DEPTHPRUNER: VISUAL TOKEN PRUNING WITH PLUGGABLE DEPTH GUIDANCE FOR EFFICIENT VISION-LANGUAGE-ACTION INFERENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-language-action (VLA) models use visual observations and language instructions to generate robot actions, but processing visual tokens is computationally expensive. Attention-based visual token pruning reduces this cost by retaining a subset of visual tokens. However, we observe that some potentially task-irrelevant tokens can receive high attention scores and compete with task-relevant tokens for a limited token budget. Depth provides geometric cues that complement attention scores for token selection. Motivated by this observation, we propose DepthPruner, a pluggable, training-free method that augments attention-based visual token pruning methods with depth guidance. It expands the candidate set provided by a base pruner and uses mean depth, residual variance, and neighborhood contrast to select tokens within the original budget. Experiments on OpenVLA-OFT across four LIBERO suites show that DepthPruner improves the average task success rate of all three base pruners at every tested token retention ratio. By replacing at most 8 of FastV's selected tokens per view, DepthPruner increases the success rate on LIBERO-Object by 9.0 percentage points at 25% token retention. Timing measurements show that depth refinement adds little inference overhead.

1 INTRODUCTION

Vision-language-action (VLA) models map visual observations and language instructions to robot actions. During closed-loop manipulation, VLA models repeatedly process new observations and generate actions, making inference latency and computational cost key deployment constraints. Recent work addresses these costs at different stages of the policy pipeline. Compact architectures such as Evo-1 (Lin et al., 2026) reduce model size, while MoLe-VLA (Zhang et al., 2026) dynamically selects which backbone layers to execute. VLA-Cache (Xu et al., 2025) reuses cached computations for visually stable regions across consecutive observations, and FASTer (Liu et al., 2026) combines compact action tokenization with block-wise decoding to reduce the number of sequential decoding steps.

Visual inputs to VLA models contain redundant tokens with overlapping information or limited task relevance (Yang et al., 2025; Pei et al., 2026). This redundancy motivates visual token pruning, which reduces computation in subsequent model layers by retaining a compact subset of tokens. Training-free methods for vision-language models (VLMs) provide a starting point for accelerating VLAs built on these backbones. FastV (Chen et al., 2025) ranks visual tokens by the average attention they receive at a selected language-model layer and removes low-scoring tokens from subsequent layers. SparseVLM (Zhang et al., 2025) uses attention from visually relevant text tokens to select visual tokens. VLA-specific methods further exploit the language and action information available during robot execution. Action-aware Dynamic Pruning (ADP) (Pei et al., 2026) combines language-guided token selection with a pruning decision based on recent end-effector motion. VLA-Pruner (Liu et al., 2025) selects candidates using current semantic attention and smoothed action attention from previous steps. SAFE-Pruner (Ma et al., 2026) matches current tokens to semantically similar tokens in a historical reference frame and transfers deep-layer saliency to guide early pruning. Across these designs, the central selection problem is to retain action-relevant information within a limited visual token budget.

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.

Attention provides a useful signal for visual token selection, but high attention scores do not always indicate informative local content. Darcet et al. (2024) identify high-norm tokens and associated attention artifacts in visually homogeneous background regions. SinkPruner (Li et al., 2026b) further shows that redundant background tokens can be over-retained by attention-based pruning, leaving fewer slots for other visual information. These findings motivate complementing attention with a criterion that helps distinguish informative local content from redundancy under a fixed token budget.

Depth, readily available in many embodied settings, provides direct geometric information that can enhance VLA models' spatial understanding. Prior work has used depth for policy representation learning (Yuan et al., 2026), visual token merging (Li et al., 2026d), and cross-view structural reallocation in token pruning (Cheng et al., 2026). To address the over-retention of redundant tokens discussed above, we study how depth geometry can complement attention-based visual token pruning under a fixed budget with little additional computation.

We propose DepthPruner, a training-free method that augments attention-based visual token pruning with pluggable depth guidance (Fig. 2). It first obtains a slightly expanded candidate set from the base pruner, then combines mean depth, residual variance, and neighborhood contrast to refine token selection while preserving the original token budget. This lets geometry reassess redundancy within regions identified by the base pruner. DepthPruner requires neither an additional learned model nor changes to the VLA architecture and incurs only quasilinear computational overhead during inference.

Our contributions are twofold:

- We introduce DepthPruner, a training-free, pluggable method that complements attention with depth geometry to refine visual token selection in VLA models while preserving the original token budget.
- We evaluate DepthPruner on OpenVLA-OFT across four LIBERO suites and three pruning methods, showing that a few depth-guided token replacements can improve task success rates for different pruning methods with little inference overhead.

2 Related Work

Vision-Language-Action Models. VLA models build on pretrained vision-language models to map visual observations and language instructions to robot actions. For example, OpenVLA (Kim et al., 2025b) predicts discretized actions as a sequence of tokens. OpenVLA-OFT (Kim et al., 2025a) combines parallel decoding, action chunking, and continuous action regression to improve policy efficiency and performance. π_0 (Black et al., 2025) uses a flow-matching action expert to generate continuous action chunks. Across these architectures, repeatedly processing visual observations during robot execution motivates reducing visual token computation while retaining information needed for control.

Visual Token Pruning for VLA Inference. Most visual token pruning methods for VLAs rely on attention scores and differ in the additional cues they use. Feature diversity complements attention-based relevance in EfficientVLA (Yang et al., 2025). Temporal information guides VLA-Pruner (Liu et al., 2025), which combines semantic candidates with candidates selected using smoothed historical action attention and then uses feature diversity to remove redundancy. Motion dynamics control pruning in ADP (Pei et al., 2026): recent end-effector trajectories determine when to apply text-guided pruning instead of processing the full set of visual tokens. Semantic-region constraints guide DTP (Li et al., 2026a), which suppresses tokens that receive abnormally high action-generation attention despite lying outside prompt-relevant regions. These methods do not explicitly use depth-derived geometry as a token selection criterion. A smaller group of methods avoids attention-score ranking: DivPrune (Alvar et al., 2025) selects tokens by feature diversity and has been evaluated on VLA models (Liu et al., 2025), while GridS (Feng et al., 2026) learns continuous sampling coordinates and interpolates visual features through joint training with the policy.

Spatial Information for VLA Models. Spatial information can enrich policy representations or guide visual token reduction. SpatialVLA (Qu et al., 2025) learns depth-derived 3D position

encodings, while DepthVLA (Yuan et al., 2026) integrates a depth expert and jointly optimizes depth prediction and action generation. These approaches incorporate geometry through additional learned components and policy training. For token reduction, SP-VLA (Li et al., 2026c) retains the union of attention-selected tokens and RGB contour tokens identified using Canny edge detection. These RGB contours cannot directly measure camera-relative distance or local 3D surface variation. DepthCache (Li et al., 2026d) uses depth to guide token merging while leaving tokens protected by attention or depth gradients unmerged. Both methods preserve attention-selected tokens, so task-irrelevant regions with high attention can remain in the reduced sequence (Li et al., 2026a). IAprune (Cheng et al., 2026) reallocates token slots within a fixed budget by projecting wrist-view RGB edges onto the main view; depth and camera calibration serve only to align the views, while the structural cue comes from image edges. DepthPruner instead derives selection cues directly from per-view depth, including distance, detrended local surface variation, and neighborhood depth contrast. It uses these cues to reselect tokens from an existing pruner's candidate set, requiring no training and little additional computation.

3 Preliminaries and Motivation

3.1 Visual Token Pruning in VLAs

A common VLA architecture connects a vision transformer (ViT) encoder to a large language model (LLM) backbone through a visual projector (Kim et al., 2025b; Black et al., 2025). The encoder extracts patch features from RGB observations. The projector maps these features into the LLM embedding space, where they are processed together with language and other nonvisual tokens. We consider token pruning at a single designated location before or within the LLM backbone. For a frozen policy with V camera views, let $\mathbf{X}^v = [\mathbf{x}_1^v, \ldots, \mathbf{x}_{N_v}^v]^\mathsf{T} \in \mathbb{R}^{N_v \times d}$ denote the visual tokens of view v at the pruning point, where N_v is the number of patches and d is the hidden dimension.

Visual token pruning retains a fraction $\rho \in (0, 1]$ of the visual tokens. The total budget is $K_{\text{tot}} = \text{round}(\rho \sum_{v=1}^{V} N_v)$, which the base pruner allocates across views as K_v with $\sum_v K_v = K_{\text{tot}}$. Let $\mathcal{B}^v(k)$ return k visual-token indices using the base pruner's native selection rule and its usual inputs and state. The selected indices and retained token sequence are

$$\mathcal{K}_{\mathcal{B}}^v = \mathcal{B}^v(K_v), \qquad \widetilde{\mathbf{X}}_{\mathcal{B}}^v = \mathbf{X}^v[\mathcal{K}_{\mathcal{B}}^v, :], \tag{1}$$

where rows are gathered in their original sequence order.

The retained visual tokens and all N_{nv} nonvisual tokens form a shortened multimodal sequence $\widetilde{\mathbf{H}}_{\mathcal{B}} \in \mathbb{R}^{(K_{\text{tot}}+N_{\text{nv}})\times d}$, preserving their original order. Visual tokens outside $\mathcal{K}_{\mathcal{B}}^v$ are removed from the sequence, so subsequent attention and feed-forward layers operate on only $K_{\text{tot}} + N_{\text{nv}}$ token states. The remaining backbone layers process this sequence as

$$\mathbf{H}_{\text{out}} = F_{\text{rem}}(\widetilde{\mathbf{H}}_{\mathcal{B}}), \tag{2}$$

where F_{rem} denotes the remaining backbone layers and $\mathbf{H}_{\text{out}}$ denotes the resulting hidden states. The policy retains its original action generation mechanism. Progressive pruning selects and removes tokens at multiple layers, using the current token representations and the retention budget at each pruning location. Our goal is to improve visual token selection under a fixed retention budget.

3.2 Motivation for Depth Guidance

Attention-based selection can assign high priority to image regions outside the current manipulation targets. DTP (Li et al., 2026a) reports that VLA policies can attend strongly to task-irrelevant regions during action generation, thereby reducing task success rates. Figure 1 shows attention ranks from OpenVLA and OpenVLA-OFT for the same LIBERO-Spatial main-camera observation. With attention ranked separately for each model, high-ranking patches appear on the wall near the upper image corners in OpenVLA and on the tabletop near the lower image corners in OpenVLA-OFT. These background regions lie outside the bowl and plate specified by the manipulation instruction.

Prior work links such high responses to locally repetitive regions. Darcet et al. (2024) associate artifacts in ViT attention maps with high-norm tokens that often occur in locally repetitive background regions. SinkPruner (Li et al., 2026b) likewise observes such tokens in MLLM vision encoders and

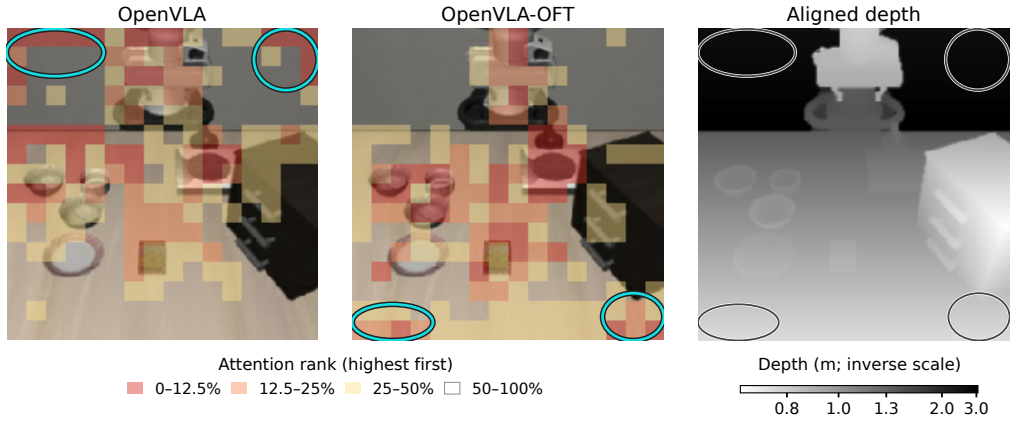

Figure 1: Attention and depth for the same LIBERO-Spatial observation. The first two columns overlay each model's attention ranks on RGB, and the third shows aligned depth with inverse-depth grayscale mapping. Circled background patches on the wall (OpenVLA) and tabletop (OpenVLA-OFT) receive high attention despite lying outside the manipulation targets. Depth reveals locally smooth, continuous wall and tabletop surfaces across neighboring patches, providing a geometric cue to potential redundancy.

shows how their high CLS attention can cause pruning methods using these scores to prioritize redundant tokens. Aligned depth provides a direct geometric view of this local repetition. The third column of Figure 1 shows the aligned depth of the same observation: in the circled wall and tabletop regions, depth varies smoothly across neighboring patches and is continuous with that of the surrounding surfaces. This continuity provides a cue to potential local redundancy that attention rank alone does not capture.

A complementary cue should also preserve the computational benefit of pruning. DTP (Li et al., 2026a) detects distracting tokens with an extra forward pass and then reruns the model with these tokens masked, incurring additional computation. In contrast, local geometric statistics can be computed directly from aligned depth without additional model passes. We therefore use depth geometry to refine the candidates of an existing attention-based pruner while preserving its original token budget. Section 4 develops this criterion, and Sec. 5.3 measures its incremental latency in complete policy calls.

4 Method

DepthPruner uses aligned depth to refine an existing selector's visual token candidates (Fig. 2). The base selector uses an expanded budget to define the candidate set, and depth geometry guides selection within this set while preserving the original per-view token budget. Depth guides token selection without changing the policy architecture or learned parameters.

4.1 Geometric Refinement of Baseline Candidates

Candidate expansion allows geometry to replace tokens within a candidate set formed by the baseline's native selection rule. For a final per-view budget K_v, an expansion factor $\alpha \geq 1$ determines the candidate budget:

$$M_v = \min\{N_v, \lceil \alpha K_v \rceil\}. \tag{3}$$

The base selector $\mathcal{B}^v$ from Sec. 3.1 returns the candidate set $\mathcal{C}^v = \mathcal{B}^v(M_v)$. Geometry then selects K_v of these candidates, refining the retained token set without increasing the final sequence length.

The additional candidate allowance should reflect the redundancy in the baseline's retained tokens, which can vary with the base selector, dataset, VLA model, and retention ratio. Accordingly, α can be fixed or follow a chosen schedule; Sec. 5.1 specifies the setting used in our experiments.

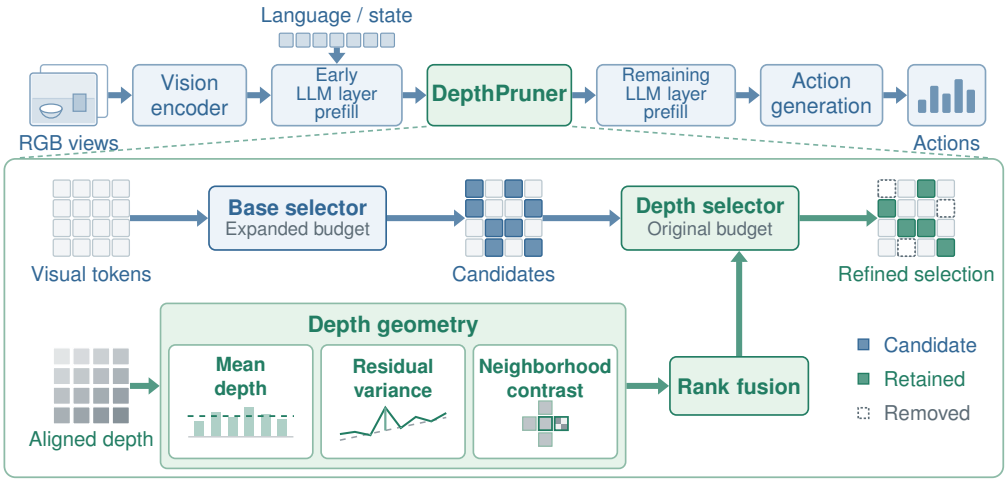

Figure 2: Overview of DepthPruner. At the base selector's pruning point, expanded baseline candidates are refined to the original per-view budget using three geometric cues ranked over the full view grid. The policy retains its native action generation mechanism. Blue, green, and dashed white tokens denote candidate, retained, and removed tokens; examples are schematic.

4.2 Local Depth Geometry

We use three complementary cues: mean depth, residual variance, and neighborhood contrast. Given finite, positive metric depth Z^v aligned with each prunable RGB view, we compute all three cues on the full patch grid. Below, we suppress the view index.

Mean depth. Mean depth provides a distance prior that lowers the retention priority of candidate tokens representing regions farther from the camera. This helps distinguish distant background regions from manipulation-relevant regions. For the m pixels Ω_i aligned with token i, we compute

$$\mu_i = \frac{1}{m} \sum_{(x,y)\in\Omega_i} Z(x,y). \tag{4}$$

This coarse preference is combined with the two structural cues below.

Residual variance. Regions whose depth follows a smooth local trend contain limited additional geometric detail, motivating a lower retention priority. Raw depth-gradient magnitude can also respond strongly to the slope of a smooth surface. To discount this slope, we fit an affine depth surface in local pixel coordinates and compute its relative residual variance:

$$\begin{aligned} (a_i, b_i, c_i) &= \arg\min_{a,b,c} \sum_{(x,y)\in\Omega_i} [Z(x,y) - ax - by - c]^2, \\ \widehat{Z}_i(x,y) &= a_i x + b_i y + c_i, \qquad r_i = \frac{\mathrm{Var}_{\Omega_i}[Z - \widehat{Z}_i]}{\max(\mu_i^2, \epsilon)}. \end{aligned} \tag{5}$$

Here $\epsilon = 10^{-8}$ and the variance is computed with divisor m. The residual variance captures variation beyond an affine trend in image coordinates, while division by squared mean depth expresses it relative to the patch's depth scale.

Neighborhood contrast. Prior studies have found that high-attention outlier tokens often occur in locally repetitive background regions (Darcet et al., 2024; Li et al., 2026b). Motivated by these findings, we use differences between neighboring depth distributions as a complementary cue to local geometric redundancy. Let $z_{i,(1)} \le \cdots \le z_{i,(m)}$ denote the sorted depths of patch i. For neighboring patches i and j of the same shape, the mean absolute difference of their sorted depths gives the Wasserstein-1 distance between their empirical distributions. We subtract the shift predicted by

the average fitted slope $\mathbf{g}_i = (a_i, b_i)$ inside the absolute difference to discount translation along a smooth sloping surface:

$$\begin{aligned} s_{ij} &= \tfrac{1}{2}(\mathbf{g}_i + \mathbf{g}_j)^{\mathsf{T}}(\mathbf{p}_j - \mathbf{p}_i), \\ d_{ij} &= \frac{m^{-1}\sum_{k=1}^{m} |z_{j,(k)} - z_{i,(k)} - s_{ij}|}{\max((\mu_i + \mu_j)/2, \epsilon)}, \qquad e_i = \max_{j \in \mathcal{N}_4(i)} d_{ij}. \end{aligned} \tag{6}$$

Here $\mathbf{p}_i$ is the patch center in image coordinates and $\mathcal{N}_4(i)$ contains valid four-connected neighbors within the view. Equally shaped patches on an exact affine depth surface have zero neighborhood contrast, up to numerical precision. The neighborhood contrast e_i preserves the strongest compensated difference, so a larger value indicates lower geometric similarity to at least one neighbor.

4.3 Geometric Ranking and Selection

We combine the three cues from Sec. 4.2 into a retention priority for selecting K_v tokens from the candidates $\mathcal{C}^v$ in Sec. 4.1. Their different units and numerical ranges motivate rank-based normalization within each view. For a fixed view v, let $N = N_v$ and omit the view superscript in the score definitions. For $N > 1$, define $R(z)_i = (\mathrm{rk}_{\uparrow}(z)_i - 1)/(N-1) \in [0,1]$, where $\mathrm{rk}_{\uparrow}$ assigns one to the smallest value and breaks ties using the original patch index; $\mathrm{rk}_{\downarrow}$ analogously assigns one to the largest value. Computing these ranks over the full patch grid keeps the geometric scores independent of the candidate-set size for a given depth map.

With nonnegative weights w_z, w_r, w_e for mean depth, residual variance, and neighborhood contrast, respectively, we form a background-oriented score β_i. The score and corresponding retention priority are

$$\begin{aligned} \beta_i &= w_z R(\mu)_i + w_r[1 - R(r)_i] + w_e[1 - R(e)_i], \\ D_i &= \mathrm{rk}_{\downarrow}(\beta)_i, \qquad \mathcal{K}^v = \mathrm{TopK}_{i \in \mathcal{C}^v}(D_i^v, K_v), \qquad w_z + w_r + w_e = 1. \end{aligned} \tag{7}$$

This fused score guides the geometric selector to favor tokens with lower mean depth, higher residual variance, and higher neighborhood contrast. Here TopK returns the indices of the K_v candidates with the largest priorities.

The refined index set $\mathcal{K}^v$ replaces $\mathcal{K}_{\mathcal{B}}^v$ in Eq. 1. We gather the selected rows in their original sequence order to obtain $\widetilde{\mathbf{X}}^v = \mathbf{X}^v[\mathcal{K}^v, :]$, retaining K_v tokens per view. These tokens and all nonvisual tokens form the shortened sequence in their original order; this sequence undergoes the same forward computation as in Eq. 2. A compatible selector provides per-view budgets and candidate indices, and accepts refined indices at its pruning point. DepthPruner introduces quasilinear computational overhead; see Appendix C for a detailed analysis.

5 Experiments

We evaluate DepthPruner by addressing three questions:

1. Does DepthPruner improve task success at matched visual token budgets?
2. Do these gains extend across different base pruning methods?
3. How much latency does DepthPruner add to policy inference?

Section 5.2 compares LIBERO success rates for different base selectors B and depth-guided variants $B{+}D$ to answer questions 1 and 2. Section 5.3 measures the additional inference time introduced by DepthPruner to answer question 3.

5.1 Experimental Setup

Model and benchmark. We evaluate OpenVLA-OFT (Kim et al., 2025a) models on the Spatial, Object, Goal, and Long suites of LIBERO (Liu et al., 2023), each containing ten tasks. The model receives two 224×224 RGB views, a language instruction, and proprioceptive information, and predicts an eight-step action chunk. DepthPruner uses aligned simulator depth to refine token selection while keeping the model weights and action-generation mechanism fixed. We run the experiments on NVIDIA H200 GPUs using BF16 inference and scaled dot-product attention.

Table 1: Success rate (%, ↑) on LIBERO. OFT is the unpruned OpenVLA-OFT reference. ρ is visual retention (%); for ADP it specifies the budget on pruning queries. +D denotes the baseline with DepthPruner. Avg. is the four-suite mean. Average success by retention ratio is computed over the three pruning methods and four suites. Δ is the +D average minus its baseline average, in percentage points.

Method	ρ (%)	Spatial	Object	Goal	Long	Avg.	Δ
OFT (unpruned)	100	95.8	98.8	97.0	94.8	96.60	–
FastV	50	95.0	97.4	97.2	91.6	95.30	–
+ DepthPruner	50	95.4	96.8	98.0	94.0	**96.05**	+0.75
FastV	25	97.2	77.4	96.4	91.0	90.50	–
+ DepthPruner	25	97.8	86.4	95.4	92.4	**93.00**	+2.50
FastV	12.5	96.2	64.8	89.8	85.6	84.10	–
+ DepthPruner	12.5	95.6	69.0	91.6	86.8	**85.75**	+1.65
SparseVLM	50	94.8	89.6	96.0	87.6	92.00	–
+ DepthPruner	50	95.4	91.4	95.0	92.2	**93.50**	+1.50
SparseVLM	25	92.8	85.4	94.4	84.8	89.35	–
+ DepthPruner	25	92.4	86.4	95.4	89.8	**91.00**	+1.65
SparseVLM	12.5	91.2	60.2	93.2	40.6	71.30	–
+ DepthPruner	12.5	91.4	60.8	91.6	50.4	**73.55**	+2.25
ADP	50	92.4	97.2	95.0	91.8	94.10	–
+ DepthPruner	50	92.6	97.2	97.2	91.4	**94.60**	+0.50
ADP	25	90.4	95.8	97.6	83.0	91.70	–
+ DepthPruner	25	90.8	97.6	96.4	85.0	**92.45**	+0.75
ADP	12.5	91.4	93.0	96.6	67.8	87.20	–
+ DepthPruner	12.5	92.2	96.2	94.6	78.4	**90.35**	+3.15

Average success by retention ratio

ρ (%)	Baseline avg.	+D avg.	Δ
50	93.80	**94.72**	+0.92
25	90.52	**92.15**	+1.63
12.5	80.87	**83.22**	+2.35

Baselines. We choose FastV (Chen et al., 2025), SparseVLM (Zhang et al., 2025), and ADP (Pei et al., 2026) as our base pruning methods. FastV and SparseVLM are representative attention-based pruning methods developed for VLMs and previously applied to VLAs (Xu et al., 2025), while ADP selects tokens using text-to-vision attention and determines when to prune based on recent end-effector trajectories. For SparseVLM, we adapt its text-rater scoring to a single pruning layer and disable token recycling. For ADP, we retain its dynamic gating and apply depth refinement only when pruning is enabled. Both variants in each $B/B{+}D$ pair use the same base token selection mechanism and pruning location.

Evaluation and metrics. We evaluate 50 episodes per task (500 per suite and method), reporting suite success rates and their equally weighted mean. We focus on the success-rate change Δ, defined as the success rate with DepthPruner minus that of the corresponding baseline. Each baseline and its depth-refined variant use the same final token budget when pruning is enabled.

Parameter settings. The choices of extra candidate budget and fusion weights depend on the dataset, VLA model, and base selector. In our LIBERO experiments with OpenVLA-OFT, the total extra candidate budget for both views is fixed at 16 tokens for FastV/SparseVLM and 17 for ADP for all three retention ratios. For success-rate evaluation, the fusion weights for mean depth, residual variance, and neighborhood contrast are $(0.35, 0.35, 0.30)$ for all three baselines. These settings are fixed across the four suites.

5.2 Task Success Rate

Improvement across selectors. DepthPruner improves four-suite average success in all nine method–budget pairs. These results show that depth guidance can improve task performance across distinct base selection mechanisms while keeping the final token budget unchanged. Because the three selectors score visual tokens with different rules, these consistent gains suggest that geometrically redundant tokens among the retained set are not specific to a single attention-based selector. Appendix F examines the contribution of each geometric cue.

Gains by suite. The gains are larger on suites where the base selectors lose more success under pruning. On Spatial and Goal, the baselines remain close to the unpruned reference, and DepthPruner preserves this success, changing it by at most 2.2 pp. On Object and Long, the baselines fall by up to 38.6 and 54.2 pp below the reference, and DepthPruner recovers part of this loss, with average gains of 2.3 and 4.1 pp across the nine method–budget pairs and a maximum gain of 10.6 pp on Long. This pattern is consistent with how DepthPruner selects tokens: it replaces some of the baseline's retained tokens with candidates ranked just below the baseline's cutoff. If the baseline already retains the task-relevant regions, these replacements change little. If its retained set includes geometrically redundant regions such as flat tabletop areas, depth guidance reassigns these slots to object candidates, as shown in Section 5.4. Object and Long require locating a specified item among distractors or handling several objects across successive subgoals, which may make them more sensitive to redundant tokens displacing object tokens.

Benefits under tighter budgets. The across-method averages in Table 1 give equal weight to the three base methods and four suites, yielding success-rate gains of 0.92, 1.63, and 2.35 pp at 50%, 25%, and 12.5% retention, respectively. As the token budget shrinks, each redundant token occupies a larger fraction of the available slots, leaving less room for task-relevant tokens. The larger average gains at lower retention ratios suggest that filtering such redundancy becomes more important under tighter budgets. At 12.5% retention, Long success improves by 9.8 pp for SparseVLM and 10.6 pp for ADP, illustrating the value of depth guidance when few visual tokens are retained.

5.3 Inference Overhead

We measure the latency of complete calls from host observations to CPU-executable eight-step action chunks on an NVIDIA H200 NVL, excluding rendering and observation acquisition. Both variants replay the same four trajectories (10–32 action chunks each) ten times, yielding 760 timed calls per configuration. We report mean latency per chunk by first averaging within each suite and then taking an equally weighted average across suites. Appendix D provides the detailed timing protocol.

Table 2: Mean latency per eight-step action chunk on H200, averaged over four suites. B denotes the baseline; B+D adds DepthPruner. Δ is the percentage change in latency relative to B. ρ is visual retention (%).

Method	ρ (%)	B (ms)	B+D (ms)	Δ (%)
FastV	50	61.107	61.604	+0.81
FastV	25	59.530	59.762	+0.39
FastV	12.5	57.448	56.684	-1.33
SparseVLM	50	64.328	65.605	+1.99
SparseVLM	25	58.206	58.767	+0.96
SparseVLM	12.5	61.243	60.145	-1.79
ADP	50	60.653	60.999	+0.57
ADP	25	61.865	63.128	+2.04
ADP	12.5	59.718	60.422	+1.18

Table 2 shows mean latency changes from -1.10 to 1.28 ms per action chunk, or only -1.79% to 2.04% relative to the paired base selector. Because depth refinement keeps the final token count

unchanged and only adds computation, the two negative changes reflect run-to-run timing variation rather than acceleration. These measurements show that depth refinement adds little latency to the base pruning methods.

5.4 Qualitative Visualization

Figure 3 shows how DepthPruner changes FastV's token selection on one LIBERO-Object observation. FastV retains several tokens on the smooth tabletop in the lower-right corner. At the same 12.5% retention ratio, DepthPruner drops these tokens and assigns the freed slots to candidates on and around the target object. Appendix E visualizes the three geometric cues for this observation.

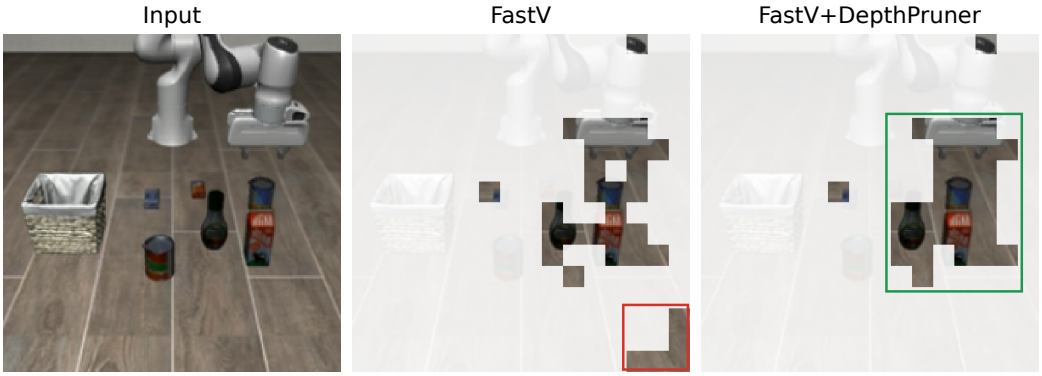

Figure 3: FastV and FastV+DepthPruner on the same main-camera observation from LIBERO-Object, each retaining 32 of 256 tokens. Pruned patches are faded. The red box marks the lower-right tabletop tokens retained by FastV, while the green box marks the object region receiving the reallocated budget with DepthPruner.

6 Conclusion

We presented DepthPruner, a training-free, pluggable method that uses depth geometry to refine attention-based visual token pruning in VLA models. Within the base pruner's token budget, DepthPruner uses local depth geometry to lower the priority of potentially task-irrelevant tokens on smooth or distant surfaces and reallocates the freed slots to other candidates. Experiments with OpenVLA-OFT on LIBERO show improved average task success across different pruning methods with little inference overhead. These findings highlight depth information as a lightweight complement to attention for token selection.

Reproducibility Statement

The experimental setup is described in Section 5.1. The complete selection procedure is given in Appendix B, the selector adaptations in Appendix A, and the latency measurement protocol in Appendix D. The source code will be released publicly.

AI Use Statement

AI tools were used to assist with drafting the paper outline, polishing the text, preparing figures, implementing code, and finding related work. The authors take full responsibility for the final content of this paper.

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 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9392–9401,

June 2025. doi: 10.1109/CVPR52734.2025.00877. URL https://openaccess.thecvf.com/content/CVPR2025/html/Alvar_DivPrune_Diversity-based_Visual_Token_Pruning_for_Large_Multimodal_Models_CVPR_2025_paper.html.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert 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, Laura Smith, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A Vision-Language-Action Flow Model for General Robot Control. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.010. URL https://www.roboticsproceedings.org/rss21/p010.html.

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 *Computer Vision – ECCV 2024*, volume 15139 of *Lecture Notes in Computer Science*, pp. 19–35. Springer Nature Switzerland, 2025. doi: 10.1007/978-3-031-73004-7_2. URL https://link.springer.com/chapter/10.1007/978-3-031-73004-7_2.

Jintao Cheng, Weibin Li, Haozhe Wang, Gang Wang, Yipu Zhang, Xiaoyu Tang, Jin Wu, Xieyuanli Chen, Yunhui Liu, and Wei Zhang. Training-Free Interaction-Aligned Visual Token Pruning for Efficient Embodied Manipulation. *arXiv preprint arXiv:2603.22991*, 2026. doi: 10.48550/arXiv.2603.22991. URL https://arxiv.org/abs/2603.22991.

Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In *International Conference on Learning Representations*, volume 2024, pp. 2632–2652, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/0b408293619f725fd30162af057e531a-Abstract-Conference.html.

Yixu Feng, Zinan Zhao, Yanxiang Ma, Chenghao Xia, Chengbin Du, Yunke Wang, and Chang Xu. See What Matters: Differentiable Grid Sample Pruning for Generalizable Vision-Language-Action Model. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://icml.cc/virtual/2026/poster/65010.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-Tuning Vision-Language-Action Models: Optimizing Speed and Success. In *Proceedings of Robotics: Science and Systems*, 2025a. doi: 10.15607/RSS.2025.XXI.017. URL https://www.roboticsproceedings.org/rss21/p017.html.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An Open-Source Vision-Language-Action Model. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 2025b. URL https://proceedings.mlr.press/v270/kim25c.html.

Chenyang Li, Jieyuan Liu, Bin Li, Bo Gao, Yilin Yuan, Yangfan He, Yuchen Li, and Jingqun Tang. DTP: A Simple yet Effective Distracting Token Pruning Framework for Vision-Language Action Models. arXiv:2601.16065, 2026a. URL https://arxiv.org/abs/2601.16065.

Shiyu Li, Zi-Yuan Hu, Shijia Huang, Yanyang Li, Yiwu Zhong, and Liwei Wang. SinkPruner: Sink-Free Visual Token Pruning for Multimodal Large Language Models. *arXiv preprint arXiv:2609.01004*, 2026b. doi: 10.48550/arXiv.2609.01004. URL https://arxiv.org/abs/2609.01004.

Ye Li, Yuan Meng, Zewen Sun, Kangye Ji, Chen Tang, Jiajun Fan, Xinzhu Ma, Shu-Tao Xia, Zhi Wang, and Wenwu Zhu. SP-VLA: A Joint Model Scheduling and Token Pruning Approach for VLA Model Acceleration. In *International Conference on Learning Representations*, volume 2026, pp. 38155–38184, 2026c. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/4072543747a14bbed76284cf2c04b9e9-Abstract-Conference.html.

Yuquan Li, Lianjie Ma, Han Ding, and Lijun Zhu. DepthCache: Depth-Guided Training-Free Visual Token Merging for Vision-Language-Action Model Inference. *arXiv preprint arXiv:2603.10469*, 2026d. doi: 10.48550/arXiv.2603.10469. URL https://arxiv.org/abs/2603.10469.

Tao Lin, Yilei Zhong, Yuxin Du, Jingjing Zhang, Jiting Liu, Yinxinyu Chen, Encheng Gu, Ziyan Liu, Hongyi Cai, Yanwen Zou, Lixing Zou, Zhaoye Zhou, Gen Li, and Bo Zhao. Evo-1: Lightweight Vision-Language-Action Model with Preserved Semantic Alignment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13397–13406, June 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Lin_Evo-1_Lightweight_Vision-Language-Action_Model_with_Preserved_Semantic_Alignment_CVPR_2026_paper.html.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking Knowledge Transfer for Lifelong Robot Learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 44776–44791. Curran Associates, Inc., 2023. doi: 10.52202/075280-1939. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/8c3c666820ea055a77726d66fc7d447f-Abstract-Datasets_and_Benchmarks.html.

Yicheng Liu, Shiduo Zhang, Zibin Dong, Baijun Ye, Tianyuan Yuan, Xiaopeng Yu, Linqi Yin, Chenhao Lu, Junhao Shi, Luca Jiang-Tao Yu, Liangtao Zheng, Jingjing Gong, Tao Jiang, Xipeng Qiu, and Hang Zhao. FASTer: Toward Powerful and Efficient Autoregressive Vision–Language–Action Models with Learnable Action Tokenizer and Block-wise Decoding. In *International Conference on Learning Representations*, volume 2026, pp. 97187–97209, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/9dc5accb1e4f4a9798eae145f2e4869b-Abstract-Conference.html.

Ziyan Liu, Yeqiu Chen, Yiming Zhang, Hongyi Cai, Tao Lin, Runquan Gui, Shuo Yang, Zheng Liu, and Bo Zhao. Bridging the Semantic-Action Gap in Visual Token Pruning for Efficient VLA Inference. *arXiv preprint arXiv:2511.16449*, 2025. doi: 10.48550/arXiv.2511.16449. URL https://arxiv.org/abs/2511.16449.

Shilin Ma, Chubin Zhang, Changyuan Wang, Yuji Wang, Yue Wu, Zixuan Wang, Jingqi Tian, Zheng Zhu, and Yansong Tang. SAFE-Pruner: Semantic Attention–Guided Future-Aware Token Pruning for Efficient Vision-Language-Action Manipulation. In *Computer Vision – ECCV 2026*, volume 17049 of *Lecture Notes in Computer Science*, pp. 163–180. Springer Nature Switzerland, 2026. doi: 10.1007/978-3-032-37362-5_9. URL https://link.springer.com/chapter/10.1007/978-3-032-37362-5_9.

Xiaohuan Pei, Yuxing Chen, Siyu Xu, Yunke Wang, Yuheng Shi, and Chang Xu. Action-aware Dynamic Pruning for Efficient Vision-Language-Action Manipulation. In *International Conference on Learning Representations*, volume 2026, pp. 10832–10851, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/1223ca2b14513106ad55db6a537337f9-Abstract-Conference.html.

Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Jiayuan Gu, Zhigang Wang, Yan Ding, Bin Zhao, Dong Wang, and Xuelong Li. SpatialVLA: Exploring Spatial Representations for Visual-Language-Action Models. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.011. URL https://www.roboticsproceedings.org/rss21/p011.html.

Siyu Xu, Yunke Wang, Chenghao Xia, Dihao Zhu, Tao Huang, and Chang Xu. VLA-Cache: Efficient Vision-Language-Action Manipulation via Adaptive Token Caching. In *Advances in Neural Information Processing Systems*, volume 38, pp. 164448–164473. Curran Associates, Inc., 2025. doi: 10.52202/085713-5484. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/f062da1973ac9ac61fc6d44dd7fa309f-Abstract-Conference.html.

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. In *Advances in Neural Information Processing Systems*,

volume 38, pp. 40891–40914. Curran Associates, Inc., 2025. doi: 10.52202/085713-1365. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/3a2ef31a1e45908901adc0ca853a8faf-Abstract-Conference.html.

Tianyuan Yuan, Yicheng Liu, Chenhao Lu, Zhuoguang Chen, Tao Jiang, and Hang Zhao. DepthVLA: Enhancing Vision-Language-Action Models with Depth-Aware Spatial Reasoning. In *2026 IEEE International Conference on Robotics and Automation (ICRA)*, 2026. URL https://ras.papercept.net/conferences/conferences/ICRA26/program/ICRA26_ContentListWeb_3.html#tui2i_104.

Rongyu Zhang, Menghang Dong, Yuan Zhang, Liang Heng, Xiaowei Chi, Gaole Dai, Li Du, Dan Wang, Yuan Du, and Shanghang Zhang. MoLe-VLA: Dynamic Layer-skipping Vision Language Action Model via Mixture-of-Layers for Efficient Robot Manipulation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 18764–18772. Association for the Advancement of Artificial Intelligence, 2026. doi: 10.1609/aaai.v40i22.38945. URL https://ojs.aaai.org/index.php/AAAI/article/view/38945.

Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis A Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, and Shanghang Zhang. SparseVLM: Visual Token Sparsification for Efficient Vision-Language Model Inference. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 74840–74857. PMLR, 2025. URL https://proceedings.mlr.press/v267/zhang25s.html.

A Selector Adaptations

FastV and SparseVLM. Our OpenVLA-OFT adaptation of FastV follows the implementation of VLA-Pruner (Liu et al., 2025). It aggregates visual attention over heads and pre-action prefill query positions, and prunes after the fourth decoder block (zero-based block index 3). The SparseVLM adaptation also follows VLA-Pruner for text-rater selection and visual-attention scoring, and applies equal per-view top-K selection without token recycling. It gathers tokens before zero-based block 3, using attention from the preceding block. Thus, the two adapters have distinct pruning locations, although each baseline/$+D$ pair uses the same pruning location and scoring path.

ADP. Our ADP baseline follows the official implementation (Pei et al., 2026). ADP uses the first decoder layer's query/key projections to score projected visual embeddings against text before the main decoder pass. It averages scaled query–key products over text tokens and attention heads and selects within each view; the scoring implementation does not apply a softmax. The controller computes net end-effector displacement over each action chunk and compares it with the displacements of the preceding two chunks using the adjacent-extrema rule. The initial state is full-token inference, including the first three queries; at most three consecutive pruned queries are allowed before a full-token query. The depth adapter preserves this controller and refines candidates only when pruning is enabled.

B Algorithm

Here we present the token selection pipeline of DepthPruner.

Algorithm 1 Token selection pipeline of DepthPruner

Require: Base selector $\mathcal{B}$, aligned depths $\{Z^v\}$, view budgets $\{K_v\}$, expansion factor $\alpha \geq 1$

1: **for** each prunable view v **do**
2: $M_v \leftarrow \min(N_v, \lceil \alpha K_v \rceil)$; $\mathcal{C}^v \leftarrow \mathcal{B}^v(M_v)$
3: Compute μ^v, r^v, e^v on the full aligned grid using Sec. 4.2
4: Compute full-view ranks and D^v using Eq. 7
5: Select the K_v highest-priority candidates from $\mathcal{C}^v$
6: Restore original sequence order and pass indices to the base adapter
7: **end for**

C Computational Cost

DepthPruner introduces only quasilinear computational overhead. Let P be the total number of pixels in the aligned depth map for one view, which is partitioned into N patches of m pixels each ($P = Nm$). Local affine fitting uses a shared fitting matrix; computing the fits, mean depth, and residual variance costs $O(P)$. Sorting depths within patches costs $O(P \log m)$, and four-neighbor comparisons of the sorted depths cost $O(P)$. Ranking and candidate selection cost $O(N \log N)$. The added geometric refinement therefore costs $O(P \log m + N \log N)$. These depth operations are lightweight compared with the attention and feed-forward computations across the VLA backbone. Section 5.3 reports the measured increment in complete policy-call latency.

D Policy Inference Timing

Replay protocol. For each LIBERO suite, we replay a trajectory collected without token pruning from task 0 and initial state 0, without filtering by task success. The Spatial, Object, Goal, and Long trajectories contain inputs for 10, 17, 17, and 32 policy calls, respectively. All variants receive the same observations, language instructions, and recorded robot states. Before each timed replay, we run 20 warm-up calls for each variant and then reset the selector state. We repeat each trajectory ten times and alternate the execution order of the baseline and its variant with DepthPruner between paired rounds.

Measurement boundary. We measure the wall-clock latency of a complete policy call, from host observations to an eight-step action chunk returned to the CPU. The timed region includes observation preprocessing, image and depth transfers, shared-memory handoff, required waiting, token selection, model execution, and CPU action conversion. CUDA is synchronized immediately before and after each call. Disk access, simulator stepping, rendering, and observation acquisition are excluded.

E VISUALIZING GEOMETRIC CUES AND TOKEN SELECTION

Figure 4 explains the selection change in Fig. 3 using the same LIBERO-Object observation. We compute the three geometric cues from its recorded, aligned depth on the 16×16 patch grid and combine their full-view ranks with weights $(0.35, 0.35, 0.30)$, following Eq. 7. FastV supplies 40 candidates, from which DepthPruner retains 32 tokens at 12.5% retention. The final mask is based on the recorded selection; offline recomputation yields the same retained indices.

The three geometric cues provide complementary guidance for token selection. Mean depth helps distinguish distant walls and other backgrounds, but does little to lower the priority of nearby tabletops. Residual variance assigns low scores to smooth tabletops and highlights deviations from a smooth local trend, although some flat object surfaces may also have low residual variance. Neighborhood contrast further compares depth distributions across adjacent patches: a locally flat patch can still receive a high score when depth differences from its neighbors exceed what the local slope can explain. This helps highlight object boundaries and nearby geometric changes while lowering the priority of large regions with similar geometry. Together, the three cues combine distance, within-patch structure, and neighborhood differences to refine candidate token selection.

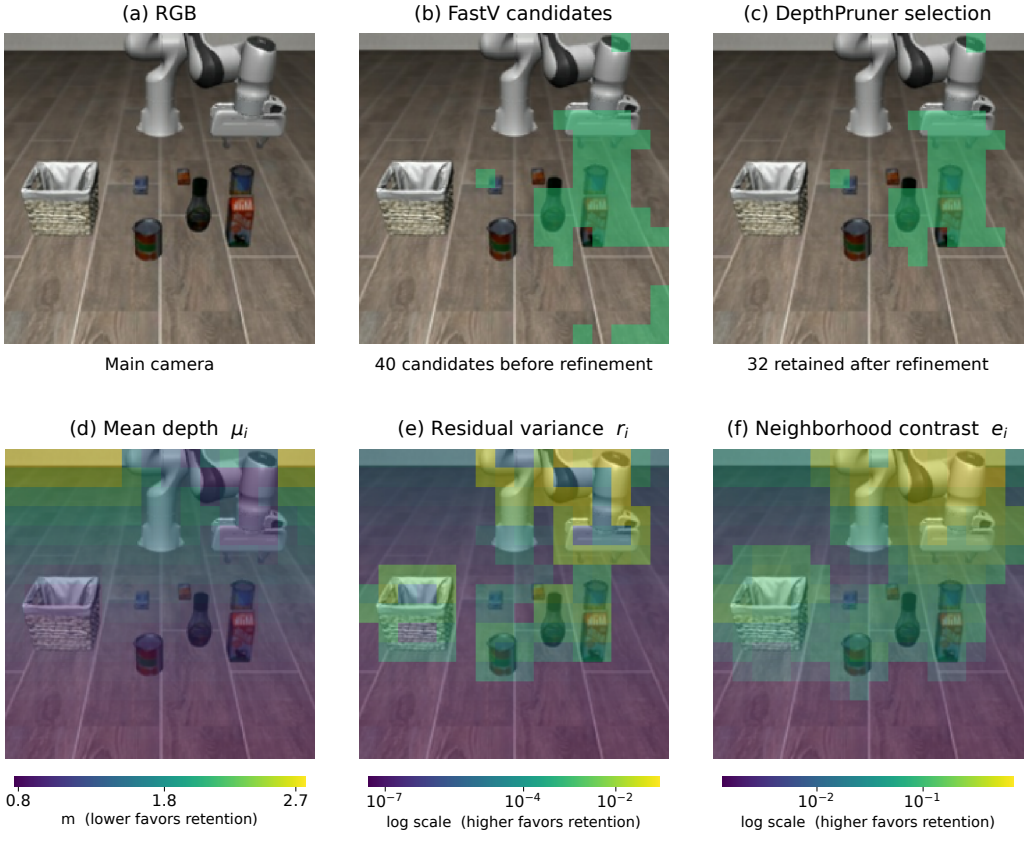

Figure 4: Visualization of token selection and geometric cues. Top: RGB, FastV candidates, and DepthPruner selections, with green overlays marking the selected patches. Bottom: mean depth, residual variance, and neighborhood contrast overlaid on the same image.

F Contribution of Geometric Cues

We isolate each cue's contribution by removing one fusion term from FastV+D at 12.5% retention with $\alpha = 1.25$. The variants omit mean depth, residual variance, or neighborhood contrast while retaining the other two terms and the same base selector, candidate-expansion rule, and final budget. Each variant is evaluated over 500 episodes per suite. Table 3 reports success rates and changes relative to the full three-cue score.

Table 3: Removing individual geometry terms from FastV + DepthPruner at $\rho = 12.5\%$ and $\alpha = 1.25$. Success rates (%, ↑) use 500 episodes per suite and variant. Avg. weights the four suites equally; Δ is the variant average minus the full-method average, in percentage points.

Variant	Spatial	Object	Goal	Long	Avg.	Δ
Full DepthPruner	95.6	69.0	91.6	86.8	**85.75**	0.00
w/o mean depth	93.4	70.0	92.2	86.4	85.50	-0.25
w/o residual variance	91.2	68.0	85.4	86.2	82.70	-3.05
w/o neighborhood contrast	90.8	65.2	87.6	87.8	82.85	-2.90

The full score achieves the highest mean success rate across the four suites, 85.75%; removing residual variance or neighborhood contrast reduces the mean by 3.05 and 2.90 percentage points, respectively, indicating that both local geometric cues help improve average task success in this setting. In comparison, removing mean depth reduces the mean success rate by only 0.25 percentage points. A plausible explanation is that distant backgrounds in LIBERO scenes are largely planar and locally continuous, so residual variance and neighborhood contrast can also lower their retention priority, leaving limited additional information for mean depth to provide. However, distant backgrounds in other benchmarks or real-world scenes may have complex geometric structures, and relying only on residual variance and neighborhood contrast may still assign these regions high retention priority. In such cases, mean depth can lower the retention priority of these background tokens using distance information, so we consider it a useful cue for token selection.

G Limitations and Future Work

DepthPruner uses depth geometry to refine attention-based visual token pruning in VLA models. Although many robotic platforms are equipped with depth sensors, missing values and outliers in real sensor measurements can distort the geometric scores used by our method, making deployment on physical robots challenging.

One direction for future work is to reduce the impact of missing values and outliers on token selection by developing geometric measures that are more robust to them. Another is to explore depth estimation models as an alternative source of depth guidance, while weighing their additional computational cost against the savings from pruning.