SEEING ACTION-CRITICAL EVIDENCE: TOKEN PRUNING WITH INTERNAL ACTION PRIORS FOR VISION-LANGUAGE-ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-Language-Action (VLA) models translate visual observations and language instructions into continuous robotic actions. However, their dense visual token sequences introduce substantial decoding overhead, hindering efficient real-world deployment. Recent *trainable* token pruning methods reduce this cost, yet their pruning criteria remain largely *action-agnostic*: they often retain task-irrelevant background tokens while discarding fine-grained visual cues essential for precise control. In this work, we investigate whether pretrained VLA models internally encode action-aware signals for token selection. Our analysis suggests that: (i) action-critical visual tokens show an association with value vectors in deep Feed-Forward Network (FFN) layers; and (ii) these value vectors define actionable residual subspaces whose projection energy approximates local action saliency. Motivated by these insights, we propose **ActPruner**, a novel Action-Anchored Dynamic Token Pruning approach, which constructs an action-aware pruning metric from intrinsic action priors encoded in deep FFN layers. Technically, ActPruner constructs an Action Information Extraction Module (AIEM) with three progressive steps: Deep Prior Extraction to extract compact action anchors from deep FFN value vectors, Semantic Space Adaptation to align them to shallow pruning layers, and Instruction Modulation to generate action-conditioned pruning queries. This enables dynamic preservation of manipulation-relevant affordance regions while suppressing redundant background tokens. Extensive experiments on VLA-based robotic manipulation benchmarks verify the superiority of our ActPruner over state-of-the-art VLA pruning methods. Code is available in the supplementary material.

1 INTRODUCTION

Vision-Language-Action (VLA) models have recently demonstrated remarkable capabilities in robotic control by integrating visual perception, language understanding, and continuous action generation into a unified policy Kim et al. (2024); Zitkovich et al. (2023); Zheng et al. (2025); Wang et al. (2026). Given embodied observations and natural language instructions, these models can directly predict low-level robotic actions, thus enabling flexible and generalizable control across diverse manipulation tasks. However, dense visual observations are represented as long sequences of visual tokens and are repeatedly processed during the decoding stage, which leads to significant memory and latency costs, making it difficult to deploy VLA models on resource-constrained robotic platforms Yu et al. (2025).

To mitigate this computational bottleneck, visual token pruning has emerged as a promising direction. Early attempts largely inherit mature token reduction strategies from Vision-Language Models (VLMs) Yang et al. (2025); Xu et al. (2025), where token importance is often estimated from visual saliency, attention responsesa and language-vision similarity. More recent VLA-specific efforts have started to tailor pruning strategies to the action-generation nature of VLA models. In the training-free regime Pei et al. (2025); Ye et al. (2025); Liu et al. (2025); Cheng et al. (2026); Huang et al. (2026), ADP Pei et al. (2025) dynamically gate visual token pruning based on end-effector trajectories to balance computational efficiency and precision. Further, trainable pruning methods Jiang et al. (2025); Qiu et al. (2026); Bendikas et al. (2025) generally achieve better efficiency-accuracy trade-offs by learning sparsity during fine-tuning. For example, LightVLA Jiang et al. (2025) introduces a

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

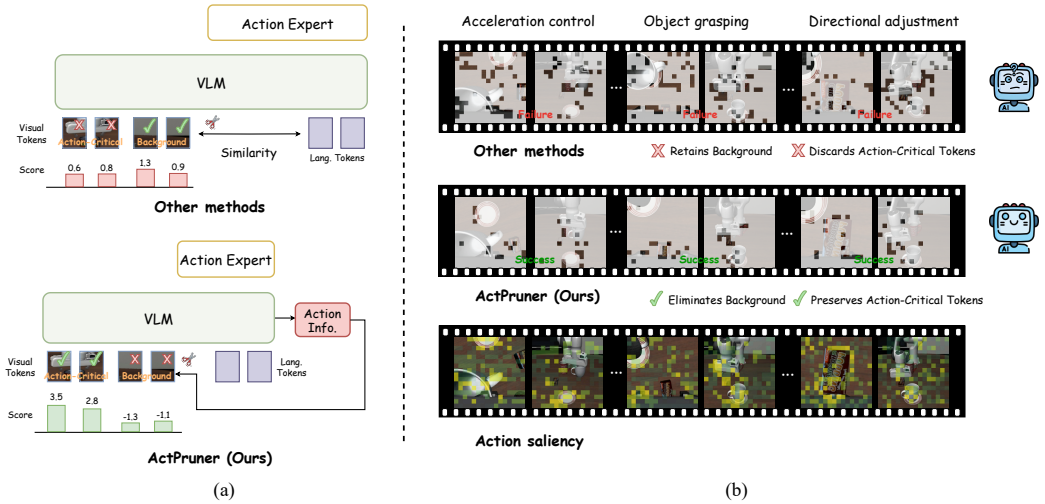

Figure 1: Visual token pruning in VLA models. **(a)** ActPruner uses internal action priors to guide token selection, whereas the compared methods rely on shallow vision-language similarity. **(b)** ActPruner preserves action-critical cues with high leave-one-out action saliency (S_i) and removes redundant background tokens, supporting successful manipulation where the compared methods discard essential cues.

differentiable pruning strategy guided by cross-attention, enabling adaptive token retention without fixed compression ratios. Meanwhile, alternative approaches Bendikas et al. (2025) introduce object- and agent-centric inductive biases into visual tokenization, selectively compressing object-level information while retaining features around the end effector to reduce redundant visual computation.

Despite these advances, most trainable VLA pruning methods still determine token importance from shallow visual representations or learned selection modules without explicitly modeling how each token affects the final continuous action. In other words, their pruning criteria remain fundamentally *action-agnostic*. We identify two critical limitations of these action-agnostic methods: (i) ***Retaining task-irrelevant background tokens***. Since the pruning process is not explicitly conditioned on the current manipulation, visually salient but control-irrelevant regions may be unnecessarily preserved, increasing computational redundancy. More critically, (ii) ***discarding fine-grained operational cues that are essential for precise action control***. During manipulation stages that require precise geometric perception, such as object approach or pre-grasp alignment, even small visual regions around the target object can have a disproportionately large impact on action prediction. However, these regions are often not visually dominant and are therefore prone to being mistakenly discarded by action-agnostic token pruning strategies. As explicitly illustrated in **Fig. 1 (Top)**, existing baselines discarded critical visual cues like the "chocolate pudding", inevitably leading to grasp failures. The above analysis raises a fundamental question: *how can we effectively identify visual tokens that are genuinely crucial for the intended robotic action?*

To answer this question, we investigate whether the internal representations of pretrained VLA models provide action-aware signals to identify action-critical visual tokens. Inspired by mechanistic interpretations of Transformer FFNs as key-value memories Geva et al. (2021) and recent evidence that deep Feed-Forward Network (FFN) layers in VLAs encode specific manipulable action intents Häon et al. (2025), we examine the relationship between action-critical visual tokens and the value vectors in deep FFN layers. Our motivation experiments (Sec. 3.1) suggest that the affinity between high-saliency visual tokens and deep FFN value vectors is more apparent in deep contextualized token representations than in shallow representations, pointing to a cross-layer representation gap. Moreover, our theoretical analysis (Sec.3.3) reveals that FFN value vectors define residual response subspaces, and that the projection energy of a token onto a task-conditioned actionable subspace locally approximates its leave-one-out action saliency. These findings suggest that deep FFN value vectors can serve as an intrinsic and efficient proxy for action-aware visual token selection.

Motivated by these findings, we propose **ActPruner**, an Action-Anchored Dynamic Token Pruning framework for efficient VLA inference. ActPruner constructs an action-aware pruning metric from the value-vector structure of deep FFN layers. Specifically, ActPruner integrates three components. First, Deep Prior Extraction applies Principal Component Analysis (PCA) to extract a compact set of

Action Anchors representing the dominant directions of the deep FFN value-vector subspace. Second, Semantic Space Adaptation uses a lightweight learnable adapter to improve the compatibility of these anchors with the shallow representations used for token pruning. Third, Instruction Modulation conditions the adapted anchors on the text instruction to produce task-sensitive action queries.

The resulting queries compute action-conditioned relevance scores over the visual tokens, allowing ActPruner to retain evidence most relevant to the current manipulation while suppressing redundant background tokens. The retained tokens are then fed into the VLA model for efficient action prediction. Extensive experiments on VLA-based robotic manipulation benchmarks demonstrate that ActPruner consistently outperforms state-of-the-art pruning methods.

Our core contributions are summarized as follows:

- We identify the action-agnostic limitation of existing trainable VLA token pruning methods and introduce a leave-one-out action saliency formulation that characterizes the local functional contribution of each visual token to continuous action prediction.
- We observe an association between action-critical visual tokens and deep FFN value vectors, and theoretically reveal that these value vectors define actionable residual subspaces whose projection energy approximates action saliency.
- We propose **ActPruner**, a dynamic token pruning framework that integrates Deep Prior Extraction, Semantic Space Adaptation, and Instruction Modulation to perform action-aware visual token selection. Extensive experiments on various benchmarks verify that ActPruner effectively reduces visual token redundancy while preserving action accuracy and downstream task success across various architectures, including $\pi_{0.5}$ (flow matching) and OpenVLA-OFT (parallel decoding).

2 Related Work

Vision-Language-Action Models. Vision-Language-Action (VLA) models Kim et al. (2024); Zitkovich et al. (2023) bridge perception and continuous robotic control. While early VLAs like OpenVLA Kim et al. (2024) predict discrete tokens, recent architectures (e.g., the π-series Black et al. (2024), CogACT Li et al. (2024), OpenVLA-OFT Kim et al. (2025)) generate continuous actions. To mitigate the severe computational bottlenecks of these massive models, visual token pruning has emerged as a promising efficiency optimization, orthogonal to lightweight model designs Shukor et al. (2025); Liu et al. (2024).

Token Pruning for VLA Models. Token pruning is pivotal for real-time VLA inference. Training-free methods typically borrow token reduction strategies from general VLMs Yang et al. (2025); Tan et al. (2025); Xu et al. (2025) or rely on heuristic trajectory rules Pei et al. (2025); Li et al. (2025); Wang et al. (2025), which often lag behind trainable approaches in success rate and latency. Consequently, trainable pruning has gained significant attention. However, representative trainable frameworks such as LightVLA Jiang et al. (2025) predominantly rely on cross-attention between instructions and visual tokens in the shallow input space. While such vision-language similarity captures high-level semantic relevance, it remains fundamentally *action-agnostic*—failing to account for the physical dynamics of continuous control and inevitably discarding fine-grained manipulation affordances (e.g., pre-grasp contact points). In sharp contrast, ActPruner departs from the conventional vision-language similarity paradigm by proposing an explicit *action information extraction* mechanism. Instead of shallow cross-modal matching, we extract high-level kinematic priors directly from deep FFN layers, project them onto the input space, and modulate them with instructions, thereby establishing a principled, action-centric criterion for dynamic token pruning.

3 Action-Anchored Dynamic Token Pruning for VLA Models

3.1 Motivation Study

We ask whether the internal representations of a pretrained VLA contain action-sensitive signals that can help identify visual tokens important to continuous action prediction. To probe this question, we conduct two diagnostic experiments, with details provided in Appendix C. The results are shown in Fig. 2.

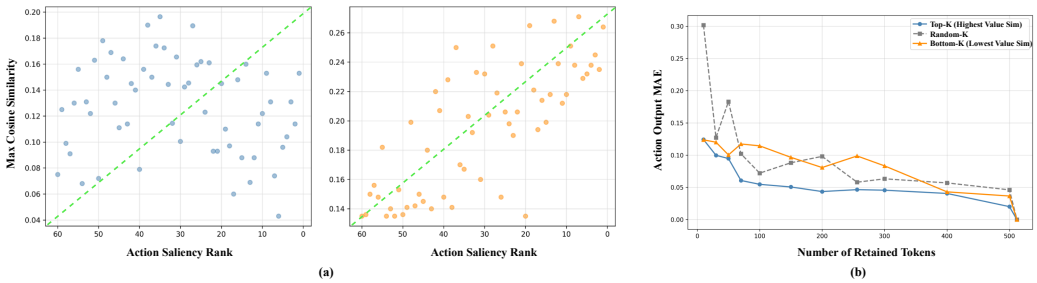

Figure 2: Affinity between FFN value vectors and action-sensitive visual tokens. **(a)** Deep contextualized token representations (right) exhibit a clearer association between leave-one-out action saliency and FFN-value affinity than shallow representations (left). **(b)** Retaining Top-K tokens ranked by FFN-value affinity generally yields lower action MAE than Random-K or Bottom-K retention.

(i) Action-critical visual tokens show a clearer affinity to deep FFN value vectors. We first estimate the importance of each visual token using its leave-one-out action saliency S_i, and compare this saliency with the token's maximum cosine similarity to the value vectors of a deep FFN layer. As shown in Fig. 2(a), tokens with higher action saliency tend to have greater FFN-value affinity when represented by deep contextualized features. In comparison, this association is less apparent for shallow token representations. This observation suggests that action-relevant information may be more accessible in deeper representations, while directly applying the same signal to shallow visual tokens may be less reliable.

(ii) FFN-value affinity provides a candidate signal for token ranking. We next examine whether this affinity is functionally useful for preserving action predictions under token pruning. We rank visual tokens by their maximum similarity to the deep FFN value vectors and construct three retained subsets: Top-K, Bottom-K, and Random-K. For each subset, we measure the action MAE relative to the prediction obtained using all visual tokens. As shown in Fig. 2(b), retaining the Top-K tokens generally produces lower action MAE than retaining random or bottom-ranked tokens. This result provides initial evidence that FFN-value affinity can serve as an action-sensitive ranking signal.

Taken together, these diagnostics suggest that deep FFN value vectors are a promising source of information for action-aware token selection. However, their direct use raises several practical questions: the full set of value vectors is large and redundant, deep representations may not be directly compatible with the shallow visual tokens used for early pruning, and the relevant action information may vary with the current instruction. These considerations motivate the method introduced in the following section.

3.2 Action-Anchored Dynamic Token Pruning

To address the three issues identified in Sec. 3.1, we propose ActPruner. As illustrated in Fig. 3, ActPruner constructs an action-aware pruning metric through an **Action Information Extraction Module (AIEM)**. The AIEM follows a three-step pipeline: Deep Prior Extraction compresses the redundant FFN value-vector space into a compact set of anchors, Semantic Space Adaptation improves their compatibility with shallow visual tokens, and Instruction Modulation conditions them on the current task.

3.2.1 Step 1: Deep Prior Extraction

The motivation study suggests that deep FFN value vectors contain action-sensitive directions that can support visual-token ranking. However, directly using the complete set of value vectors would introduce substantial redundancy. We therefore extract a compact set of *Action Anchors* from the down-projection matrix of a deep FFN layer via Principal Component Analysis (PCA).

Formally, let $\mathbf{W}_{\text{down}} \in \mathbb{R}^{D\times M}$ denote the FFN down-projection matrix, where M is the intermediate hidden dimension and D is the model dimension. We center $\mathbf{W}_{\text{down}}$ by subtracting the mean of its value vectors and compute the Singular Value Decomposition (SVD), yielding

$$\bar{\mathbf{W}}_{\text{down}} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\top}. \tag{1}$$

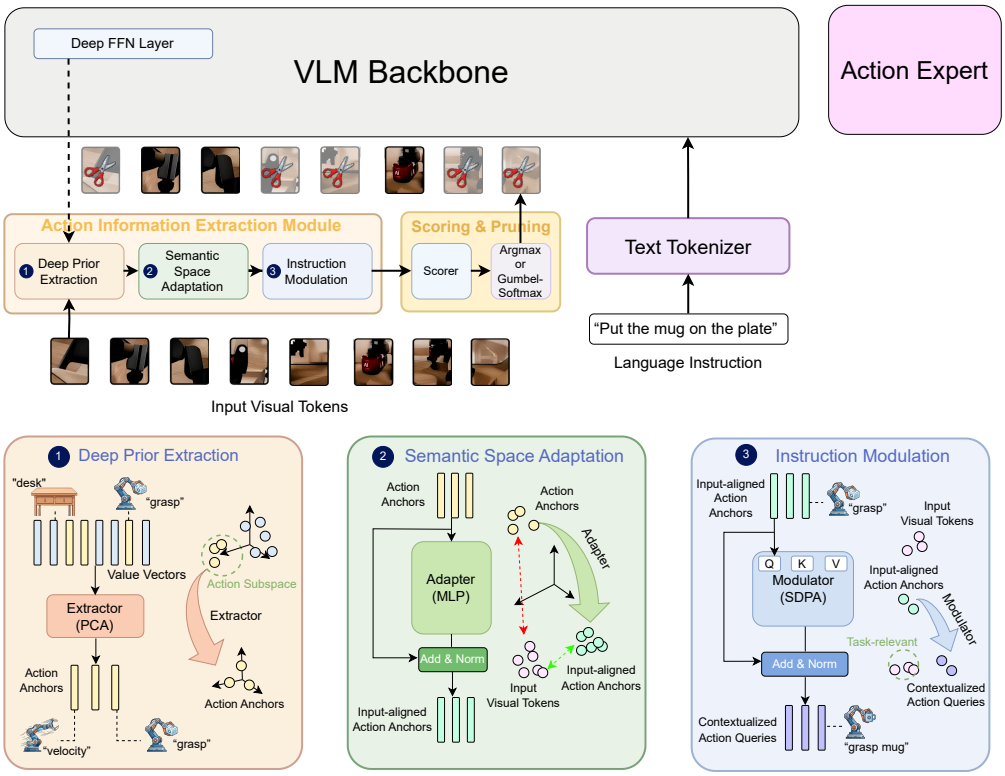

Figure 3: Overall architecture of ActPruner. The Action Information Extraction Module (AIEM) generates action queries through three stages: Deep Prior Extraction extracts action anchors from deep FFN value vectors; Semantic Space Adaptation aligns them to the pruning layer; and Instruction Modulation conditions them on task instructions. Query–patch affinity scores then guide differentiable visual token pruning, preserving action-critical regions while removing redundant background tokens.

Here, $\mathbf{U} \in \mathbb{R}^{D\times D}$ and $\mathbf{V} \in \mathbb{R}^{M\times M}$ are the left and right singular-vector matrices, respectively, and $\mathbf{\Sigma}$ contains the singular values.

We select the top-K principal directions associated with the largest singular values by taking the first K columns of $\mathbf{U}$:

$$\mathbf{P} = \mathbf{U}^{\top}_{[:,:K]} \in \mathbb{R}^{K\times D}. \tag{2}$$

The resulting PCA components form the *Action Anchors*, providing a compact approximation of the dominant directions in the FFN value-vector subspace. The choice of leading PCA directions is evaluated against low-variance and random alternatives under matched token-retention budgets in Appendix E.

3.2.2 Step 2: Semantic Space Adaptation

The extracted anchors $\mathbf{P}$ are derived from a deep FFN layer, whereas token pruning operates on shallow visual representations. Although their dimensions are compatible, directly computing their affinity may be suboptimal because of the cross-layer representation discrepancy. We therefore introduce a lightweight learnable adapter, together with a residual connection and RMSNorm:

$$\mathbf{A} = \mathrm{RMSNorm}\left(\mathbf{P} + \mathrm{Adapter}(\mathbf{P})\right), \tag{3}$$

where $\mathbf{A}$ denotes the adapted anchors used for subsequent visual-token scoring.

3.2.3 Step 3: Instruction Modulation

The adapted anchors $\mathbf{A}$ encode generic action-related information but do not specify which visual evidence is relevant to the current task. For example, different instructions may require attending to different objects, contact regions, or geometric cues. We therefore modulate the adapted anchors

using the instruction tokens $\mathbf{T} \in \mathbb{R}^{L\times D}$ through parameter-free cross-attention. The resulting task-conditioned queries $\mathbf{Q}_{\text{task}} \in \mathbb{R}^{K\times D}$ are computed as

$$\begin{aligned}\mathbf{Q}_{\text{task}} &= \text{RMSNorm}\left(\mathbf{A} + \mathbf{Task}\right),\\ \mathbf{Task} &= \text{Softmax}\left(\frac{\mathbf{A}\mathbf{T}^\top}{\sqrt{D}}\right)\mathbf{T}.\end{aligned} \tag{4}$$

Given normalized visual tokens $\mathbf{V} \in \mathbb{R}^{N\times D}$, we compute the query–token affinity matrix

$$\mathbf{S} = \frac{1}{\sqrt{D}}\mathbf{Q}_{\text{task}}\mathbf{V}^\top, \qquad \mathbf{S} \in \mathbb{R}^{K\times N}. \tag{5}$$

Each row of $\mathbf{S}$ scores the visual tokens with respect to one task-conditioned query. During training, we apply row-wise hard Gumbel-Softmax with a straight-through estimator, allowing each query to select one visual-token slot while preserving gradient propagation. During inference, we apply row-wise `argmax` to $\mathbf{S}$ and merge duplicated selections to obtain the final token mask. The retained visual tokens are then concatenated with the class and task tokens for subsequent VLA inference.

End-to-End Training Objective. The Gumbel-Softmax straight-through estimator allows gradients to propagate through the token-selection scores. We jointly optimize the AIEM and VLA parameters using the standard action-learning objective of the corresponding base model:

$$\mathcal{L} = \begin{cases} \|\hat{\mathbf{a}} - \mathbf{a}_{gt}\|_1, & \text{for OpenVLA-OFT},\\ \|\mathbf{v}_\theta(\mathbf{x}_t, t) - (\mathbf{a}_{gt} - \mathbf{x}_0)\|_2^2, & \text{for } \pi_{0.5}. \end{cases} \tag{6}$$

3.3 Theoretical Analysis: Connecting FFN Value Vectors to Action-Anchored Pruning

We provide a local analysis connecting deep FFN value vectors as action-aware priors for token pruning.

Action saliency. Let $\mathbf{V} \in \mathbb{R}^{N\times D}$ and $\mathbf{T} \in \mathbb{R}^{L\times D}$ denote visual and instruction tokens, respectively, and let $\mathbf{h}_i \in \mathbb{R}^D$ be the contextualized representation of token i at a deep decoding layer. For a VLA policy $\mathbf{a} = f(\mathbf{V}, \mathbf{T})$, we define leave-one-out action saliency as

$$S_i^\star := \|f(\mathbf{V}, \mathbf{T}) - f(\mathbf{V}_{-i}, \mathbf{T})\|_2^2, \tag{7}$$

where $\mathbf{V}_{-i}$ removes or masks token i.

FFN value-vector subspace. A deep FFN with down-projection columns $\{\mathbf{w}_m\}_{m=1}^M$ satisfies

$$\text{FFN}(\mathbf{x}) = \sum_{m=1}^{M} \alpha_m(\mathbf{x})\mathbf{w}_m, \qquad \alpha_m(\mathbf{x}) := \phi(\mathbf{u}_m^\top \mathbf{x}), \tag{8}$$

where $\mathbf{u}_m^\top$ is a row of $\mathbf{W}_{\text{up}}$. Thus, its residual updates lie in

$$\mathcal{S}_{\text{ffn}} := \text{span}\{\mathbf{w}_1, \ldots, \mathbf{w}_M\}. \tag{9}$$

Lemma 1 (FFN Response Subspace)**.** *Where the FFN is differentiable,*

$$\text{Im}\left(\frac{\partial\, \text{FFN}(\mathbf{x})}{\partial \mathbf{x}}\right) \subseteq \mathcal{S}_{\text{ffn}}. \tag{10}$$

The proof is provided in Appendix B.1. This identifies a candidate space for action-relevant residual directions, without establishing which directions affect actions.

Local saliency surrogate. Connecting token removal to this space requires two assumptions.

Assumption 1 (Local Perturbation Model)**.** *For the current input and instruction t, the action deviation $\Delta\mathbf{a}_i := f(\mathbf{V}, \mathbf{T}) - f(\mathbf{V}_{-i}, \mathbf{T})$ satisfies*

$$\Delta\mathbf{a}_i = \mathbf{J}_i^{(t)}\mathbf{h}_i + \mathbf{r}_i, \qquad \|\mathbf{r}_i\|_2 \le \kappa\|\mathbf{h}_i\|_2^2, \tag{11}$$

where $\mathbf{J}_i^{(t)} := \partial\mathbf{a}/\partial\mathbf{h}_i$. This assumes that the effect of input-token removal is captured by the corresponding deep-token perturbation up to the stated residual.

Assumption 2 (Task-Conditioned Action Subspace)**.** *There exists a low-dimensional subspace $\mathcal{U}_t \subseteq \mathcal{S}_{\text{ffn}}$ such that*

$$\mathbf{J}_i^{(t)} = \mathbf{B}_i^{(t)} \mathbf{\Pi}_{\mathcal{U}_t} + \mathbf{R}_i^{(t)}, \qquad \|\mathbf{R}_i^{(t)}\|_2 \le \varepsilon, \tag{12}$$

where $\mathbf{\Pi}_{\mathcal{U}_t}$ is the orthogonal projector onto $\mathcal{U}_t$, and $\mathbf{B}_i^{(t)}$ maps its components to action perturbations.

Proposition 1 (Projection Energy as a Local Saliency Surrogate)**.** *Under Assumptions 1 and 2,*

$$\sqrt{S_i^\star} \le \|\mathbf{B}_i^{(t)}\|_2 \|\mathbf{\Pi}_{\mathcal{U}_t} \mathbf{h}_i\|_2 + \delta_i, \qquad \delta_i := \varepsilon \|\mathbf{h}_i\|_2 + \kappa \|\mathbf{h}_i\|_2^2. \tag{13}$$

If, additionally, $c\|\mathbf{z}\|_2 \le \|\mathbf{B}_i^{(t)}\mathbf{z}\|_2 \le C\|\mathbf{z}\|_2$ for all $\mathbf{z} \in \mathcal{U}_t$, with uniform constants $0 < c \le C$, and $\delta_i \le \eta c \|\mathbf{\Pi}_{\mathcal{U}_t} \mathbf{h}_i\|_2$ for some $0 \le \eta < 1$, then

$$S_i^\star \asymp \|\mathbf{\Pi}_{\mathcal{U}_t} \mathbf{h}_i\|_2^2, \tag{14}$$

where $\asymp$ denotes equivalence up to positive multiplicative constants.

The upper bound follows from the triangle inequality (Appendix B.2); the reverse triangle inequality gives $\sqrt{S_i^\star} \ge c\|\mathbf{\Pi}_{\mathcal{U}_t} \mathbf{h}_i\|_2 - \delta_i$, yielding the conditional equivalence above.

Implications for AIEM. The preceding analysis clarifies the conceptual roles of the three steps in AIEM. **Step 1 (Deep Prior Extraction)** is motivated by Lemma 1, which shows that local FFN responses lie within the value-vector span $\mathcal{S}_{\text{ffn}}$, making it a natural source of candidate action-related directions; ActPruner uses its leading PCA directions as a practical compact approximation of this potentially large space. **Step 2 (Semantic Space Adaptation)** follows from Proposition 1, which relates the projection energy of a deep contextualized token representation onto a task-conditioned actionable subspace to its local action saliency. Since pruning operates on shallow visual representations, a learned cross-layer transformation is required to transfer this signal to the pruning layer. **Step 3 (Instruction Modulation)** is motivated by Assumption 2, which models the actionable subspace as task-conditioned and thus calls for conditioning the adapted anchors on the current instruction before visual-token selection.

Overall, the theoretical analysis motivates the structure of the AIEM pipeline, but not implying that the leading PCA directions or the learned adapter are theoretically optimal. These implementation choices are further examined through controlled action-preservation diagnostics in Appendix E.

4 Experiments

4.1 Experimental Setup and Details

To evaluate our ActPruner, we conduct both Simulation and real-world experiments on a single NVIDIA A800 GPU using the AdamW optimizer.

Simulation. We consider two standard continuous-control benchmarks: LIBERO Liu et al. (2023) and Meta-World Yu et al. (2020). For LIBERO, we evaluate 50 rollouts per task across four suites and report the average task success rate (%). We use OpenVLA-OFT as the base model and integrate ActPruner with LoRA adaptation ($r = 32$, batch size 64). The model is trained for 40,000 steps, with a learning rate of 5×10^{-4} for the first 30,000 steps and 5×10^{-5} for the remaining 10,000 steps. For Meta-World, we use the $\pi_{0.5}$-base model with batch size 32, warmup, and cosine learning-rate decay from 2.5×10^{-5} to 2.5×10^{-6}. We fully fine-tune the dense VLA for 30,000 steps, then jointly optimize all VLA parameters and the trainable ActPruner parameters for another 10,000 steps under the same behavioral cloning objective. We report the dense baseline at 30,000 steps because another 10,000 steps of dense fine-tuning reduced task success. We evaluate 10 rollouts per task across four difficulty levels and report average success rates; $\pm$ denotes standard deviation across five independent evaluation rollout seeds.

Real-World. We further deploy ActPruner on a 6-DoF AgileX PiPer robotic arm with wrist-mounted and overhead RGB-D cameras. We use the $\pi_{0.5}$ base model with an action chunk size of 50 and follow the Meta-World training configuration. Each task is evaluated over 30 independent test rollouts, and additional hardware and control details are provided in Appendix F.

4.2 Quantitative & Qualitative Simulation Results

Table 1: Task success rates on the LIBERO benchmark. ActPruner retains only 4% of visual tokens while achieving an average task success rate comparable to the dense OpenVLA-OFT baseline.

Method	Success Rate (%, ↑)					Token Ret. (%, ↓)
	Spatial	Object	Goal	Long	Average	
OpenVLA-OFT Kim et al. (2025)	98.6	98.4	97.0	94.2	97.1	100
FastV Chen et al. (2024)	94.6	95.8	94.0	88.8	93.3	30
SparseVLM Zhang et al. (2024)	96.8	94.2	**97.6**	93.6	95.6	20
DivPrune Alvar et al. (2025)	92.4	91.2	89.0	84.8	89.4	30
EfficientVLA Yang et al. (2025)	96.5	91.1	96.0	72.1	88.9	22
ADP Pei et al. (2025)	**99.4**	98.0	96.4	91.2	96.3	50
LightVLA Jiang et al. (2025)	98.2	97.6	97.4	92.9	96.5	15
SpecPrune-VLA Wang et al. (2025)	97.4	95.8	97.7	93.4	96.1	30
ActPruner (Ours)	97.2 ± 0.6	**99.0 ± 0.1**	97.0 ± 0.4	**94.6 ± 0.3**	**97.0 ± 0.3**	**4**

Table 2: Task success rates on the Meta-World benchmark.

Method	Success Rate (%, ↑)					Token Ret. (%, ↓)
	Easy	Medium	Hard	Very Hard	Average	
$\pi_{0.5}$ Intelligence et al. (2025)	68.6	36.4	26.7	54.0	55.0	100
FastV Chen et al. (2024)	**63.6**	33.6	18.3	30.0	48.2	50
DivPrune Alvar et al. (2025)	54.6	30.0	26.7	24.0	42.8	30
EfficientVLA Yang et al. (2025)	42.9	7.3	3.3	0.0	26.0	30
LightVLA Jiang et al. (2025)	62.1	29.1	25.0	32.0	47.4	30
ActPruner (Ours)	62.5 ± 2.2	**41.1 ± 2.1**	**32.3 ± 5.0**	**46.0 ± 6.2**	**52.5 ± 1.3**	**14**

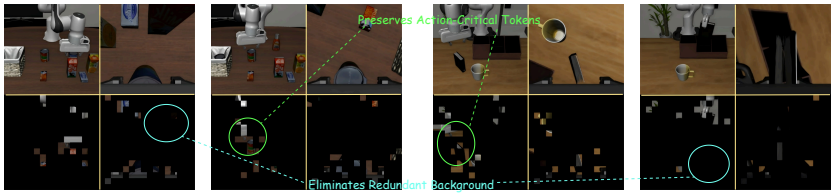

Figure 4: Qualitative visualization on LIBERO tasks.

To verify the effectiveness of ActPruner, we compare it with typical VLA pruning methods, including: FastV Chen et al. (2024), SparseVLM Zhang et al. (2024), DivPrune Alvar et al. (2025), and EfficientVLA Yang et al. (2025), which inherit token reduction strategies from VLMs; ADP Pei et al. (2025) and SpecPrune-VLA Wang et al. (2025), which serve as training-free VLA-specific methods; and LightVLA Jiang et al. (2025), which studies the trainable pruning paradigm.

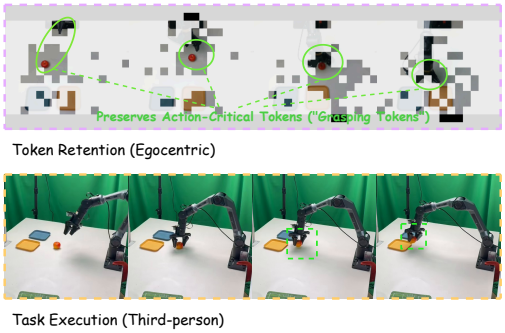

Figure 5: Real-world visualization: Put Apple.

Quantitative Performance. On **LIBERO** (Table 1), ActPruner achieves a 97.0% average success rate, comparable to the dense OpenVLA-OFT baseline (97.1%) and higher than LightVLA Jiang et al. (2025) (96.5%), while retaining only 4% of visual tokens and providing a 1.65× per-step inference speedup. On **Meta-World** (Table 2), ActPruner achieves 52.5% success with 14% token retention, compared with 55.0% for the dense baseline. LightVLA achieves 47.4% success at 30% token retention. We hypothesize that brief task instructions induce homogeneous text-conditioned queries in LightVLA, whereas FFN-derived action anchors help preserve token diversity. Further analysis of operating points and computational efficiency are provided in Appendix H, where ActPruner achieves a 1.65× inference speedup.

Qualitative Performance. The qualitative results in Fig. 4 further show that ActPruner removes redundant background regions (blue circles) while preserving fine-grained, action-critical tokens (green circles) via intrinsic action priors. Unlike existing action-agnostic baselines that often mis-

takenly discard subtle but essential manipulation affordances (e.g., pre-grasp contact points), our action-aware approach dynamically isolates task-relevant visual evidence. This visually demonstrates how our action-aware approach directly resolves the primary defects of action-agnostic methods.

4.3 Results on Real-World Tasks

As shown in Table 3 and Fig. 5, ActPruner transfers effectively from simulation to real-world manipulation. It achieves an impressive 93.3% average success rate while retaining only 12% of visual tokens, confirming that aggressive token reduction does not compromise physical execution. Despite real-world challenges such as lighting changes, sensor noise, and spatial uncertainty, ActPruner reliably preserves fine-grained visual cues required for precise continuous control, leading to robust task completion in physical environments.

Table 3: Success rates on 4 real-world tasks.

Method	Success Rate (%, ↑)					Token Ret. (%, ↓)
	Move Tennis	Remove Cuboid	Pick Red Cube	Put Apple	Average	
$\pi_{0.5}$ Intelligence et al. (2025)	83.3	96.7	90.0	93.3	90.8	100
ADP Pei et al. (2025)	86.7	93.3	90.0	**93.3**	90.8	50
ActPruner (Ours)	**90.7** $\pm$ **1.5**	**96.7** $\pm$ **2.4**	**94.0** $\pm$ **3.7**	92.0 $\pm$ 3.0	**93.3** $\pm$ **1.7**	**12**

4.4 Ablation Studies

We conduct ablation studies of ActPruner's key design choices on the LIBERO-Long suite. Additional analyses of the layer used to extract action priors, anchor dimensionality, cross-suite transferability of action anchors are provided in Appendix G.

Table 4: Ablation of core components and anchor initialization strategies on LIBERO-Long. We compare our PCA-derived principal directions against alternative anchor formulations (including end-to-end learnable queries), and evaluate the necessity of semantic space adaptation and instruction modulation.

Anchor Initialization	**Architectural Components**		**Success Rate (%, ↑)**	**Token Ret. (%, ↓)**
	Semantic Space Adapt.	**Instruct. Mod.**		
Learnable Queries	✓	✓	93.4 $\pm$ 0.3	6
Random Selected Value Vectors	✓	✓	93.6 $\pm$ 0.2	7
Principal Directions	✗	✓	93.0 $\pm$ 0.3	5
Principal Directions	✓	✗	93.5 $\pm$ 0.2	9
Principal Directions (Ours)	✓	✓	**94.6** $\pm$ **0.3**	**6**

Table 4 shows that PCA-derived principal directions outperform randomly selected value vectors (94.6% vs. 93.6% success) while retaining fewer tokens (6% vs. 7%). Removing instruction modulation reduces success to 93.5% despite increasing retention to 9%, suggesting that additional tokens cannot compensate for weaker task-specific selection. Removing semantic space adaptation instead reduces both success and retention to 93.0% and 5%, potentially indicating that cross-layer misalignment causes useful visual cues to be discarded. Together, these opposite shifts in retention suggest complementary roles: instruction modulation filters task-irrelevant evidence, while semantic adaptation helps preserve action-relevant cues.

5 Conclusion

We address the action-agnostic limitation of existing visual token pruning in VLA models, which often discards critical visual cues. By revealing that deep FFN value vectors intrinsically encode manipulation intents, we propose ActPruner, an action-anchored dynamic pruning framework. ActPruner sequentially performs Deep Prior Extraction, Semantic Space Adaptation, and Instruction Modulation to precisely isolate task-critical visual tokens. Extensive evaluations across LIBERO, Meta-World, and real-world deployments demonstrate that ActPruner achieves extreme compression while preserving kinematic accuracy. With negligible computational overhead, ActPruner delivers state-of-the-art inference speedups, providing a highly efficient, plug-and-play solution for real-time VLA deployment in resource-constrained systems.

AI USE STATEMENT

In this work, we used generative AI tools to assist with translation and the implementation of auxiliary code. We have not used generative AI tools to develop the theoretical model or conceptual framework; formulate the assumptions, lemmas, propositions, or other mathematical claims; provide critical ingredients for, or assist with, the proofs; propose or refine the research hypotheses; design or provide feedback on research methodology or experiments; support the qualitative analysis of token-selection and robotic-execution visualizations; or interpret the experimental results. These tasks were performed by the authors.

The generation of synthetic datasets and the cleaning or reformatting of datasets were not applicable to this work, since the study used established robotic benchmarks and physical-robot experiments and did not introduce a new synthetic dataset or involve a dataset-cleaning or reformatting task.

Additionally, we used generative AI tools for proposing candidate titles and keywords and for sourcing and searching for information. We have reviewed all AI-assisted work. Specifically, AI-assisted translations were checked against the original text, candidate titles and keywords were reviewed and revised by the authors, and information identified through AI-assisted searches was independently verified against the original sources. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Section 3 provides the formulation of ActPruner, including its architecture, token-selection mechanism, training objective, and theoretical assumptions. Appendices B–F provide the complete proofs, algorithmic pseudocode, diagnostic evaluation protocols, implementation details, and experimental hyperparameters. We will also submit an anonymous source code as part of the supplementary materials.

REFERENCES

Saeed Ranjbar Alvar, Gursimran Singh, Mohammad Akbari, and Yong Zhang. Divprune: Diversity-based visual token pruning for large multimodal models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 9392–9401, 2025.

Rokas Bendikas, Daniel Dijkman, Markus Peschl, Sanjay Haresh, and Pietro Mazzaglia. Focusing on what matters: Object-agent-centric tokenization for vision language action models. *arXiv preprint arXiv:2509.23655*, 2025.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control, 2024.

Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. In *European Conference on Computer Vision*, pp. 19–35. Springer, 2024.

Jintao Cheng, Haozhe Wang, Weibin Li, Gang Wang, Yipu Zhang, Xiaoyu Tang, Jin Wu, Xieyuanli Chen, Yunhui Liu, and Wei Zhang. Vla-iap: Training-free visual token pruning via interaction alignment for vision-language-action models. *arXiv preprint arXiv:2603.22991*, 2026.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5484–5495, 2021.

Bear Häon, Kaylene Caswell Stocking, Ian Chuang, and Claire Tomlin. Mechanistic interpretability for steering vision-language-action models. In *Conference on Robot Learning*, pp. 2743–2762. PMLR, 2025.

Yuting Huang, Leilei Ding, Zhipeng Tang, Zenghuan Zhu, Jiajun Deng, Xinrui Lin, Shuo Liu, Haojie Ren, Jianmin Ji, and Yanyong Zhang. Environment-aware adaptive pruning with interleaved inference orchestration for vision-language-action models. *arXiv preprint arXiv:2602.00780*, 2026.

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

Titong Jiang, Xuefeng Jiang, Yuan Ma, Xin Wen, Bailin Li, Kun Zhan, Peng Jia, Yahui Liu, Sheng Sun, and Xianpeng Lang. The better you learn, the smarter you prune: Towards efficient vision-language-action models via differentiable token pruning. *arXiv preprint arXiv:2509.12594*, 2025.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025.

Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, et al. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *arXiv preprint arXiv:2411.19650*, 2024.

Ye Li, Yuan Meng, Zewen Sun, Kangye Ji, Chen Tang, Jiajun Fan, Xinzhu Ma, Shutao Xia, Zhi Wang, and Wenwu Zhu. Sp-vla: A joint model scheduling and token pruning approach for vla model acceleration. *arXiv preprint arXiv:2506.12723*, 2025.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023.

Jiaming Liu, Mengzhen Liu, Zhenyu Wang, Pengju An, Xiaoqi Li, Kaichen Zhou, Senqiao Yang, Renrui Zhang, Yandong Guo, and Shanghang Zhang. Robomamba: Efficient vision-language-action model for robotic reasoning and manipulation. *Advances in Neural Information Processing Systems*, 37:40085–40110, 2024.

Ziyan Liu, Yeqiu Chen, Hongyi Cai, Tao Lin, Shuo Yang, Zheng Liu, and Bo Zhao. Vla-pruner: Temporal-aware dual-level visual token pruning for efficient vision-language-action inference. *arXiv preprint arXiv:2511.16449*, 2025.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Xiaohuan Pei, Yuxing Chen, Siyu Xu, Yunke Wang, Yuheng Shi, and Chang Xu. Action-aware dynamic pruning for efficient vision-language-action manipulation. *arXiv preprint arXiv:2509.22093*, 2025.

Weikang Qiu, Tinglin Huang, and Rex Ying. Efficient long-horizon vision-language-action models via static-dynamic disentanglement. *arXiv preprint arXiv:2602.03983*, 2026.

Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, et al. Smolvla: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025.

Xudong Tan, Yaoxin Yang, Peng Ye, Jialin Zheng, Bizhe Bai, Xinyi Wang, Jia Hao, and Tao Chen. Think twice, act once: Token-aware compression and action reuse for efficient inference in vision-language-action models. *arXiv preprint arXiv:2505.21200*, 2025.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Hanzhen Wang, Jiaming Xu, Yushun Xiang, Jiayi Pan, Yongkang Zhou, Yong-Lu Li, and Guohao Dai. Specprune-vla: Accelerating vision-language-action models via action-aware self-speculative pruning. *arXiv preprint arXiv:2509.05614*, 2025.

Yihao Wang, Pengxiang Ding, Lingxiao Li, Can Cui, Zirui Ge, Xinyang Tong, Wenxuan Song, Han Zhao, Wei Zhao, Pengxu Hou, et al. Vla-adapter: An effective paradigm for tiny-scale vision-language-action model. In *Proceedings of the AAAI conference on artificial intelligence*, volume 40, pp. 18638–18646, 2026.

Siyu Xu, Yunke Wang, Chenghao Xia, Dihao Zhu, Tao Huang, and Chang Xu. Vla-cache: Towards efficient vision-language-action model via adaptive token caching in robotic manipulation. *arXiv e-prints*, pp. arXiv–2502, 2025.

Yantai Yang, Yuhao Wang, Zichen Wen, Luo Zhongwei, Chang Zou, Zhipeng Zhang, Chuan Wen, and Linfeng Zhang. Efficientvla: Training-free acceleration and compression for vision-language-action models. *arXiv preprint arXiv:2506.10100*, 2025.

Yifan Ye, Jiaqi Ma, Jun Cen, and Zhihe Lu. Token expand-merge: Training-free token compression for vision-language-action models. *arXiv preprint arXiv:2512.09927*, 2025.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pp. 1094–1100. PMLR, 2020.

Zhaoshu Yu, Bo Wang, Pengpeng Zeng, Haonan Zhang, Ji Zhang, Zheng Wang, Lianli Gao, Jingkuan Song, Nicu Sebe, and Heng Tao Shen. A survey on efficient vision-language-action models. *arXiv preprint arXiv:2510.24795*, 2025.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 11975–11986, 2023.

Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, et al. Sparsevlm: Visual token sparsification for efficient vision-language model inference. *arXiv preprint arXiv:2410.04417*, 2024.

Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, et al. X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. *arXiv preprint arXiv:2510.10274*, 2025.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pp. 2165–2183. PMLR, 2023.

A Limitations

ActPruner focuses on exploiting action-aware structure encoded in the internal FFN geometry of pretrained VLA models. While our current instantiation constructs a compact action subspace from a single deep FFN layer, action-relevant structure may be distributed across multiple depths and may vary with model architecture, embodiment, and manipulation stage. An interesting direction is therefore to explore multi-layer or input-adaptive action subspaces, as well as the transferability of such internal action priors across VLA backbones and robotic embodiments with minimal adaptation. Moreover, ActPruner primarily performs spatial token selection for each observation. Incorporating temporal redundancy and control-stage-aware computation may enable joint spatiotemporal pruning across observations and action chunks, potentially providing further efficiency gains while preserving fine-grained control.

B Proofs

In this section, we provide the detailed proofs of Lemma 1 and Proposition 1 in Sec. 3.3. We follow the notation defined in the main text.

B.1 Proof of Lemma 1

Proof. Recall that the deep FFN block is defined as

$$\mathrm{FFN}(\mathbf{x}) = \mathbf{W}_{\mathrm{down}}\phi(\mathbf{W}_{\mathrm{up}}\mathbf{x}), \tag{15}$$

where $\mathbf{W}_{\mathrm{up}} \in \mathbb{R}^{M\times D}$ and $\mathbf{W}_{\mathrm{down}} \in \mathbb{R}^{D\times M}$. Taking the Jacobian of the FFN output with respect to $\mathbf{x}$ gives

$$\frac{\partial\,\mathrm{FFN}(\mathbf{x})}{\partial\mathbf{x}} = \mathbf{W}_{\mathrm{down}}\,\mathrm{Diag}(\phi'(\mathbf{W}_{\mathrm{up}}\mathbf{x}))\;\mathbf{W}_{\mathrm{up}}. \tag{16}$$

For any perturbation direction $\mathbf{d} \in \mathbb{R}^D$, the corresponding Jacobian-vector product is

$$\frac{\partial\,\mathrm{FFN}(\mathbf{x})}{\partial\mathbf{x}}\mathbf{d} = \mathbf{W}_{\mathrm{down}}\left[\mathrm{Diag}(\phi'(\mathbf{W}_{\mathrm{up}}\mathbf{x}))\,\mathbf{W}_{\mathrm{up}}\mathbf{d}\right]. \tag{17}$$

The bracketed term is an M-dimensional coefficient vector. Denote it as $\mathbf{c} \in \mathbb{R}^M$. Then

$$\frac{\partial\,\mathrm{FFN}(\mathbf{x})}{\partial\mathbf{x}}\mathbf{d} = \mathbf{W}_{\mathrm{down}}\mathbf{c} = \sum_{m=1}^{M} c_m\mathbf{w}_m, \tag{18}$$

where $\mathbf{w}_m$ is the m-th column of $\mathbf{W}_{\mathrm{down}}$. Therefore, for any direction $\mathbf{d}$, the local FFN response lies in

$$\mathrm{span}\{\mathbf{w}_1,\ldots,\mathbf{w}_M\} = \mathcal{S}_{\text{ffn}}. \tag{19}$$

Hence,

$$\mathrm{Im}\left(\frac{\partial\,\mathrm{FFN}(\mathbf{x})}{\partial\mathbf{x}}\right) \subseteq \mathcal{S}_{\text{ffn}}. \tag{20}$$

This completes the proof. □

B.2 Proof of Proposition 1

Proof. Let

$$\Delta\mathbf{a}_i := f(\mathbf{V},\mathbf{T}) - f(\mathbf{V}_{-i},\mathbf{T}) \tag{21}$$

denote the action deviation caused by removing token i.

By Assumption 1, masking the corresponding deep representation $\mathbf{h}_i$ admits a local first-order approximation:

$$\Delta\mathbf{a}_i = \mathbf{J}_i^{(t)}\mathbf{h}_i + \mathbf{r}_i, \qquad \|\mathbf{r}_i\|_2 \le \kappa\|\mathbf{h}_i\|_2^2. \tag{22}$$

By Assumption 2, the action Jacobian can be decomposed as

$$\mathbf{J}_i^{(t)} = \mathbf{B}_i^{(t)}\mathbf{\Pi}_{\mathcal{U}_t} + \mathbf{R}_i^{(t)}, \qquad \|\mathbf{R}_i^{(t)}\|_2 \le \varepsilon. \tag{23}$$

Substituting Eq. equation 23 into Eq. equation 22, we obtain

$$\Delta\mathbf{a}_i = \mathbf{B}_i^{(t)}\mathbf{\Pi}_{\mathcal{U}_t}\mathbf{h}_i + \mathbf{R}_i^{(t)}\mathbf{h}_i + \mathbf{r}_i. \tag{24}$$

Taking the ℓ_2 norm and applying the triangle inequality gives

$$\|\Delta\mathbf{a}_i\|_2 \le \|\mathbf{B}_i^{(t)}\mathbf{\Pi}_{\mathcal{U}_t}\mathbf{h}_i\|_2 + \|\mathbf{R}_i^{(t)}\mathbf{h}_i\|_2 + \|\mathbf{r}_i\|_2. \tag{25}$$

Using submultiplicativity of the operator norm, we have

$$\|\mathbf{B}_i^{(t)}\mathbf{\Pi}_{\mathcal{U}_t}\mathbf{h}_i\|_2 \le \|\mathbf{B}_i^{(t)}\|_2\|\mathbf{\Pi}_{\mathcal{U}_t}\mathbf{h}_i\|_2, \tag{26}$$

and

$$\|\mathbf{R}_i^{(t)}\mathbf{h}_i\|_2 \le \|\mathbf{R}_i^{(t)}\|_2\|\mathbf{h}_i\|_2 \le \varepsilon\|\mathbf{h}_i\|_2. \tag{27}$$

Together with $\|\mathbf{r}_i\|_2 \le \kappa\|\mathbf{h}_i\|_2^2$, this yields

$$\|\Delta\mathbf{a}_i\|_2 \le \|\mathbf{B}_i^{(t)}\|_2\|\mathbf{\Pi}_{\mathcal{U}_t}\mathbf{h}_i\|_2 + \varepsilon\|\mathbf{h}_i\|_2 + \kappa\|\mathbf{h}_i\|_2^2. \tag{28}$$

Since $\Delta\mathbf{a}_i = f(\mathbf{V},\mathbf{T}) - f(\mathbf{V}_{-i},\mathbf{T})$, this proves Eq. equation 13. □

B.3 JUSTIFICATION OF EQ. EQUATION 14

Eq. equation 14 states that, in the local regime,

$$S_i^\star \approx \|\mathbf{\Pi}_{\mathcal{U}_t}\mathbf{h}_i\|_2^2. \tag{29}$$

Here we make this approximation precise.

Let

$$\mathbf{z}_i := \mathbf{\Pi}_{\mathcal{U}_t}\mathbf{h}_i \tag{30}$$

denote the actionable component of the deep token representation. From Assumptions 1–2, we have

$$\Delta\mathbf{a}_i = \mathbf{B}_i^{(t)}\mathbf{z}_i + \boldsymbol{\eta}_i, \tag{31}$$

where the residual term is

$$\boldsymbol{\eta}_i := \mathbf{R}_i^{(t)}\mathbf{h}_i + \mathbf{r}_i. \tag{32}$$

Its norm is bounded by

$$\|\boldsymbol{\eta}_i\|_2 \le \varepsilon\|\mathbf{h}_i\|_2 + \kappa\|\mathbf{h}_i\|_2^2. \tag{33}$$

To justify the projection-energy surrogate, we assume that the local action mapping $\mathbf{B}_i^{(t)}$ is well-conditioned on the actionable subspace $\mathcal{U}_t$. That is, there exist positive constants c_t and C_t such that for all $\mathbf{z} \in \mathcal{U}_t$,

$$c_t\|\mathbf{z}\|_2 \le \|\mathbf{B}_i^{(t)}\mathbf{z}\|_2 \le C_t\|\mathbf{z}\|_2. \tag{34}$$

This condition states that the actionable component is neither collapsed nor arbitrarily amplified when propagated to the action space.

Using the reverse triangle inequality and Eq. equation 34, we obtain

$$\|\Delta\mathbf{a}_i\|_2 = \|\mathbf{B}_i^{(t)}\mathbf{z}_i + \boldsymbol{\eta}_i\|_2 \ge \|\mathbf{B}_i^{(t)}\mathbf{z}_i\|_2 - \|\boldsymbol{\eta}_i\|_2 \ge c_t\|\mathbf{z}_i\|_2 - \|\boldsymbol{\eta}_i\|_2. \tag{35}$$

Similarly,

$$\|\Delta\mathbf{a}_i\|_2 \le \|\mathbf{B}_i^{(t)}\mathbf{z}_i\|_2 + \|\boldsymbol{\eta}_i\|_2 \le C_t\|\mathbf{z}_i\|_2 + \|\boldsymbol{\eta}_i\|_2. \tag{36}$$

Therefore,

$$(c_t\|\mathbf{z}_i\|_2 - \|\boldsymbol{\eta}_i\|_2)_+^2 \le S_i^\star \le \left(C_t\|\mathbf{z}_i\|_2 + \|\boldsymbol{\eta}_i\|_2\right)^2, \tag{37}$$

where $(x)_+ = \max(x, 0)$.

When the residual term $\|\boldsymbol{\eta}_i\|_2$ is small compared with the dominant actionable component $\|\mathbf{z}_i\|_2$, Eq. equation 37 implies that

$$S_i^\star \asymp \|\mathbf{z}_i\|_2^2 = \|\mathbf{\Pi}_{\mathcal{U}_t}\mathbf{h}_i\|_2^2, \tag{38}$$

where $\asymp$ denotes equivalence up to local multiplicative constants. Thus, the projection energy onto $\mathcal{U}_t$ serves as a first-order surrogate for the true leave-one-out action saliency.

C DETAILED SETUPS FOR THE MOTIVATION STUDY

In this section, we provide the detailed experimental setups and execution procedures for the motivation studies presented in Sec. 3.1 of the main paper.

C.1 AFFINITY BETWEEN VALUE VECTORS AND VISION TOKENS

To quantify the importance of individual visual tokens to the final mechanical action, we conduct a leave-one-out perturbation analysis. By systematically masking each visual patch and measuring the resulting deviation in the continuous action trajectory, we assign an Action Saliency score (S_i) to each token. To rigorously evaluate the relationship between these tokens and the deep action priors, we isolate the top-60 most critical vision tokens (i.e., those with the highest S_i) during the approach phase of a manipulation task.

We then extract the complete set of value vectors from the down-projection matrix (W_{down}) at a deep decoding layer of the base VLA model. For each of the top-60 critical vision tokens, we compute its maximum cosine similarity against all value vectors in W_{down}. To understand the evolution of this representation, we compare the similarity distributions using token hidden states extracted from two distinct stages: a shallow input layer and the targeted deep layer.

C.2 Zero-shot Action Preservation via Value Vectors

To answer whether the semantic affinity functionally translates to accurate action prediction, we conduct a zero-shot token pruning experiment using the value vectors as a proxy scoring mechanism.

Specifically, we rank all visual tokens based on their maximum cosine similarity with the deep value vectors. We then construct three sequence masking conditions: retaining the Top-K highest-scored tokens, the Bottom-K lowest-scored tokens, and K randomly selected tokens. For each condition, we mask out the unselected tokens during the forward pass and compute the Mean Absolute Error (MAE) between the resulting predicted action and the unperturbed baseline action (where all tokens are visible). This functional proof-of-concept solidifies our motivation for the dynamic, end-to-end ActPruner framework.

D Algorithmic Details of ActPruner

This section details token selection and gradient propagation, complementing Sec. 3.2.

Token scoring. Pruning is applied before the first language-model decoder layer. Let $X = [C; V; R]$, where C is the leading special token, $V \in \mathbb{R}^{N\times D}$ contains visual tokens, and R contains instruction tokens T and, where applicable, action noise tokens or action placeholders. Only T participates in instruction modulation; all non-visual tokens are retained unchanged. Given fixed PCA anchors $P \in \mathbb{R}^{K\times D}$ ($K = 50$) and token-wise RMSNorm $\mathcal{N}$, we compute

$$\begin{aligned} A &= \mathcal{N}\big(P + \mathrm{Adapter}_\phi(P)\big), \\ Q &= \mathcal{N}\left(A + \mathrm{Softmax}\left(\frac{AT^\top}{\sqrt{D}}\right)T\right), \\ S &= \frac{Q\mathcal{N}(V)^\top}{\sqrt{D}} \in \mathbb{R}^{K\times N}. \end{aligned} \tag{39}$$

Normalization is used for scoring; selected features are taken from the original V.

Differentiable selection and gradient propagation. Following the query-based selection design of LightVLA Jiang et al. (2025), each query selects one visual token during training. With independent standard Gumbel noise G_{ki} and temperature $\tau = 1$, we define

$$\begin{aligned} M^{\mathrm{soft}} &= \mathrm{Softmax}((S+G)/\tau), \\ j_k &= \operatorname*{argmax}_i M^{\mathrm{soft}}_{ki}, \qquad M^{\mathrm{hard}}_{ki} = \mathbb{I}[i = j_k], \\ M^{\mathrm{ST}} &= M^{\mathrm{hard}} - \mathrm{sg}(M^{\mathrm{soft}}) + M^{\mathrm{soft}}, \\ V_{\mathrm{train}} &= M^{\mathrm{ST}} V, \end{aligned} \tag{40}$$

where softmax is row-wise and sg denotes stop-gradient. The forward pass selects exactly K token slots in query order, including duplicates. During backpropagation, M^{ST} uses the derivative of M^{soft}, allowing the VLA loss to update the adapter through $V_{\mathrm{train}} \to M^{\mathrm{ST}} \to S \to Q \to \mathrm{Adapter}_\phi$. Thus, training selects features by matrix multiplication; discrete indices are used only to gather position IDs and attention-mask entries.

The adapter and the VLA parameters enabled by the fine-tuning configuration are jointly optimized using the standard action loss: L1 regression for OpenVLA-OFT and flow matching for $\pi_{0.5}$. The PCA anchors remain fixed, and no auxiliary pruning loss is introduced.

Compact inference. At inference, each query selects its highest-scoring token without Gumbel noise:

$$j_k = \operatorname*{argmax}_i S_{ki}, \qquad b_i = \mathbb{I}[\exists k : j_k = i], \qquad V_{\mathrm{infer}} = V[b]. \tag{41}$$

The Boolean mask removes duplicates and preserves the original visual-token order, yielding an input-dependent count $1 \le n_{\mathrm{keep}} \le \min(K, N)$ for $N \ge 1$. Reported retention rates and speedups use this deduplicated sequence. In both stages, the backbone receives $X' = [C; V'; R]$, with original position IDs and token-mask entries gathered or filtered consistently with the selected visual features. Table 8 evaluates the combined effect of inference-time deduplication and reordering on LIBERO-Long, showing that this sequence transformation improves ActPruner's mean task success rate compared with retaining duplicate tokens in query order.

Algorithm 1 Token Selection and Training in ActPruner

Require: Sequence $X = [C; V; R]$ with instruction tokens T contained in R; fixed anchors P; adapter parameters ϕ; VLA parameters θ

$A \leftarrow \mathcal{N}(P + \mathrm{Adapter}_\phi(P))$
$Q \leftarrow \mathcal{N}\Big(A + \mathrm{Softmax}(AT^\top/\sqrt{D})T\Big)$
$S \leftarrow Q\mathcal{N}(V)^\top/\sqrt{D}$
if training **then**
  Sample $G_{ki} \sim \mathrm{Gumbel}(0, 1)$
  $M^{\mathrm{soft}} \leftarrow \mathrm{Softmax}((S + G)/\tau)$, with $\tau = 1$ and row-wise softmax
  $j_k \leftarrow \mathrm{argmax}_i\, M^{\mathrm{soft}}_{ki}$; $M^{\mathrm{hard}}_{ki} \leftarrow \mathbb{I}[i = j_k]$
  $M^{\mathrm{ST}} \leftarrow M^{\mathrm{hard}} - \mathrm{sg}(M^{\mathrm{soft}}) + M^{\mathrm{soft}}$
  $V' \leftarrow M^{\mathrm{ST}}V$ {Keep K slots, including duplicates}
else
  $j_k \leftarrow \mathrm{argmax}_i\, S_{ki}$; $b_i \leftarrow \mathbb{I}[\exists k : j_k = i]$
  $V' \leftarrow V[b]$ {Unique tokens in original order}
end if
$X' \leftarrow [C; V'; R]$
Gather or filter original position IDs and token-mask entries consistently with V'
if training **then**
  Compute the standard VLA action loss $\mathcal{L}_{\mathrm{VLA}}$ using X'
  Backpropagate through $V' = M^{\mathrm{ST}}V$ to update ϕ and enabled parameters in θ; keep P fixed
end if
return X' and its position IDs and token mask

E Diagnostic Evaluation of Deep Prior Extraction and Semantic Space Adaptation

We examine two hypotheses underpinning ActPruner: (1) the leading PCA directions of FFN value vectors provide useful action-related priors for visual-token selection, and (2) learned semantic space adaptation makes these priors more effective in the input visual-token space. To isolate these contributions, we compare anchor-based token rankings under a common scoring rule and matched token-retention budgets, using the same dense OpenVLA-OFT checkpoint throughout. For each observation, we compare this model's action predictions with all visual tokens retained against its predictions with a selected subset of visual tokens. The OpenVLA-OFT backbone and action head receive no pruning-specific training or parameter updates in this diagnostic. For the semantic space adaptation experiment, we use the previously trained adapter solely to transform the anchors for token scoring; the evaluated policy remains the original dense checkpoint. This diagnostic measures action-output preservation along dense-policy trajectories; closed-loop results for the complete trained ActPruner pipeline are reported separately in the main experiments.

Evaluation protocol. For a policy query q, let $\hat{\mathbf{a}}^q_{\mathrm{dense}} \in \mathbb{R}^{H\times d_a}$ denote the denormalized action chunk predicted by the dense model, where H and d_a are the action horizon and action dimension, respectively. Given a token-selection condition m and a vision-token retention rate r, we retain $k_r = \min\{N, \max\{1, \mathrm{round}(rN)\}\}$ out of the N input vision tokens and obtain the corresponding pruned prediction $\hat{\mathbf{a}}^q_{m,r}$. Here, k_r denotes the retained-token budget, distinct from the number of anchors. We measure the deviation from the dense prediction using

$$\mathrm{MAE}^q(m, r) = \frac{1}{Hd_a}\sum_{h=1}^{H}\sum_{j=1}^{d_a}\left|\hat{a}^q_{m,r,h,j} - \hat{a}^q_{\mathrm{dense},h,j}\right|. \tag{42}$$

We first average query-level MAEs within each episode, then average the episode means with equal weight. For E evaluated episodes, we report approximate pointwise 95% confidence intervals of the form $\overline{\mathrm{MAE}}(m, r)\pm 1.96\, s_{m,r}/\sqrt{E}$, where $s_{m,r}$ is the sample standard deviation of the episode means. Thus, temporally correlated queries within a rollout are not treated as independent observations when computing these intervals. The PCA experiment uses **500 episodes** (50 per task; 16,844 policy

queries), whereas the semantic space adaptation experiment uses **200 episodes** (20 per task; 6,652 policy queries), both on the ten LIBERO-10 tasks.

Common anchor–token affinity. Token relevance is computed using the projected vision tokens at the input of the language model. Let $\mathbf{V}^q = \{\mathbf{v}_i^q\}_{i=1}^N$ denote these input vision tokens and let $\mathcal{V}_m$ denote the anchor vectors used by condition m. For all anchor-based conditions, we use the same relevance score:

$$s_i^q(m) = \max_{\mathbf{u}\in\mathcal{V}_m} \frac{|\langle \mathbf{v}_i^q, \mathbf{u}\rangle|}{\|\mathbf{v}_i^q\|_2 \|\mathbf{u}\|_2}. \tag{43}$$

This score measures the strongest magnitude of normalized alignment between a token and the evaluated anchors. It is closely related to the normalized dot-product affinity used by ActPruner. Specifically, for nonzero D-dimensional vectors and parameter-free RMSNorm, ignoring the numerical stabilization term,

$$\frac{\mathrm{RMSNorm}(\mathbf{x})^\top \mathrm{RMSNorm}(\mathbf{y})}{\sqrt{D}} = \sqrt{D}\,\frac{\langle \mathbf{x}, \mathbf{y}\rangle}{\|\mathbf{x}\|_2\|\mathbf{y}\|_2}. \tag{44}$$

Thus, normalized dot products and cosine similarities induce the same pairwise-affinity ordering up to a positive constant under these normalization conditions.

For this diagnostic, we take the absolute value to measure alignment with an anchor direction regardless of its orientation. This is particularly useful for PCA anchors, whose signs are arbitrary: replacing a supplied anchor $\mathbf{u}$ with $-\mathbf{u}$ leaves Eq. equation 43 unchanged. We apply this convention consistently to all anchor-based conditions. The resulting maximum absolute affinity is a common diagnostic score for comparing anchor representations; it is not identified with the full projection energy onto their span.

Matched-budget selection. For each anchor-based condition, we select the k_r tokens with the highest scores in Eq. equation 43 and restore their original spatial/token order before the pruned forward pass. The Random Tokens baseline uses the same retained-token budget.

The fixed-budget Top-k_r rule serves a different experimental purpose from the dynamic selection rule in ActPruner. In the complete method, instruction-conditioned queries select tokens individually, and their selections determine the retained set. Applying that rule separately to different anchor conditions could produce different retained-token counts, confounding anchor quality with the amount of visual information available to the policy. Here, aggregating anchor affinities into a token ranking and selecting a common number of tokens allows us to compare action preservation at the same budget. The diagnostic therefore retains the normalized anchor–token affinity underlying the method while using an orientation-insensitive score and controlled selection budget to isolate the contributions of prior extraction and semantic space adaptation.

For each query, all conditions use the same observation and reuse the exact projected vision tokens captured from the corresponding dense forward pass. Only the dense actions advance the environment, ensuring that comparisons within each experiment use the same trajectory observations. At $r = 100\%$, all methods have zero reported action MAE, consistent with the expected equivalence when all vision tokens are retained.

PCA action-subspace baselines. We extract the value vectors from the columns of the FFN down-projection matrix $\mathbf{W}_{\mathrm{down}}^{(\ell)} \in \mathbb{R}^{D\times d_{\mathrm{ff}}}$ at layer $\ell = 27$ of the frozen policy, where D is the residual-stream dimension and d_{ff} is the FFN intermediate dimension. Let $\boldsymbol{\mu}$ be the mean of these column vectors. We compute

$$\widetilde{\mathbf{W}}_{\mathrm{down}}^{(\ell)} = \mathbf{W}_{\mathrm{down}}^{(\ell)} - \boldsymbol{\mu}\mathbf{1}^\top = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top. \tag{45}$$

With this column-vector convention, the leading $M = 50$ left singular vectors form the PCA anchors, stored as rows of $\mathbf{P} = [\mathbf{u}_1, \ldots, \mathbf{u}_M]^\top$. We compare five token-selection strategies:

- **Full Values**: all original, uncentered value vectors from $\mathbf{W}_{\mathrm{down}}^{(\ell)}$, serving as the uncompressed value-vector reference;
- **Top PCA**: the M singular directions associated with the largest singular values of the centered matrix;

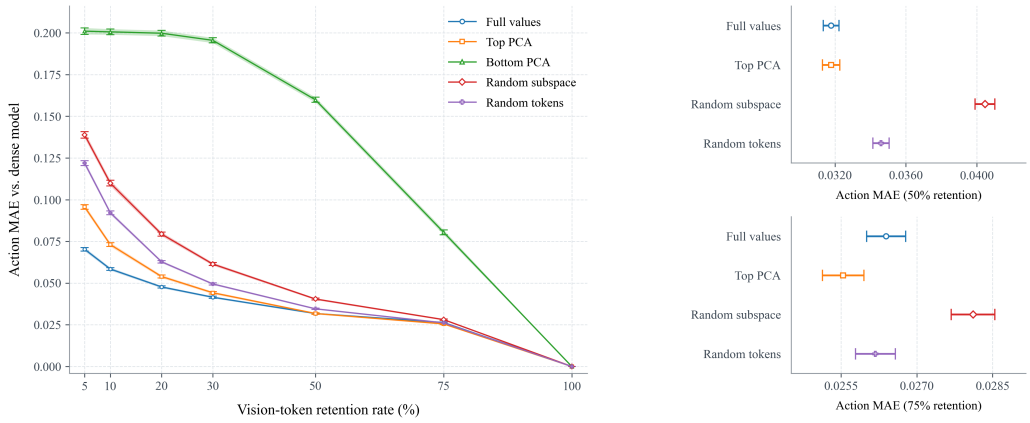

Figure 6: **Action-output preservation under vision-token pruning (PCA Subspace).** PCA action-subspace comparison over **500 episodes** (50 per LIBERO-10 task; 16,844 policy queries), using Full Values, Top PCA, Bottom PCA, a dimension-matched Random Subspace, and Random Tokens. Each point is the mean of episode-level mean action MAEs relative to the dense policy; lower is better. Shaded bands show approximate pointwise 95% confidence intervals.

- **Bottom PCA**: the M directions associated with the smallest singular values of the centered matrix;
- **Random Subspace**: M random orthonormal directions in the same representation space; and
- **Random Tokens**: uniformly sampled vision tokens without anchor-based scoring.

Top PCA, Bottom PCA, and Random Subspace use the same number of anchors. The reported Random Subspace baseline uses one fixed randomly generated basis across all queries. Random Tokens uses one random permutation per query, with prefixes of that permutation defining nested subsets across retention rates. We evaluate $r \in \{5\%, 10\%, 20\%, 30\%, 50\%, 75\%, 100\%\}$.

Semantic Space Adaptation baselines. We conduct the same action-MAE diagnostic using three matched sets of $M = 50$ anchors:

- **Raw PCA Anchors** ($\mathbf{P}$): the PCA directions before semantic space adaptation, bypassing both the adapter and RMSNorm;
- **Adapted Anchors** ($\mathbf{A}$): the output $\mathbf{A} = \mathrm{RMSNorm}(\mathbf{P} + \mathrm{Adapter}(\mathbf{P}))$ of the learned semantic space adaptation module; and
- **Random Adapter**: anchors $\mathbf{A}_{\mathrm{rand}} = \mathrm{RMSNorm}(\mathbf{P} + \mathrm{Adapter}_{\mathrm{rand}}(\mathbf{P}))$ produced by an adapter with the same architecture, randomly initialized and left untrained.

All three variants use the same anchor count, scoring function, vision tokens, retention rates, and dense action references within this experiment. Instruction modulation is bypassed for all three conditions. The learned and random adapters use the same parameter-free RMSNorm operation. The Random Adapter baseline uses one fixed initialization with seed 1007 throughout evaluation. This comparison tests whether the learned adapter outperforms both the unadapted anchors and the evaluated untrained architectural control.

Leading PCA directions support action-preserving token selection. As shown in Fig. 6, Top PCA achieves lower action MAE than Bottom PCA and both random baselines at every tested retention rate below 100%. At **10%** retention, Top PCA obtains an action MAE of **0.0731**, compared with **0.0584** for Full Values, **0.0921** for Random Tokens, **0.1099** for Random Subspace, and **0.2006** for Bottom PCA. Taking an unweighted mean of MAE over retention rates $\{5\%, 10\%, 20\%, 30\%\}$, Top PCA obtains **0.06671**, versus **0.08156** for Random Tokens and **0.09736** for Random Subspace. These correspond to reductions of **18.2%** and **31.5%**, respectively, relative to the corresponding baseline mean MAEs.

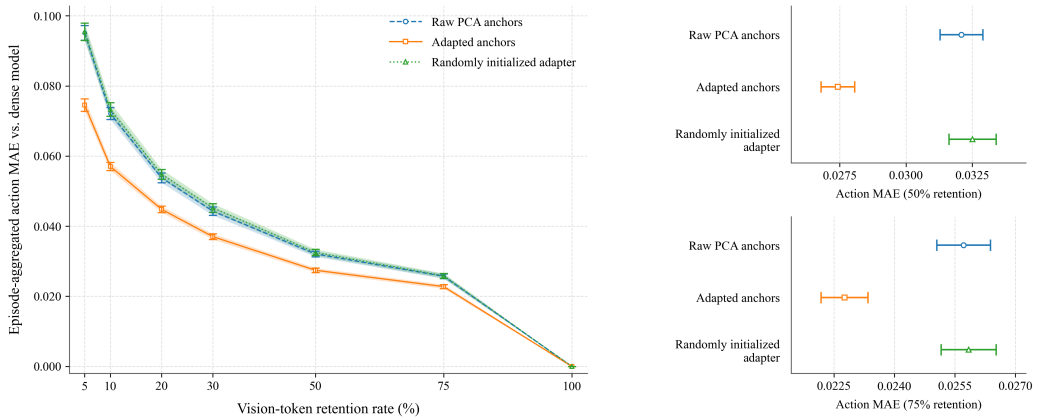

Figure 7: **Action-output preservation under vision-token pruning (Semantic Space Adaptation).** Semantic Space Adaptation comparison over **200 episodes** (20 per LIBERO-10 task; 6,652 policy queries), using Raw PCA Anchors, learned Adapted Anchors, and a Randomly Initialized Adapter. Each point is the mean of episode-level mean action MAEs relative to the dense policy; lower is better. Shaded bands and capped error bars show approximate pointwise 95% confidence intervals.

The separation between Top PCA and Bottom PCA shows that the leading directions are substantially more useful for action-preserving token ranking than the low-variance directions of the same weight matrix. The advantage over the evaluated dimension-matched Random Subspace also suggests that anchor count alone does not explain the observed benefit. Together, these results support the use of leading PCA directions as action-related priors for zero-shot visual-token selection.

Compression is not lossless across all retention rates: Top PCA has higher MAE than Full Values at 5%–30% retention. The gap narrows as retention increases, with nearly identical MAEs at 50% retention (0.031777 for Top PCA versus 0.031774 for Full Values). At 75% retention, Top PCA has a slightly lower mean MAE (0.0255 versus 0.0264). These results show that 50 PCA anchors can approach the uncompressed reference at moderate-to-high retention, while incurring a measurable mean-MAE penalty under more aggressive pruning. The small advantage at 75% does not by itself establish general superiority over Full Values.

Semantic space adaptation improves action preservation with input-space scoring. Fig. 7 compares raw PCA anchors with their semantically adapted counterparts. Learned Adapted Anchors achieve lower mean action MAE than both baselines at every tested retention rate below 100%. At **10%** retention, Adapted Anchors obtain an action MAE of **0.0570**, compared with **0.0721** for Raw PCA Anchors and **0.0733** for the Random Adapter, corresponding to reductions of **20.9%** and **22.2%**, respectively. The largest absolute MAE reductions occur at 5% retention, where Adapted Anchors achieve 0.0746 versus 0.0952 and 0.0954 for the two baselines. The improvement persists at 75% retention, with MAEs of 0.0228, 0.0257, and 0.0258 for Adapted Anchors, Raw PCA Anchors, and the Random Adapter, respectively.

The improvement over Raw PCA Anchors indicates that the extracted directions benefit from learned adaptation before they are used to score input visual tokens. The evaluated random adapter remains close to the raw PCA baseline and does not reproduce the learned adapter's improvement, supporting a contribution from the learned parameters beyond the tested architecture and normalization alone. These observations are consistent with improved compatibility between the adapted anchors and the input visual-token space, as measured by action-output preservation.

F Detailed Experimental Setup and Hyperparameters

In this section, we provide the comprehensive implementation details, hardware specifications, and real-world task descriptions that were omitted from the main text for brevity.

F.1 Simulation Implementation Details

Benchmarks and Metrics. We evaluate ActPruner on two widely-used continuous control robotic benchmarks: LIBERO Liu et al. (2023) and Meta-World Yu et al. (2020). The LIBERO benchmark features a simulated Franka Emika Panda arm, comprising four suites (Spatial, Object, Goal, and Long), each containing 10 tasks and 500 expert demonstrations. To assess policy generalization across various layouts and horizons, we conduct 50 evaluation rollouts per task. Meta-World provides 50 diverse manipulation tasks using a Sawyer arm, spanning four difficulty levels (Easy, Medium, Hard, Very Hard). For this benchmark, we conduct 10 independent rollouts per task to evaluate multi-task learning and rapid generalization. For both environments, we report the average task success rate (%).

Implementation Details. All training and evaluation are conducted on a single NVIDIA A800 GPU. Unless otherwise specified, we employ the AdamW optimizer with a weight decay of 0.01, gradient clipping at 1.0, and a cosine decay learning rate scheduler (peak 2.5×10^{-5}, minimum 2.5×10^{-6}). For the LIBERO benchmark, we adopt the OpenVLA-OFT as our base model (dual-branch visual encoder Zhai et al. (2023); Oquab et al. (2023) + LLaMA-2-7B Touvron et al. (2023)) and directly integrate our pruning method using LoRA ($r = 32$) for 40,000 steps (effective batch size 64). In contrast, the $\pi_{0.5}$-base model used for Meta-World lacks prior exposure to the target environments. Thus, we employ a two-stage pipeline: we first establish a multi-task baseline by fully fine-tuning the raw model for 30,000 steps, and subsequently initialize our framework with this adapted baseline, fine-tuning the complete ActPruner pipeline for an additional 10,000 steps.

F.2 Real-world Experimental Setup and Tasks

Experimental Setup. All real-world evaluations are deployed on a 6-DoF AgileX PiPer robotic arm equipped with a 1-DoF parallel gripper, which is shown in Fig. 8. Visual observations are captured via two Intel RealSense D435 RGB-D cameras: one wrist-mounted for an egocentric view and one mounted overhead for a third-person perspective. The control loop operates at 10 Hz with a maximum episode length of 1,000 steps. We adopt the $\pi_{0.5}$ base model with a fixed action chunk size of 50. The training regimen and optimization hyperparameters strictly follow our Meta-World configuration. Models are trained on a single NVIDIA A800 GPU, while real-time inference is executed on an NVIDIA RTX 4090.



Figure 8: Real-world experimental workspace. The setup features a 6-DoF AgileX PiPer robotic arm equipped with a 1-DoF parallel gripper. Two Intel RealSense D435 RGB-D cameras are deployed to capture continuous visual observations: one mounted on the wrist for an egocentric view, and another mounted overhead for a third-person perspective.

Table 5: Impact of the FFN source layer used for anchor extraction on LIBERO-Long.

Source Layer	Success Rate (%, ↑)	Token Ret. (%, ↓)
10	93.6	**6**
20	94.2	7
25	94.0	7
27 (Ours)	**94.6**	**6**
29	94.0	7

For evaluation, we report the task success rate averaged over 30 independent test rollouts per task.

Detailed Real-world Tasks. We evaluated the policies across four distinct physical manipulation tasks requiring precise geometric perception and localization:

1. **Move Tennis (*move the tennis from yellow plate to blue plate*):** The robot must grasp a tennis ball resting on a yellow plate and accurately transport it to a designated blue plate.
2. **Remove Cuboid (*remove the cuboid from blue plate*):** The robot needs to precisely pinch a wooden cuboid located on a blue plate and remove it from the receptacle.
3. **Pick Red Cube (*pick the red cube into the yellow plate*):** The robot is instructed to pick up a small red cube from the table and carefully place it inside a yellow plate.
4. **Put Apple (*put the apple on the yellow plate*):** The robot must navigate to a plastic apple, grasp it, and place it stably onto a yellow plate.

G Additional Ablation Studies and Analysis

In this section, we provide further experimental analyses that complement the ablation studies presented in Sec. 4.4 of the main text.

Table 6: Effect of the number of extracted Action Anchors (K). Retaining $K = 50$ anchors captures sufficient variance to maintain high success rates without incurring unnecessary computational overhead during the adaptation phase.

K	Success Rate (%, ↑)	Token Ret. (%, ↓)
25	94.2	4
50 (Ours)	94.6	6
100	**95.0**	8

Source Layer of Action Priors We evaluate the effect of the FFN source layer used for anchor extraction on LIBERO-Long. As shown in Table 5, Layer 27 achieves the highest success rate (94.6%) with 6% token retention. Layer 10 retains the same proportion of tokens but achieves 93.6% success, while Layers 20, 25, and 29 retain 7% of tokens and achieve 94.0–94.2% success. These results support selecting Layer 27 among the tested layers.

Dimensionality of Action Anchors. To determine the optimal compactness of the extracted action anchors, we evaluate the impact of the number of retained principal components (K). As shown in Table 6, we vary K to balance the capture of kinematic variance against computational efficiency. While retaining $K = 100$ components achieves the highest success rate (95.0%), it incurs a higher token retention rate (8%) and an increased computational footprint during the adaptation phase. Conversely, aggressively reducing K to 25 sacrifices essential action representations, dropping the success rate to 94.2%. Therefore, $K = 50$ strikes the ideal "sweet spot," capturing sufficient variance to robustly guide precise token selection while maximizing inference efficiency.

Validity and Transferability. To evaluate the usefulness and cross-suite transferability of action anchors, we compare different anchor sources in Table 7. For the Cross-task setting, we extract anchors from an OpenVLA-OFT checkpoint trained on LIBERO-Object and use them in ActPruner initialized from the corresponding target-suite checkpoints for Spatial, Goal, and Long. Each model

Table 7: Validity and transferability of Action Anchors across LIBERO suites. We compare the extracted anchors against random noise (sanity check) and evaluate cross-suite generalization by applying anchors explicitly extracted from the Object suite to other unseen environments.

Anchor Type	Anchor Source	Spatial	Object	Goal	Long	Average
Random Noise	$\mathcal{N}(0,1)$	96.0	98.4	96.2	92.8	95.9
Action Anchors (Cross-task)	Object	96.6	99.0	**97.4**	93.6	96.7
Action Anchors (In-domain)	Respective Target Suites	**97.2**	**99.0**	97.0	**94.6**	**97.0**

Table 8: Effect of inference-time deduplication and reordering on LIBERO-Long. For each method, both inference modes use the same checkpoint and deterministic argmax selection. Success rates are reported as mean $\pm$ standard deviation across evaluation seeds. Δ denotes the success rate with deduplication minus that without deduplication, in percentage points.

Method	No Deduplication Query Order	Deduplication Original Token Order	Difference Δ (pp)
LightVLA	92.5 ± 0.4	92.7 ± 0.4	+ 0.2
ActPruner (Ours)	93.1 ± 0.3	94.6 ± 0.3	+ 1.5

is then trained on its target suite, jointly optimizing the adapter and enabled VLA parameters while keeping the anchors fixed. The In-domain setting follows the same protocol using anchors extracted from each target-suite checkpoint. For the Random Noise baseline, we replace the PCA anchor matrix P with entries sampled independently from $\mathcal{N}(0,1)$ and keep this matrix fixed throughout the same target-suite training procedure.

The extracted anchors outperform Random Noise across all suites, supporting the benefit of FFN-derived priors over random anchors under the evaluated training settings. Cross-task anchors achieve a four-suite average success rate of 96.7%, close to the 97.0% obtained with In-domain anchors. These results support the reuse of Object-derived action anchors across suites with target-suite training.

Effect of Inference-Time Deduplication and Reordering. We evaluate whether the difference between training and inference token-sequence construction affects closed-loop task success on LIBERO-Long. For each method, we fix the trained checkpoint and use deterministic argmax selection in both inference modes. The first retains one token per query, including duplicates, in query order, matching the training sequence layout. The second removes duplicates and restores the original token order, as used in standard inference.

As shown in Table 8, deduplication and reordering increase the mean success rate of LightVLA from 92.5% to 92.7% and that of ActPruner from 93.1% to 94.6%, corresponding to gains of 0.2 and 1.5 percentage points, respectively. Thus, matching the training sequence layout does not improve the observed mean success rate for either method. ActPruner also achieves a higher mean success rate than LightVLA under both inference modes. These results provide empirical evidence that the deployment-time sequence transformation does not impair task success in the evaluated setting and is beneficial for ActPruner.

H Computational Efficiency and Overhead Analysis

In this section, we provide a detailed analysis of ActPruner's computational efficiency, focusing on both the algorithmic token sparsity and the resulting system-level hardware acceleration.

Token Sparsity and Efficiency–Accuracy Trade-off. The token retention rates reported for the baselines in Table 1 follow the configurations recommended in their papers. Importantly, training-free and trainable pruning methods support different forms of retention control. Training-free methods generally allow the retention ratio to be adjusted directly at inference time, whereas trainable dynamic pruning methods jointly optimize the token selector and the policy. Consequently, under a fixed training configuration, methods such as ActPruner and LightVLA converge to native operating points

that cannot be reliably rescaled post hoc without retraining. We therefore evaluate these methods at their respective native operating points rather than forcing them to match a prescribed retention ratio.

At its native operating points, ActPruner retains only 4% of visual tokens on LIBERO and 14% on Meta-World. On LIBERO, LightVLA retains 15% of tokens; ActPruner therefore uses approximately one quarter as many visual tokens as LightVLA. To characterize the efficiency–accuracy trade-off over a broader range of computational budgets, Figure 9 reports retention curves for inference-scalable baselines together with the native operating points of trainable methods. At 4% retention, ActPruner achieves both higher success and lower token retention than all evaluated pruned baselines shown in the figure. The dense OpenVLA-OFT baseline attains a marginally higher success rate (97.1% vs. 97.0%) but retains all visual tokens.

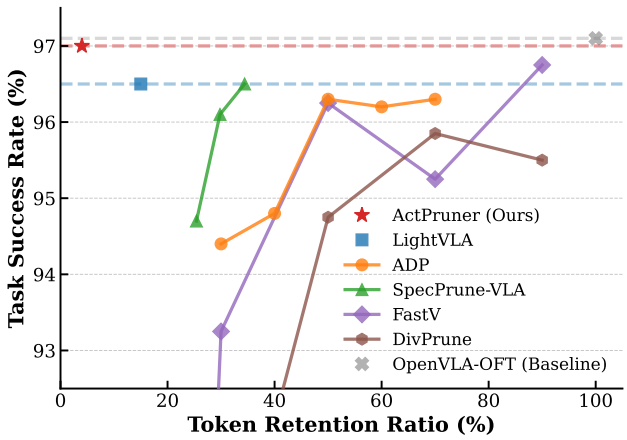

Figure 9: Accuracy–retention operating points on LIBERO. For inference-scalable methods, we report multiple retention settings; for trainable dynamic pruning methods, we report their native operating points under the corresponding trained configurations. ActPruner achieves a favorable Pareto operating point among the evaluated pruning methods.

System-Level Acceleration and Overhead. Building upon this extreme visual compression, Table 9 breaks down the system-level advantages of ActPruner. To ensure a rigorous evaluation, all FLOPs, latency, and inference speedup measurements are conducted on the LIBERO-Long suite using a single NVIDIA 4090 GPU with a batch size of 1, strictly mimicking the real-time continuous control scenario. By employing a remarkably lightweight semantic space adapter and parameter-free cross-attention for instruction modulation, the introduced computational overhead is nearly negligible (+0.0032% FLOPs). Consequently, ActPruner achieves the lowest total FLOPs and the highest end-to-end per-step inference speedup (1.65×) among the evaluated baselines, solidifying its practicality for real-time robotic deployment. Note that this speedup is measured on a per-step basis rather than per-episode, explicitly evaluating the latency reduction of a single forward pass.

Table 9: Computational efficiency analysis on the LIBERO benchmark. Despite the minor overhead from the adaptation modules, ActPruner achieves the lowest total FLOPs and the highest inference speedup.

Method	Overhead FLOPs (%, ↓)	Total FLOPs (%, ↓)	Per-Step Speedup (↑)	Latency (ms, ↓)	Frequency (Hz, ↑)	Memory Usage (GB, ↓)
OpenVLA-OFT	-	100	1.00	106	9.46	14.98
SparseVLM	-	77	1.25	85	11.76	-
SpecPrune-VLA	-	43	1.41	75	13.52	-
ADP	+ 0.23%	81	1.23	86	11.67	-
LightVLA	+ 0.033%	35	1.47	72	13.89	-
Ours	**+ 0.0032%**	**26**	**1.65**	**64**	**15.63**	**14.54**

I ADDITIONAL QUALITATIVE RESULTS ON REAL-WORLD TASKS

To further demonstrate the robustness and effectiveness of our proposed ActPruner in physical environments, we provide comprehensive qualitative visualizations across the four real-world manipulation tasks.

As shown in Figure 10, ActPruner successfully preserves fine-grained, action-critical visual cues essential for continuous control while effectively eliminating redundant background regions. This enables the policy to smoothly and accurately execute the full manipulation trajectories for Move Tennis, Remove Cuboid, Pick Red Cube, and Put Apple.

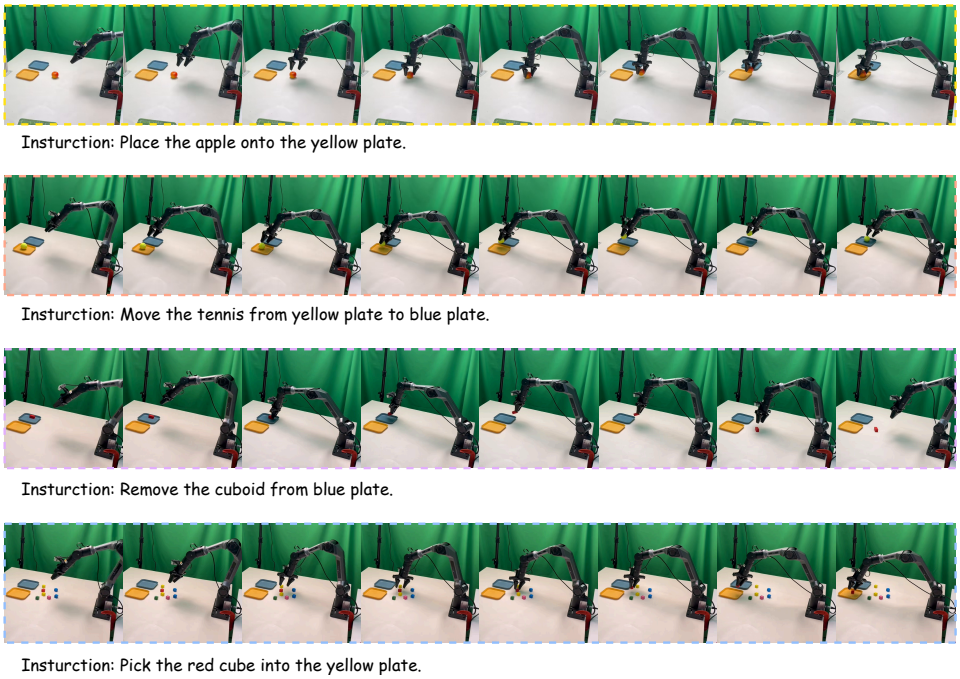

Figure 10: Qualitative execution sequences of ActPruner on four diverse real-world manipulation tasks. From top to bottom: Put Apple, Move Tennis, Remove Cuboid, and Pick Red Cube.