual structure from DINOv2, providing a semantically rich representation-shaping signal beyond task-specific dense reconstruction alone.

Dense Spatial Reconstruction We explore dense spatial reconstruction of CeDiRNet [21] center-direction maps, which point towards identified cloth corners. The objective encourages the visual-thought tokens to preserve cloth-relevant spatial structure by fully reconstructing these dense outputs, and constitutes the main task-specific supervision used to shape the visual-thought workspace for cloth manipulation.

Let the CeDiRNet teacher be denoted by Ψ_ϕ^{CED}. Given an image I_t, the teacher produces several dense output fields at resolution $H_{\text{CED}} \times W_{\text{CED}}$; of these, we use only the two direction fields (the sine and cosine components of the predicted center direction), which suffice for cloth-corner localization, and discard the remaining outputs. The supervision target is therefore the two-channel map

$$Y_t^{\text{CED}} = \Psi_\phi^{\text{CED}}(I_t) \in \mathbb{R}^{C_{\text{CED}} \times H_{\text{CED}} \times W_{\text{CED}}}, \qquad C_{\text{CED}} = 2.$$

On the prediction side, the visual-thought decoder output $T_t = f_\psi(Z_t) \in \mathbb{R}^{M \times D_T}$, with $M = H_g W_g$ tokens arranged on a spatial grid, is linearly projected and reshaped into a latent feature map of size $D_H \times H_g \times W_g$. A single dense multi-channel head g_ω then jointly predicts both direction channels at latent-grid resolution, and the result is bilinearly upsampled to the teacher resolution, yielding the prediction $\hat{Y}_t^{\text{CED}} \in \mathbb{R}^{C_{\text{CED}} \times H_{\text{CED}} \times W_{\text{CED}}}$. The dense reconstruction loss is the mean squared error over all channels and spatial positions:

$$\mathcal{L}_{\text{CED}} = \frac{1}{C_{\text{CED}} H_{\text{CED}} W_{\text{CED}}} \sum_{c=1}^{C_{\text{CED}}} \sum_{u=1}^{H_{\text{CED}}} \sum_{v=1}^{W_{\text{CED}}} \left(\hat{Y}_{t,c,u,v}^{\text{CED}} - Y_{t,c,u,v}^{\text{CED}}\right)^2. \tag{27}$$

4. Guidance Integration into the VLA Policy

As visual-thought tokens are learned representations trained to recover expert-derived perceptual structure from the VLA backbone, we now introduce how these representations can be integrated into the policy. The central question is whether expert-aligned visual-thought information should only shape the representation during training, or whether it should also be exposed directly to the action-generation module.

We therefore distinguish two integration regimes: an *implicit exploitation* regime, where visual-thought supervision is used only as an auxiliary objective during action learning, and an *explicit exploitation* regime, where the learned visual-thought representation is injected into the action-conditioning pathway, allowing it to influence the predicted action chunk directly.

4.1. Implicit Exploitation. Implicit exploitation uses expert supervision only as a training-time objective to shape the model representation, without exposing expert-aligned signals directly to the action-generation module. In the spirit of SG-VLA [8], this is implemented by training the policy together with an attached visual-thought module to reconstruct expert information from the VLA representation while leaving the action-conditioning interface unchanged.

The objective is to encourage the VLM to preserve enough structure in its internal features for expert-derived information to remain recoverable, including geometric cues such as depth or segmentation maps, as well as representative semantic structure such as DINOv2 [5] tokens. In this way, the VLM representation is regularized toward a more informative and better aligned conditioning space, reducing the likelihood that task-relevant perceptual structure is lost during training.

In this work, we consider two expert-supervised visual-thought settings: one aligned to DINOv2 patch-level representations, and one aligned to CeDiRNet grasp-direction maps. In both cases, the expert signal is used only as supervision for an auxiliary reconstruction objective, while the policy continues to use the original action-conditioning interface. Concretely, training proceeds in two stages: the VLA is first pretrained for the target task, after which the visual-thought decoder is trained on top of the frozen VLM representation, and finally the full architecture is optimized jointly under both the action objective and the expert-guidance objective. The corresponding optimization objectives are specified in Section 6.

4.2. Explicit Exploitation. Explicit exploitation refers to exposing the learned visual-thought representation directly to the action-generation module. Unlike implicit exploitation, in which the guidance objective only regularizes the learned representation during training, explicit exploitation allows the expert-aligned information to condition action prediction directly.

Notably, at inference time, the policy does not receive raw or projected outputs from an external expert. Instead, it uses visual-thought tokens produced by a decoder trained through feature-level distillation during implicit training. This allows the policy to exploit model-native, expert-aligned structure without requiring the external expert during deployment.

Let

$$G_t = f_{\text{extract}}(Z_t) \tag{28}$$

denote the expert-aligned guidance representation extracted from the contextualized VLM sequence Z_t. In the explicit regime, this representation is fused with the native X-VLA action-conditioning sequence to form a guided input

$$X_{t,g}^{(0)} = F_{\text{fuse}}\left(X_t^{(0)},\, G_t\right), \tag{29}$$

which replaces the unguided conditioning sequence at the input of the action transformer. The concrete explicit-guidance architecture is described and visualized in Section 5, while the corresponding optimization objectives are introduced in Section 6.

5. Explicit Guidance Architecture

Having defined the explicit exploitation regime, we now specify the architecture used to expose visual-thought guidance to the flow matching action module. We first describe the native X-VLA action-conditioning sequence, then the fusion design space considered for explicit guidance exploitation, and finally the concrete guidance instantiation used in this work.

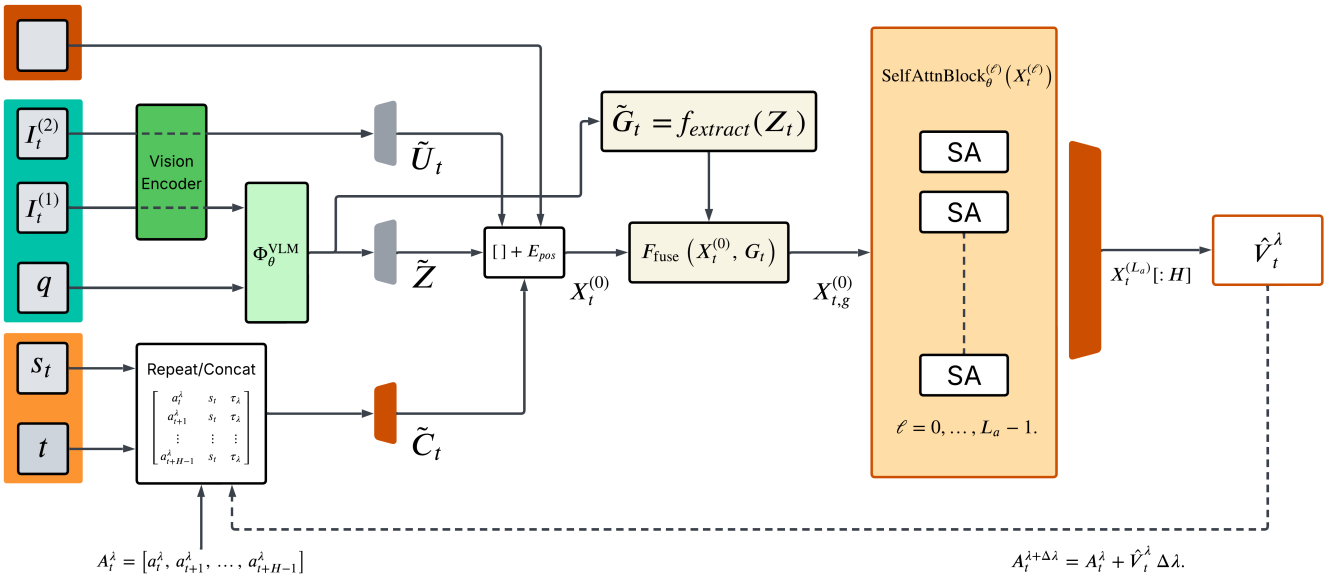


Figure 3. Late-fusion stage in the guided X-VLA pipeline. A visual-thought decoder extracts a guidance representation from the contextualized VLM features, which is then fused with the native action-conditioning sequence before being processed by the action transformer.

5.1. Native X-VLA Action Conditioning. Under the standard X-VLA pathway, the action transformer uses the native conditioning sequence $X_t^{(0)}$ defined in Eq. 14. This sequence contains the projected control tokens $\tilde{C}_t$, the projected contextualized multimodal tokens $\tilde{Z}_t$, the projected auxiliary-view tokens $\tilde{U}_t$, and the domain-specific soft prompts S_d.

Equivalently, the native action-conditioning pathway can be summarized as

$$X_t^{(0)} = \mathcal{C}_{\mathrm{native}}\left(\tilde{C}_t,\, \tilde{Z}_t,\, \tilde{U}_t,\, S_d\right), \tag{30}$$

where the exact sequence construction, including positional encoding of the non-prompt tokens and subsequent prompt concatenation, is given in Eq. 14.

5.2. Fusion Design Space. When using explicit fusion of the guidance representation, the remaining design question is how the extracted visual-thought representation should be combined with the native action-conditioning sequence. The goal is to expose expert-aligned guidance to the action transformer while controlling how strongly it modifies the original conditioning stream.

We consider two families of fusion operators: *concatenative fusion*, which appends guidance tokens to the action-transformer input sequence; and *layer-selective fusion*, which appends the guidance tokens only at a chosen subset of transformer blocks.

Let

$$\tilde{G}_t = P_{\mathrm{g}}(G_t) \tag{31}$$

denote the projection of the guidance representation into the action-transformer hidden dimension D_h. The guidance information is then combined with the native conditioning sequence through a fusion operator $F_{\mathrm{fuse}}(X_t^{(0)}, G_t)$, whose specific forms are described in the following subsections. Figure 3 illustrates the stage at which this fusion occurs in the X-VLA pipeline.

5.2.1. Concatenative Fusion. Concatenative fusion is the most direct way of incorporating visual-thought guidance into the action-conditioning framework. We concatenate the projected guidance representation after the action-conditioning control tokens and before the native VLM visual tokens, yielding

$$X_{t,g}^{(0)} = \left[\tilde{C}_t;\, \tilde{G}_t;\, \tilde{Z}_t;\, \tilde{U}_t;\, S_d\right]. \tag{32}$$

We further explore both ungated and gated guidance insertion. In the ungated case, the projected guidance tokens are injected directly into the action-transformer input. In the gated case, a scalar relevance weight is predicted from pooled native visual features and pooled guidance features,

$$\alpha_t = \sigma\left(h_{\mathrm{gate}}\left(\mathrm{Pool}([\tilde{Z}_t; \tilde{U}_t]), \mathrm{Pool}(\tilde{G}_t)\right)\right), \tag{33}$$

and used to scale the inserted guidance tokens:

$$X_{t,g}^{(0)} = \left[\tilde{C}_t;\, \alpha_t \tilde{G}_t;\, \tilde{Z}_t;\, \tilde{U}_t;\, S_d\right]. \tag{34}$$

This gives a simple way to compare unconditional guidance exposure against a learned mechanism that modulates how strongly the expert-derived representation enters the policy token stream.

The main advantage of concatenative fusion is its simplicity and generality. The policy receives direct token-level access to the guidance representation throughout the entire action-transformer stack. Its main limitation is that the sequence length is increased for every transformer block, so the policy must learn how strongly and where to use the inserted guidance.

5.2.2. Selected-Layer Fusion. Selected-layer fusion exposes guidance only at a chosen subset of transformer blocks, rather than throughout the full action-transformer stack. The native conditioning input is first formed as usual without guidance (Eq. 14) and processed normally by the action transformer. Let $\mathcal{I}_{\text{sel}}$ denote the set of selected transformer-block indices. For each block $\ell \in \mathcal{I}_{\text{sel}}$, the projected guidance tokens are inserted temporarily after the action-conditioning tokens and before the native visual tokens, yielding

$$X_{t,g}^{(\ell)} = \left[\tilde{C}_t^{(\ell)};\, \tilde{G}_t;\, \tilde{Z}_t^{(\ell)};\, \tilde{U}_t^{(\ell)};\, S_d^{(\ell)}\right]. \tag{35}$$

The selected block is then applied to this expanded sequence, allowing the native policy tokens and the injected guidance tokens to interact through the block's standard self-attention and feedforward operations. However, the guidance tokens are not propagated as a permanent part of the sequence. After the selected block is evaluated, only the updated native token streams are retained, while the temporary guidance token segment is removed. Thus, guidance affects only the designated subset of transformer layers, while the remaining layers operate on the native action-conditioning stream.

Similarly as in the concatenative version, within this selected-layer formulation, we explore both ungated and gated guidance insertion into the selected blocks (see equations 33 and 34).

The main potential advantage of selected-layer fusion is that it localizes the intervention: guidance is made available only at chosen depths, reducing the structural disturbance to the pretrained action-transformer stack. Its main limitation is that guidance is not persistently present across all layers, so the effectiveness of the method depends on selecting transformer depths at which the additional information is most useful.

5.3. Dense Spatial-Workspace Guidance. In this work, the explicit guidance representation is derived from the CeDiRNet-based visual-thought workspace. Although the supervision target is dense, the injected representation is the decoder token grid rather than the final reconstructed dense map:

$$G_t \in \mathbb{R}^{M_g \times D_g}, \qquad M_g = H_g W_g.$$

These tokens are projected into the action-transformer hidden space as

$$\tilde{G}_t = P_{\text{g}}(G_t) \in \mathbb{R}^{M_g \times D_h},$$

where P_{g} denotes the learned guidance projection. The resulting projected guidance tokens are then included in the conditioning sequence using one of the operators defined in Section 5.2.

6. Training Objectives

This section specifies the optimization objectives used across the integration settings considered in this work.

6.1. Base Policy Objective. All policy variants retain the X-VLA action objective described in Section 2 and defined in Eq. 5. It consists of scaled mean-squared error terms for end-effector position and 6D rotation, together with a binary cross-entropy term for the gripper state. We denote this objective by $\mathcal{L}_{\text{act}}$.

The unguided baseline is optimized using only $\mathcal{L}_{\text{act}}$. In the guided variants, this remains the primary policy objective and is combined with the visual-thought supervision objectives defined below.

6.2. Expert-Aligned Guidance Objective. Let $\mathcal{L}_{\text{guide}}$ denote the task-specific supervision objective used to train the visual-thought decoder and its associated guidance head. Its exact form depends on the expert signal considered:

$$\mathcal{L}_{\text{guide}} \in \{\mathcal{L}_{\text{feat}}, \mathcal{L}_{\text{CED}}\}, \tag{36}$$

where $\mathcal{L}_{\text{feat}}$ and $\mathcal{L}_{\text{CED}}$ were defined in Section 3.3 as the losses of representative-feature alignment and dense spatial reconstruction tasks, respectively.

This objective is responsible for shaping the decoded visual-thought representation to preserve expert-aligned information, independently of whether that representation is later exposed explicitly to the policy.

6.3. Objectives by Integration Regime. The final optimization objective depends on how guidance is used.

Unguided baseline. For the standard X-VLA baseline, no visual-thought module is used and the optimization reduces to

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{act}}. \tag{37}$$

Implicit exploitation. In the implicit setting, the visual-thought module is trained under expert supervision, but its output is not exposed to the action-generation module. Instead, the expert objective acts only as a representation-shaping signal. Starting from a X-VLA model pretrained for a given task under its basic objective, training proceeds in two more stages:

First, the VLM backbone is frozen and only the visual-thought decoder and guidance head are optimized:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{guide}}. \tag{38}$$

Secondly, the X-VLA representation and the visual-thought module are optimized jointly under both the action objective and the expert-guidance objective:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{act}} + \lambda_{\text{guide}}\mathcal{L}_{\text{guide}}. \tag{39}$$

In this regime, the expert objective regularizes the learned representation, but does not modify the action-conditioning interface seen by the policy.

Explicit exploitation. In the explicit setting, the learned visual-thought representation is fused into the native conditioning stream and therefore affects action prediction directly.

The procedure starts with complete implicit exploitation training, allowing the visual-thought decoder to learn an expert-aligned guidance representation from the X-VLA latent space.

After the implicit-guidance procedure, explicit conditioning is introduced in a subsequent stage, where the learned guidance representation is fused into the action-conditioning input and the guided policy is further optimized.

During this stage, the VLM backbone remains frozen, while the action-generation module and fusion components are optimized to exploit the learned guidance representation. The intention is to isolate the effect of explicit guidance exposure at the policy level, without further modifying the underlying multimodal representation, which is assumed to be correct and representative at this stage.

The total objective is then

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{act}} + \lambda_{\text{guide}}\mathcal{L}_{\text{guide}}, \tag{40}$$

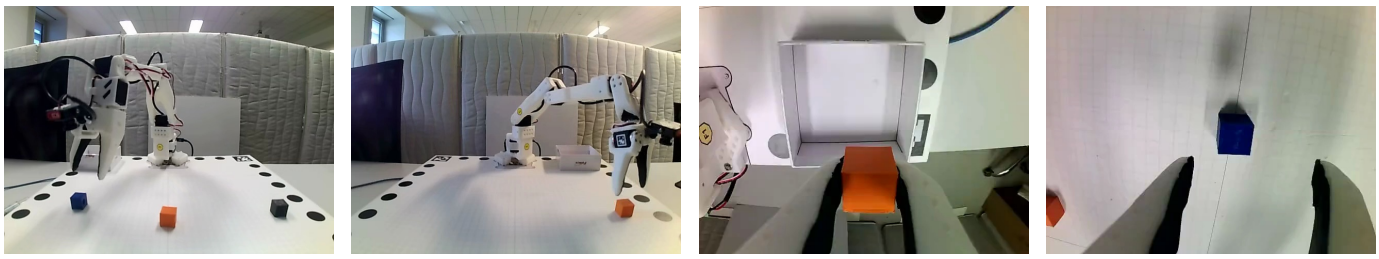
where the action loss is evaluated under the guided conditioning input $X_{t,g}^{(0)}$.

7. Experimental Setup

We evaluate the proposed visual-thought decoder and the integration strategies described in Section 4 across two task families. Cube pick-and-place tasks provide a controlled setting for analyzing spatial generalization and assessing whether the X-VLA policy preserves task-relevant semantic information. Cloth-manipulation tasks constitute the primary focus of the study, as they introduce the additional challenges associated with deformable-object control. Successful execution requires accurate identification of the cloth structure, particularly its corners, making these tasks well suited for evaluating explicit guidance based on CeDiRNet-derived information. The detailed descriptions of each task are discussed in Subsection 7.1.

Our physical setup for policy learning consists of a single SO-ARM101 manipulator positioned at a fixed location within a white workspace, as shown in Figure 4. Two wide-angle cameras are set. The first one is facing the robot and provides the primary view, capturing the entire manipulator and a large portion of the workspace. The second camera is mounted on the robot's wrist and provides an auxiliary close-range view of the gripper tip and its immediate surroundings. In the datasets used in this thesis, both camera streams are RGB videos at 640×480 resolution and 30 fps. The dataset versions used for training are resampled and stored at 7.5 Hz, matching the standard X-VLA action horizon of 30 predicted actions over 4 seconds of real robot motion.

Depending on the task, the workspace contains either one or more wooden cubes used as manipulation targets in the cube pick-and-place experiments, or a square gray cloth together with cardboard boxes that define the target placement regions. This is shown in each task's specifications in figures 5, 6, and 7.



(a) Main-camera view. **(b)** Main-camera view. **(c)** Wrist-camera view. **(d)** Wrist-camera view.

Figure 4. Representative observations from cube-manipulation demonstrations. The first two images show examples captured by the primary view camera, while the last two show the corresponding wrist-mounted camera view.

Type	Task (dataset)	Instruction pattern	Ep.	Avg. dur. (s)
Cube	`pickplace-orange`	"Pick up orange cube and place inside white box"	196	22
	`pickplace-multicolor`	"Pick up *{color}* cube and place inside white box"	184	30
Cloth	`cloth-corner-box`	"Pick the cloth by a visible corner and drop the cloth into the box"	70	30
	`cloth-corner-fold`	"Pick up a visible corner of the cloth and fold it over to the opposite side"	50	26

Table 1. Overview of the demonstration datasets used for policy training. For each task we report the dataset identifier, the language instruction pattern provided to the policy, the number of recorded demonstration episodes (Ep.), and the average duration of a single episode in seconds (Avg. dur.). In the multicolor dataset, *{color}* is instantiated per episode with the commanded cube color (orange, blue, or black). The two cloth tasks share the same physical setup but differ in the commanded post-grasp behavior (transport to a box vs. folding).

7.1. Tasks and Datasets. Table 1 summarizes the datasets and task families used throughout the study. All datasets were recorded on the SO-ARM101 manipulator using the LeRobot dataset recording pipeline [22]. The `pickplace-orange` dataset is used during the initial adaptation stage of the base model before finetuning for each task evaluated in this thesis.

These task datasets are used both for policy learning and for decoder distillation. We conduct and report decoder-distillation runs across the task settings to evaluate reconstruction fidelity. These diagnostic runs should not be interpreted as every decoder being used in downstream policy evaluation: the selected policies use DINOv2-aligned guidance for *Multi-Color Cube Semantic Generalization* (`pickplace-multicolor`) and CeDiRNet-aligned guidance for *Cloth Corner-to-Box* (`cloth-corner-box`) and *Cloth Corner-Folding* (`cloth-corner-fold`). The joint CeDiRNet+DINOv2 configurations are retained as optimization-sweep comparisons but are not selected for final policy configurations. During decoder distillation, the pretrained X-VLA backbone remains frozen and supervision is obtained from expert predictions computed during optimization on the corresponding task frames.

Across the study, training and validation are conducted under fixed task-specific data partitions. Accordingly, the decoder-distillation results should be understood as being obtained under the same general task-specific data regime as the subsequent policy-learning experiments.

7.1.1. Single-Color Cube Generalization. This setting considers a single orange cube. Its purpose is to test the policy's spatial generalization capabilities by testing pickup locations that were not part of the training data while still operating within the same known spatial distribution. Four evaluation positions are defined for the orange cube, all of which are excluded from the training positions (Figure 5a). The cube is placed separately at each of these four held-out positions and for every position, four pickup trials are performed.

Specialized dataset We use the self-collected `pickplace-multicolor` dataset, which contains 184 pick-and-place demonstrations involving blue, orange, and black cubes. The robot starts from a resting position, approaches the target with a vertical end effector pose, grasps, and transports to a white box that is placed either left or right of the manipulator. After placement, robot is sent back to the resting position. In each demonstration, the language instruction specifies the target cube color accordingly (Table 1).

Success criteria. Evaluation is based on two outcomes: (i) successful target acquisition and (ii) successful drop. Successful target acquisition indicates that the policy reaches

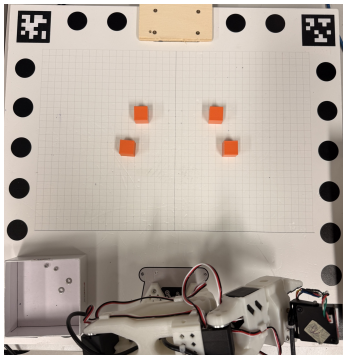
and securely lifts the cube from the evaluation position. Successful drop indicates that the cube is subsequently placed inside the target box.

7.1.2. Multi-Color Cube Semantic Generalization. This setting is designed to evaluate whether the policy preserves color-conditioned semantic reasoning after task adaptation. The instruction always follows the same template: *"Pick up {color} cube and place inside white box."* The semantic evaluation is formulated as a binary decision problem. Each scene contains exactly two cubes, one placed on the left side of the workspace and one on the right side, and the policy must identify, approach, and grasp the cube whose color matches the language command. This setting is visualized in Figure 5b.

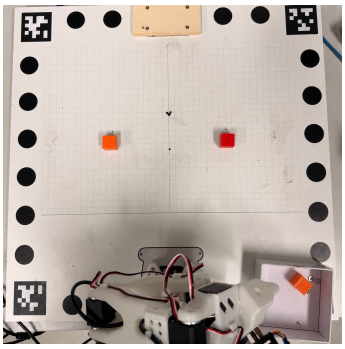
In order to test color-based semantic reasoning, as well as semantic representation preservation after finetuning, three color-pair families are evaluated: (i) two colors seen during training, orange and blue; (ii) one seen and one unseen color, orange and red; and (iii) two unseen colors, red and yellow. For each pair family, both colors are used as commanded targets. Cube placement is balanced left-right across trials, so that each commanded color is tested equally often from both sides of the workspace. Four grasp attempts are performed for each side assignment, resulting in eight evaluation attempts per commanded-color.

Dataset details. The same dataset as for *Single-Color Cube Generalization* (`pickplace-multicolor`).

Success criteria. Evaluation is based on two outcomes: (i) successful target identification and (ii) successful grasping. Successful target identification records whether the policy selects and attempts to grasp the cube whose color matches the instruction. Successful grasping records whether that commanded cube is then securely grasped.



(a) Single-color cube generalization. A single cube is positioned in each of the four visualized points at a time.



(b) Multi-color cube semantic generalization binary positioning example. The orange cube is seen during training while the red cube is not.

Figure 5. Evaluation layouts for the cube-manipulation tasks.

7.1.3. Cloth Corner-to-Box. In this setting, the policy must pick up the cloth by a visible corner and drop it into a box. The task emphasizes grasp-point selection via corner localization. It is used as one of the main benchmarks for evaluating whether explicit spatial guidance improves downstream control for precise cloth-grasping maneuvers. Four evaluation cloth configurations are defined for this task and are shown in Figure 6.

Dataset details. We use the self-collected `cloth-corner-box` dataset, in which each demonstration consists of grasping the cloth by the visible corner closest to the primary camera, lifting it, and placing it into a box located on the left side of the workspace. Similarly to the cube dataset, the robot starts and returns to a common resting position after task completion. This dataset contains 70 demonstrations.

Success criteria. Evaluation is based on three outcomes: (i) target identification, (ii) grasp success score, and (iii) successful placement. Target identification records whether the policy selected the correct visible cloth corner as the intended grasp target. The grasp success score measures the spatial accuracy of the executed grasp with respect to the targeted corner: a score of 1.0 is assigned if the grasp is within 2 cm of the target corner, 0.5 if the grasp is between 2 and 5 cm from the target corner, and 0 if the cloth is grasped at another position or not grasped. Successful placement records whether the cloth is deposited inside the target placement area.

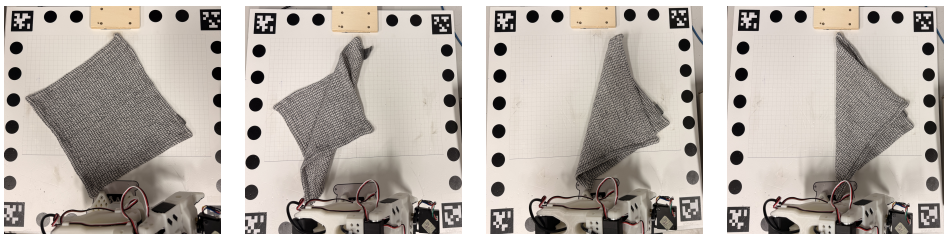


Figure 6. Evaluation positions for *Cloth Corner-to-Box*. Each position is attempted four times, resulting in 16 total attempts.

7.1.4. Cloth Corner Folding. The policy must pick up a visible corner and fold the cloth toward the opposite side, matching corners. This task places stronger demands on structured spatial reasoning over the cloth geometry from grasp to placement. It is therefore particularly relevant for evaluating the usefulness of dense visual-thought guidance. The four evaluation configurations used for this task are shown in Figure 7.

Dataset details. We use the self-collected `cloth-corner-fold` dataset. It contains 50 demonstrations in which the cloth is grasped by the visible corner closest to the primary camera and folded by placing the grasped corner onto the opposite corner, producing a triangular folded shape. Each demonstration begins and ends with the robot in a common resting position.

Success criteria. Evaluation is based on three outcomes: (i) target identification, (ii) grasp success score, and (iii) fold completion score. Target identification records whether the policy successfuly selected a visible cloth corner as the intended target for attempting the grasp. The grasp score measures the spatial accuracy of the executed grasp with respect to the target corner: a score of 1.0 is assigned if the grasp is within 2 cm of the target corner, 0.5 if the grasp is between 2 and 5 cm from the target corner, and 0 if the cloth is grasped at another position or not grasped. The fold completion score indicates whether the grasped corner is brought toward the opposite side corner, following the same distance-based scoring than that of the grasping.

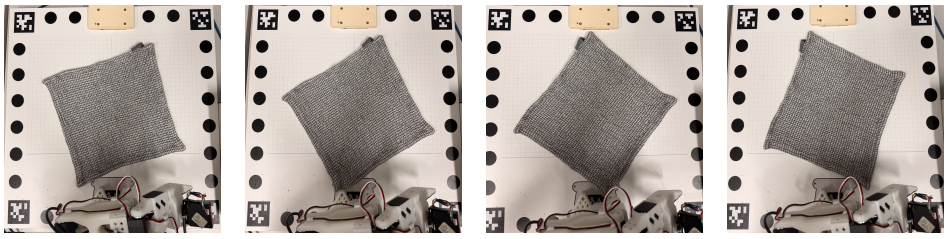


Figure 7. Evaluation positions for *Cloth Corner-Folding*. Each position is attempted four times, resulting in 16 total attempts.

7.2. Experimental Regimes. We consider four experimental regimes with different objectives and trainable components which isolate complementary questions about the role of visual-thought guidance: how the policy performs in its standard configuration, and how the extraction and reinjection of visual-thought tokens affect its performance. Their corresponding training objectives are detailed in Section 6.

Base X-VLA Policy Training This regime provides the unguided X-VLA reference used throughout the empirical study.

We use the LeRobot framework [22, 23], with modifications to its X-VLA preprocessing and action-representation configurations. The standard LeRobot implementation of X-VLA proved difficult to use without modification. For single-arm SO-101 datasets, LeRobot typically relies on the dataset-defined action representation, which in standard SO-101 datasets corresponds to joint positions and gripper state rather than end-effector pose. We were unable to obtain reliable training and execution under this configuration, potentially given that the mismatch between the pretrained model's general end-effector action representation, and the robot-specific joint-space representation increases the difficulty of policy re-adaptation, particularly when only relatively small self-collected datasets are available.

We also found that the default normalization configuration was insufficient for our setup, as it applied no normalization to the proprioceptive state. Reliable training required overriding this behavior and applying mean-based normalization using statistics computed from the training dataset.

After these modifications, all evaluated policies follow a two-stage training procedure. We first initialize the model from the pretrained X-VLA checkpoint provided by LeRobot on Hugging Face, `lerobot/xvla-base`, and train it for 45K steps on a dataset containing 196 demonstrations. This stage adapts the checkpoint to our robotic setup; further details are provided in Subsection 7.1. We use a batch size of 64 and a learning rate of 1×10^{-4}.

Following this adaptation stage, a task-specific baseline policy is trained for each evaluated task for 15K steps, using an initial learning rate of 1×10^{-4} and a cosine decay schedule toward 1×10^{-5} over 30K steps. Effective batch sizes of 32, 64, and 128 were evaluated on a representative task. Since a batch size of 64 produced the best performance, it was adopted for the remaining baseline training runs.

Expert-Aligned Visual-Thought Distillation Before introducing guidance into policy training, we first evaluate whether the visual-thought decoders presented in Section 3 can reliably reconstruct the expert-aligned representations from the pretrained X-VLA latent features. Decoders are trained to reconstruct the expert token representation of DINOv2 and the center direction maps of CeDiRNet.

For the DINOv2 target, each task-specific decoder consists of 4 layers and is distilled for 3K steps using a batch size of 64 and a learning rate of 10^{-3}. The hidden dimension of the tokens matches that of the expert (768) and the FFN layer has a 8 to 1 ratio with the hidden dimension.

For CeDiRNet-aligned distillation, ta single decoder is distilled from the vanilla X-VLA cloth-folding policy for 2.5K steps using a batch size of 64 and a learning rate of 10^{-3}. The decoder is single layer and has a hidden dimension of 32×32 tokens, which represent a 32×32 grid from which the predictions are up-sampled to match the expert's dense maps size on both training and visualization.

Implicit Visual-Thought Guidance Training This regime corresponds to the implicit exploitation setting. It focuses on learning expert-aligned visual-thought representations from X-VLA features under auxiliary supervision, without exposing the learned guidance explicitly to the action-conditioning interface. We train implicit-guidance experiments for 3K steps, a learning rate of 5×10^{-5} and batch size of 16.

Explicit Guided X-VLA Integration Training This regime corresponds to the explicit exploitation setting. Starting from a pretrained X-VLA backbone and a previously learned visual-thought guidance module, the learned guidance representation is fused into the action-conditioning stream and the guided policy is optimized for control considering the injected information. Experiments are trained for 4K steps, learning rate 5×10^{-5} and batch size of 16.

The three policy-training regimes introduced above are successive views on the same underlying X-VLA-based manipulation setups. They therefore share common task families, datasets, pretrained base-model initialization, and core policy configuration.

8. Results

This section reports results of the whole experimental pipeline used in this work: (i) vanilla X-VLA policy training, (ii) expert-aligned visual-thought distillation, and guidance inclusion into policy training through (iii) implicit joint training and (iv) explicit guidance integration.

8.1. Vanilla X-VLA Policy Training. This is the reference baseline. We establish the unguided X-VLA performance by training the policy under the standard flow-matching objective and optimization settings.

8.1.1. Optimization Results. Table 2 reports the training and validation losses for the selected baseline runs. Across all tasks, the models fit the training data consistently, while the larger validation losses indicate a noticeable degree of overfitting. *Multi-Color Cube Semantic Generalization* and *Cloth Corner-Folding* exhibit broadly comparable position and gripper losses, although the latter shows a somewhat higher orientation loss. By contrast, *Cloth Corner-to-Box* is substantially more difficult to fit, with considerably larger validation losses across all reported components.

This train–validation gap was observed across the X-VLA training regimes considered in this work, though explicit guidance regimes managed to decrease it to a considerable degree when compared with the equivalent base model results. The gap is likely attributable to the combination of relatively small task-specific datasets and the use of the standard X-VLA regularization settings, which were kept largely unchanged from the original implementation and provide limited protection against overfitting. Validation is performed on held-out episodes rathe, making it a strict estimate of generalization to unseen trajectories.

We explored synthetic image and lighting augmentations and weight decay, but these did not produce a clear improvement in validation performance. While under-training

could achieve lower validation losses, the corresponding checkpoints performed poorly during real-world execution, suggesting that validation loss alone was not a fully reliable model-selection criterion. We therefore trained the policies for 15K steps, where validation performance had broadly stabilized and the policies remained effective on the physical robot.

Task	Dataset / Setting	BS	Step	Train Losses			Validation Losses		
				Pos.	Orient.	Grip.	Pos.	Orient.	Grip.
Multi-Color Cube	`pickplace-multicolor`	32	15k	3.640	0.180	0.104	47.304	1.285	0.073
Multi-Color Cube	`pickplace-multicolor`	64	15k	3.635	0.276	0.040	**46.270**	**1.242**	**0.065**
Multi-Color Cube	`pickplace-multicolor`	128	15k	1.466	0.118	0.071	68.130	1.490	0.146
Cloth Corner-Folding	`cloth-corner-fold`	64	15k	0.977	0.132	0.027	69.954	2.159	0.080
Cloth Corner-to-Box	`cloth-corner-box`	64	15k	0.800	0.106	0.042	208.673	4.611	0.839

Table 2. Training and validation component losses for the *Vanilla X-VLA* policies after $15\mathrm{K}$ training steps. Position loss corresponds to a weighted mean-squared error on mean/std-normalized end-effector translation targets. The underlying coordinates are expressed in meters and the translation term is weighted by 500, following the original X-VLA implementation. Orientation loss corresponds to a weighted mean-squared error on the 6D continuous rotation, weighted by 10. Gripper loss corresponds to binary cross-entropy on open/closed gripper targets. For *Multi-Color Cube Semantic Generalization*, batch sizes of 32, 64, and 128 were evaluated. Batch size 64 achieved the best overall performance and was therefore used as the baseline configuration in the remaining experiments.

8.1.2. Evaluation Results.

Single-Color Cube Generalization. Table 3 summarizes the vanilla X-VLA results on the single-color cube task. The policy achieves 12/16 successful target acquisitions and 10/16 successful drops, providing evidence of moderate but incomplete spatial generalization to unseen cube positions. Performance is not uniform across the workspace: some positions are handled reliably, while others remain difficult despite not presenting obvious visual or geometric differences. Moreover, when the cube is placed near previously observed training positions, the policy occasionally collapses toward the closest familiar grasp configuration rather than adapting precisely to the new target location. This suggests that the policy collapses towards locally interpolative behavior rather than a completely general, position-invariant.

Nonetheless, the stages of the task, including pick up, grasp, transfer, drop, and return to the home pose, could be easily identified in the policy behavior. The intention of grasping was consistently clear when the target was inside the workspace. The gripper closing action occurred reliably anytime the object was aligned below the robot gripper, and once the object had been grasped, it was successfully transported to the placement box. The principal challenge was grasping precision, especially for cube positions farthest from the robot.

Notably, the policy demonstrated robust recovery maneuvers, consistently readjusting its position to re-attempt grasping after an unsuccessful attempt. While single-target identification was generally successful, executing precise, slow-displacement adjustments required to re-adjust the end effector for cube displacement caused by gripper collision proved more difficult, as the robot was not explicitly trained on recovery maneuvers.

Successful acquisitions that were not followed by a drop usually occurred because repeated acquisition attempts exhausted the available evaluation time.

Model	Position	Acquisition	Drop	$\mathbf{P(Drop\|Acq.)}$
Vanilla X-VLA	Position 1	4/4	3/4	75.0%
	Position 2	3/4	2/4	66.7%
	Position 3	4/4	4/4	100.0%
	Position 4	1/4	1/4	100.0%
	Total	12/16 (75.0%)	10/16 (62.5%)	83.3%

Table 3. Performance on *Single-Color Cube Generalization*. Acquisition reports successful target acquisition. Drop reports successful placement of the cube in the target area. Successful acquisitions with uncompleted drops occurred because the robot exceeded the episode's operational timeframe due to fail-and-recover grasp operations. The conditional drop rate reports the probability of a successful drop given that acquisition was successful.

Multi-Color Cube Semantic Generalization. Table 4 summarizes the vanilla X-VLA results on the binary semantic cube evaluation per commanded color pair, split by the left/right position of the commanded cube and annotated with the familiarity of the commanded and distractor colors at training time (K = known, U = unknown). Aggregate results by familiarity and position are given in Tables 5 and 6.

Overall, the baseline achieves 40/48 successful target identifications and 28/48 successful grasps. Approach accuracy is not directly affected by color familiarity (83.3% for both known and unknown commanded colors, Table 5), but appears strongly affected by position (Table 6): the policy identifies the commanded cube with perfect accuracy when

Multi-Color Cube Semantic Generalization — Vanilla X-VLA

Commanded / distractor	Fam.	Approach L	Approach R	Approach Tot.	Grasp L	Grasp R	Grasp Tot.
Orange / Blue	K/K	4/4	4/4	8/8	4/4	4/4	8/8
Blue / Orange	K/K	4/4	4/4	8/8	2/4	3/4	5/8
Orange / Red	K/U	4/4	0/4	4/8	3/4	0/4	3/8
Red / Orange	U/K	4/4	0/4	4/8	1/4	0/4	1/8
Red / Yellow	U/U	4/4	4/4	8/8	3/4	3/4	6/8
Yellow / Red	U/U	4/4	4/4	8/8	4/4	1/4	5/8
Total		**24/24** (100%)	**16/24** (66.7%)	**40/48** **(83.3%)**	**17/24** (70.8%)	**11/24** (45.8%)	**28/48** **(58.3%)**

Table 4. Vanilla X-VLA on the binary semantic cube evaluation (8 trials per commanded color: 4 with the commanded cube on the left, 4 on the right). **Approach** records whether the policy selected and approached the commanded cube; **Grasp** records whether that cube was then successfully grasped; both are split by the commanded cube's position (**L**/**R**). **Fam.** gives the familiarity of the commanded/distractor colors at training time (K = known, U = unknown). Across all 48 trials the policy never approaches or grasps the non-commanded cube (Table 6): every identification failure is a failure to commit to either cube, not confusion between them. Aggregate results by familiarity and position follow in Tables 5 and 6.

it is on the left (24/24, 100.0%) versus only 66.7% (16/24) when it is on the right, and a comparable left-side advantage appears in grasping (70.8% vs. 45.8%).

Although this may seem to suggest that position is the critical factor for identification, the per-pair breakdown in Table 4 reveals that every identification failure is sourced exclusively from a single condition: the mixed familiarity (K/U) pairings, in which the policy must choose between a color seen during training and one that was not. Right-side targets in these pairings were never approached (0/8), whereas right-side targets in the pure K/K and U/U pairings were always approached (16/16).

Importantly, in these failure cases the policy did not approach the wrong target either; instead, the gripper remained aligned midway between the two cubes, as if in doubt as to which one to grasp. Across all 48 trials the baseline neither approaches nor grasps the non-commanded cube even once (0/48), meaning every identification failure is a failure to commit to either cube rather than a failure of color discrimination.

We note that in this evaluation the mixed-familiarity pairings, orange–red pairings, are additionally two colors are chromatically adjacent, so the hesitation cannot be unambiguously attributed to the familiarity asymmetry only, versus the color similarity of the candidates.

In contrast to identification, the left-side advantage in grasping is not confined to the mixed pairings (4/4 left vs. 1/4 right for yellow among two unknown colors), indicating a positional bias at the acquisition stage given a more familiar grasping position.

Cmd. color	Approach	Grasp	Wrong cube
Known	20/24	16/24	0/24
Unknown	20/24	12/24	0/24
Total	**40/48** (83.3%)	**28/48** (58.3%)	**0/48** (0%)

Table 5. Vanilla X-VLA aggregate performance by color familiarity (24 trials per row). **Approach** = correct target identification; **Grasp** = successful acquisition of the commanded cube; **Wrong cube** = approaches toward the non-commanded cube (no wrong grasp ever occurred). Identification is identical for known and unknown commanded colors (20/24, 83.3%); acquisition is moderately lower for unknown colors (50.0% vs. 66.7%).

Cmd. position	Approach	Grasp	Wrong cube
Left	24/24	17/24	0/24
Right	16/24	11/24	0/24
Total	**40/48** (83.3%)	**28/48** (58.3%)	**0/48** (0%)

Table 6. Vanilla X-VLA aggregate performance by target position (24 trials per row), with columns as in Table 5. Identification is perfect when the commanded cube is on the left (24/24, 100%) and drops to 66.7% (16/24) on the right, with acquisition following the same pattern (70.8% vs. 45.8%). Since the policy never interacts with the wrong cube, all right-side identification failures are failures to commit rather than incorrect selections.

Cloth Corner-to-Box. Table 7 summarizes the vanilla X-VLA results on *Cloth Corner-to-Box*. This setting was harder for the policy to learn successfully.

The language instruction to pick up a corner and the corner-grasp demonstrations were not sufficient for the policy to target cloth corners specifically during evaluation. Only in Configuration 2, where the target corner was the most visible part of the cloth and located near the center of the workspace, did the policy correctly identify the target and attempt a corner grasp. Again, grasping success was hindered by precision issues. This explains the variation observed in this configuration's results: placement success is low because only one of the four correct target identifications led to a successful grasp.

In contrast to the cube-grasping task, after a failed grasp attempt the policy would try to transport and drop the cloth in the placement area regardless of whether the grasp had actually occurred. Few recovery attempts were made. This indicates an over-reliance

on the position of the gripper to determine the dropping stage of the task, ignoring the visual cues indicating that the object of interest had not yet been grasped. This hypothesis of hindered visual reliance is further supported below by the results of the cloth-folding task.

Model	Config.	Target ID	Grasp Score	Placement
Vanilla X-VLA	Config 1	0/4	0/4	4/4
	Config 2	4/4	1/4	1/4
	Config 3	0/4	0/4	4/4
	Config 4	0/4	0/4	4/4
	Total	4/16 (25.0%)	1/16 (6.3%)	13/16 (81.3%)

Table 7. Performance on *Cloth Corner-to-Box*. Target ID reports whether the correct cloth corner was identified. Grasp Score is based on the grasp location: 1.0 if the grasp was within 2 cm of the target corner, 0.5 if it was between 2 and 5 cm, and 0 otherwise. Grasp Score percentages represent the normalized score relative to the maximum possible score. Placement indicates whether the cloth was successfully placed in the target area, **regardless of whether the grasp was successful** (thus includes arbitrary non-corner grasps).

Cloth Corner-Folding. Table 8 summarizes the vanilla X-VLA results on *Cloth Corner-Folding*. The folding task again showed moderately successful performance. The corners were consistently identified (15 out of 16 times), and although the grasping score was hurt (down to 9.5) due to precision issues, the intention was evident. Interestingly, Configurations 2, 3, and 4 have low fold-score values. Even though the robot folded the cloth to the intended side, the corners were usually not brought together; instead, the robot would easily overshoot the placement and simply place the corner somewhere on the opposite side of the cloth. It was only in Configuration 1 that most executions were carried out successfully.

It is important to note that the dataset distribution of starting cloth positions was not highly varied, since it was ensured that both target ends remained visible in the camera view throughout execution. For this reason, the evaluation positions can be considered close to in-distribution, which likely contributed to the successful task executions, even with 30% less data than the Cloth-to-Box task. This suggests that the policy can relate familiar visual configurations to learned action sequences, but is far from demonstrating language-based 'corner-to-corner folding' reasoning. Instead, the behavior appears to rely more on visual correlations between coarse action regions, the task structure, and the learned motor stages. For example, if the grasp displaced the cloth and caused the target opposite corner to move, the policy would not react to bring the cloth to the new corner location.

This is consistent with the robot-pose over-reliance observed in *Cloth Corner-to-Box*, and suggests that a large and highly representative collection of in-distribution task demonstrations is required for the policy's regression head to predict precise and successful actions. Few-shot demonstrations did not transfer pre-trained knowledge to new tasks in a reliable, general way.

Model	Config.	Target ID	Grasp Score	Fold
Vanilla X-VLA	Config 1	4/4	3/4	3/4
	Config 2	4/4	2/4	0/4
	Config 3	3/4	2/4	0/4
	Config 4	4/4	2.5/4	0/4
	Total	15/16 (93.8%)	9.5/16 (59.4%)	3/16 (18.8%)

Table 8. Performance on *Cloth Corner-Folding*. Target ID reports whether the correct cloth corner was identified. Grasp Score is graded according to grasp location: 1.0 if the grasp was within 2 cm of the target corner, 0.5 if it was between 2 and 5 cm from the target corner, and 0 if the cloth was grasped at another position. Grasp Score percentages report the score relative to the maximum possible score. Fold reports whether the cloth was successfully folded making grasped and opposite edges meet to a distance of no more than 2 cm after placement.

8.2. Expert-Aligned Visual-Thought Distillation.

DINO-Aligned Decoder Results Table 9 reports the corresponding distillation losses, and Figure 8 shows qualitative visualizations of the decoded DINOv2 representations. For visualization, the expert and student tokens associated with each visual patch are projected into a shared three-dimensional PCA space and rendered as RGB values. Across both cube and cloth-manipulation scenes, semantically similar regions exhibit similar colors in the expert and reconstructed student representations. These results indicate that the querying-based decoder is able to recover the structure of the expert feature space with good fidelity.

Regime	Dataset	BS	Step	Train loss	Val loss
DINO Distillation	`pickplace-multicolor`	64	3K	0.416	0.418
DINO Distillation	`cloth-corner-fold`	64	3K	0.426	0.412
DINO Distillation	`cloth-corner-box`	64	3K	0.475	0.408

Table 9. DINO-aligned decoder distillation losses across the three task settings.

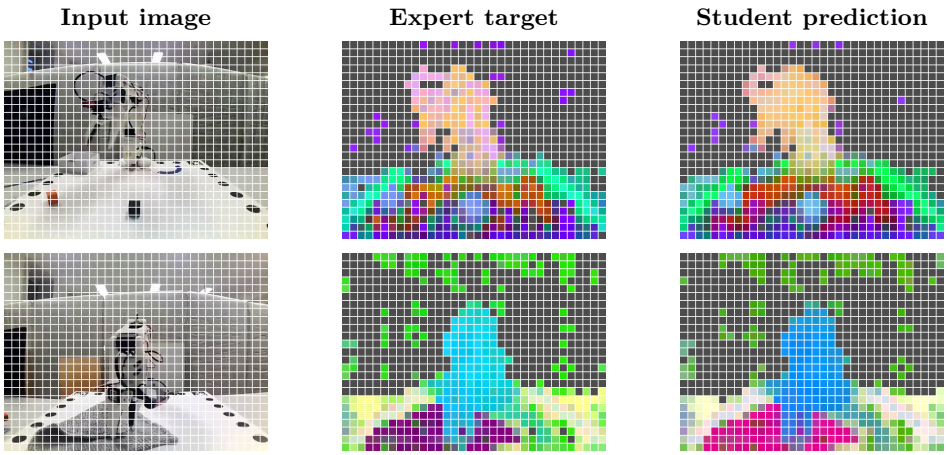


Figure 8. Qualitative DINO-aligned decoder outputs for the cube and cloth-folding settings. For each row, the columns show the input image, the expert DINO target representation, and the student decoder prediction.

CeDiRNet-Aligned Decoder Results Table 10 reports the corresponding distillation losses, and Figure 9 shows qualitative visualizations of the decoded guidance maps. The resulting reconstructions preserve the coarse spatial organization of the scene and recover the task-relevant directional structure needed for cloth manipulation. This shows that the visual-thought decoder is also capable of recovering the spatial expert signal required for CeDiRNet-based guidance.

Dataset	BS	Step	Train loss	Val loss
`cloth-corner-fold`	64	2.5K	0.009	0.015

Table 10. CeDiRNet-aligned visual-thought decoder distillation results across the cloth-manipulation settings.

8.3. Guidance Inclusion into Policy Training. We next evaluate how expert-aligned guidance is introduced into policy training. This part of the chapter is organized around optimization stages only: first implicit joint training, then explicit guidance integration. The downstream rollout comparisons are deferred to Section 8.4, where each task is reported once across the full training lineage.

8.3.1. Implicit Guidance Joint Training. We first evaluate the implicit-guidance setting, in which the X-VLA policy and pre-trained Visual-Thought decoders are jointly trained for an additional 3K steps under both the action objective and the expert-guidance objective. In this setting, the learned guidance remains internal to the model and is not injected explicitly into the action-transformer sequence at inference time. An additional control experiment is run, training the same baseline under the same optimization parameters than those of the implicit guidance, to isolate the effects of the decoder-based regularization.

Optimization Results. Relative to the 15K *Vanilla X-VLA* baselines, both continuation paths substantially reduce validation losses, showing that the policies continue to benefit from optimization under the restarted schedule. For the CeDiRNet Decoder-only runs, *Implicit Guidance* further reduces both validation position and orientation losses for the *Cloth Corner-Folding* and *Cloth Corner-to-Box* tasks relative to the matched control. This reduction is, however, not reproduced by the joint CeDiRNet+DINOv2 decoder runs. Thus, this stage establishes that expert-aligned guidance can be incorporated while maintaining competitive optimization and, in the CeDiRNet-only setting, improved the validation losses. The effects on evaluation are observed and discussed in Section 8.4.

8.3.2. Explicit Guidance Integration. We finally evaluate explicit guidance integration, in which the learned visual-thought representation is exposed directly to the policy by fusing it into the X-VLA action-conditioning stream. All explicit-guidance runs are initialized from the corresponding implicit joint-training checkpoints and then trained for a further 4K steps under the guided-policy objective.

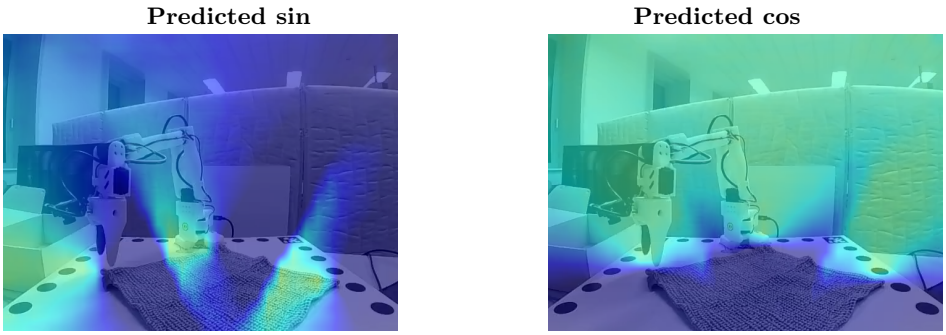


Figure 9. Qualitative CeDiRNet-aligned decoder output, shown as predicted sine and cosine directional maps.

Task	Setting	BS	Step	Train Losses				Validation Losses			
				Pos.	Orient.	DINO	CED	Pos.	Orient.	DINO	CED
Multi-Color Cube	Vanilla baseline	64	15k	3.635	0.276	–	–	46.270	1.242	–	–
Multi-Color Cube	Control run	16	15+3k	6.270	0.266	0.923	–	26.168	1.204	0.821	–
Multi-Color Cube	Implicit (DINO)	16	15+3k	8.532	0.356	0.456	–	**19.084**	**0.778**	**0.350**	–
Cloth Corner-Folding	Vanilla baseline	64	15k	0.977	0.132	–	–	69.954	2.159	–	–
Cloth Corner-Folding	Control run	16	15+3k	14.799	0.406	–	–	28.297	1.136	–	–
Cloth Corner-Folding	Implicit (CeDiRNet)	16	15+3k	13.828	0.248	–	0.034	**27.426**	**1.059**	–	**0.034**
Cloth Corner-Folding	Implicit (CeDiRNet + DINO)	16	15+3k	6.319	0.293	0.430	0.012	30.899	1.197	0.419	0.050
Cloth Corner-to-Box	Vanilla baseline	64	15k	0.800	0.106	–	–	208.673	4.611	–	–
Cloth Corner-to-Box	Control run	16	15+3k	5.123	0.198	–	–	19.790	0.490	–	–
Cloth Corner-to-Box	Implicit (CeDiRNet)	16	15+3k	4.650	0.551	–	0.006	**7.561**	**0.310**	–	0.052
Cloth Corner-to-Box	Implicit (CeDiRNet + DINO)	16	15+3k	8.205	0.286	0.417	0.017	22.909	0.638	0.445	**0.049**

Table 11. Implicit-stage optimization results together with the corresponding vanilla baseline checkpoints used as initialization. Vanilla baseline rows report the unguided X-VLA baseline after 15K steps with batch size 64. Control run rows report matched continuation without auxiliary guidance for cloth manipulation tasks, while implicit rows report the corresponding guided continuation. The cube implicit run uses DINO supervision, while the cloth implicit runs include both CeDiRNet-only and joint CeDiRNet+DINO supervision.

For each task, we evaluate six configurations: (i) a baseline continuation without explicit guidance, (ii) a hidden-guidance control that retains the auxiliary regularization of the implicit guidance strategy, and the explicit guidance strategies: (iii) concatenative, (iv) gated concatenative, and (v) selected-layer using layers $6, 12, 18$ of the 24-layer X-VLA action transformer, and (iv) its gated version. We evaluate only the cloth-manipulation tasks and use CeDiRNet decoder tokens as guidance. The runs were initialized from the implicit training checkpoints trained with CeDiRNet and CeDiRNet + DINOv2 supervision for each task.

Optimization Results. Table 12 compares the validation losses of the explicit-guidance fusion strategies.

For *Cloth Corner-Folding*, the best results overall are obtained by the selected-layer fusion strategy using layers $6, 12, 18$. This strategy achieves the lowest validation position and orientation losses, and also matches the best gripper loss. In this setting, plain concatenation performs worst, while gated concatenation improves over plain concatenation but remains slightly weaker than selected-layer fusion.

For *Cloth Corner-to-Box*, the pattern is different. Here, gated concatenation is the strongest strategy, achieving the best position validation loss. Selected-layer fusion gives the best orientation loss, but is much weaker on position.

The baseline and hidden-guidance control runs are also highly informative. Their validation values remain consistently aligned and are higher than those of the best explicit guidance approache for each task. This suggests that the gains at this stage do not come from further training or retaining the auxiliary regularization pathway. Rather, the benefit comes from making the learned guidance representation explicitly available to the policy.

Overall, the results indicate that explicit guidance is helpful, but that the most effective fusion mechanism depends on the task.

8.4. Real-world Evaluation. This subsection consolidates the rollout comparisons once per task on *Multi-Color Cube Semantic Generalization*, *Cloth Corner-to-Box*, and *Cloth Corner-Folding*, following the progression from the base X-VLA model, the implicit guidance training, and finally training exposed to explicit guidance.

8.4.1. Cube Semantic Reasoning. Tables 13 and 14 report the binary semantic cube evaluation, separating semantic target selection (approach) from successful acquisition (grasp). Each row corresponds to one commanded target within a color-pair family, with the balanced left- and right-side trials shown separately ($\mathbf{L}/\mathbf{R}$) so that positional effects are visible per pair. Tables 15 and 16 aggregate the same trials by color familiarity and by target position, respectively.

Task	Implicit init	Strategy	Val Position	Val Orientation
Cloth Corner-Folding	CeDiRNet	Baseline	46.946	1.212
		Hidden	47.260	1.182
		Concat	48.183	1.389
		Concat Gated	44.218	1.313
		Selected 6-12-18	**44.050**	**1.138**
		Selected 6-12-18 Gated	44.543	1.176
Cloth Corner-Folding	CeDiRNet + DINO	Baseline	61.250	1.376
		Hidden	61.350	1.379
		Concat	56.986	1.523
		Concat Gated	52.783	1.479
		Selected 6-12-18	57.778	1.326
		Selected 6-12-18 Gated	58.057	1.366
Cloth Corner-to-Box	CeDiRNet	Baseline	43.217	1.161
		Hidden	42.933	1.148
		Concat	37.699	1.175
		Concat Gated	**36.133**	1.180
		Selected 6-12-18	42.284	**1.125**
		Selected 6-12-18 Gated	42.595	1.143
Cloth Corner-to-Box	CeDiRNet + DINO	Baseline	52.222	1.352
		Hidden	50.870	1.327
		Concat	43.101	1.321
		Concat Gated	42.199	1.282
		Selected 6-12-18	50.561	1.286
		Selected 6-12-18 Gated	51.043	1.301

Table 12. Validation losses for explicit-guidance fusion strategies on the cloth-manipulation tasks using CeDiRNet guidance. Each run is initialized from the implicit-training checkpoint and trained for a further 4K steps at batch size 16. A larger validation set than that used for the baseline and implicit-training runs was employed in this evaluation; therefore, the loss magnitudes are not directly comparable with those reported for pure implicit training in the previous tables. However, the continued implicit-training run was also extended by 4K steps, providing an aligned reference for direct comparison with the explicit-guidance strategies.

Relative to the baseline, the effect of implicit DINO guidance is mixed but revealing. First, on the two ambiguous conditions in which the commanded cube is on the right within a known–unknown orange–red pairing: commanding the known orange cube and commanding the unknown red cube. In both, the baseline achieves zero correct identifications (0/4), whereas the implicit variant recovers 2/4 and 1/4, respectively. Although these gains are small, the more telling change is qualitative: the remaining failures are no longer the midway-stopping, non-committal behavior seen in the baseline, but attempts in which the model commits to a target with greater confidence. This shift is also reflected in the acquisition results. In the baseline, the same lack of confidence appeared to affect even left-side grasps, where the robot was slower and less decisive; under implicit guidance these become more reliable. Most notably, the left-positioned red cube in the red–orange pairing (unknown commanded color) improves its grasp success from 1/4 to 3/4.

8.4.2. Cloth Corner-to-Box. As described in 7.1.3, we evaluate each stage of the task separately. **Target ID** is a binary count of trials in which the policy approached the commanded corner, **Grasp Score** uses spatial tolerance: a trial scores 1 if the gripper contacts the corner within 2 cm, 0.5 if within 2–5 cm, and 0 otherwise, and **Placement** scores whether cloth is transferred into the box.

Table 17 shows that implicit and explicit selected-layer guidance both substantially improve target identification over the control run (75.0% and 87.5% vs. 50.0%), indicating that the gains result from the guidance signal rather than from the additional training alone, which the control already receives relative to the original baseline.

We focus on target identification and on grasp score given correct identification (Grasp | ID, Table 18), as these measure whether the policy locates the commanded corner and then grasps at it. Placement is reported for completeness but is not a reliable success measure on its own: as shown in Table 18, it credits any transfer of cloth into the box without conditioning on a correct grasp. The baseline is the clearest case, placing cloth in 13/16 trials while attaining a grasp score of only 1/16, which reflects arbitrary grasping rather than proper task competence.

Between the two guidance variants, implicit guidance shows the most balanced improvement: it attains the highest grasp score overall (50.0%) and, more importantly, the highest grasp score given correct identification (66.7%, Table 18), well above explicit selected-layer guidance (39.3%). Explicit selected-layer guidance identifies the target slightly more often (87.5%) but grasps less accurately on those identifications, so its identification advantage does not translate into reliable manipulation. Explicit concate-

Binary-Decision Target Identification

Commanded / distractor	Fam.	Vanilla X-VLA L	Vanilla X-VLA R	Vanilla X-VLA Tot.	Implicit X-VLA L	Implicit X-VLA R	Implicit X-VLA Tot.
Orange / Blue	K/K	4/4	4/4	8/8	4/4	4/4	8/8
Blue / Orange	K/K	4/4	4/4	8/8	4/4	4/4	8/8
Orange / Red	K/U	4/4	0/4	4/8	4/4	2/4	6/8
Red / Orange	U/K	4/4	0/4	4/8	4/4	1/4	5/8
Red / Yellow	U/U	4/4	4/4	8/8	4/4	4/4	8/8
Yellow / Red	U/U	4/4	4/4	8/8	4/4	4/4	8/8
Total		**24/24** (100%)	**16/24** (66.7%)	**40/48** **(83.3%)**	**24/24** (100%)	**19/24** (79.2%)	**43/48** **(89.6%)**

Table 13. Target identification (approach). Trials in which the policy selected and approached the commanded cube, split by the commanded cube's position: **L** = commanded cube on the left (4 trials per row), **R** = on the right (4 trials). **Fam.** gives the familiarity of the commanded/distractor colors at training time (K = known, U = unknown). Both models identify the target perfectly when it lies on the left and degrade only on the right, with the drop confined to the orange–red pairings. Aggregate results by familiarity and position are reported in Tables 15 and 16.

Binary-Decision Target Acquisition

Commanded / distractor	Fam.	Vanilla X-VLA L	Vanilla X-VLA R	Vanilla X-VLA Tot.	Implicit X-VLA L	Implicit X-VLA R	Implicit X-VLA Tot.
Orange / Blue	K/K	4/4	4/4	8/8	4/4	1/4	5/8
Blue / Orange	K/K	2/4	3/4	5/8	4/4	3/4	7/8
Orange / Red	K/U	3/4	0/4	3/8	3/4	0/4	3/8
Red / Orange	U/K	1/4	0/4	1/8	3/4	1/4	4/8
Red / Yellow	U/U	3/4	3/4	6/8	4/4	0/4	4/8
Yellow / Red	U/U	4/4	1/4	5/8	4/4	2/4	6/8
Total		**17/24** (70.8%)	**11/24** (45.8%)	**28/48** **(58.3%)**	**22/24** (91.7%)	**7/24** (29.2%)	**29/48** **(60.4%)**

Table 14. Target acquisition (grasp). Trials in which the commanded cube was successfully grasped, in the same format as Table 13. Overall acquisition is nearly identical between models (58.3% vs. 60.4%), but the totals hide opposite side profiles: the implicit variant is far stronger on the left (91.7% vs. 70.8%) and far weaker on the right (29.2% vs. 45.8%).

nation is weak on both metrics (12.5% identification, 3.1% grasp score), suggesting that appending an extra input dimension is less beneficial than controlled, selected guidance applied at staged points of the attention pipeline.

8.4.3. Cloth Corner-Folding. Scoring follows *Cloth Corner-to-Box*: Target ID is a binary count, Grasp Score uses the same 2 cm / 5 cm spatial tolerance, and **Fold** scores the quality of the resulting fold. Grasp and Fold are therefore mean scores rather than counts of exact successes.

Table 19 shows a different pattern from the *Cloth Corner-to-Box*. All variants except explicit concatenation (which again collapsed policy performance) identify the target near-perfectly (93.8–100%), so identification is not the distinguishing factor. Grasp score given correct identification is. The control and explicit selected-layer guidance are strongest (70.0% and 65.6%, Table 20), whereas implicit guidance attains the lowest conditional grasp score (34.4%) despite perfect identification (16/16). End-to-end fold scores are low across all variants (18.8–34.4%), with the control highest (34.4%), indicating that the folding stage remains difficult for every model and is not resolved by any of the guidance schemes.

9. Discussion

Task-relevant perceptual structure is present in the X-VLA backbone. The central premise of this work, and an important principle of the X-VLA architecture more generally, is supported by the decoder results: task-relevant information can be recovered, organized, and exploited using lightweight decoders that read from the VLA backbone, without external modules and by reusing the existing X-VLA encoding pipeline.

Guidance improves target identification, not merely through additional training. Across cloth tasks, implicit and explicit selected-layer guidance improved target identification and either improved or maintained overall task performance relative to the base X-VLA model. In *Cloth Corner-to-Box*, guidance raised identification from 50.0% (control) to 75.0% (implicit) and 87.5% (explicit selected layers). In *Cloth Corner-Folding*, both guidance variants reached perfect identification (16/16), only a slight improvement over

Performance by Color Familiarity

	Vanilla X-VLA			Implicit X-VLA		
Commanded color	**Approach**	**Grasp**	**Wrong cube**	**Approach**	**Grasp**	**Wrong cube**
Known	20/24	16/24	0/24	22/24	15/24	2/24
Unknown	20/24	12/24	0/24	21/24	14/24	3/24
Total	**40/48**	**28/48**	**0/48**	**43/48**	**29/48**	**5/48**
	(83.3%)	(58.3%)	(0%)	(89.6%)	(60.4%)	(10.4%)

Table 15. Aggregate performance by color familiarity (24 trials per row per model). **Approach** = correct target identification; **Grasp** = successful acquisition of the commanded cube; **Wrong cube** = approaches toward the non-commanded cube (no wrong grasp ever occurred, for either model). Familiarity has little effect on identification for either model; the vanilla baseline shows a moderate acquisition gap for unknown colors (12/24 vs. 16/24) that the implicit variant nearly closes (14/24 vs. 15/24). Wrong-cube approaches by the implicit model occur at similar rates for known and unknown commanded colors.

Performance by Target Position

	Vanilla X-VLA			Implicit X-VLA		
Commanded cube position	**Approach**	**Grasp**	**Wrong cube**	**Approach**	**Grasp**	**Wrong cube**
Left	24/24	17/24	0/24	24/24	22/24	0/24
Right	16/24	11/24	0/24	19/24	7/24	5/24
Total	**40/48**	**28/48**	**0/48**	**43/48**	**29/48**	**5/48**
	(83.3%)	(58.3%)	(0%)	(89.6%)	(60.4%)	(10.4%)

Table 16. Aggregate performance by target position (24 trials per row per model), with columns as in Table 15. Both models exhibit a clear left-side bias: identification is perfect on the left and drops on the right, and acquisition follows the same pattern, with the implicit variant amplifying the asymmetry (22/24 left vs. 7/24 right for grasping). The implicit model's wrong-cube approaches occur *exclusively when the commanded cube is on the right*, i.e., every wrong approach is toward the left-positioned distractor. This is the same positional bias expressed as a wrong choice rather than a failure to commit.

the baseline, while slightly affecting the final, precision-critical folding step. Notably, guidance produced a substantial gain on the identification of targets on the task whose training data had a sparser distribution of target positions, indicating that it can promote generalization during policy optimization.

Identification and manipulation are dissociable. A policy can locate the commanded target reliably and still fail to act on it. Explicit selected-layer guidance is the clearest case: it obtains the best target identification on both tasks (14/16 on *Cloth Corner-to-Box* and 16/16 on *Cloth Corner-Folding*) but converts those identifications into grasps far less effectively than implicit guidance in *Cloth Corner-to-Box* (39.3% vs. 66.7% conditional grasp score). This failure is not necessarily coming from perception errors, but is likely related also to imprecise pose estimations and grasp executions. We therefore argue that conditional, staged metrics (Grasp | ID) are necessary to diagnose when and how guidance helps, and we recommend them for evaluating VLA policies more generally.

The benefit of implicit guidance is task-dependent. The two cloth tasks dissociate. In *Cloth Corner-to-Box*, implicit guidance yields the best grasp score (50.0%) and the best conditional grasp score (66.7%). In *Cloth Corner-Folding*, however, it attains the *lowest* conditional grasp score (34.4%) despite perfect identification, while the control and explicit selected-layer guidance perform best (70.0% and 65.6%). End-task fold scores remain low for every variant (18.8–34.4%), indicating that none of the guidance schemes considered here resolves the folding stage. We attribute the absence of benefit in folding to the high variability of the task, in which the cloth's own motion perturbs the expected action execution, although verifying this remains open. The conclusion we draw is that implicit guidance improves simple, clear semantic representation to a relatively high level given the easier identification of targets, and particularly for an underrepresented training distribution. The guidance strategies employed however, did not on their own improve fine-grained deformable manipulation.

The injection mechanism matters. Explicit concatenation degrades overall performance across every stage of both tasks, despite receiving the same recovered guidance information as the selected-layer variant; the two differ only in *how* guidance enters the action-generation module. This failure does not imply that concatenation is inherently unsuitable as a conditioning mechanism: RT-Trajectory [11] successfully conditions a policy by concatenating a trajectory sketch channel-wise with the visual input, where the concatenated signal is spatially registered with the observation. In our case, however, the guidance is appended as *unaligned feature tokens* to the input of a pretrained action

Model	Config.	Target ID	Grasp Score	Placement
Baseline	Config 1	0/4	0/4	4/4
	Config 2	4/4	1/4	1/4
	Config 3	0/4	0/4	4/4
	Config 4	0/4	0/4	4/4
	Total	4/16 (25.0%)	1/16 (6.3%)	13/16 (81.3%)
Control (+4K)	Config 1	2/4	2/4	4/4
	Config 2	4/4	2/4	4/4
	Config 3	0/4	0/4	3/4
	Config 4	2/4	0/4	4/4
	Total	8/16 (50.0%)	4/16 (25.0%)	**15/16 (93.8%)**
Implicit	Config 1	4/4	2.5/4	4/4
	Config 2	4/4	3/4	3/4
	Config 3	0/4	0/4	1/4
	Config 4	4/4	2.5/4	4/4
	Total	12/16 (75.0%)	**8/16 (50.0%)**	12/16 (75.0%)
Explicit Concat	Config 1	1/4	0.5/4	1/4
	Config 2	0/4	0/4	0/4
	Config 3	1/4	0/4	0/4
	Config 4	0/4	0/4	0/4
	Total	2/16 (12.5%)	0.5/16 (3.1%)	1/16 (6.3%)
Explicit Selected Layers	Config 1	4/4	2/4	4/4
	Config 2	4/4	1/4	1/4
	Config 3	3/4	1.5/4	2/4
	Config 4	3/4	1/4	4/4
	Total	**14/16 (87.5%)**	5.5/16 (34.4%)	11/16 (68.8%)

Table 17. *Cloth Corner-to-Box* results across the reported comparison variants: baseline, further-trained baseline control (+4K), implicit guidance, explicit concat, and explicit selected-layer guidance. Target ID is a binary count; Grasp Score is a spatial-tolerance score (1 within 2 cm, 0.5 within 2–5 cm, 0 otherwise), so its totals are mean scores rather than counts of exact grasps. Bold in each Total row marks the best score per column across all five models. The baseline results are the same of Table 7 repeated here for clarity.

generator, which must learn the correspondence between guidance and observation from plain attention; this suggests that the failure originates from injecting spatially unaligned information into an already-structured pipeline. Our selective injection through attention at chosen layers avoids this disruption by letting the model attend to guidance where it is useful, without displacing the pretrained input structure. Further work is needed to determine whether hard concatenation could become viable given sufficient task-specific training data.

X-VLA is highly sensitive to environment variations. The most challenging and remarkable observation during the deployment of the base X-VLA model on our experimental hardware was the model's high sensitivity to environmental variation. Particularly, even small changes in camera orientation and lighting conditions considerably affected the model's estimations of depth and horizontal positioning. This suggests a strong correlation between the perceived size and shape of the robot or target and the camera viewpoint, independently of each other, with a weaker relation between each other under varying camera conditions. Sensitivity to lighting was mitigated by training on a dataset with greater heterogeneity of lighting conditions. Whether collecting data from the same fixed setup with slight camera inclination variations would similarly improve robustness to imprecise control remains to be seen.

Furthermore, since X-VLA predicts end-effector poses (position and orientation) relative to a fixed axis (physically the base of the robot), operational precision is highly conditioned on keeping the poses estimated via inverse kinematics consistent between training and real-world execution. Variations in joint-angle sensing or motor miscalibrations therefore degrade performance immediately by widening the gap between intended and executed actions. In our setup, using the SO-ARM101 platform, we consistently observed shaky behavior slightly during training and especially during policy evaluation, arising from the momentum of the joint movements. To reduce this effect we applied an exponential moving average (EMA) filter to the predicted motor commands at evaluation time, which proved beneficial to a certain extent.

Limitations. Our evaluation uses task-specific trial counts: *Single-Color Cube Generalization* uses 16 trials per model (4 positions $\times$ 4 trials), *Multi-Color Cube Semantic Generalization* uses 48 trials per model (6 commanded-color conditions $\times$ 8 trials), and

Model	Task-stage score			Grasp | ID	Uncredited
	Target ID	Grasp	Placement	(cond. score)	placement
Baseline	4/16 (25.0%)	1/16 (6.3%)	13/16 (81.3%)	25.0%	12/16
Control (+4K)	8/16 (50.0%)	4/16 (25.0%)	15/16 (93.8%)	50.0%	11/16
Implicit	12/16 (75.0%)	**8/16 (50.0%)**	12/16 (75.0%)	**66.7%**	**4/16**
Explicit Concat	2/16 (12.5%)	0.5/16 (3.1%)	1/16 (6.3%)	25.0%	0.5/16
Explicit Sel. Layers	**14/16 (87.5%)**	5.5/16 (34.4%)	11/16 (68.8%)	39.3%	5.5/16

Table 18. Conditional analysis of *Cloth Corner-to-Box*. Task-stage columns repeat the scores from Table 17. Grasp is a partial-credit score, whereas Target ID and Placement are binary counts. **Grasp | ID** is the mean grasp score restricted to correctly identified targets (grasp score divided by the number of correct identifications), isolating grasp accuracy from identification. **Uncredited placement** is placement not backed by a correct grasp (placement minus grasp score); because placement does *not* condition on grasping the commanded corner, this reflects transfers that most plausibly followed a wrong-corner or indiscriminate grasp. Bold marks the best value per column. *Implicit guidance* attains the highest conditional grasp score (66.7%) and, excluding the collapsed *Explicit Concatenation* model (which makes only 1/16 placements overall), by far the least uncredited placement (4/16), whereas the models with the highest raw placement (*Continuation Control*, *Baseline*) accumulate it largely without correctly identifying the target (corners).

Cloth Corner-to-Box and *Cloth Corner-Folding* each use 16 trials per model (4 configurations $\times$ 4 trials). Individual per-configuration figures are therefore noisy. The observations demonstrate the potential benefit of guidance, but firmer conclusions would require a more rigorous testing protocol. Physical robot evaluations are inherently noisy, given variation in object placement, sensor noise, and the stochastic nature of the model itself. Finally, guidance improvements are directly dependent on the quality of the guidance prediction itself: a reliable expert and a stable training procedure are therefore essential to optimize the self-guided policy robustly.

Future work. Future work should first evaluate the proposed guidance strategies at larger scale: more physical and simulation-environment trials, additional task configurations, and repeated training runs, so that systematic effects can be more clearly separated from experimental noise.

A second direction is robustness, studying X-VLA under controlled variations in camera pose, lighting, and object placement. Closely related is the choice of primary-camera viewpoint: evaluating alternative viewpoints would reveal how the visibility of object boundaries, grasp regions, and robot–object contact affects target identification and grasping robustness. Views that expose contact geometry more clearly may provide stronger spatial cues than our frontal setup, improving grasping and control precision.

For deformable manipulation, further experiments should determine whether the limited improvements arise from insufficient perceptual guidance, under-representative training data, or inaccurate pose prediction.

Finally, alternative mechanisms for incorporating the recovered perceptual structure should be explored. Particularly, they should clarify whether the observed failure of concatenation is intrinsic to the mechanism itself on the X-VLA architecture, a consequence of limited task-specific data, or caused by a mismatch between unaligned guidance tokens and the pretrained action-generation pathway in which the visual bias is not explicit enough.

10. Conclusion

We investigated how task-relevant perceptual structure can be recovered directly from a pretrained VLA backbone in a way that it can be reintroduced as an internal guidance signal, comparing the structured guidance as a representation-level training signal against direct conditioning of the action generator. Across colored cube selection and two deformable cloth tasks, guidance consistently improves target identification relative to a compute-matched control, confirming that the backbone encodes exploitable structure that the action expert does not fully use. However, identification gains do not automatically translate into manipulation gains, particularly since a VLA remains highly dependent on a calibrated hardware configuration and heterogeneous training demonstrations that completely model a execution distribution. Implicit guidance converts identification into successful grasps most effectively in *Cloth Corner-to-Box* (66.7% conditional grasp score), whereas explicit selected-layer guidance identifies targets best yet follows through less reliably, showing increased difficulty on making the policy be highly robust throughout every stage of the executions.

By comparing guidance injection strategies, we conclude that *how* recovered structure is injected into the action pathway is as consequential as *whether it is recovered and used at all*, and that implicit guidance offers a balanced, useful improvement in semantic representation that can be translated into more confident policy decisions. Future work should examine whether more heterogeneous datasets allow encoded information to be

Model	Config.	Target ID	Grasp Score	Fold
Baseline	Config 1	4/4	3/4	3/4
	Config 2	4/4	2/4	0/4
	Config 3	3/4	2/4	0/4
	Config 4	4/4	2.5/4	0/4
	Total	15/16 (93.8%)	9.5/16 (59.4%)	3/16 (18.8%)
Control (+4K)	Config 1	3/4	0/4	0/4
	Config 2	4/4	3.5/4	2/4
	Config 3	4/4	3/4	0/4
	Config 4	4/4	4/4	3.5/4
	Total	15/16 (93.8%)	**10.5/16 (65.6%)**	**5.5/16 (34.4%)**
Implicit	Config 1	4/4	3/4	2/4
	Config 2	4/4	1/4	0/4
	Config 3	4/4	0/4	0/4
	Config 4	4/4	1.5/4	1/4
	Total	**16/16 (100.0%)**	5.5/16 (34.4%)	3/16 (18.8%)
Explicit Concat	Config 1	1/4	0/4	0/4
	Config 2	3/4	1/4	0.5/4
	Config 3	3/4	1.5/4	0/4
	Config 4	4/4	4/4	2.5/4
	Total	11/16 (68.8%)	6.5/16 (40.6%)	3/16 (18.8%)
Explicit Selected Layers	Config 1	4/4	0/4	0/4
	Config 2	4/4	4/4	2.5/4
	Config 3	4/4	4/4	0/4
	Config 4	4/4	2.5/4	2/4
	Total	**16/16 (100.0%)**	**10.5/16 (65.6%)**	4.5/16 (28.1%)

Table 19. *Cloth Corner-Folding* results across the reported comparison variants. Scoring matches Table 17: Target ID is a binary count, Grasp Score is the spatial-tolerance score (1 within 2 cm, 0.5 within 2–5 cm, 0 otherwise), and Fold scores the resulting fold. Grasp and Fold totals are mean scores. Bold in each Total row marks the best score per column across all five models. The baseline results are the same of Table 8, repeated here for clarity.

exploited more robustly for general control, and explore whether alternative guidance strategies preserve the performance trend observed here, given the selective, careful inclusion of guidance information at different stages of the action-generation attention pipeline.

Bibliography

1. Zheng J et al. (2025) X-VLA: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. *arXiv preprint arXiv:2510.10274*.
2. Chen B et al. (2024) SpatialVLM: Endowing vision-language models with spatial reasoning capabilities in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. pp. 14455–14465.
3. Cheng AC et al. (2024) SpatialRGPT: Grounded spatial reasoning in vision language models in *Advances in Neural Information Processing Systems (NeurIPS)*.
4. Caron M et al. (2021) Emerging properties in self-supervised vision transformers in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. pp. 9650–9660.
5. Oquab M et al. (2024) DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research (TMLR)*.
6. Shang J et al. (2024) Theia: Distilling diverse vision foundation models for robot learning in *Conference on Robot Learning (CoRL)*.
7. Qin Y et al. (2025) Chain-of-visual-thought: Teaching vlms to see and think better with continuous visual tokens. *arXiv preprint arXiv:2511.19418*.
8. Tu R et al. (2026) SG-VLA: Learning spatially-grounded vision-language-action models for mobile manipulation. *arXiv preprint arXiv:2603.22760*.
9. Shukla A, Tao S, Su H (2025) ManiSkill-HAB: A benchmark for low-level manipulation in home rearrangement tasks in *International Conference on Learning Representations (ICLR)*.
10. Yu Q et al. (2026) Affordancevla: A vision-language-action model empowering action generation through affordance-aware understanding. *arXiv preprint arXiv:2606.06155*.
11. Gu J et al. (2024) RT-Trajectory: Robotic task generalization via hindsight trajectory sketches in *International Conference on Learning Representations (ICLR)*.
12. Perez E, Strub F, de Vries H, Dumoulin V, Courville A (2018) FiLM: Visual reasoning with a general conditioning layer in *Proceedings of the AAAI Conference on Artificial Intelligence*.
13. Black K et al. (2025) π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*.
14. Brohan A et al. (2023) RT-1: Robotics transformer for real-world control at scale in *Robotics: Science and Systems (RSS)*.
15. Zitkovich B et al. (2023) RT-2: Vision-language-action models transfer web knowledge to robotic control in *Proceedings of the 7th Conference on Robot Learning*, Proceedings of Machine Learning Research. Vol. 229, pp. 2165–2183.
16. Kim MJ et al. (2025) OpenVLA: An open-source vision-language-action model in *Proceedings of the 8th Conference on Robot Learning*, Proceedings of Machine Learning Research. Vol. 270.
17. Chi C et al. (2023) Diffusion policy: Visuomotor policy learning via action diffusion in *Robotics: Science and Systems (RSS)*.
18. Zhou Y, Barnes C, Lu J, Yang J, Li H (2019) On the continuity of rotation representations in neural networks in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. pp. 5745–5753.
19. Xiao B et al. (2024) Florence-2: Advancing a unified representation for a variety of vision tasks in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.
20. Ding M et al. (2022) Davit: Dual attention vision transformers in *European Conference on Computer Vision (ECCV)*. (Springer), pp. 74–92.
21. Tabernik D, Muhovič J, Urbas M, Skočaj D (2024) Center direction network for grasping point localization on cloths. *IEEE Robotics and Automation Letters* 9(10):8913–8920.
22. Hugging Face (2026) Lerobot (https://github.com/huggingface/lerobot). GitHub repository, accessed 13 July 2026.
23. Cadene R et al. (2026) Lerobot: An open-source library for end-to-end robot learning in *The Fourteenth International Conference on Learning Representations (ICLR)*.

Model	Task-stage score			Grasp | ID
	Target ID	Grasp	Fold	(cond. score)
Baseline	15/16 (93.8%)	9.5/16 (59.4%)	3/16 (18.8%)	63.3%
Control (+4K)	15/16 (93.8%)	**10.5/16 (65.6%)**	**5.5/16 (34.4%)**	**70.0%**
Implicit	**16/16 (100%)**	5.5/16 (34.4%)	3/16 (18.8%)	34.4%
Explicit Concat	11/16 (68.8%)	6.5/16 (40.6%)	3/16 (18.8%)	59.1%
Explicit Sel. Layers	**16/16 (100%)**	**10.5/16 (65.6%)**	4.5/16 (28.1%)	65.6%

Table 20. Conditional analysis of *Cloth Corner-Folding*. Task-stage columns repeat the scores from Table 19; Grasp and Fold are partial-credit scores, reported as mean scores. **Grasp | ID** is the mean grasp score restricted to correctly identified targets (grasp score divided by the number of correct identifications), isolating grasp accuracy from identification. Bold marks the best value per column. The control and explicit selected-layer guidance attain the highest grasp scores given identification (70.0% and 65.6%), whereas implicit guidance, despite perfect identification (16/16), attains the lowest (34.4%): on this task its identification advantage does not carry through to grasping.