PROTOP: REGION-ANCHORED VISUAL TOKEN PRUNING FOR VISION-LANGUAGE-ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-Language-Action (VLA) models translate the reasoning capabilities of large vision-language backbones into continuous robotic control, but their deployment is bottlenecked by the cost of processing dense visual token sequences through Transformer backbones. Token pruning offers a natural remedy, yet we identify a critical failure mode of existing attention-based methods: spatial collapse. At high compression ratios, global Top-k selection concentrates the retained tokens onto one or two high-attention hotspots – typically the gripper or the currently manipulated object – even though the budget would suffice to cover every region of the scene, leaving the model blind to the rest of it and causing catastrophic task failure. The failure is one of *coverage*, not of token importance. We introduce ProtoP (Prototype-Preserving Token Pruning), a training-free pruning strategy that structurally prevents spatial collapse. ProtoP partitions the visual attention map into spatially coherent regions around local attention peaks ("prototypes"), then allocates the token budget across regions proportionally to prototype importance, with a one-token-per-region floor that guarantees every semantically distinct region survives. At moderate pruning ratios, where sufficient tokens remain to cover the scene, ProtoP matches the strongest baselines; as the ratio increases and spatial collapse becomes the dominant failure mode, the gap widens sharply, reaching 93.3% (OpenVLA-OFT) and 86.0% ($\pi_{0.5}$) task success at 90% pruning versus 82.2% and 72.8% for the best competing method.

1 INTRODUCTION

Vision-Language-Action (VLA) models have emerged as a powerful paradigm for robotic manipulation, translating the broad reasoning capabilities of large vision-language models into continuous motor control (Brohan et al., 2022; Kim et al., 2024; 2025; Black et al., 2024; Team et al., 2024). Despite their promise in generalized tasks, deploying VLA models in real-world scenarios exposes a critical computational bottleneck. Real-time robotic control demands high-frequency inference (e.g., 10-20 Hz); however, processing dense visual streams through a heavy Transformer backbone incurs prohibitive latency, significantly hindering practical application and edge deployment.

Token pruning presents a natural solution to this inefficiency, motivated by the inherent redundancy of visual representations. Recent studies on VLM and VLA models (Chen et al., 2024; Zhang et al., 2024; Xu et al., 2025; Liu et al., 2025; Yang et al., 2025a; Xing et al., 2024) have consistently shown that visual tokens are highly redundant—removing up to 50% causes negligible performance degradation. Many pruning methods leverage this redundancy by selecting top-k tokens based on attention scores or learned importance (Bolya et al., 2022; Liang et al., 2022). While effective at moderate pruning ratios, these global selection strategies suffer from a critical failure mode: **spatial collapse**. At high compression ratios, global top-k selection collapses all retained tokens onto a few high-attention hotspots—typically the gripper or the currently manipulated object—discarding essential environmental context (such as the basket in Fig. 1(B)) and causing catastrophic task failure despite high individual token attention scores.

In contrast, preserving local attention peaks (prototypes) identified from the 3D attention map and retaining tokens from each local region maintains spatial diversity across task-relevant regions, achieving high success rates even under aggressive compression (Fig. 1(C)).

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.

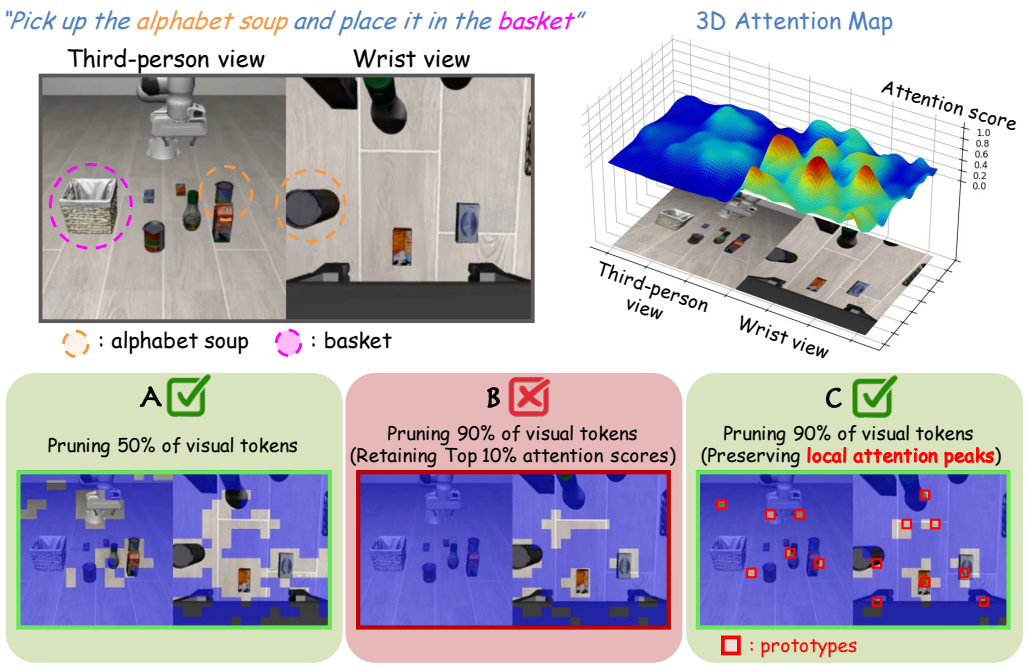


Figure 1: **Overview of prototype-preserving token pruning.** (Top) Multi-view images (third-person and wrist views) and task instruction given to the VLA model, with the corresponding 3D attention map visualizing spatial distribution of visual attention. (Bottom) Comparison of three pruning strategies: (A) retaining top-50% tokens by attention score, (B) retaining top-10% tokens globally, (C) retaining 10% tokens while preserving local attention peaks (red boxes mark prototypes). (C) matches (A)'s performance at 5× compression, while (B) fails by missing critical regions.

To address this limitation, we propose **ProtoP** (Prototype-Preserving Token Pruning), a simple yet structure-preserving training-free pruning strategy that maintains spatial diversity across all pruning ratios—from moderate (60%) to extreme (90%). The key insight is to partition the visual attention map into spatially coherent regions using watershed segmentation, anchoring each region with a *prototype*—a local attention peak (Fig. 1(C)). By allocating the token budget proportionally across these regions while *guaranteeing every region retains at least one token*, ProtoP structurally prevents spatial collapse. High-importance regions (e.g., the primary manipulation target) receive more tokens to preserve fine-grained detail, while secondary regions (e.g., the basket) retain fewer tokens but maintain presence. This ensures that the model retains spatial awareness of the entire scene, not just the current hotspot.

In summary, our main contributions are as follows:

- **Diagnosis.** We identify and characterize spatial collapse in attention-based pruning for VLA models as a *coverage* failure: as the pruning ratio increases, naïve global top-k selection concentrates tokens onto a small number of attention hotspots and discards entire task-relevant regions, producing catastrophic failure even when individual surviving tokens have high attention.
- **Method.** We propose ProtoP, a training-free pruning algorithm that prevents spatial collapse by partitioning the attention map into watershed regions anchored at local peaks and allocating tokens across regions with a one-per-region floor.
- **Empirical evidence.** Across three manipulation benchmarks (LIBERO, VLABench, DROID) and two VLA backbones (OpenVLA-OFT, $\pi_{0.5}$), ProtoP matches existing baselines at moderate pruning ratios and substantially outperforms them at extreme ratios ($\geq$75%), where spatial collapse dominates, achieving 93.3% task success on OpenVLA-OFT and 86.0% on $\boldsymbol{\pi_{0.5}}$ at 90% pruning – substantially outperforming VLA-Pruner (82.2% and 72.8%, respectively) and other methods.

2 Related Work

Vision-Language-Action models. Building on the success of large vision-language models (VLMs) (Liu et al., 2023b; Alayrac et al., 2022; Radford et al., 2021; Li et al., 2023), recent work has explored diverse approaches for robotic action prediction. RT-2 (Zitkovich et al., 2023) demonstrated that VLMs can be fine-tuned to output robot actions tokenized as text; OpenVLA (Kim et al., 2024) extended this with an open-source 7B-parameter model, and OpenVLA-OFT (Kim et al., 2025) introduced continuous action heads with action chunking for improved control fidelity. π_0 (Black et al., 2024) instead uses a flow-matching action expert to generate continuous action chunks. Despite these architectural differences, all such models process visual observations as dense token sequences, making visual token efficiency a critical bottleneck for real-time deployment.

Token pruning in Vision-Language models. Reducing the number of tokens processed by transformers has been widely studied in vision and language domains (Shi et al., 2024). Early work focused on learning-based approaches (Ryoo et al., 2021; Yin et al., 2022): DynamicViT (Rao et al., 2021) learns to predict token importance for progressive pruning, ToMe (Bolya et al., 2022) merges similar tokens via bipartite matching, and AdaViT (Meng et al., 2022) adaptively adjusts tokens per layer. Recent training-free methods for VLMs—FastV (Chen et al., 2024) (pruning based on later-layer attention), SparseVLM (Zhang et al., 2024) (text-visual relevance), and HiPrune (Liu et al., 2026) (hierarchical attention within the vision encoder)—are effective at moderate compression but apply global importance thresholds without spatial awareness, leading to spatial concentration at high pruning ratios.

Token pruning in Vision-Language-Action models. The computational demands of VLA models have motivated recent work on efficient inference. VLA-Cache (Xu et al., 2025) reuses cached visual tokens across timesteps, while pruning-based approaches take diverse strategies: adjusting pruning ratio by end-effector motion (VLA-ADP (Pei et al., 2025)), exploiting inter-/intra-view similarity across multi-view observations (BFA++ (Li et al., 2026)), layer-specific self-speculative pruning (SpecPrune-VLA (Wang et al., 2025)), Gumbel-Softmax token selection (LightVLA (Jiang et al., 2025)), joint prefill/action-decoding selection (VLA-Pruner (Liu et al., 2025)), joint temporal-spatial reduction via model scheduling (SP-VLA (Li et al., 2025)), holistic acceleration of the full VLA pipeline (EfficientVLA (Yang et al., 2025b)), and geometric-prior structural anchors with motion-aligned pruning intensity (VLA-IAP (Cheng et al., 2026)). While these methods effectively reduce computation, they rely on global selection criteria without explicitly preserving spatial diversity across distinct objects, making them susceptible to spatial collapse at high pruning ratios, which our work addresses.

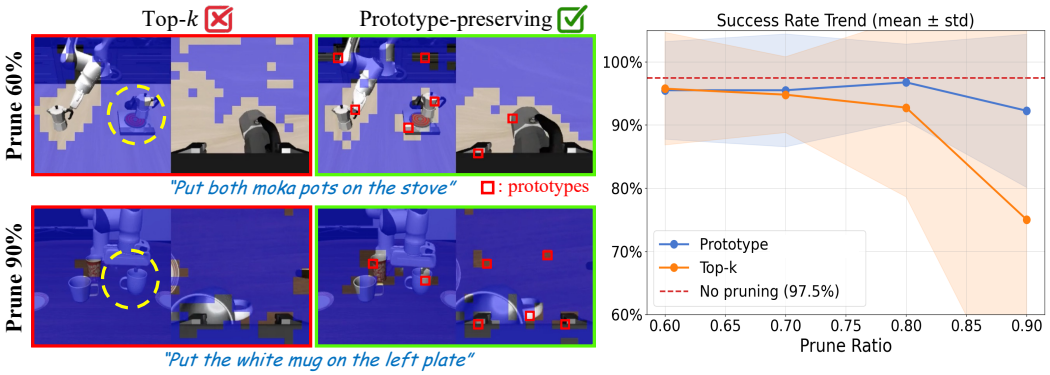
3 Empirical Analysis

In this section, we first show that the failure of global top-k pruning at high ratios is a *coverage* failure – the surviving tokens describe too little of the scene, even though the budget would suffice to cover all of it (Sec. 3.1). We then show that, once every region is represented, the tokens surrounding each attention peak are highly redundant (Sec. 3.2).

3.1 Spatial collapse is a coverage failure

Existing top-k pruning methods, which retain only the globally highest-scoring tokens, exhibit severe performance degradation as the pruning ratio increases. Fig. 2(a) illustrates a representative failure case caused by **spatial collapse**. By greedily selecting tokens with the highest attention scores, top-k pruning collapses all retained tokens onto a few hotspots—typically the gripper or the currently manipulated object.

In the illustrated example, even at 60% pruning, the robot must locate and manipulate a second moka pot (highlighted in yellow) after placing the first one on the stove. However, the top-k method already prunes all visual tokens corresponding to this next target object, leaving the model spatially blind to it despite the high attention scores of the surviving tokens. The same collapse persists and worsens at 90% pruning. This spatial collapse discards essential environmental context, leading to catastrophic task failure.



(a) Spatial concentration in Top-k vs. Prototype-preserving (b) Success rate vs. pruning ratio

Figure 2: **Comparison of top-k and prototype-preserving token pruning strategies.** (a) Spatial collapse at 60% (top) and 90% (bottom) pruning: top-k retains only the highest-scoring tokens, concentrating on the gripper and first moka pot while discarding the next target object (highlighted in yellow), although the retained budget would suffice to represent every region. Prototype-preserving maintains tokens across all local peaks, preserving critical regions. (b) Success rate vs. pruning ratio on LIBERO benchmark: prototype-preserving maintains robust performance across all ratios, while top-k degrades catastrophically beyond 80% pruning.

We describe this failure in terms of region coverage: partition the attention map into spatially coherent regions around its local peaks (the watershed procedure of Sec. 4.2, which involves no pruning), and ask what fraction of regions retain at least one token after pruning. Since regions correspond to distinct peaks – in practice, distinct objects or interaction sites – coverage measures how much of the scene's semantic layout survives, as opposed to how much attention mass survives. Global top-k makes no coverage guarantee. Crucially, coverage loss is not a consequence of budget scarcity: the attention map exhibits only $K \approx 24$–30 peaks per observation (Appendix B.5), so even the 51 tokens retained at 90% pruning would suffice to cover every region. Fig. 2(a) shows what top-k does with that budget instead – it spends it inside one or two regions and leaves the next target object with no token at all.

The simplest way to enforce coverage is to retain at least one token per region – a *prototype-preserving* rule, which is the constraint ProtoP builds on in Sec. 4. As shown in Fig. 2(a), this strategy preserves tokens for both the gripper and the next target object, enabling the VLA model to successfully complete the task by maintaining awareness of all task-relevant entities.

Fig. 2(b) quantifies the consequence on LIBERO. At moderate pruning (60-70%), both methods perform comparably: enough tokens remain that even global selection covers most regions by chance. Beyond 80% pruning, top-k success collapses to 75.0% at 90% – a 22.5-point drop from the unpruned baseline (97.5%) – even though, as noted above, its budget could still have covered every region. Prototype-preserving pruning, which guarantees coverage by construction, degrades gracefully to 92.8%, only 4.7% below the unpruned model. The difference between the two methods is not how many tokens they keep, nor how much attention those tokens carry – it is only whether every region is represented. This identifies spatial collapse as a coverage failure, not an importance failure: the tokens that survive under top-k still carry high attention, but they describe too little of the scene for the policy to act on.

3.2 Redundancy around local peaks

Sec. 3.1 established that coverage – one token per region – is what prevents collapse. It remains unclear how much redundancy exists within each region beyond that one token: if these regions contain substantial redundancy, we can further compress them while preserving the peaks themselves – achieving both spatial diversity and aggressive compression.

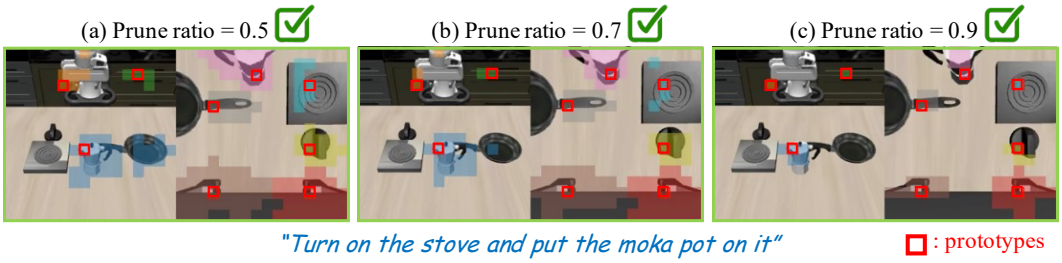


Figure 3: **Visualization of retained tokens under prototype-preserving pruning at three pruning ratios for the task**. Red boxes indicate local attention peaks (prototypes). Each colored region represents the segment associated with its respective prototype, while uncolored regions denote pruned tokens. As the pruning ratio increases from (a) 0.5 to (b) 0.7 to (c) 0.9, tokens with lower attention scores are progressively removed, yet the salient structure of each region is preserved—all task-relevant objects (stove, moka pot, gripper) maintain representative tokens across all ratios.

To investigate this, we apply the prototype-preserving pruning strategy and progressively increase the pruning ratio from 0.5 to 0.9, eliminating additional tokens *within* each region in ascending order of their attention scores while keeping the local peaks intact.

Fig. 3 visualizes this process at three representative ratios. Even at 90% pruning (Fig. 3c), the spatial layout of salient regions remains largely intact—each task-relevant object (the stove, moka pot, and gripper) maintains representative tokens around its prototype. This suggests that the removed tokens contribute only marginally to the overall representation, as long as the peaks are preserved.

To quantify the effect on action quality, we compare the trajectories generated by the pruned model against those of the unpruned baseline using Dynamic Time Warping (DTW) (Sakoe & Chiba, 1978).

As shown in Tab. 1, the success rate remains consistently high across all pruning ratios ($> 94\%$), and the mean DTW stays below 0.7 cm at 80% pruning and 0.9 cm at 90% pruning. This indicates that the predicted trajectories are virtually identical to those of the full model, despite discarding the vast majority of visual tokens.

Pruning ratio	Success rate	DTW (m, ↓)
No pruning	97.5%	-
0.5	95.8%	0.00533 ±0.00410
0.6	94.8%	0.00585 ±0.00458
0.7	94.2%	0.00649 ±0.00567
0.8	96.7%	0.00651 ±0.00446
0.9	95.0%	0.00832 ±0.00526

Table 1: Dynamic Time Warping (DTW) distance between trajectories produced by the unpruned baseline and the prototype-preserving pruned model across pruning ratios. At 90% pruning, the average DTW distance remains below 0.9 cm, indicating that the model retains trajectory fidelity despite aggressive token removal.

These results reveal a key insight: **regions around local peaks contain substantial redundancy, but the peaks themselves are essential**. By preserving the peaks as structural anchors, we can aggressively compress the surrounding tokens without degrading action quality. This finding, combined with the observation from Sec. 3.1 that peaks prevent spatial collapse, motivates ProtoP: we partition the visual field into regions anchored by local attention peaks, then allocate the pruning budget across regions to balance aggressive compression with guaranteed spatial diversity.

4 Method

4.1 Problem Setting

Consider a VLA model that encodes a visual observation into a grid of $P = H_g \times W_g$ patch tokens per camera view (*e.g.*, $P = 256$ for a 16×16 grid). With C camera views (*e.g.*, $C = 2$ for agent and wrist cameras), the total number of visual tokens is $N_v = C \cdot P$. These visual tokens, together with language tokens encoding the task instruction and special action tokens, form the input sequence to the LLM backbone.

Algorithm 1 ProtoP: Prototype-Preserving Token Pruning

Require: Attention scores $\{s_i\}_{i=1}^{N_v}$, keep budget N_{keep}, grid dimensions $\{H_g, W_g\}$, min peak distance $d_{\min}$
Ensure: Pruned token set $\mathcal{K}_{\text{ProtoP}}$

for each camera c **do**
  Reshape scores into grid $\mathbf{M}_c \in \mathbb{R}^{H_g \times W_g}$
  $\tau_c \leftarrow \text{median}(\mathbf{M}_c)$ ▷ Foreground/background threshold
  $\mathcal{F}_c \leftarrow \{(r,q) : \mathbf{M}_c[r,q] \geq \tau_c\}$ ▷ Keep top 50%
  $\mathcal{P}_c \leftarrow \text{LocalPeaks}(\mathbf{M}_c, \mathcal{F}_c, d_{min})$ ▷ Identify prototypes
  $\{\mathcal{R}_c^1, \ldots, \mathcal{R}_c^{K_c}\} \leftarrow \text{Watershed}(-\mathbf{M}_c, \mathcal{P}_c, \mathcal{F}_c)$
end for
$K \leftarrow \sum_c K_c$ ▷ Total regions across all cameras
for each region $\mathcal{R}^k$ with prototype p_k **do**
  $n_k \leftarrow \max\left(1, \left\lfloor N_{\text{keep}} \cdot \frac{s_{p_k}}{\sum_{j=1}^{K} s_{p_j}} \right\rfloor\right)$ ▷ Allocate budget
  $\mathcal{K}^k \leftarrow \text{Top-}n_k\left(\{s_i : i \in \mathcal{R}^k\}\right)$ ▷ Select by attention
end for
return $\mathcal{K}_{\text{ProtoP}} = \bigcup_{k=1}^{K} \mathcal{K}^k$

Given a target pruning ratio $\rho \in [0, 1)$, the goal is to select a subset of $N_{\text{keep}} = \lfloor N_v \cdot (1 - \rho) \rfloor$ visual tokens that maximally preserves the information needed for accurate action prediction, while discarding the remaining $N_v - N_{\text{keep}}$ tokens before the costly LLM forward pass.

Attention score computation. Following prior work (Chen et al., 2024; Chefer et al., 2021), we compute token importance via a partial forward pass through the first L_s layers of the LLM. At a scoring layer l_s (we use $l_s = 3$), we extract the attention weights $\mathbf{A} \in \mathbb{R}^{H \times S \times S}$ where H is the number of attention heads and S is the sequence length. The importance score of each visual token v_i is the mean attention it receives from a designated set of query token positions $\mathcal{Q}$:

$$s_i = \frac{1}{H \cdot |\mathcal{Q}|} \sum_{h=1}^{H} \sum_{j \in \mathcal{Q}} A_{h,j,i}, \quad i \in \{1, \ldots, N_v\} \tag{1}$$

This score reflects how much the designated query tokens attend to each visual patch and serves as the foundation for both standard attention-based pruning and our method. The choice of $\mathcal{Q}$ is studied in Sec. 5.3, where combining text and action tokens as query positions yields the best performance.

4.2 Prototype-preserving token pruning

ProtoP distributes the token budget across semantically coherent regions rather than selecting tokens globally. The key insight is to partition the attention map into regions anchored by local peaks (prototypes), then allocate tokens proportionally to each region's importance while guaranteeing every region retains at least one representative. Algorithm 1 summarizes the procedure.

Spatial grouping via watershed segmentation. Prior works (Liu et al., 2025; Pei et al., 2025; Li et al., 2026) show that removing roughly 50% of visual tokens causes negligible performance degradation in VLA models; we thus first separate foreground from background by thresholding at the median attention score, discarding the bottom 50%.

Within the foreground, we apply watershed segmentation (Meyer, 1994; Beucher & Lantuejoul, 1979) to partition tokens into spatially coherent regions: we reshape each camera's attention scores into a spatial grid $\mathbf{M}_c \in \mathbb{R}^{H_g \times W_g}$, treat it as a topographic surface, and flood from each local attention peak (corresponding to task-critical objects such as the target, gripper, or supporting surfaces) on the negated map $-\mathbf{M}_c$. This assigns every foreground token to its nearest prototype, yielding K_c non-overlapping regions $\{\mathcal{R}_c^1, \ldots, \mathcal{R}_c^{K_c}\}$ per camera, ensuring distinct objects occupy separate regions and laying the foundation for spatial diversity guarantees.

Budget allocation and token selection. Given $K = \sum_c K_c$ total regions and keep budget N_{keep}, each region with prototype score s_{p_k} receives a token budget n_k proportional to its share of total prototype attention (Eq. 2), with a $\max(1, \cdot)$ floor that guarantees every region—however low its

prototype attention—retains at least one token. This is exactly the coverage guarantee identified as decisive in 3.1: every region is represented regardless of the global ratio (recall Fig. 2, where the second moka pot vanished under Top-k).

$$n_k = \max\left(1,\ \left\lfloor N_{\text{keep}} \cdot \frac{s_{p_k}}{\sum_{j=1}^{K} s_{p_j}} \right\rfloor\right). \tag{2}$$

Within each region $\mathcal{R}^k$, we select the Top-n_k tokens by attention, $\mathcal{K}^k = \text{Top-}n_k\left(\{s_i : i \in \mathcal{R}^k\}\right)$, and the final pruned set is their union: $\mathcal{K}_{\text{ProtoP}} = \bigcup_{k=1}^{K} \mathcal{K}^k$. Since prototypes are local maxima by construction, they are always included, guaranteeing that each region's most informative token survives. The overall effect: high-importance regions receive more tokens for fine-grained spatial detail, while secondary regions contribute fewer tokens but maintain presence—balancing efficiency with coverage, unlike global methods that sacrifice entire regions for marginal attention gains elsewhere.

5 Experiments

5.1 Experimental setup

We evaluate ProtoP on OpenVLA-OFT (Kim et al., 2025) (7B) and $\pi_{0.5}$ (Intelligence et al., 2025), comparing against training-free baselines: FastV (Chen et al., 2024),SparseVLM (Zhang et al., 2024), VLA-Cache (Xu et al., 2025), and VLA-Pruner. OpenVLA-OFT experiments are conducted on an NVIDIA RTX 3090 GPU, while $\pi_{0.5}$ experiments are performed on an NVIDIA RTX PRO 6000 Blackwell Max-Q GPU. For $\pi_{0.5}$, we additionally apply model compilation to the denoising step so that prefix encoding constitutes a non-trivial share of total latency, allowing pruning-induced speedups to be meaningfully measured alongside TFLOPs.

Benchmarks. We test on three simulation benchmarks at pruning ratios of 60%, 75%, and 90%. LIBERO (Liu et al., 2023a) contains four suites (Spatial, Object, Goal, Long) with 10 tasks each, 50 episodes per task. VLABench (Zhang et al., 2025) evaluates performance averaged across 2 tasks (100 trials each). DROID (Khazatsky et al., 2024) tests diverse manipulation scenarios, with performance averaged across 2 tasks (100 trials each). We report task success rate following the standard OpenVLA evaluation protocol (Kim et al., 2024; 2025).

5.2 Main results

LIBERO benchmark. Tab. 2 presents results across four LIBERO suites using OpenVLA-OFT and $\boldsymbol{\pi_{0.5}}$ backbones at varying pruning ratios. FastV, SparseVLM, and VLA-Cache exhibit severe performance degradation as the pruning ratio increases, with success rates dropping sharply on both models. VLA-Pruner demonstrates more robust behavior, but still suffers significant drops under aggressive pruning – particularly at 90% pruning.

At 60% pruning, ProtoP is on par with the strongest baselines (within $\sim$1 point on both backbones), as sufficient tokens remain to cover all regions even under global selection. As the ratio increases, the gap widens sharply: at 90% pruning on OpenVLA-OFT, ProtoP achieves 93.3% average success, outperforming VLA-Pruner by 11.1% and other baselines by substantial margins (*e.g.*, 60.0% over VLA-Cache); on $\boldsymbol{\pi_{0.5}}$, ProtoP achieves 86.0%, outperforming VLA-Pruner by 13.2%. This crossover is exactly what the spatial-collapse hypothesis predicts – coverage guarantees only matter once the budget is too small to cover all regions by chance. Notably, ProtoP is the only method that explicitly preserves local attention peaks through prototype-based region anchoring. This demonstrates that retaining local peak structure is critical for maintaining task performance under high compression ratios—a property overlooked by global pruning strategies.

VLABench and DROID benchmarks. Tab. 3 shows results on VLABench and DROID benchmarks. While all methods experience performance degradation relative to the unpruned baseline, ProtoP consistently demonstrates the most robust behavior under aggressive pruning. At 90% pruning on VLABench, ProtoP achieves 28.0%, substantially outperforming VLA-Pruner (19.0%), SparseVLM (19.0%), and FastV (2.0%). On DROID, ProtoP achieves 41.5% at 90% pruning, compared to 38.5% for SparseVLM, 21.5% for VLA-Pruner, and 0.0% for FastV. These results confirm

Table 2: Performance comparison of token pruning methods on the LIBERO benchmark with OpenVLA-OFT and $\pi_{0.5}$ backbones. We evaluate at three pruning ratios (0.6, 0.75, 0.9). Unlike baseline methods that suffer catastrophic performance drops at high pruning ratios, ProtoP (Ours) demonstrates robust task success rates even under aggressive compression. S, O, G, and L denote the four LIBERO task suites: Spatial, Object, Goal, and Long, respectively.

Method	OpenVLA-OFT								$\pi_{0.5}$							
	S	O	G	L	Avg.↑	TFLOPs↓	Lat(ms)↓	Speed.↑	S	O	G	L	Avg.↑	TFLOPs↓	Lat(ms)↓	Speed.↑
Vanilla	99.0	98.2	97.4	95.4	97.5	8.650	174.9	1.00×	99.2	97.6	97.2	92.6	96.65	4.715	159.5	1.00×
Prune **60%** of Visual Tokens																
FastV	**99.2**	98.2	96.8	93.4	96.9	4.772	146.0	1.20×	92.6	97.8	90.8	86.8	92.0	1.970	97.6	1.63×
SparseVLM	99.0	96.6	**97.8**	93.0	96.6	4.897	148.0	1.18×	**98.4**	98.2	**95.8**	91.8	96.1	2.141	120.5	1.32×
VLA-Cache	93.6	96.8	96.6	90.2	94.3	4.936	157.4	1.11×	85.0	95.0	85.8	69.2	83.8	2.793	148.9	1.07×
VLA-Pruner	99.2	**98.6**	97.4	92.4	96.9	4.897	156.7	1.12×	98.0	**98.2**	96.0	**92.6**	**96.2**	2.141	132.3	1.21×
ProtoP (Ours)	99.0	98.0	97.4	**94.0**	**97.1**	4.373	137.4	1.27×	97.6	97.0	95.6	90.6	95.2	2.106	111.6	1.43×
Prune **75%** of Visual Tokens																
FastV	**99.2**	97.0	95.0	87.4	94.7	3.837	116.2	1.51×	82.4	94.6	81.2	63.2	80.35	1.693	94.7	1.68×
SparseVLM	98.8	90.8	**97.2**	92.0	94.7	3.993	118.9	1.47×	92.2	96.2	91.4	86.0	91.5	1.882	114.6	1.39×
VLA-Cache	90.8	85.8	95.8	79.0	87.9	3.959	123.9	1.41×	60.2	65.8	68.4	33.4	57.0	2.477	138.5	1.15×
VLA-Pruner	98.8	**97.8**	96.0	92.8	96.4	3.993	124.0	1.41×	93.8	**98.2**	93.4	87.4	93.2	1.882	118.8	1.34×
ProtoP (Ours)	98.8	97.4	96.2	**95.2**	**96.9**	3.906	124.8	1.40×	**96.4**	94.8	**94.6**	**89.2**	**93.8**	1.988	106.8	1.49×
Prune **90%** of Visual Tokens																
FastV	97.0	53.4	90.8	22.6	66.0	2.914	99.0	1.76×	23.2	55.4	43.8	18.0	35.1	1.422	84.2	1.90×
SparseVLM	98.4	52.2	94.0	40.6	71.3	3.101	100.8	1.74×	64.2	78.6	64.8	36.4	61.0	1.627	108.9	1.46×
VLA-Cache	47.4	8.4	56.4	20.8	33.3	2.981	116.6	1.50×	14.2	4.6	23.4	1.8	11.0	2.161	133.0	1.20×
VLA-Pruner	97.4	74.8	**95.6**	61.0	82.2	3.101	102.4	1.71×	74.8	90.2	72.8	53.4	72.8	1.627	110.2	1.45×
ProtoP (Ours)	**98.8**	**88.4**	94.4	**91.4**	**93.3**	3.251	107.5	1.63×	**91.8**	**91.2**	**88.2**	**72.8**	**86.0**	1.821	99.9	1.60×

Table 3: Performance comparison on VLABench and DROID benchmarks at two pruning ratios. ProtoP maintains robust performance across both benchmarks while baselines degrade significantly at high pruning ratios.

Method	VLABench				DROID			
	Prune **75%**		Prune **90%**		Prune **80%**		Prune **90%**	
	Succ.(%)↑	TFLOPs↓	Succ.(%)↑	TFLOPs↓	Succ.(%)↑	TFLOPs↓	Succ.(%)↑	TFLOPs↓
Vanilla	47.0	4.715	47.0	4.715	76.5	4.729	76.5	4.729
FastV	33.0	1.880	2.0	1.469	15.5	1.705	0.0	1.521
SparseVLM	27.0	2.056	19.0	1.671	**64.0**	1.894	38.5	1.718
VLA-Pruner	39.0	2.056	19.0	1.671	52.0	1.893	21.5	1.716
ProtoP (Ours)	**39.0**	2.056	**28.0**	1.671	62.5	2.050	**41.5**	1.905

that prototype-preserving pruning generalizes across diverse manipulation environments beyond LIBERO, particularly under high compression ratios where spatial collapse becomes the dominant failure mode. We further validate these findings on a physical robot and under controlled visual perturbations in Appendix C.

5.3 Ablation Studies

Score layer index. Tab. 4(a) compares attention scores extracted from different LLM layers. Deeper layers generally capture more refined visual-language-action alignment but at higher computation and latency. Layer 3 strikes the best balance (92.8% average success at moderate cost); Layers 1–2 underperform due to insufficient semantic processing, while Layer 4 offers only marginal gains (93.0%) at higher cost.

Attention source. In OpenVLA-OFT, the input sequence consists of BOS, visual, text, and action tokens, so we can compute visual token importance from attention over different token subsets; Tab. 4(c) compares all combinations. Visual-to-visual attention alone underperforms (90.5% average), suggesting it does not reliably indicate task relevance, whereas text- and action-token attention provide much stronger signals (92.0% and 94.2%, respectively). Combining text and action yields the best performance (94.3%), capturing both task semantics (what to do) and motion intent (how to do it); adding visual attention back degrades performance to 89.2%, indicating it introduces noise rather than useful signal. We therefore adopt text + action attention as our default scoring mechanism.

Table 4: Ablation study on prototype-preserving visual token pruning. We report task success rate (%) on four LIBERO benchmarks. **Bold** indicates the best result per group; underline indicates our default setting. Detailed experimental settings are provided in the Appendix A.4.

Setting	TFLOPs	Spatial	Object	Goal	Long	Avg.
(a) Score layer index						
Layer 1	2.811	**98.8**	73.2	94.4	69.6	84.0
Layer 2	3.053	97.0	74.2	95.6	60.2	81.7
Layer 3	3.251	**98.8**	**88.4**	92.4	**91.4**	92.8
Layer 4	3.443	97.6	87.8	**96.2**	90.6	**93.0**
(b) Minimum peak distance						
Distance = 1	3.251	**100.0**	**97.3**	**98.3**	95.0	**97.7**
Distance = 2	–	99.0	91.3	98.0	**95.3**	95.9
Distance = 3	–	97.3	89.7	95.3	91.7	93.5
Distance = 4	–	95.3	66.3	96.3	89.3	86.8
(c) Attention source						
Visual	3.251	99.3	80.0	96.7	86.0	90.5
Text	–	94.0	87.3	**98.7**	88.0	92.0
Action	–	99.3	88.7	96.0	**92.7**	94.2
Visual + Text	–	99.3	75.3	96.7	89.3	90.2
Visual + Action	–	**100.0**	76.0	96.7	85.3	89.5
Text + Action	–	98.7	**89.3**	98.0	91.0	**94.3**
All	–	**100.0**	76.0	94.0	86.7	89.2

Minimum peak distance. Fig. 4 visualizes this trade-off: as $d_{\min}$ increases, prototypes become sparser and regions grow, and Tab. 4(b) shows $d_{\min} = 1$ achieves the best average performance. With $d_{\min} = 1$, nearby attention peaks—even those separated by a single grid cell—are treated as distinct prototypes, so closely positioned objects (e.g., multiple targets on a table) or small interaction sites (handles, buttons, switches) occupy separate regions, letting ProtoP allocate tokens to each entity independently even when objects are spatially close. Larger values ($d_{\min} \geq 2$) instead merge nearby objects into a single region; this reduces region count and simplifies computation but pools the token budget across objects, so lower-attention objects may receive insufficient tokens or be pruned entirely—reintroducing the spatial concentration problem ProtoP aims to solve. We therefore adopt $d_{\min} = 1$ as our default, prioritizing fine-grained spatial coverage over computational simplicity.

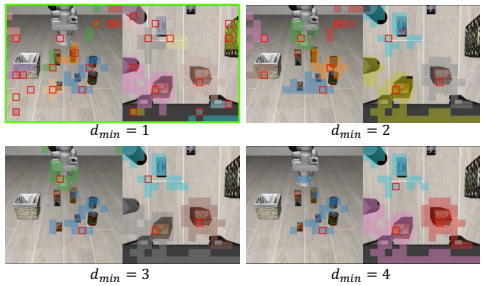


Figure 4: **Comparison of prototype detection and region segmentation under varying minimum peak distance $d_{\min}$.** Red boxes indicate detected prototypes (local attention peaks), and each colored region shows tokens assigned to the nearest prototype via watershed segmentation. As $d_{\min}$ increases, fewer prototypes are detected and regions become larger.

6 Conclusion

We have introduced ProtoP, a training-free visual token pruning method for Vision-Language-Action (VLA) models that structurally prevents spatial collapse by guaranteeing region coverage. By partitioning the attention map into watershed regions anchored at local peaks and enforcing a one-token-per-region floor, ProtoP guarantees that every semantically distinct region of the scene retains representation, regardless of the global pruning ratio. Experiments across LIBERO, VLABench, and DROID show that ProtoP consistently outperforms existing baselines under aggressive compression, achieving 93.3% task success on OpenVLA-OFT at 90% pruning—an 11.1-point gain over the strongest baseline—at a modest additional compute cost. Our key insight, that preserving local attention peaks is critical for maintaining spatial awareness, enables robust deployment of high-performance VLA models even under extreme token budgets.

AI USE STATEMENT

In this work, we used generative AI tools to assist with parts of code implementation, organizing mathematical formulations and algorithm descriptions, proofreading the manuscript text, and searching for and summarizing related work during the literature review. We did not use generative AI tools for any other purpose requiring disclosure. We have reviewed all AI-assisted work: AI-assisted code and the mathematical formulations/algorithm descriptions were reviewed by all authors for correctness, and AI-assisted literature search results were manually verified against the original sources. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

Our work focuses on improving the computational efficiency of Vision-Language-Action models for robotic manipulation. Our experiments are primarily conducted in simulation benchmarks (LIBERO, VLABench, DROID) using publicly released models and datasets; we additionally conduct a small-scale real-robot evaluation (Appendix C) involving only inanimate objects, with no human subjects, crowdsourcing, or personally identifiable data. As ProtoP is a training-free inference-time technique rather than a capability-expanding method, we do not anticipate risks beyond those already associated with the underlying VLA models it accelerates. We discuss potential societal implications, including possible labor-market effects of more efficient robotic automation, in Appendix F.

REPRODUCIBILITY STATEMENT

We provide an algorithmic specification of ProtoP in Algorithm 1, including the attention-scoring procedure, watershed-based region partitioning, and the budget allocation rule. Full experimental settings—model configurations, benchmark protocols, and baseline adaptations (including our adaptation of VLA-Pruner to OpenVLA-OFT)—are detailed in Appendix A, with hyperparameter configurations for all ablation studies summarized in Tab. S1. Computational cost and hardware are reported in Sec. 5.1 and Appendix A.5.

REFERENCES

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736, 2022.

Serge Beucher and Ch Lantuejoul. Use of watersheds in contour detection. int. In *Workshop on Image Processing, CCETT/IRISA, Rennes, France*, 1979.

Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, et al. Paligemma: A versatile 3b vlm for transfer. *arXiv preprint arXiv:2407.07726*, 2024.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. *arXiv preprint arXiv:2210.09461*, 2022.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

Hila Chefer, Shir Gur, and Lior Wolf. Transformer interpretability beyond attention visualization. In *CVPR*, 2021.

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.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.

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.

Siddharth Karamcheti, Suraj Nair, Ashwin Balakrishna, Percy Liang, Thomas Kollar, and Dorsa Sadigh. Prismatic vlms: Investigating the design space of visually-conditioned language models. In *Forty-first International Conference on Machine Learning*, 2024.

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

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.

Haosheng Li, Weixin Mao, Zihan Lan, Hongwei Xiong, Hongan Wang, Chenyang Si, Ziwei Liu, Xiaoming Deng, and Hua Chen. Bfa++: Hierarchical best-feature-aware token prune for multi-view vision language action model. *IEEE Robotics and Automation Letters*, 2026.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *ICML*, 2023.

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.

Youwei Liang, Chongjian Ge, Zhan Tong, Yibing Song, Jue Wang, and Pengtao Xie. Not all patches are what you need: Expediting vision transformers via token reorganizations. In *ICLR*, 2022.

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, 2023a.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023b.

Jizhihui Liu, Guangdao Zhu, and Feiyi Du. Hiprune: Training-free visual token pruning via hierarchical attention in vision-language models (student abstract). In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 41275–41277, 2026.

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.

Lingchen Meng, Hengduo Li, Bor-Chun Chen, Shiyi Lan, Zuxuan Wu, Yu-Gang Jiang, and Ser-Nam Lim. Adavit: Adaptive vision transformers for efficient image recognition. In *CVPR*, 2022.

Fernand Meyer. Topographic distance and watershed lines. *Signal processing*, 38(1):113–125, 1994.

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.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification. *Advances in neural information processing systems*, 34:13937–13949, 2021.

Michael S Ryoo, AJ Piergiovanni, Anurag Arnab, Mostafa Dehghani, and Anelia Angelova. Tokenlearner: What can 8 learned tokens do for images and videos? *arXiv preprint arXiv:2106.11297*, 2021.

Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing*, 26(1):43–49, 1978.

Dachuan Shi, Chaofan Tao, Anyi Rao, Zhendong Yang, Chun Yuan, and Jiaqi Wang. Crossget: Cross-guided ensemble of tokens for accelerating vision-language transformers. In *International Conference on Machine Learning*, 2024.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

Stefan Van der Walt, Johannes L Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D Warner, Neil Yager, Emmanuelle Gouillart, and Tony Yu. scikit-image: image processing in python. *PeerJ*, 2:e453, 2014.

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.

Long Xing, Qidong Huang, Xiaoyi Dong, Jiajie Lu, Pan Zhang, Yuhang Zang, Yuhang Cao, Conghui He, Jiaqi Wang, Feng Wu, et al. Pyramiddrop: Accelerating your large vision-language models via pyramid visual redundancy reduction. *arXiv preprint arXiv:2410.17247*, 2024.

Siyu Xu, Yunke Wang, Chenghao Xia, Dihao Zhu, Tao Huang, and Chang Xu. Vla-cache: Efficient vision-language-action manipulation via adaptive token caching. *arXiv preprint arXiv:2502.02175*, 2025.

Senqiao Yang, Yukang Chen, Zhuotao Tian, Chengyao Wang, Jingyao Li, Bei Yu, and Jiaya Jia. Visionzip: Longer is better but not necessary in vision language models. In *CVPR*, 2025a.

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*, 2025b.

Hongxu Yin, Arash Vahdat, Jose M Alvarez, Arun Mallya, Jan Kautz, and Pavlo Molchanov. A-vit: Adaptive tokens for efficient vision transformer. In *CVPR*, 2022.

Shiduo Zhang, Zhe Xu, Peiju Liu, Xiaopeng Yu, Yuan Li, Qinghui Gao, Zhaoye Fei, Zhangyue Yin, Zuxuan Wu, Yu-Gang Jiang, et al. Vlabench: A large-scale benchmark for language-conditioned robotics manipulation with long-horizon reasoning tasks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11142–11152, 2025.

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.

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 Implementation Details

A.1 Model Setup

OpenVLA-OFT. OpenVLA-OFT (Kim et al., 2024) is a 7B-parameter Vision-Language-Action model built upon the Prismatic-7B backbone (Karamcheti et al., 2024), which combines a fused SigLIP-DINOv2 vision encoder with a Llama-2-based language model. The model is fine-tuned on downstream robot manipulation tasks using Low-Rank Adaptation (LoRA) (Hu et al., 2022) with a continuous action head. Specifically, we use an L1 regression action head that predicts an action chunk of 8 timesteps per inference step, with a LoRA rank of 32. The vision encoder produces 256 patch tokens from a 256×256 image, which are projected into the language model's embedding space and prepended to the task instruction tokens. During action prediction, the model predicts the full action chunk in a single forward pass via parallel decoding, conditioned on the visual and language context. All experiments use the publicly available checkpoints fine-tuned on each LIBERO task suite.

$\pi_{0.5}$. $\pi_{0.5}$ (Intelligence et al., 2025)is a flow-matching VLA built on the PaliGemma-3B (Beyer et al., 2024) backbone. The model uses a diffusion-style action head that generates continuous action trajectories. For evaluation, we use the publicly released checkpoint without modification, following the original inference protocol with classifier-free guidance. Visual tokens are produced by a SigLIP (400M) vision encoder and processed by a Gemma (2B) language backbone, and actions are predicted by a 300M-parameter flow-matching action expert.

A.2 Benchmarks

LIBERO. LIBERO (Liu et al., 2023a) is a tabletop robot manipulation benchmark comprising four task suites: LIBERO-SPATIAL (10 tasks testing spatial reasoning), LIBERO-OBJECT (10 tasks testing object generalization), LIBERO-GOAL (10 tasks testing goal conditioning), and LIBERO-10 (10 long-horizon tasks). Each suite is evaluated over 500 trials (10 tasks $\times$ 50 episodes per task). We report the mean success rate across tasks, following the standard evaluation protocol. Observations consist of a 256×256, a 256×256 wrist-camera RGB image, and the 7-DOF end-effector state. Actions are 7-dimensional (6-DOF pose + gripper) and are predicted in chunks of 8 timesteps.

VLABench. VLABench (Zhang et al., 2025) is a language-conditioned robot manipulation benchmark that evaluates fine-grained instruction following across diverse manipulation scenarios, including object rearrangement, tool use, and multi-step assembly. We evaluate on two benchmark suites (`add_condiment`, `select_drink`) and report task success rate averaged over 50 trials per suite.

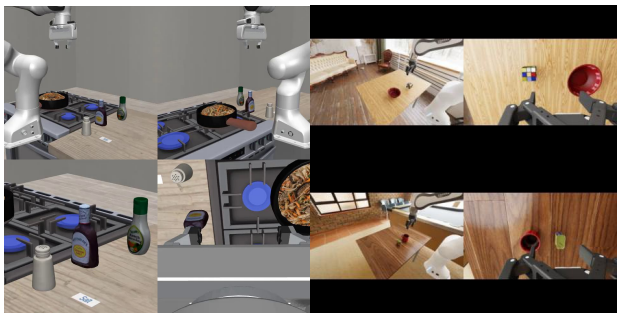


Figure S1: Example simulation environments for VLAbench (left) and DROID (right).

DROID. DROID (Khazatsky et al., 2024) is a large-scale real-robot dataset and benchmark covering diverse everyday manipulation tasks collected across multiple robot platforms. We evaluate on two benchmark suites (*"put the cube in the bowl"*, *"put the can in the mug"*) and report task success rate averaged over 100 trials per suite.

A.3 Baselines

FastV. FastV (Chen et al., 2024) reduces visual token count by identifying uninformative tokens based on the attention scores computed at an early layer. Specifically, FastV retains the K-th transformer layer's attention from the last text token to each visual token, and discards the lowest-scoring

$(1 - R) \times 100\%$ of visual tokens after that layer. Tokens are dropped once and remain absent for all subsequent layers. We use the recommended hyperparameters of $K = 2$ for all evaluations.

SparseVLM. SparseVLM (Zhang et al., 2024) applies progressive text-guided visual token sparsification across multiple transformer layers. In the original formulation, visual tokens are pruned at several intermediate layers (e.g., layers 8, 16, and 24 in a 32-layer LLM) using text-token attention as relevance scores, allowing the model to progressively refine which visual tokens to retain. For a fair comparison with single-shot pruning baselines such as FastV and single-step pruning methods, we adapt SparseVLM to perform a single pruning step at layer 2, retaining only the top-$(1 - r)$ fraction of visual tokens ranked by the text-to-visual attention at that layer. This ensures that all methods operate under the same computational constraint and pruning schedule.

VLA-Cache. VLA-Cache (Xu et al., 2025) exploits temporal redundancy across consecutive inference steps in robot manipulation by caching and reusing key-value (KV) states of visual tokens that have not changed significantly between steps. A change-detection module computes the difference between consecutive image features and selects a subset of visual tokens whose features have shifted, updating only those tokens' KV states while reusing the cached states for the rest. This approach reduces redundant computation without discarding visual information permanently. We use the publicly available implementation with default hyperparameters.

VLA-Pruner. VLA-Pruner (Liu et al., 2025) proposes a dual-level importance scoring strategy that combines vision-language prefill attention and action decoding attention with temporal smoothing to identify important visual tokens. Specifically, at each inference step, the method scores tokens using a weighted combination of the prefill-phase last-token attention (at layer 2) and the action-decode-phase attention, smoothed over a temporal window of $w = 3$ steps with decay factor $\gamma = 0.8$. The lowest-scoring tokens are then removed prior to the action decoding phase. In our experiments, we set the pruning layer to $K = 2$.

The original VLA-Pruner was implemented and evaluated on OpenVLA (Kim et al., 2024), which generates discrete action tokens autoregressively during decoding, making action-phase attention straightforward to extract. However, OpenVLA-OFT fundamentally differs in its inference architecture: action tokens are fed *jointly* into the language model alongside visual and instruction tokens in a single forward pass (rather than being generated autoregressively), so there is no separate action decoding phase from which to extract attention scores.

To adapt VLA-Pruner to OpenVLA-OFT, we replace the action decoding attention with the attention from the *action tokens* within the joint forward pass. Specifically, we extract the attention maps from each action token to each visual token at the scoring layer and aggregate them across action tokens and heads. We then form a union of the top-ranked visual tokens identified by this action-token attention and those identified by the last-instruction-token attention (following FastV), and prune tokens not appearing in either set. This adaptation preserves the dual-level spirit of VLA-Pruner while respecting the architectural constraints of OpenVLA-OFT's chunk-prediction paradigm. While this adaptation approximates the original dual-level spirit, we acknowledge that it may not fully reproduce the intended behavior of VLA-Pruner on autoregressive models.

A.4 Detailed Experimental Settings

Top-k vs. Prototype-Preserving The experiment in Fig. 2(b) compares our prototype-preserving method against the top-k baseline across varying pruning ratios. Due to the computational cost of sweeping over multiple ratios, we set the number of trials per task to 15 at each pruning ratio.

Ablation Study Settings. Unless otherwise noted, all ablation studies employ the prototype-preserving pruning strategy across all four LIBERO task suites. Tab. S1 summarizes the specific hyperparameter configurations for each experiment.

Score Layer Index (l) We investigate the impact of the layer depth from which attention scores are extracted. Since higher layers in the transformer typically capture more task-relevant semantic information, this study evaluates how the choice of layer influences the identification of critical attention peaks.

Table S1: Hyperparameter configurations for ablation studies. Default values ($l = 3, d_{\min} = 1$, Text + Action) are applied unless specified as a variable. l denotes the score layer index, $d_{\min}$ is the minimum peak distance, and r is the pruning ratio.

Ablation Target	Variable Range	Fixed / Default Parameters	Trials/Task
Score Layer Index	$l \in \{1, 2, 3, 4\}$	$d_{\min} = 1, r = 0.9$, Attn: Text + Action	50
Min. Peak Distance	$d_{\min} \in \{1, 2, 3, 4\}$	$l = 3, r \in \{0.6, 0.75, 0.9\}$, Attn: Text + Action	15
Attention Source	7 Configurations	$l = 3, d_{\min} = 1, r = 0.9$	15

Minimum Peak Distance ($d_{\min}$) This parameter $d_{\min}$ controls the spatial density of the structural anchors. We evaluate the trade-off between ensuring spatial diversity (larger $d_{\min}$) and maintaining fine-grained local representations (smaller $d_{\min}$) across varying pruning intensities.

Attention Source ($\mathcal{A}$) We identify the most informative cues for anchoring spatial regions by comparing seven distinct configurations of the attention score map, derived from different token modalities:

- **Visual:** Scores derived solely from visual-to-visual self-attention.
- **Text:** Scores derived solely from visual-to-text cross-attention.
- **Action:** Scores derived from visual-to-action-query attention.
- **Visual + Text:** Union of visual and text-based attention scores.
- **Visual + Action:** Union of visual and action-based attention scores.
- **Text + Action (Default):** Union of text and action attention; used as the default configuration for its robust performance.
- **All:** Holistic combination of Visual, Text, and Action attention sources.

This comprehensive comparison allows us to determine which modality-specific attention maps provide the most reliable structural anchors for maintaining spatial and semantic fidelity.

A.5 Computational Overhead of ProtoP

We measure the computational cost of ProtoP by profiling its components during inference on LIBERO tasks. The additional overhead consists of: (a) peak detection via local maxima ($\sim$0.25 ms), (b) watershed segmentation on a 16×16 grid per camera ($\sim$0.39 ms), and (c) per-region Top-k selection ($\sim$0.1 ms). The total overhead is approximately 0.75 ms, representing less than 1% of the total inference time—negligible compared to the LLM forward pass, which dominates the computational budget at $\sim$103 ms. Thus, ProtoP achieves significant token reduction and performance gains with minimal computational penalty.

B Additional Experiments

B.1 Budget Allocation Strategy Comparison

Our ProtoP framework assigns a token budget to each watershed region proportional to its importance score. As formalized in **Line 10** of Algorithm 1, the budget n_k for each region $\mathcal{R}^k$ is determined by an importance weight w_k, such that $n_k \propto w_k \cdot N_{\text{keep}}$. Crucially, we apply a $\max(1, \cdot)$ operator to ensure that every identified region retains at least its prototype (the local peak). To determine the best allocation strategy, we compare three candidates for defining w_k:

Peak-proportional (Default) allocates tokens in proportion to the raw peak attention score of each region: $w_k = s_{p_k}$. This strategy prioritizes highly salient areas for denser representation.

Uniform assigns an equal number of tokens to each region regardless of its attention magnitude: $w_k = 1/K$. This ensures that all detected objects are represented with equal token density.

Mean-proportional is motivated by the notion of *sharpness*. A region where the peak attention is much higher than its surrounding mean is considered *sharp* (concentrated info), requiring fewer surrounding tokens. Conversely, a *flat* region requires more tokens to represent its distributed signal. Formally, sharpness is peak/mean; inverting this and multiplying by the peak score reduces to $w_k = \text{mean}(\mathcal{R}^k)$.

Table S2: Success rate (%) comparison of budget allocation strategies across LIBERO task suites and pruning ratios. Results are averaged over 15 trials per task (10 tasks per suite). Peak-prop: peak-proportional (ours); Uniform: equal budget per region; Mean-prop: mean-proportional allocation.

Ratio	Method	Spatial	Object	Goal	Long	Avg
$r = 0.7$	Peak-prop (Ours)	99.3	90.0	97.3	94.7	95.3
	Uniform	98.7	93.3	97.3	91.3	95.2
	Mean-prop	99.3	93.3	96.7	94.7	96.0
$r = 0.8$	Peak-prop (Ours)	99.3	92.7	96.7	94.7	95.8
	Uniform	98.7	96.7	95.3	92.7	95.8
	Mean-prop	99.3	90.0	94.7	91.3	93.8
$r = 0.9$	Peak-prop (Ours)	98.0	88.0	92.7	90.7	92.3
	Uniform	95.3	92.0	93.3	84.7	91.3
	Mean-prop	97.3	92.0	90.0	85.3	91.2

We evaluate the three allocation strategies across pruning ratios $r \in \{0.7, 0.8, 0.9\}$ on all four LIBERO task suites. As shown in Tab. S2, the strategies perform comparably across all settings, with differences within 2% across all settings, and peak-proportional allocation achieves the best or comparable performance in nearly all conditions. This suggests that the precise allocation of tokens within each watershed region has little impact on task performance, provided that the **Line 10** constraint is met. The key factor is the preservation of local attention peaks (prototypes) themselves: as long as each identified prototype and its surrounding region retain a sufficient number of tokens to serve as structural anchors, the model's manipulation capability remains robust.

B.2 Extending to Low Pruning Ratios ($r < 0.5$)

Table S3: Success rate (%) and computational cost of our method at low pruning ratios ($r < 0.5$) on LIBERO. At these ratios, we set `fg_fraction`$= 1.0$ to use the full attention map for peak detection and watershed segmentation. Results show that retaining 60–80% of visual tokens incurs negligible performance degradation compared to the unpruned baseline.

OpenVLA-OFT						
Pruning ratio	**Spatial**	**Object**	**Goal**	**Long**	**Avg.↑**	**TFLOPs↓**
No pruning	99.0	98.2	97.4	95.4	97.5	8.650
0.2	99.6	98.4	98.4	94.6	97.8	5.899
0.3	98.6	98.6	97.0	93.0	96.8	5.655
0.4	98.8	98.0	96.8	94.8	97.1	5.348

At low pruning ratios ($r < 0.5$), where most tokens are retained, our method handles structural constraints by setting `fg_fraction` $= 1.0$. This ensures the entire attention map is used for watershed segmentation when the required token count exceeds the default foreground (top 50%) size. Evaluations at $r \in \{0.2, 0.3, 0.4\}$ show performance nearly indistinguishable from the unpruned baseline, confirming that our approach gracefully scales to low-pruning regimes without degradation.

B.3 Sensitivity to the Foreground Threshold

ProtoP separates foreground from background before watershed segmentation by thresholding at the median attention score (Sec. 4.2), retaining the top 50% of tokens as candidates for prototype detection. ProtoP applies this threshold only when the pruning ratio is aggressive: we set `fg_fraction`

= 1.0 (no filtering) when prune_ratio < 0.5, and fg_fraction = 0.5 otherwise (see also Appendix B.2). The threshold is computed as a percentile of each image's own attention distribution rather than a fixed absolute value, so a scene with concentrated attention yields a high cutoff and a diffuse scene yields a low one, keeping the retained foreground fraction at 50% regardless of how peaked or flat the attention map is.

We evaluate sensitivity to this design choice by sweeping fg_fraction over $\{0.25, 0.50, 0.75, 1.00\}$ at $r = 0.9$ across all four LIBERO suites, measuring both task success rate and the stability of the detected prototype set relative to our default (fg_fraction $= 0.50$).

Table S4: Success rate (%) across foreground threshold values at $r = 0.9$ (250 trials per cell).

fg_fraction	Spatial	Object	Goal	Long	Avg. ↑
0.25	98.8	89.6	92.8	92.4	93.4
0.50 (ours)	98.4	88.8	95.2	90.8	93.3
0.75	98.0	88.0	96.8	89.2	93.0
1.00 (no filter)	98.0	86.8	96.4	91.2	93.1

Table S5: Prototype-set identity across foreground threshold values, measured over 50 observations relative to fg_fraction $= 0.50$.

fg_fraction	Avg. #prototypes	Overlap w/ fg=0.5	Spatial shift (patches)
0.25	22.3	0.722 ± 0.095	0.00
0.50 (ours)	31.3	1.00 (ref.)	0.00
0.75	35.9	1.000 ± 0.000	0.00
1.00 (no filter)	36.5	1.000 ± 0.000	0.00

Average success rate varies by at most 0.4 points across the sweep (Tab. S4). Tab. S5 shows why: the detected peak sets are strictly monotone in fg_fraction—looser thresholds (0.75, 1.00) form a strict superset of the default peak set, while a stricter threshold (0.25) retains a stable 72% subset, with zero spatial shift throughout. The threshold never introduces *different* peaks; it only changes how many are retained. Together, these results indicate that ProtoP does not depend on a delicately tuned foreground threshold: the peak set is either identical to or a stable subset of the default across $[0.25, 1.00]$, and task success rate is essentially flat.

B.4 Effect of Peak Detection Method

We investigate whether the choice of local peak detection algorithm affects task performance and inference latency. Our method identifies prototype tokens as local maxima of the attention score map; the default implementation uses skimage.feature.peak_local_max (Van der Walt et al., 2014), which finds peaks via non-maximum suppression (NMS) within a neighborhood of radius $d_{\min}$. We compare it against three alternatives:

- **Baseline (peak_local_max):** The default method from *scikit-image*. For each candidate pixel, it checks whether the pixel is strictly greater than all neighbors within a $d_{\min}$-radius window, retaining only true local maxima.
- **Scipy maximum filter (scipy_maxfilt):** Applies scipy.ndimage.maximum_filter with a $(2d_{\min} + 1) \times (2d_{\min} + 1)$ kernel, then identifies pixels that equal their local maximum. This is functionally equivalent to the baseline but utilizes a different implementation path.
- **Connected-component centroid (cc_centroid):** Thresholds the attention map to obtain foreground blobs, extracts connected components, and uses the centroid of each component as the peak. Notably, this approach does not search for actual maxima within each region, meaning centroids may fall on low-attention pixels.
- **Mean shift (mean_shift):** Runs mean-shift clustering on the 2D attention map and treats cluster centers as peaks. This is a non-parametric density estimation approach that naturally identifies the modes of the attention distribution.

Table S6: Success rate (%) comparison of different peak detection methods on LIBERO suites ($r = 0.9$, 10 trials per task). Our method uses `peak_local_max` (Baseline).

OpenVLA-OFT							
Method	**Spatial**	**Object**	**Goal**	**Long**	**Avg.↑**	**TFLOPs↓**	**Lat(ms)↓**
Peak local max (baseline)	**98.0**	**88.0**	**97.0**	**94.0**	**94.3**	3.251	107.5
Scipy maxfilt	96.0	85.0	97.0	94.0	93.0	3.222	101.9
Mean shift	99.0	86.0	97.0	90.0	93.0	3.211	198.8
CC centroid	76.0	17.0	69.0	23.0	46.3	**3.076**	**95.9**

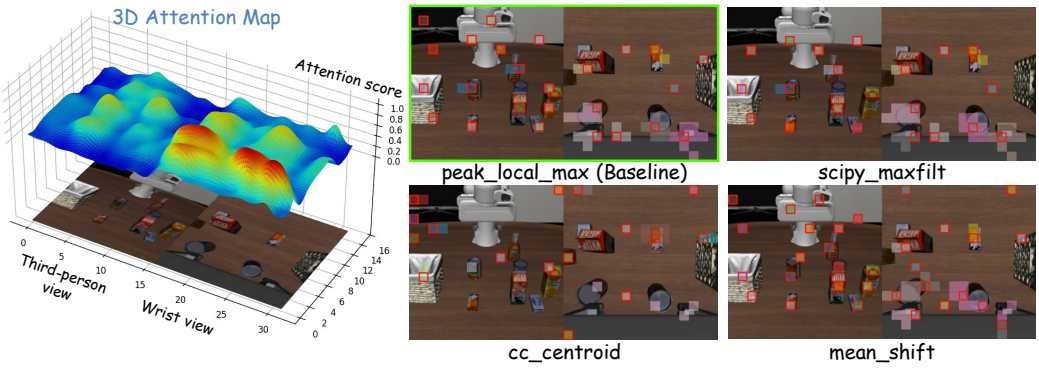


Figure S2: **Qualitative comparison of peak detection methods.** We visualize the detected peaks (prototypes) under a $d_{\min} = 1$ and 90% pruning ratio configuration. While the baseline and other local maximum-based methods (`scipy_maxfilt`, `mean_shift`) identify consistent and semantically relevant attention peaks, the `cc_centroid` approach fails to locate the genuine attention modes, leading to the significant performance degradation observed in our experiments.

As shown in Tab. S6, the baseline and the two alternative local maximum search methods (`scipy_maxfilt` and `mean_shift`) achieve comparable performance, with average success rates staying within 1.3% of each other. In contrast, the connected-component centroid approach (`cc_centroid`) suffers a dramatic degradation (46.3% average), as region centroids often fail to capture the semantically critical attention peaks.

However, a significant discrepancy arises in terms of computational efficiency. While `mean_shift` identifies valid peaks, it incurs a substantial latency of 198.8ms, which is considerably higher than both the baseline and `scipy_maxfilt`. On the other hand, `scipy_maxfilt` maintains nearly identical performance to the baseline while offering a noticeable reduction in inference latency. These results suggest that `scipy_maxfilt` serves as a compelling alternative to our default `peak_local_max` implementation, particularly for scenarios where minimizing latency is prioritized with a negligible trade-off in success rate.

Figure S2 provides a qualitative visualization of the prototypes identified by different peak detection methods under the $d_{\min} = 1$ and 90% pruning ratio configuration. While `peak_local_max`, `scipy_maxfilt`, and `mean_shift` all consistently identify the genuine local peaks of the 3D attention map—explaining their strong task performance—`cc_centroid` is observed to select starkly different points as prototypes. This visual evidence further substantiates the performance gap discussed above; by failing to locate the actual modes of the attention distribution, `cc_centroid` misses semantically critical tokens, leading to the observed failure in manipulation tasks.

B.5 Number of Detected Prototypes

Fig. S3 shows the number of detected prototypes k at each policy query step across all four LIBERO task suites. The value of k remains remarkably stable throughout each episode and across tasks, predominantly falling in the range of 24–30, with occasional outliers in some tasks, which corresponds to approximately 5% of the total 512 visual patches. Such consistent behavior suggests that

the attention score map reliably exhibits a moderate number of local peaks regardless of the specific task, manipulation phase, or the underlying policy model—with no extreme cases of a near-flat map or an excessively noisy one. We take this as evidence that our prototype-based token selection is well-posed and robust across diverse model and task settings.

C Real-Robot and Robustness Validation

While our main evaluation is conducted in simulation—which allows precise control over object sets, camera geometry, and pruning ratio, enabling us to attribute failures cleanly to token selection rather than sensor noise or hardware variability—we additionally validate ProtoP under two complementary conditions it was not designed for: physical deployment (Sec. C.1) and synthetic visual perturbations that probe robustness to real-world attention noise (Sec. C.2).

C.1 Real-Robot Deployment

Setup. We evaluate on an XLeRobot platform (v0.4.0), equipped with dual SO-101 arms and three RGB camera views. The policy is fine-tuned for 15,000 steps on a pick-up task. To test whether ProtoP identifies the correct attention peak under visual clutter, the scene deliberately includes multiple task-irrelevant distractor objects alongside the target. We evaluate the task *"Pick up the doll"*, where a doll (target), a small carton, and a snack package are placed together in the scene, over 10 trials per configuration.

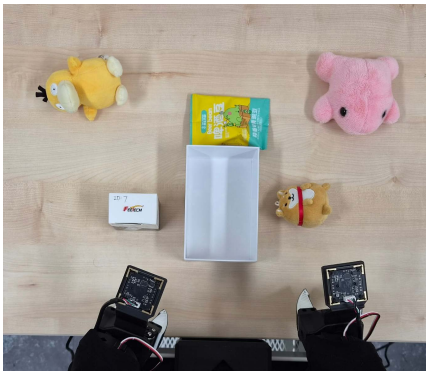


Figure S4: **Real-robot experimental setup.** Top-down view of the XLeRobot platform and the cluttered tabletop scene used for the *"Pick up the doll"* task. The target object (doll) is placed alongside task-irrelevant distractors (a small carton and a snack package) to test whether ProtoP's prototype detection remains robust under visual clutter.

Results. Table S7 reports success rate (SR) and control frequency across pruning ratios of 60%, 75%, and 90%. The unpruned vanilla model achieves 70% SR at a control frequency of 3.35 Hz. Consistent with our simulation results, baseline pruning methods degrade substantially as the pruning ratio increases—FastV and SparseVLM drop to 20% SR at 90% pruning, and VLA-Pruner falls to 30%. ProtoP achieves the highest or tied-highest SR at every ratio, maintaining 70% SR at both 60% and 75% pruning (matching the unpruned baseline) and 50% SR at 90% pruning, while maintaining control frequencies comparable to the baselines (3.63–3.84 Hz). These results confirm that the spatial-collapse failure mode identified in simulation also manifests on physical hardware under cluttered backgrounds, and that prototype-preserving pruning mitigates it under real-world sensor noise and scene complexity, corroborating our simulation findings without altering any of the paper's conclusions.

Latency breakdown. The reported control frequencies (3.35–3.84 Hz) remain below the 10–20 Hz range typically cited for real-time VLA control. This reflects our deployment architecture rather than a limitation of ProtoP: policy inference runs on a remote GPU server communicating with the robot's onboard controller over Wi-Fi, and of the $\sim$280 ms end-to-end latency, only $\sim$110 ms is spent on model inference—the remainder is network round-trip and robot-side I/O, neither of which is affected by token pruning. This real-robot study is designed as a feasibility check—whether the robustness observed systematically in simulation transfers to a physical, cluttered environment—rather than a demonstration of real-time deployment; closing the systems-level latency gap (e.g., on-device inference, a wired connection) is an orthogonal engineering direction independent of the pruning method itself.

Table S7: Real-robot results on the task *"Pick up the doll"* using an XLeRobot platform (10 trials per configuration). Vanilla (no pruning): SR = 70%, Control Freq. = 3.35 Hz.

	Prune 60%		Prune 75%		Prune 90%	
Method	SR (%)	Ctrl. Freq.	SR (%)	Ctrl. Freq.	SR (%)	Ctrl. Freq.
FastV	50	3.68	40	3.91	20	3.94
SparseVLM	20	3.55	20	3.56	20	3.64
VLA-Pruner	50	3.57	70	3.67	30	3.71
ProtoP (Ours)	**70**	3.63	**70**	3.74	**50**	**3.84**

C.2 Robustness to Visual Perturbations

The real-robot deployment above tests ProtoP under uncontrolled real-world conditions, but does not isolate *which* sources of visual disturbance prototype detection tolerates. We complement it with a controlled diagnostic analysis targeting conditions common in real-world deployment: lighting variation, sensor noise, occlusion, and specular reflections.

Setup. We construct a stability analysis over 50 diverse LIBERO observations (5 per task $\times$ 10 tasks), comparing prototype detection under clean vs. perturbed inputs across four perturbation types: Gaussian noise ($\sigma = 0.05$), brightness scaling ($\times 1.30$), occlusion (14% of image area), and a synthetic specular highlight. We report three metrics: the prototype overlap ratio (fraction of clean-image prototypes preserved under perturbation), the mean spatial shift of matched prototypes on the 16×16 patch grid (computed via Hungarian assignment between clean and perturbed prototype sets), and the cosine similarity of the score-layer attention distribution that feeds watershed segmentation.

Table S8: Robustness of prototype detection under visual perturbations, measured over 50 LIBERO observations. Attention cosine similarity is computed on the score-layer attention distribution (Eq. 1) before watershed segmentation.

Perturbation	Prototype overlap	Spatial shift (patches)	Attention cos-sim
Gaussian noise ($\sigma = 0.05$)	0.754 ± 0.083	0.585 ± 0.303	0.997 ± 0.002
Brightness ($\times 1.30$)	0.909 ± 0.041	0.190 ± 0.212	0.998 ± 0.000
Occlusion (14% area)	0.804 ± 0.065	0.507 ± 0.289	0.995 ± 0.006
Specular highlight	0.830 ± 0.062	0.404 ± 0.259	0.998 ± 0.001

Results. Prototype detection is robust to this class of perturbation. This is structurally expected: prototype scores come from LLM decoder attention at layer 3, after passing through the vision backbone and two prior decoder layers that already attenuate pixel-level noise, and watershed itself operates on a 16×16 semantic attention grid (each cell aggregating a 14×14 patch) rather than raw pixels. Table S8 confirms this empirically: (a) the score-layer attention distribution stays exceptionally stable, with cosine similarity ≥ 0.995 under every perturbation; (b) matched-prototype spatial shift averages under 1 patch (out of a 16×16 grid) across all conditions, indicating that the detected foreground regions barely move under perturbation; and (c) 75–91% of prototypes are preserved exactly across all four perturbation types, showing that the majority of the pruning topology is invariant. Together with the real-robot results in Sec. C.1, this indicates that ProtoP's robustness derives from operating on semantically compressed attention features rather than raw pixel statistics, and is not specific to the clean visual conditions of simulation.

D Additional Qualitative Results

D.1 Qualitative Analysis on LIBERO Task Suites

Figure S5 provides an extensive qualitative analysis of ProtoP across various LIBERO task suites and pruning ratios. The visualization illustrates the temporal evolution of the retained tokens throughout

an episode, demonstrating how our method consistently preserves task-relevant objects over time. Furthermore, it highlights the adaptive nature of our budget allocation: as the pruning ratio r increases, the number of tokens assigned to the regions surrounding each prototype is reduced in a granularity-aware manner, while the core structural anchors (the prototypes themselves) remain intact. This visual evidence confirms that ProtoP effectively maintains the semantic layout of the scene even under extreme compression.

D.2 Comparative Qualitative Analysis on VLABench

To further evaluate the discriminative power of ProtoP in complex environments, we conduct a comparative study on the VLABench suite, which requires processing multi-view (3-view) camera inputs. As shown in Fig. S6, we analyze the performance of various methods using the task "Add ketchup to the dish" as a representative case study.

We observe distinct failure modes in existing baselines at a high pruning ratio ($r = 0.9$). FastV exhibits severe *spatial collapse* (highlighted by yellow circles), failing even to localize the initial object (the ketchup bottle). While SparseVLM and VLA-Pruner successfully manage the initial grasp of the ketchup, they lose track of the target "dish" in the subsequent frames. This is because their pruning strategies often discard low-attention but semantically vital regions, leading to a failure in spatial reasoning where the ketchup is dispensed at incorrect locations.

In contrast, ProtoP maintains the spatial information of both the ketchup and the dish by preserving their respective prototypes. By ensuring that structural anchors for all potential interactive objects are retained, ProtoP enables the policy to achieve precise localization and successfully execute the multi-stage task from grasping to final placement.

Actual Token Count and Compute Transparency. The gap between the nominal target N_{keep} and the realized token count in ProtoP arises from two primary sources, as formalized in Algorithm 1 (Line 10) where per-region budgets are allocated using a floor operation $\lfloor \cdot \rfloor$ with a $\max(1, \cdot)$ constraint.

Table S9: Nominal vs. actual retained token counts for ProtoP on OpenVLA-OFT. The gap arises from the floor operation in Algorithm 1 and a group-size cap that prevents over-allocation to small watershed regions. Despite retaining fewer tokens, ProtoP consistently outperforms baselines, confirming that spatial diversity matters more than absolute count.

Pruning Ratio (r)	Nominal N_{keep}	Actual (ProtoP)	Gap
0.6	205	∼150	∼55
0.75	128	∼110	∼18
0.9	51	∼48	∼3

The first source is the **floor effect**: truncating fractional allocations across the $k \approx 24$–30 detected regions results in a loss of up to one token per region, contributing a total gap of approximately k.

The second source is the **group-size cap**: while peak-proportional allocation assigns budgets based on attention scores, the number of tokens actually selectable is bounded by the region's physical watershed area. When the allocated budget exceeds the region size, the surplus is discarded rather than redistributed. This effect is most pronounced at low pruning ratios ($r = 0.6$), where per-region budgets are large relative to region sizes, and diminishes at high ratios ($r = 0.9$) where each region typically receives only 1–2 tokens.

As reported in Tab. S9, this discrepancy is reported explicitly for transparency. Crucially, ProtoP matches or outperforms all baselines at every pruning ratio (Tab. 2) despite the lower token count, suggesting that performance gains stem from *where* tokens are preserved—across spatially diverse, attention-salient regions—rather than simply *how many* are retained.

Regarding **scoring layer depth**: ProtoP extracts attention at layer 3, one layer deeper than SparseVLM and VLA-Pruner (layer 2). This incurs a fixed overhead visible in the TFLOPs at $r = 0.9$ (Tab. 2).

E LIMITATIONS

While ProtoP demonstrates significant efficiency gains across various architectures, its latency benefit for diffusion-based VLA models like $\pi_{0.5}$ depends on the inference setup. Without optimization, the primary bottleneck in such models is the iterative denoising process within the action expert, so reducing prefix-encoding TFLOPs alone translates poorly into wall-clock speedup. Under compiled inference, where prefix encoding constitutes a larger share of total latency (Sec. 5), pruning does yield a measurable speedup (up to 1.60× at 90% pruning), though still smaller than the gains observed on autoregressive models such as OpenVLA-OFT (1.63×). Further optimizing the denoising schedule itself remains complementary to token pruning for diffusion-based VLA models.

ProtoP relies on the quality of attention scores extracted from a specific scoring layer to identify prototypes. As shown in Tab. 4, the choice of layer significantly affects performance, and the optimal layer may vary across VLA architectures beyond those evaluated here. Additionally, the median-threshold foreground separation unconditionally removes the bottom 50% of tokens before prototype detection, which may discard weakly-attended but task-relevant context in scenes with diffuse attention distributions.

While our main evaluation is conducted in simulation, we additionally validate ProtoP on a real-robot platform (Appendix C); broader real-robot evaluation across more diverse tasks and environments remains an important direction for future work.

F BROADER IMPACTS

This work improves the computational efficiency of Vision-Language-Action (VLA) models, enabling faster and more accessible deployment of robotic manipulation systems on resource-constrained hardware. By reducing inference latency without sacrificing task performance, ProtoP may facilitate broader adoption of intelligent robots in real-world settings such as manufacturing, healthcare, and assistive robotics. On the negative side, more efficient robotic systems could accelerate automation in ways that displace human workers in certain labor markets. Additionally, improved robotic manipulation capabilities could potentially be misused in unintended or harmful applications. However, as ProtoP is a general efficiency technique rather than a capability-expanding method, we do not anticipate direct pathways to serious misuse beyond those already associated with the underlying VLA models.

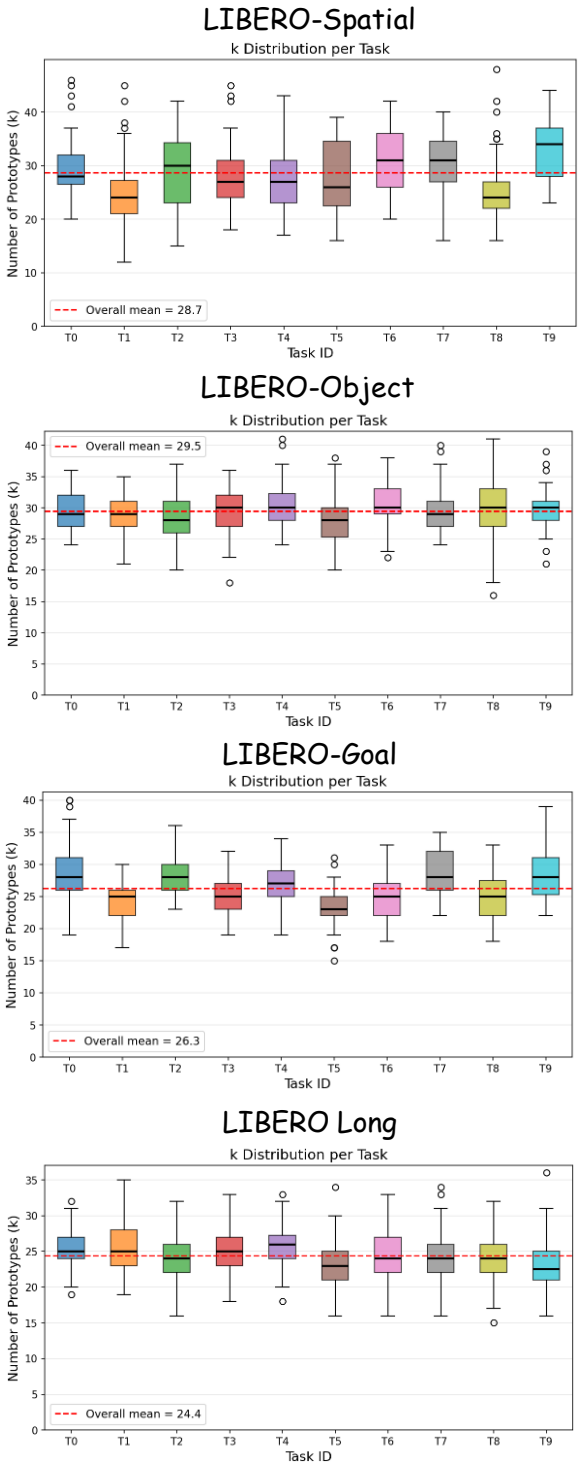


Figure S3: **Stability of the number of detected prototypes k.** We plot the number of detected prototypes k across all policy query steps for the four LIBERO task suites and OpenVLA-OFT. Throughout the episodes, k remains remarkably consistent within the range of 24–30 (approximately 5% of the 512 visual patches). This stability across diverse tasks and manipulation phases indicates that the attention-based structural anchors are reliably and robustly identified without extreme variations in attention map density.



Figure S5: **Qualitative comparison of ProtoP across pruning ratios and LIBERO task suites.** We visualize the retained tokens for various tasks and ratios ($r \in \{0.6, 0.75, 0.9\}$) to demonstrate the stability of our prototype-preserving selection mechanism.

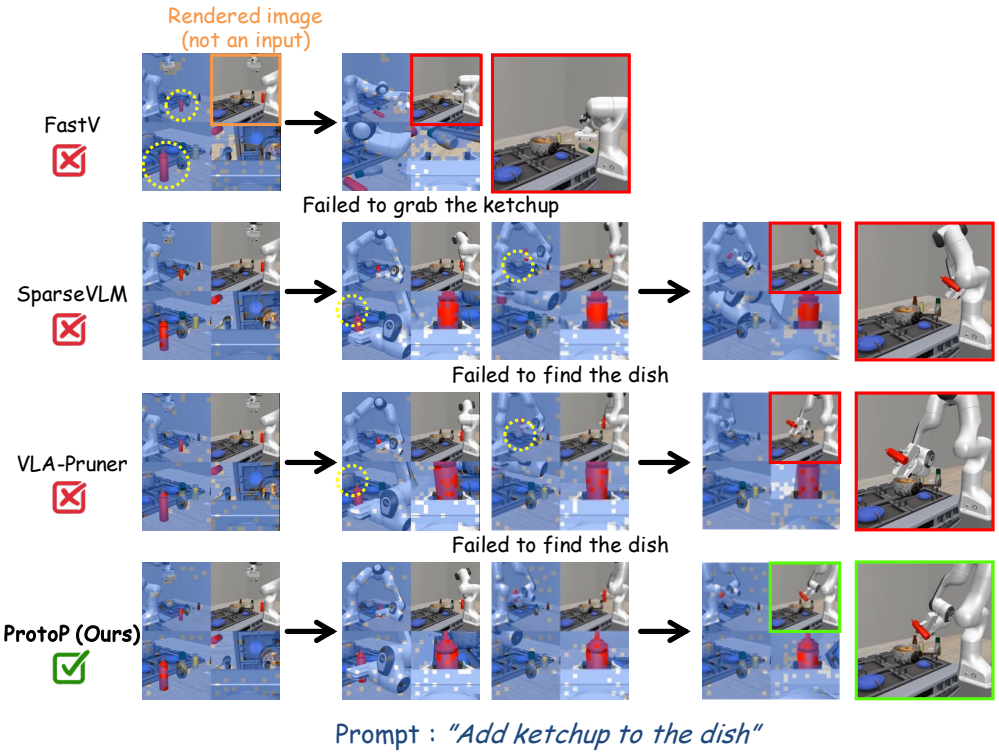


Figure S6: **Comparative qualitative analysis on VLABench** ($r = 0.9$)**.** Yellow circles indicate instances of *spatial collapse* or localization failure. Unlike baselines that lose track of the target dish, ProtoP preserves the necessary spatial anchors to successfully complete the "Add ketchup" task.