Visual Thought Tokens: Decoding and Querying Frozen VLM Representations for Spatial Grounding and Guidance

Jose Edgar Hernandez Cancino Estrada
jh65312@student.uni-lj.si
GitHub Repository

May 2026

Abstract

Pretrained vision-language models (VLMs) are increasingly used as perceptual backbones in multimodal reasoning and vision-language-action (VLA) systems. Given an image and a language instruction, a VLM encoder produces a sequence of latent tokens that combines visual, spatial, and semantic information.

This work studies the encoded representation of a frozen VLM as an information substrate. The central question is whether task-relevant visual information can be recovered from the internal token sequence without modifying the VLM itself.

We introduce a lightweight Visual-Thought representation mechanism that maps frozen VLM tokens into task-specific latent tokens. These visual-thought tokens are trained to align the frozen representation with downstream spatial objectives while keeping the VLM encoder fixed. We use this mechanism for two purposes:

- reconstructing expert visual signals, including patch-level representations and dense spatial maps;
- grounding the VLM image tokens associated with instruction-conditioned, action-relevant regions for possible VLA guidance.

Through these directions, we frame the VLM token sequence as a structured representation space that can be decoded and queried. Since many VLA policies already consume VLM representations, extracting guidance from the same latent space may support visual-token selection, attention reweighting, and improved interpretability of action generation.

1 Frozen X-VLA Representation

We use the pretrained X-VLA model [5] as the frozen source of multimodal representations. X-VLA is a vision-language-action policy built around a Florence-2-based vision-language backbone [4]. Although the full checkpoint includes action-generation modules, this work uses only the intermediate VLM encoder representation. All X-VLA parameters are kept fixed; the trainable modules introduced in this work operate only on top of the frozen token sequence.

Let the input at observation time t be

$$u_t = (I_t, \ell), \tag{1}$$

where I_t is the image observation and ℓ is the language instruction. The frozen VLM encoder is denoted by

$$\Phi_\theta^{\mathrm{VLM}}, \tag{2}$$

with pretrained parameters θ. It maps the image-language input into a contextualized multimodal token sequence of length N and hidden dimension D_{VLM}:

$$Z_t = \Phi_\theta^{\mathrm{VLM}}(I_t, \ell) = [z_{t,1}, z_{t,2}, \ldots, z_{t,N}] \in \mathbb{R}^{N \times D_{\mathrm{VLM}}}. \tag{3}$$

The sequence contains 50 image-side tokens and N_L language tokens. The image-side tokens appear first in the cached VLM feature sequence. They consist of one pooled, non-spatial image token followed by a spatial patch-token grid of 49 tokens:

$$50 = 1 + 49, \qquad 49 = 7 \times 7. \tag{4}$$

Thus, for the X-VLA cache used here, the token sequence can be viewed as

$$Z_t = \left[z_t^{\mathrm{pool}}; Z_t^I; Z_t^L \right], \tag{5}$$

where

$$z_t^{\mathrm{pool}} \in \mathbb{R}^{1 \times D_{\mathrm{VLM}}}, f \;\; Z_t^I \in \mathbb{R}^{49 \times D_{\mathrm{VLM}}}, \;\; Z_t^L \in \mathbb{R}^{50 \times D_{\mathrm{VLM}}}. \tag{6}$$

The X-VLA image feature sources include one global pooled token and one patch-grid source. After removing the pooled token, the remaining image-side tokens form a square grid of 49 patch tokens, which we reshape as

$$Z_t^I \in \mathbb{R}^{7 \times 7 \times D_{\mathrm{VLM}}}. \tag{7}$$

These 7×7 tokens provide the native spatial support used for token-grounding supervision.

The cached hidden states are produced by a frozen X-VLA forward pass that receives both language tokens and image pixels:

$$Z_t = \Phi_\theta^{\mathrm{VLM}}(I_t, \ell). \tag{8}$$

Therefore, Z_t is the output token sequence of the frozen multimodal encoder conditioned on the current image-language input. For this reason the spatial image tokens Z_t^I are treated as instruction-conditioned image-region representations.

Finally, in addition to Z_t, the X-VLA/Florence visual backbone exposes intermediate visual feature maps from

its DaViT vision tower. We denote such an intermediate visual feature grid by

$$R_t = [r_{t,1}, \ldots, r_{t,N_R}] \in \mathbb{R}^{N_R \times D_R}. \tag{9}$$

Unlike Z_t, which is produced after multimodal fusion with the language instruction, R_t is extracted from an intermediate visual stage of the frozen vision tower. It therefore provides higher-resolution visual structure, but is not itself the final language-conditioned VLM token sequence.

2 Methodology

The objective of this work is to study whether useful spatial and task-relevant information can be extracted from Z_t without updating Φ_θ^{VLM}. We therefore treat Z_t as a fixed representation substrate and train lightweight modules above it.

For all tasks, we use the same representation-alignment architectural template, which we call the *Visual-Thought Decoder* f_ψ. The decoder maps the frozen VLM token sequence into a learned task-specific token sequence:

$$T_t = f_\psi(Z_t), \qquad T_t = [\tau_{t,1}, \tau_{t,2}, \ldots, \tau_{t,M}] \in \mathbb{R}^{M \times D_T}, \tag{10}$$

where M is the number of visual-thought tokens and D_T is their hidden dimension. We refer to T_t as the *visual-thought token sequence*.

We use these aligned representations in three different settings:

- **Expert feature distillation**: The visual thought tokens T_t are directly matched to approximate an external expert model's representations, in this case DINOv2 [2] (Section 2.2.1).
- **Dense map reconstruction**: T_t reorganizes information from the frozen VLM token sequence into a spatial representation that can be decoded by a reconstruction head. In this work, we reconstruct CeDiRNet [3] maps for cloth-corner grasp localization (Section 2.2.2).
- **VLM-token grounding**: Visual-thought tokens are projected into grounding queries and scored against the frozen VLM image-token grid to produce instruction-conditioned spatial relevance maps. This setting tests whether the model can learn to identify action-relevant image tokens directly within the representation space used by X-VLA (Section 2.2.3).

2.1 The Visual-Thought Decoder

The visual-thought tokens are obtained by iteratively querying projected versions of the frozen VLM token sequence Z_t.

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

$$\tilde{Z}_t = p_\rho(Z_t), \qquad \tilde{Z}_t \in \mathbb{R}^{N \times D_T}. \tag{11}$$

More explicitly, the projection is implemented as

$$p_\rho(Z_t) = Z_t W_s^\top + b_s + \sigma\left(\text{LN}(Z_t)W_1^\top + b_1\right) W_2^\top + b_2. \tag{12}$$

Here,

$$W_s \in \mathbb{R}^{D_T \times D_{\text{VLM}}}, \quad W_1 \in \mathbb{R}^{D_H \times D_{\text{VLM}}}, \quad W_2 \in \mathbb{R}^{D_T \times D_H}. \tag{13}$$

The matrix W_s defines a learned linear skip projection, σ is GELU, and the MLP branch maps from D_{VLM} to an intermediate hidden dimension D_H and then to D_T.

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

$$Q^{(0)} = \left[q_1^{(0)}, \ldots, q_M^{(0)}\right] \in \mathbb{R}^{M \times D_T}, \tag{14}$$

that start the L layers query-refinement decoding process. In each iteration:

1. Each decoder layer applies pre-normalized self-attention over the visual-thought queries:

$$\bar{Q}^{(\ell)} = Q^{(\ell-1)} + \text{SelfAttn}\left(\text{LN}(Q^{(\ell-1)})\right). \tag{15}$$

2. The updated queries attend over the projected VLM tokens and are added back through a residual connection:

$$U^{(\ell)} = \bar{Q}^{(\ell)} + \text{CrossAttn}\left(\bar{Q}^{(\ell)}, \tilde{Z}_t, \tilde{Z}_t\right). \tag{16}$$

3. The queries are refined using a pre-normalized Transformer-style feed-forward network:

$$Q^{(\ell)} = U^{(\ell)} + \text{FFN}\left(\text{LN}(U^{(\ell)})\right). \tag{17}$$

After L stacked decoder layers, the final visual-thought token sequence is

$$T_t = Q^{(L)} = [\tau_{t,1}, \ldots, \tau_{t,M}] \in \mathbb{R}^{M \times D_T}. \tag{18}$$

The number of visual-thought tokens M depends on the downstream task. For DINOv2 feature distillation, M is chosen to match the DINOv2 teacher token sequence length. For dense prediction tasks, M is chosen as a spatial decoder grid. For token-grounding, we instantiate the decoder with $M = 1$, producing a single compact visual-thought token per observation.

The general architecture diagram is shown in Fig. 1.

2.2 Task Heads and Objectives

2.2.1 Expert Feature Distillation

In expert feature distillation, the visual-thought decoder is trained to directly reproduce the token sequence of a frozen external vision model. In this work, the expert is DINOv2 ViT-B/14 [2]. Let

$$E_t = [e_{t,1}, \ldots, e_{t,N_{\text{DINO}}}] \in \mathbb{R}^{N_{\text{DINO}} \times D_{\text{DINO}}} \tag{19}$$

denote the frozen DINOv2 teacher token sequence for image I_t.

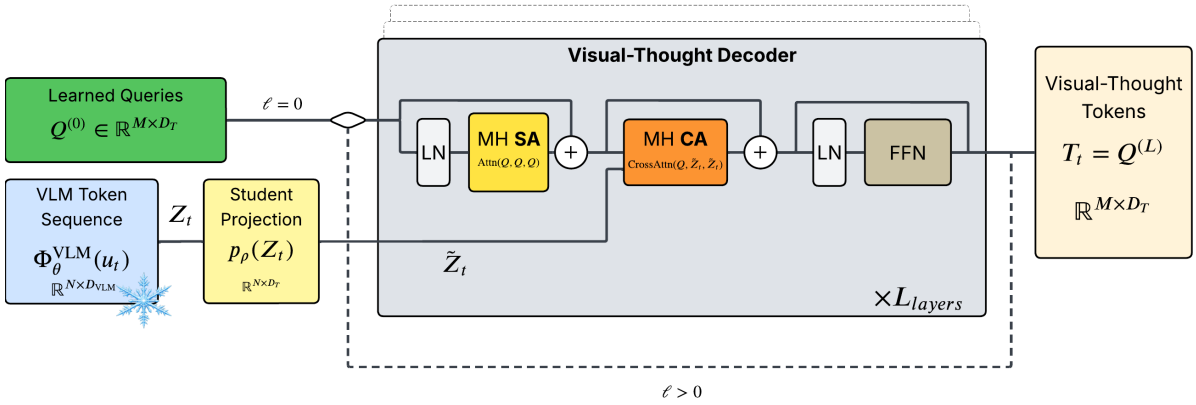


Figure 1: Visual-thought decoder architecture. The VLM token sequence Z_t is projected into the decoder dimension. The query tokens pass through stacked decoder blocks with query self-attention, residual cross-attention, and an FFN refinement. The final output T_t is the visual-thought token sequence. The frozen VLM encoder Φ_θ^{VLM} remains fixed during training.

For this objective, the visual-thought decoder output space is chosen to match the teacher token sequence:

$$M = N_{\text{DINO}}, \qquad D_T = D_{\text{DINO}}. \tag{20}$$

No additional prediction head is used. The decoder output is compared directly against the DINOv2 tokens:

$$\hat{E}_t = T_t = f_\psi(Z_t), \qquad \hat{E}_t \in \mathbb{R}^{N_{\text{DINO}} \times D_{\text{DINO}}}. \tag{21}$$

The feature-distillation objective is the mean squared error over the full token sequence:

$$\mathcal{L}_{\text{feat}} = \frac{1}{N_{\text{DINO}} D_{\text{DINO}}} \left\| \hat{E}_t - E_t \right\|_F^2. \tag{22}$$

This experiment tests whether the frozen X-VLA token sequence contains enough visual information to reconstruct the representation space of a strong external vision backbone.

2.2.2 Dense Map Reconstruction

In dense map reconstruction, the visual-thought tokens are decoded into a spatial expert map. In this work, we use CeDiRNet [3] as the dense expert for cloth manipulation. Let the frozen CeDiRNet teacher be denoted by

$$\Psi_\phi^{\text{CED}}, \tag{23}$$

with parameters ϕ kept fixed. Given image observation I_t, the teacher produces a dense map

$$Y_t^{\text{CED}} = \Psi_\phi^{\text{CED}}(I_t) \in \mathbb{R}^{C_{\text{CED}} \times H \times W}. \tag{24}$$

We supervise a subset of the CeDiRNet output channels corresponding to the geometric fields used for cloth-corner localization:

$$\mathcal{C} = \{\texttt{sin\_y}, \texttt{cos\_x}, \texttt{radius}\}. \tag{25}$$

The dense reconstruction target is therefore

$$Y_t^\star = Y_t^{\text{CED}}[\mathcal{C}] \in \mathbb{R}^{3 \times H \times W}. \tag{26}$$

The visual-thought decoder maps the X-VLA sequence Z_t into a spatial token grid:

$$T_t = f_\psi(Z_t) \in \mathbb{R}^{M \times D_T}, \qquad M = H_g W_g. \tag{27}$$

Here, $H_g \times W_g$ determines the latent grid resolution used by the dense reconstruction head. Each of the M visual-thought tokens is projected into the dense-head hidden dimension:

$$H_t = p_\gamma(T_t) \in \mathbb{R}^{M \times D_H}, \tag{28}$$

and then reshaped into a spatial feature map:

$$\bar{H}_t = \text{reshape}(H_t) \in \mathbb{R}^{D_H \times H_g \times W_g}. \tag{29}$$

A lightweight convolutional dense head refines this latent grid and predicts the selected CeDiRNet channels at the latent grid resolution:

$$\tilde{Y}_t^{\text{CED}} = g_\omega(\bar{H}_t) \in \mathbb{R}^{3 \times H_g \times W_g}. \tag{30}$$

The latent-grid prediction is then bilinearly upsampled to the teacher resolution:

$$\hat{Y}_t^{\text{CED}} = \text{Upsample}\left(\tilde{Y}_t^{\text{CED}}, H, W\right) \in \mathbb{R}^{3 \times H \times W}. \tag{31}$$

The model is trained with mean squared reconstruction error:

$$\mathcal{L}_{\text{CED}} = \frac{1}{3HW} \left\| \hat{Y}_t^{\text{CED}} - Y_t^\star \right\|_F^2. \tag{32}$$

This objective tests whether the frozen X-VLA token sequence Z_t contains enough cloth-relevant geometric information to reconstruct dense CeDiRNet supervision through the visual-thought representation.

2.2.3 VLM-Token Grounding

Inspired by token-level pointing methods such as MolmoPoint [1], the grounding head is trained to identify language-conditioned relevant image regions by scoring the spatial image-token subset of the frozen X-VLA representation.

As described in Section 1, the spatial image-token subset of Z_t is

$$Z_t^I = \left[z_{t,1}^I, \ldots, z_{t,49}^I\right] \in \mathbb{R}^{49 \times D_{\text{VLM}}}. \tag{33}$$

The grounding objective is to produce a relevance score map over this spatial image-token subset, indicating which internal X-VLA image regions are relevant for a given language-conditioned task.

The full VLM token sequence is first processed by the visual-thought decoder with $M = 1$, producing a single visual-thought token:

$$T_t = f_\psi(Z_t) = [\tau_t] \in \mathbb{R}^{1\times D_T}. \tag{34}$$

This visual-thought token is projected into a grounding-query space:

$$q_t^G = g_q(\tau_t) \in \mathbb{R}^{D_G}. \tag{35}$$

Each spatial image token is projected into the same grounding space as a key, including fixed two-dimensional sinusoidal positional encodings added to the spatial image-token positions:

$$k_{t,j}^G = g_k\left(z_{t,j}^I + e_j^I\right) \in \mathbb{R}^{D_G}, \qquad j = 1, \ldots, 49. \tag{36}$$

Here, g_q and g_k are implemented as layer normalization followed by learned linear projections, and e_j^I denotes the positional encoding associated with the j-th spatial image token.

The relevance logit for image token j is computed by scaled dot product over the 49 spatial image tokens:

$$\begin{aligned} a_{t,j} &= \frac{(q_t^G)^\top k_{t,j}^G}{\sqrt{D_G}}. \\ a_t &= [a_{t,1}, \ldots, a_{t,49}] \in \mathbb{R}^{49}. \end{aligned} \tag{37}$$

These logits define a relevance map over the native X-VLA image-token grid and can be reshaped for visualization:

$$A_t = \mathrm{reshape}(a_t) \in \mathbb{R}^{7\times 7}. \tag{38}$$

Token refinement for improved scoring. The coarse X-VLA image-token grid has resolution 7×7, so a single token may cover multiple nearby objects or action-relevant regions. To improve localization, we refine the top coarse candidates using intermediate DaViT features from the frozen X-VLA vision tower:

$$R_t \in \mathbb{R}^{14\times 14\times D_R}. \tag{39}$$

Each coarse cell corresponds to a 2×2 region in this refinement grid.

Given the coarse logits a_t, the model selects K candidate coarse cells:

$$\mathcal{C}_t = \mathrm{TopK}(a_t). \tag{40}$$

For each selected candidate, we gather a local DaViT neighborhood $R^{\mathrm{loc}} \in \mathbb{R}^{S\times D_R}$. In the final configuration this neighborhood is 4×4, so $S = 16$ visual feature tokens per coarse candidate.

The local features are projected into refinement keys,

$$H^R = g_R(R^{\mathrm{loc}}) \in \mathbb{R}^{S\times D_G}, \tag{41}$$

and the visual-thought token is projected into a refinement query,

$$q^R = g_r(\tau_t) \in \mathbb{R}^{D_G}. \tag{42}$$

The refinement query is updated by local cross-attention over the candidate neighborhood:

$$\bar{q}^R = \mathrm{LN}\left(q^R + W_R\mathrm{CrossAttn}\left(q^R, H^R, H^R\right)\right). \tag{43}$$

Although the query attends over the full local context H^R, the refinement score is evaluated only on the $U = 4$ fine positions corresponding to the 2×2 region inside the selected coarse cell. Let h_u^R denote the projected key for one of these four fine positions.

$$r_u = \frac{(\bar{q}^R)^\top h_u^R}{\sqrt{D_G}}, \qquad u = 1, \ldots, 4. \tag{44}$$

For candidate c, the final selected-candidate score combines its coarse logit with its local refinement support:

$$b_c = a_{t,c} + \log \sum_{u=1}^{4} \exp(r_u). \tag{45}$$

Collecting these scores over the K selected candidates gives

$$b_t = [b_{t,1}, \ldots, b_{t,K}] \in \mathbb{R}^K. \tag{46}$$

During training, candidate selection is initially teacher-forced with the ground-truth coarse cell. Later, candidates are selected from the predicted top-K, but the ground-truth coarse cell is inserted if absent so that the refinement target remains defined.

Training objective. The coarse target is a binary relevance map over the 7×7 X-VLA image-token grid. Let $y_{t,j} \in \{0, 1\}$ indicate whether coarse cell j is positive. We train the coarse logits with a sigmoid-margin loss. Defining

$$\mu_+ = \mathrm{logit}(\pi_+), \qquad \mu_- = \mathrm{logit}(\pi_-), \tag{47}$$

the loss penalizes positive cells below μ_+ and hard negative cells above μ_-:

$$\begin{aligned} \mathcal{L}_{\mathrm{coarse}} = \frac{1}{N_t^{\mathrm{loss}}}\Bigg[&\sum_{j:y_{t,j}=1} w_+ \max(0, \mu_+ - a_{t,j}) \\ &+ \sum_{j\in\mathcal{H}_t} \max(0, a_{t,j} - \mu_-)\Bigg]. \end{aligned} \tag{48}$$

Here, $\mathcal{H}_t$ contains the K_{neg} negative cells with the largest negative-margin violations, i.e. the negative cells whose logits most exceed μ_-, and N_t^{loss} is the number of cells included in the loss.

For refinement, the model predicts two discrete targets. First, it predicts which of the K selected coarse candidates contains the ground-truth point. Second, inside the correct coarse candidate, it predicts which of the four fine positions in the corresponding 2×2 refinement region contains the point. Let $b_t \in \mathbb{R}^K$ be the candidate scores and let $r_t^\star \in \mathbb{R}^4$ be the fine-subpatch scores for the correct candidate. The refinement loss is

$$\mathcal{L}_{\mathrm{ref}} = \frac{1}{2}\,\mathrm{CE}(b_t, c_t^\star) + \frac{1}{2}\,\mathrm{CE}(r_t^\star, u_t^\star). \tag{49}$$

Here, $c_t^\star$ denotes the position of the ground-truth coarse cell within the selected top-K candidate list, and $u_t^\star$ is the ground-truth fine subpatch inside that coarse cell.

The total grounding objective is

$$\mathcal{L}_{\mathrm{ground}} = \lambda_{\mathrm{coarse}}\mathcal{L}_{\mathrm{coarse}} + \lambda_{\mathrm{ref}}\mathcal{L}_{\mathrm{ref}}. \tag{50}$$

3 Experiments

The Visual-Thought framework is implemented in three experimental settings: (i) DINOv2 token feature distillation, (ii) dense-map reconstruction of CeDiRNet maps, (iii) VLM-token grounding with visual thought queries.

3.1 General Experimental Setup

All experiments use the Hugging Face checkpoint `lerobot/xvla-base` as the frozen X-VLA source model. The cached VLM token sequence Z_t is used as the student input. Training updates only the visual-thought decoder and the task-specific output modules.

We evaluate three settings. DINOv2 feature distillation uses the cube-manipulation dataset `soarm101_pickplace_multicolor_v1_7p5hz`. CeDiRNet dense reconstruction uses the cloth-manipulation dataset `soarm101_square_cloth_corner_to_box_7p5hz`.

For VLM-token grounding, we use a reviewed cube pick-and-place grounding set consisting of 2,191 unique images and 4,069 image-instruction grounding examples: 2,545 positive examples and 1,524 negative examples. The positive labels cover orange cube, blue cube, black cube, and white box targets. The examples are generated from cube pick-and-place instructions: `"Pick up {color} cube and place inside white box"`, with negative rows used when the instructed target is absent or marked as non-relevant.

Each run uses a held-out validation split. Reconstruction experiments report validation loss against frozen expert outputs, while grounding experiments report validation loss against held-out grounding targets.

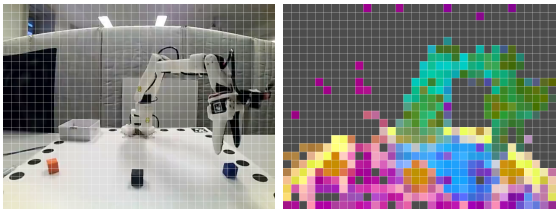
3.2 DINOv2 Token Feature Distillation

For DINOv2 feature distillation, we train the visual-thought decoder on the cube-manipulation dataset using frozen DINOv2 ViT-B/14 tokens as targets. The decoder output dimension is 768, matching the expert's dimension. No additional prediction head is used.

Figure 2 shows the spatial DINOv2 and distilled tokens projected into the same 3-D PCA space, with the first three components interpreted as RGB. These spatial feature maps provide a qualitative comparison of whether the distilled tokens preserve the expert representation's coarse object boundaries and semantic structure.

Run	Layers	Dim.	Params	Train loss	Val loss
1	2	768	**28.34M**	0.3593	0.3422
2	3	768	37.79M	0.3425	0.3299
3	4	768	47.24M	0.3353	0.3221
4	8	768	85.04M	**0.3176**	**0.3080**

Table 1: DINOv2 token-distillation results for different decoder depths. All runs use 32 attention heads and token dimension 768. Deeper decoders reduce train and validation MSE, with the 8-layer model performing best at the cost of 85.04M trainable parameters.



Input image Expert DINOv2

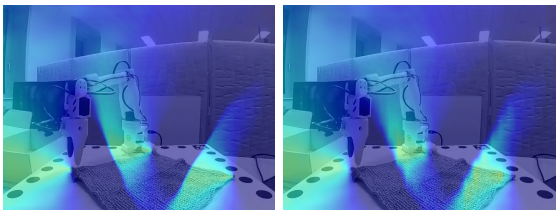
Distilled

Figure 2: Qualitative visualization of DINOv2 feature distillation. Expert and predicted patch tokens are projected into the same 3-D PCA space and visualized as RGB feature maps. Consistent region coloring indicates preservation of the expert feature organization.

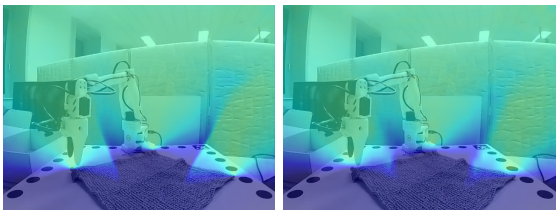
3.3 CeDiRNet Dense Reconstruction

This experiment evaluates dense geometric-map reconstruction on cloth-graspable regions provided by frozen CeDiRNet outputs over the three supervised channels `sin_y`, `cos_x`, and `radius`. We vary the visual-thought decoder depth and output grid resolution, while keeping X-VLA and the CeDiRNet teacher frozen.

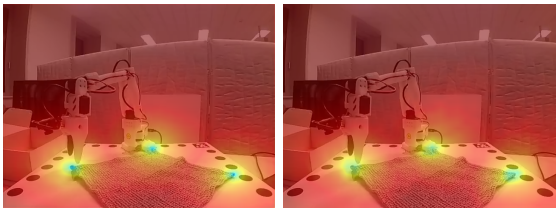
Figure 3 compares predicted maps against the teacher maps. The predictions recover the main spatial structure of the CeDiRNet outputs.



Teacher `sin_y` Predicted `sin_y`



Teacher `cos_x` Predicted `cos_x`



Teacher `radius` Predicted `radius`

Figure 3: Qualitative CeDiRNet dense-map reconstruction. Each row compares a frozen teacher channel against its corresponding visual-thought prediction.

Num. Layers	Grid Size	Decoder Params	Head Params	Train MSE	Val MSE
1	16×16	5.15M	0.31M	0.0109	0.0162
1	32×32	5.25M	0.31M	**0.0093**	**0.0154**
2	16×16	5.42M	0.31M	0.0110	0.0156
2	32×32	5.52M	0.31M	0.0097	0.0157
4	16×16	5.95M	0.31M	0.0114	0.0181
4	32×32	6.04M	0.31M	0.0112	0.0177
8	16×16	7.00M	0.31M	0.0113	0.0183
8	32×32	7.10M	0.31M	0.0106	0.0174

Table 2: CeDiRNet dense reconstruction ablation over decoder depth and visual-thought grid resolution. The best validation MSE is obtained with a 1-layer decoder and a 32×32 grid.

3.4 VLM-Token Grounding

The token-grounding experiment shows that a single visual-thought token can identify instruction-relevant image tokens inside the frozen X-VLA representation.

We evaluate coarse grounding over the 7×7 X-VLA image-token grid. The primary validation metrics are false positives, false negatives, precision, recall, and, because false positives are especially costly for downstream action guidance, $F_{0.25}$, a precision-weighted F_β-score with $\beta = 0.25$:

$$F_{0.25} = (1 + 0.25^2)\frac{\text{Precision} \cdot \text{Recall}}{0.25^2\text{Precision} + \text{Recall}}. \qquad (51)$$

Table 3 reports the coarse-scale ablation at $60k$ training steps. The best precision-weighted score is obtained by increasing the decoder depth to four layers and using a stricter negative margin. The selected configuration uses decoder dimension 256, four decoder layers, $p^+ = 0.60$, $p^- = 0.15$, and learning rate 5×10^{-6}.

Table 4 compares checkpoints for the selected configuration. The $40k$ checkpoint gives the lowest false-positive count and validation loss, while the $51k$ checkpoint gives the highest precision and $F_{0.25}$.

The qualitative examples in Figures 4 and 5 illustrate two complementary behaviors of the grounding head. Figure 4 shows view consistency: for the same pre-grasp moment, the same model identifies the instructed cube as the relevant region from both camera perspectives. Figure 5 shows task-phase sensitivity: in later executions where the cube has already been grasped, the predicted relevance shifts from the source object to the box region.

References

[1] Christopher Clark, Yue Yang, Jae Sung Park, Zixian Ma, Jieyu Zhang, Rohun Tripathi, Mohammadreza Salehi, Sangho Lee, Taira Anderson, Winson Han, and Ranjay Krishna. MolmoPoint: Better pointing for VLMs with grounding tokens. *arXiv preprint arXiv:2603.28069*, 2026.

[2] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[3] Domen Tabernik, Jon Muhovič, Matej Urbas, and Danijel Skočaj. Center direction network for grasping point localization on cloths. *IEEE Robotics and Automation Letters*, 2024.

[4] Bin Xiao, Haiping Wu, Weijian Xu, Xiyang Dai, Houdong Hu, Yumao Lu, Michael Zeng, Ce Liu, and Lu Yuan. Florence-2: Advancing a unified representation for a variety of vision tasks. *arXiv preprint arXiv:2311.06242*, 2023.

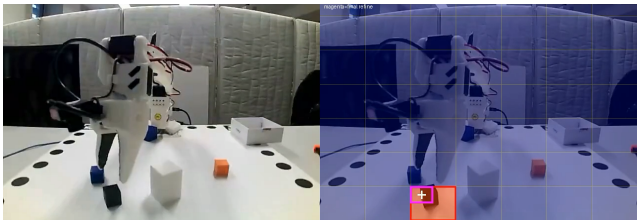
[5] Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, Ya-Qin Zhang, Jiangmiao Pang, Jingjing Liu, Tai Wang, and Xianyuan Zhan. X-VLA: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. *arXiv preprint arXiv:2510.10274*, 2025.

	Group	LR	Dec	Layers	p^+	p^-	Params (M)	FP ↓	FN ↓	Prec. ↑	Rec. ↑	$F_{0.25}$ ↑	Val loss ↓
	LR	$\mathbf{8\times10^{-6}}$	256	2	0.60	0.25	10.68	154	**476**	0.843	**0.634**	0.827	1.094
→	LR	$\mathbf{5\times10^{-6}}$	256	2	0.60	0.25	10.68	**147**	481	**0.848**	0.631	**0.831**	**0.936**
	Decoder dim	5×10^{-6}	**64**	2	0.60	0.25	7.58	164	539	0.823	0.586	0.804	**0.825**
→	Decoder dim	5×10^{-6}	**256**	2	0.60	0.25	10.68	147	481	0.848	0.631	0.831	0.936
	Decoder dim	5×10^{-6}	**512**	2	0.60	0.25	18.49	**138**	**467**	**0.858**	**0.641**	**0.841**	1.016
	Layers	5×10^{-6}	256	**1**	0.60	0.25	9.63	165	485	0.832	0.627	0.816	0.939
	Layers	5×10^{-6}	256	**2**	0.60	0.25	10.68	147	481	0.848	0.631	0.831	0.936
	Layers	5×10^{-6}	256	**3**	0.60	0.25	11.73	181	**439**	0.827	**0.663**	0.815	0.940
→	Layers	5×10^{-6}	256	**4**	0.60	0.25	12.79	**137**	476	**0.858**	0.634	**0.840**	**0.884**
	Margin p^+	5×10^{-6}	256	4	**0.65**	0.25	12.79	164	**444**	0.840	**0.659**	0.826	0.916
→	Margin p^+	5×10^{-6}	256	4	**0.60**	0.25	12.79	**137**	476	**0.858**	0.634	**0.840**	0.884
	Margin p^-	5×10^{-6}	256	4	0.60	**0.25**	12.79	137	**476**	0.858	**0.634**	0.840	0.884
→	Margin p^-	5×10^{-6}	256	4	0.60	**0.15**	12.79	**122**	499	**0.868**	0.617	**0.848**	0.997

Table 3: Coarse-scale validation ablation at 60k training steps. All runs use sigmoid-margin coarse loss, batch size 32, decoder dropout 0.3, hard-negative top-$k = 10$, refinement weight 1.0, and positive weight $p_w = 2.0$. Bold hyperparameter values mark the variable within each ablation block; bold metrics mark the best result within each block. Since false positives are most costly downstream, $F_{0.25}$ is used as the primary aggregate metric because it weights precision more strongly than recall.

Checkpoint	Step	FP ↓	FN ↓	Prec. ↑	Rec. ↑	$F_{0.25}$ ↑	Val loss ↓
20k	20000	105	715	0.848	0.451	0.806	0.939
40k	40000	**98**	571	0.882	0.561	0.853	**0.937**
best	51000	104	518	**0.883**	0.602	**0.859**	0.976
60k	60000	122	**499**	0.868	**0.617**	0.848	0.997

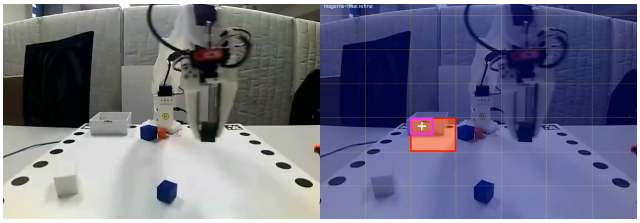
Table 4: Checkpoint comparison for the selected 4-layer grounding model with decoder dimension 256, $p^+ = 0.60$, $p^- = 0.15$, and learning rate 5×10^{-6}. The 40k checkpoint gives the lowest false-positive count and validation loss, while the 51k best checkpoint gives the highest precision-weighted $F_{0.25}$.



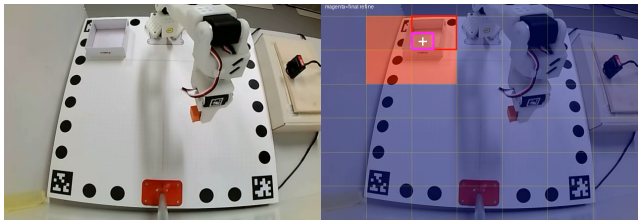
Main-view prediction

Top-view prediction

Figure 4: Qualitative token-grounding predictions from the same model on two synchronized views of the same moment. For the instruction "`Pick up black cube and place inside white box`," the grounding head assigns high relevance to the black cube region from both the main camera view and the top camera view. The overlay shows the coarse 7×7 X-VLA image-token grid and the local refinement prediction inside the selected coarse cell.



Post-grasp execution, main view



Post-grasp execution, top view

Figure 5: Qualitative token-grounding predictions for two different post-grasp executions. In both examples, the cube has already been grasped, so the action-relevant region is no longer the source cube but the placement area specified by the instruction. The same grounding model correctly shifts relevance to the box region (regardless of the camera view), showing that the predicted token relevance depends on the current trajectory state and instruction.