BEYOND RELEVANCE: RETHINKING WHAT ROBOTS NEED TO SEE ACROSS TASKS AND STATES

Anonymous authors
Paper under double-blind review

ABSTRACT

Visual relevance does not determine observation precision or visual capacity. Motivated by action sensitivity to state errors, we study action-conditioned measurement sufficiency beyond the Base policy's existing context. Controlled ACT analyses show task- and state-dependent location–precision allocation under a fixed budget, no uniformly monotonic benefit from increasing the precision ceiling or residual capacity, and different preferred configurations across tasks. We propose Action–Visual Precision Matching: action context jointly ranks location–precision candidates, Global Top-K selects them under a shared budget, and only selected high-precision observations are encoded. Their feature residuals relative to base-aligned references refine native action generation. The main ACT configuration improves mean success rate by 2.65 percentage points over its matched Base and outperforms same-budget random allocation. A post-hoc task-wise configuration combination reaches 84.20%, exceeding Base and the best uniform configuration by 4.03 and 1.38 percentage points. Diffusion Policy and π_0 further test integration across action generation structures. These results support matching visual resources to current action requirements rather than uniformly maximizing them.

1 INTRODUCTION

ACT jointly predicts future action chunks (Zhao et al., 2023), Diffusion Policy generates actions through conditional denoising (Chi et al., 2023), and π_0 combines pretrained vision–language representations with flow matching (Black et al., 2024). These policies typically use preset image resolutions, spatial sampling densities, and visual capacities, although a single specification may not suit different tasks and execution states.

Prior work compresses redundant visual tokens (Rao et al., 2021; Pei et al., 2025), preserves dense spatial features (Zhou et al., 2026), or acquires local observations as needed (Cui et al., 2026; Liu et al., 2026), addressing which information can be reduced, which details to retain, and when and where to observe further. These methods improve access to task-relevant information, but regional relevance does not determine the required observation precision.

Even within the same spatial region, actions may have different sensitivities to state errors across tasks and execution states. Locally, this relationship can be written as (Goodfellow et al., 2016)

$$\delta a \approx J_{\pi,x}\, \delta x$$

where $J_{\pi,x}$ describes the local sensitivity of policy outputs to state changes. When a state perturbation δx causes a substantial change in action, the policy needs sufficiently fine visual measurements to distinguish that perturbation. When the same observation error barely changes the current action, further increasing precision at that location may be unnecessary. We do not explicitly estimate or optimize this Jacobian; we use it only to explain why visual measurement precision should match the effect of state errors on actions.

Active perception links information acquisition to the task and current scene interpretation (Bajcsy, 1988). If an action is sensitive to only a few local state differences, limited fine-grained observations may suffice; if several state factors affect control simultaneously, more additional visual capacity

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.

may be needed. The base policy's existing visual, task, and action context may already supply some of this information. Visual resource allocation therefore depends on which state differences the current action still requires the policy to distinguish, rather than solely on whether a region is relevant.

We define visual precision as spatial sampling density at a fixed local input size: higher precision corresponds to a smaller spatial support in the original scene and a denser grid of discrete observation locations.

We therefore ask: given the existing Base policy context, where does the current action require additional visual observations, at what precision, and with how much additional visual capacity?

We first address the fixed-budget form of this problem: given a total residual budget K_{res}, we learn action-conditioned location–precision allocation. The precision ceiling and residual capacity serve as controlled resource limits for examining how different tasks respond to visual configurations.

Using ACT as a controlled analysis platform, we observe that precision composition and spatial allocation vary across tasks and execution states under a fixed budget; neither a higher precision ceiling nor greater residual capacity provides a uniformly monotonic advantage across tasks; and different tasks favor different precision–capacity configurations. These findings are consistent with the action-sensitivity perspective: when existing visual measurements already distinguish the state differences to which the current action is sensitive, additional precision or capacity need not provide new information useful for control.

Action–Visual Precision Matching uses Base visual and action context to jointly rank location–precision candidates, then selects them under a shared budget with one Global Top-K operation. Only selected high-precision observations are acquired and encoded. Their feature differences from base-aligned references form a residual memory read by the native action generation pathway to refine actions.

Our contributions are as follows:

- Problem formulation. We study visual precision and capacity requirements beyond the base policy's existing context through action sensitivity and measurement sufficiency.
- Controlled analysis and method. We analyze closed-loop responses to the precision ceiling, residual capacity, and conditional allocation, and propose joint location–precision selection with base-aligned residual refinement under a fixed budget.
- Closed-loop evaluation. We examine integration into ACT, Diffusion Policy, and π_0, test the role of learned routing against random allocation under the same budget, and characterize the scope for resource matching through task-wise configuration combinations.

2 Related Work

2.1 Adaptive Visual Representations for Robot Policies

Adaptive visual representations reduce existing processing costs by pruning, aggregating, or merging tokens, as in DynamicViT, TokenLearner, and Token Merging (Rao et al., 2021; Ryoo et al., 2021; Bolya et al., 2023). ADP switches between full and pruned visual inputs using recent action trajectories (Pei et al., 2025); the recent StackTok preprint shares a token budget across encoded image crops (Wang et al., 2026).

Patch Policy retains dense pretrained patch features, preserving local spatial information that global aggregation may discard (Zhou et al., 2026). Its focus is access to fine-grained evidence.

Selective visual access also has broad precedents. Recurrent attention selects multi-resolution glimpses, while Spatial Transformer Networks learn input-dependent spatial transformations (Mnih et al., 2014; Jaderberg et al., 2015). Attention Sampling and PatchDrop select high-resolution patches from low-resolution inputs (Katharopoulos & Fleuret, 2019; Uzkent & Ermon, 2020). Differentiable Patch Selection selects patches before expensive encoding, including Top-K selection among multiple scales and aspect ratios in fine-grained recognition (Cordonnier et al., 2021). TNet traverses scale space, while MS-GFNet adapts local patch size and stopping (Papadopoulos et al.,

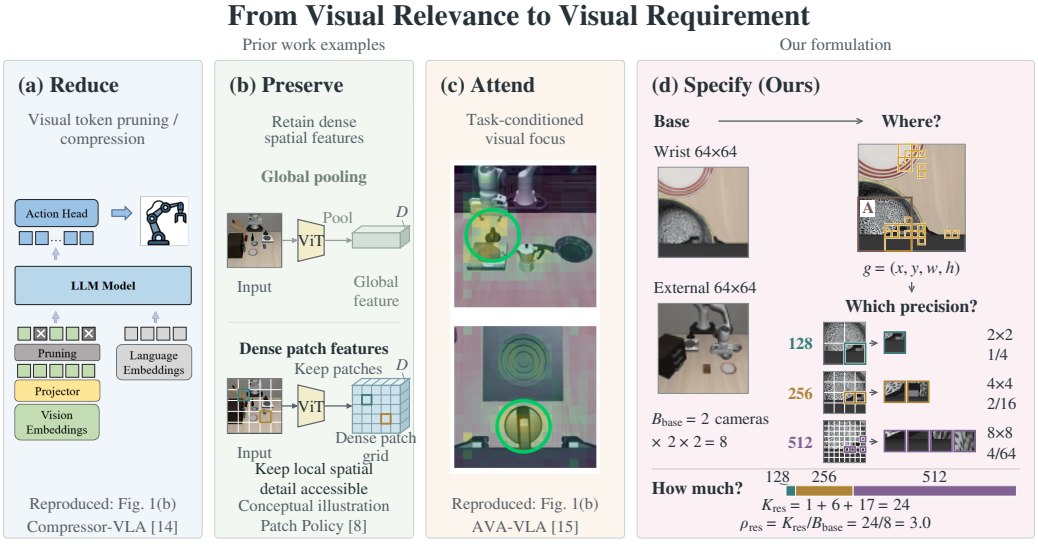

Figure 1: From visual relevance to visual precision requirements. (a) A pruning pipeline illustrated in Compressor-VLA (Gao et al., 2025). (b) Global pooling versus dense patch features, based on Patch Policy (Zhou et al., 2026). (c) Dual-view attention in AVA-VLA (Xiao et al., 2025). (d) Location–precision selection under a fixed capacity. The ACT example uses dual-camera Base64 with 8 Base tokens and selects 24 patches under a 3B residual budget.

2021; Huang et al., 2023). In robot policies, AtVLA combines attention, uncertainty, and local re-encoding; ActiveVLA combines three-dimensional localization, rendered-view selection, and zooming (Cui et al., 2026; Liu et al., 2026). These precedents connect spatial selection, local scale, and selective high-resolution processing.

2.2 Multi-Scale Vision and Visual Precision Allocation

Multi-scale methods differ in selection granularity and budgets. FPN constructs feature pyramids, MSViT varies regional token scales, and DRNet selects image resolution (Lin et al., 2017; Havtorn et al., 2023; Zhu et al., 2021a). CF-ViT refines selected regions while reusing coarse features; Mixed-Resolution Tokenization subdivides patches under a token limit (Chen et al., 2023; Ronen et al., 2023). Deformable DETR weights samples across feature levels, Sparse DETR selects tokens for updating, and Focus-DETR already pools cross-level scores for Top-K selection (Zhu et al., 2021b; Roh et al., 2022; Zheng et al., 2023). QuadTree Attention and BiFormer select coarse regions before finer interactions, whereas MST compares scales after per-scale Top-K selection (Tang et al., 2022; Zhu et al., 2023; Xu et al., 2024). In control, Multi-Resolution Sensing combines spatial and temporal observations, while Dynamic-Resolution Model Learning adjusts particle representation granularity (Saxena et al., 2023; Wang et al., 2023). The detection and attention methods control computation over encoded features; their budgets differ from our additional observation capacity.

Residual representations have distinct precedents. LapSRN predicts image residuals, and Side-Tuning learns a complementary branch for a frozen model (Lai et al., 2017; Zhang et al., 2020). Mixed-Resolution Tokenization scores subdivision using original-versus-blurred feature error; video compression can subtract temporal reference features (Ronen et al., 2023; Shin et al., 2025). The recent Visual Token Coding preprint uses inter-frame residuals for selection and budgeting (Fang et al., 2026). Here, overlapping location–precision candidates compete under action context for an additional observation budget beyond Base. Feature differences between selected patches and current base-aligned references refine native action generation. Precision levels differ in spatial support and sampling density; Section 4 defines the candidates and capacity ratios.

2.3 Action Generation Context for Visual Allocation

ACT predicts action chunks, Diffusion Policy generates actions through conditional denoising, and π_0 uses flow matching (Zhao et al., 2023; Chi et al., 2023; Ho et al., 2020; Black et al., 2024; Lipman et al., 2023). In our integrations, action hidden representations, denoising states, or flow generation states provide allocation context. Candidate-scoring and residual-reading interfaces preserve each policy's native generation pathway. RT-2 and OpenVLA also adapt pretrained vision–language representations for action generation (Brohan et al., 2023; Kim et al., 2024).

Mixture of Horizons combines prediction horizons to balance local control and longer-term anticipation (Jing et al., 2026). We draw on multi-horizon representations to condition visual allocation through cumulative action representations while retaining the native prediction horizon.

3 Mechanistic Analysis of Visual Precision and Capacity Matching

3.1 Conditional Visual Precision Allocation under Fixed Resource Limits

With the FULL512 candidate space, we fix residual capacity and examine how the limited visual budget is allocated across locations and precision levels. Figure 2 shows the 3B configuration, in which precision levels 128, 256, and 512 are all available for selection.

Precision composition differs across tasks. Spatial uses relatively more 512-precision information, whereas LONG favors 256; tasks within Goal also differ. Although 512-precision candidates constitute 76.19% of the FULL512 pool, they account for only 16.02% of selected tokens. Thus, the resulting composition is not directly determined by candidate counts but emerges from selection under a limited budget.

Precision allocation also varies during execution of a single task. In one successful Goal T03 trajectory, the total budget remains fixed at 24 tokens, yet the number of selected 512 tokens falls from 12 at step 50 to 0 at step 110, alongside changes in spatial locations. Because the task, model, and resource limits remain unchanged, this variation reflects adaptation of the visual configuration to execution state.

Under the same budget, the policy changes precision composition and spatial locations across tasks and states. This supports state-conditioned precision allocation, but the selected composition should not be treated as an intrinsic precision label for a task.

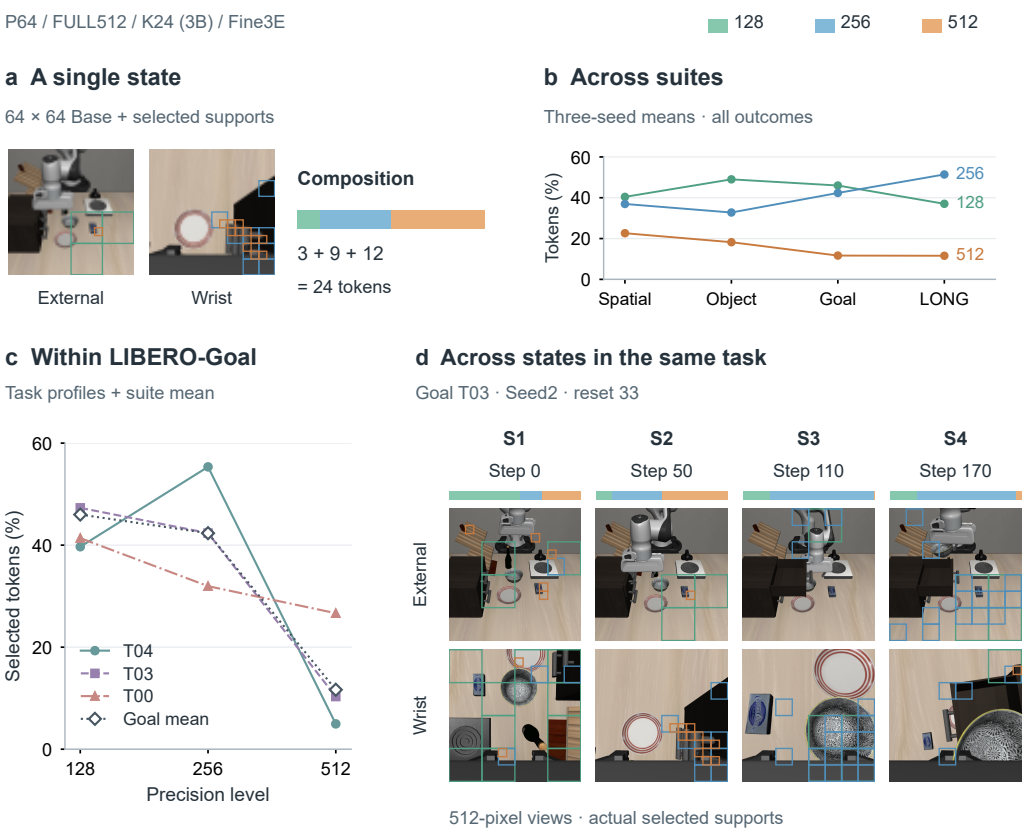

Figure 2: Precision allocation under a fixed budget, using the FULL512 candidate space and a 3B budget. (a) Base inputs from two cameras and selected supports. (b) Precision composition across four suites. (c) Goal T04, T03, T00, and the Goal mean. (d) Four states from the same successful Goal T03 rollout.

3.2 Matching Visual Capacity under Precision Allocation

To test whether conditional allocation still requires matching the overall visual resource budget, we compare two precision ceilings, MAX256 and FULL512, and two residual capacities, 3B and 6B, under the same allocation mechanism, yielding four precision–capacity configurations.

In Table 1, all four configurations improve mean success rate over Base, but the effects of the precision ceiling and capacity are interdependent. Under MAX256, increasing capacity from 3B to 6B raises Macro from 82.45% to 82.75%; under FULL512, the same increase lowers Macro from 82.82% to 82.10%. Making 512 precision available improves overall performance at 3B but reduces it at 6B. Suites also respond differently to these configuration changes.

Table 1: Closed-loop performance of the base policy and different precision–budget configurations.

Configuration	Budget	Spatial	Object	Goal	LONG	Macro
Base64 20E	—	66.27 ± 0.31	92.53 ± 1.86	85.27 ± 3.93	76.60 ± 0.92	80.17 ± 0.29
MAX256	3B	69.80 ± 1.56	95.47 ± 0.23	86.73 ± 2.05	77.80 ± 0.53	82.45 ± 0.72
MAX256	6B	69.13 ± 0.50	94.47 ± 0.81	88.13 ± 2.84	79.27 ± 1.22	82.75 ± 0.93
FULL512	3B	69.80 ± 1.56	94.47 ± 0.99	88.40 ± 2.36	78.60 ± 1.31	82.82 ± 1.09
FULL512	6B	69.47 ± 1.63	94.07 ± 0.12	87.47 ± 3.14	77.40 ± 0.87	82.10 ± 1.25

Note: Results are mean success rates $\pm$ sample standard deviation across three seeds. Macro weights the four suites equally.

The effect of visual capacity depends on the available precision and task conditions. Even with conditional allocation, access to higher precision or greater residual capacity does not necessarily improve control. Both how precision is allocated and how much capacity is allocated must match action requirements.

3.3 Scope for Task-Level Precision–Capacity Matching

For each task, we select the best of the four trained configurations above by mean task success rate, then average tasks equally to form the Task-wise Oracle. This characterizes the scope for matching beyond a uniform configuration.

Figure 3 compares task-wise selection with uniform configurations. Spatial, Object, Goal, and LONG improve by 1.33, 0.33, 0.80, and 1.40 percentage points over their respective best uniform configurations. Macro, the equally weighted mean across the four suites, reaches 84.20%, exceeding Base64's 80.17% by 4.03 percentage points and the best uniform configuration, FULL512/3B at 82.82%, by 1.38 percentage points.

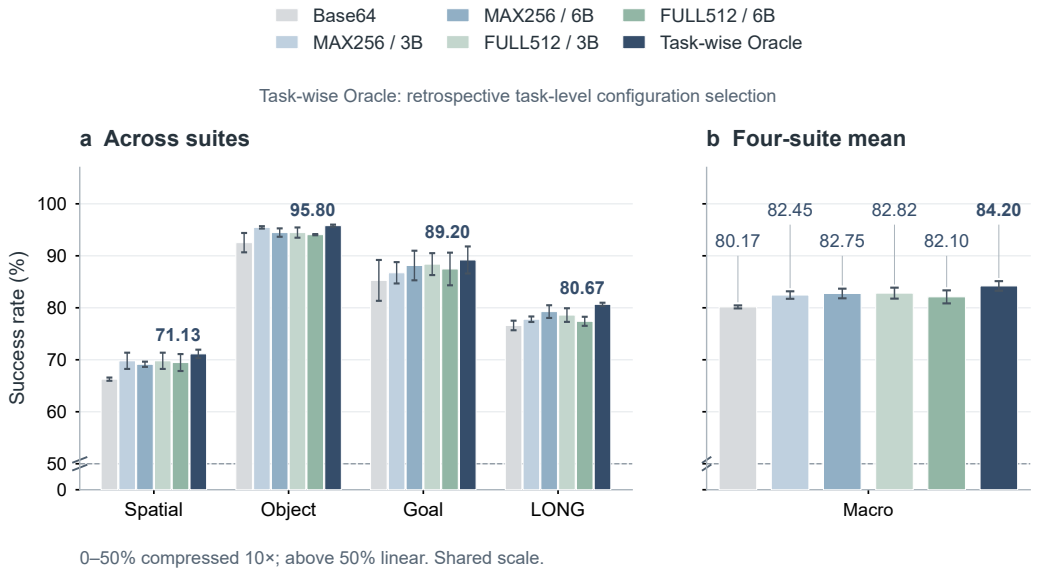

Figure 3: Fixed configurations and the task-wise best-configuration envelope. (a) Base, four uniform configurations, and the Task-wise Oracle for each suite. (b) Equally weighted Macro across the four suites.

Task-level complementarity among the tested configurations reveals room for further matching beyond conditional allocation; this post-hoc result does not constitute a dynamic selector.

4 Action–Visual Precision Matching

Action–Visual Precision Matching jointly selects location–precision candidates under a fixed budget K_{res} and refines action generation with base-aligned visual residuals (Figure 4).

4.1 Cross-Precision Spatial Candidates

ACT uses 64×64 Base images and fixed-size 32×32 Fine patches, with precision determined by spatial support in the source image. At precision levels 128, 256, and 512, each Base parent has 2×2, 4×4, 8×8 children, respectively; higher precision yields smaller support and a denser location grid.

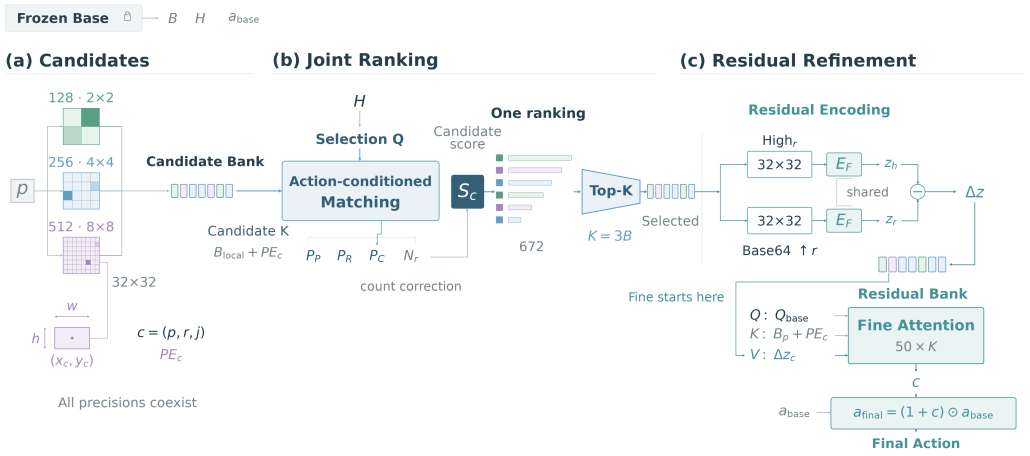

Figure 4: Action–Visual Precision Matching. (a) Each Base region defines multi-precision candidates. (b) Action context H conditions candidate scoring; count-corrected scores feed Global Top-K selection. (c) High and spatially aligned Reference patches share an encoder, and their features are subtracted. Fine Attention uses Q_b (Qbase in the figure) as its query, Base parent features and geometry as keys, and residuals as values. The readout produces coefficients to refine Base actions.

We denote a candidate by $\boxed{c = (p, r, j)}$, where p, r, and j denote the Base parent, precision, and corresponding child. Each candidate has the normalized spatial identity $g_c = (x_c, y_c, w_c, h_c, \text{cam}_c)$, encoded as PE_c and combined with local Base features for scoring.

4.2 Action-Conditioned Joint Ranking under a Residual Budget

Base action representations H condition candidate scoring. Using candidate local features and geometry, we factorize the probability of candidate $c = (p, r, j)$ as

$$\boxed{P(c \mid H) = P_P(p \mid H) P_R(r \mid p, H) P_C(j \mid p, r, H).}$$

The parent, conditional precision, and conditional child factors P_P, P_R, and P_C involve no intermediate hard selection.

Child scoring matches action representations H to candidate information and corrects for differing child counts across precisions using $N_r = \left(\frac{r}{r_B}\right)^2$ to obtain the ranking score

$$\boxed{S(c) = \log P_P + \log P_R + \log P_C + \log N_r.}$$

N_r corrects only the ranking, not the joint probability.

All cameras, parents, precisions, and children enter one global selection:

$$\boxed{\mathcal{S}_t = \text{GlobalTopK}(\{S(c)\}, K_{\text{res}}).}$$

No precision quotas are imposed. Relative to the Base spatial token count N_B, residual capacity is $\boxed{\rho_{\text{res}} = \frac{K_{\text{res}}}{N_B},}$ where K_{res} is fixed within each training configuration and $\sum_r K_r = K_{\text{res}}$. Routing learns only precision composition and spatial allocation, not the total count. Precision composition is $\pi_r = K_r / K_{\text{res}}$, where K_r counts selected tokens at precision r.

Following MoH's multi-horizon representations (Jing et al., 2026), we introduce action context over different future horizons. ACT retains its native 50-step prediction horizon and constructs cumulative representations from $H = [h_1, \ldots, h_{50}]$:

$$D_L = \frac{1}{L} \sum_{k=1}^{L} h_k, \qquad L \in \{10, 20, 30, 40, 50\},$$

Candidate-specific temporal mixing uses D_L to condition allocation scores (Appendix A.4).

4.3 BASE-ALIGNED RESIDUAL VISUAL REFINEMENT

Only after Global Top-K selects candidate c do we read and encode its high-precision patch I_c^{high}. We upsample the same camera's Base image to the corresponding precision and crop I_c^{ref} at matching spatial support. A shared Fine encoder produces their feature difference:

$$\boxed{\delta z_c = E_F(I_c^{\text{high}}) - E_F(I_c^{\text{ref}}).}$$

Fine Attention uses Q_b (Qbase in the figure) to query the memory of selected residuals. Keys encode Base parents and geometry, while values carry residuals:

$$\boxed{Q = Q_b, \qquad K_c = W_K B_{p(c)} + W_{\text{geo}} PE_c, \qquad V_c = W_V \delta z_c.}$$

The Fine attention readout is (Vaswani et al., 2017)

$$M_F = \text{Attention}(Q_b, K_F, V_F).$$

ACT uses the Fine context to produce refinement coefficients:

$$\boxed{a_t^{\text{final}} = (1 + c_t) \odot a_t^{\text{base}}.}$$

Diffusion Policy and π_0 integrate residual memory into their native denoising and flow generation pathways, preserving their output forms.

Training freezes Base and optimizes the added modules with the native action objective. Hard Top-K uses a straight-through surrogate (Bengio et al., 2013); Appendices A.5 and C give implementation and training settings.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

We evaluate Diffusion Policy (task-level visual control), ACT (task-instruction-conditioned) (Haldar et al., 2024; Bharadhwaj et al., 2023), and the pretrained VLA π_0 on LIBERO (Liu et al., 2023) Spatial, Object, Goal, and LONG. ACT uses FULL512/3B for the main analysis. All retain native action generation; DP and π_0 incorporate residuals into denoising and flow latents. Appendices B and C give budgets and training/evaluation settings.

5.2 DOES VISUAL PRECISION MATCHING DEPEND ON ACT?

In Table 2, ACT, Diffusion Policy, and π_0 show Macro changes of +2.65, +0.43, and +0.12 percentage points relative to their matched Base policies. The mechanism thus supports integration into different action generation structures for closed-loop control, with gains that vary by policy.

Table 2: Closed-loop results of Action–Visual Precision Matching across policy architectures.

Policy	Configuration	Seeds	Spatial	Object	Goal	LONG	Macro
ACT	Base64 / 20E	3	66.27 ± 0.31	92.53 ± 1.86	85.27 ± 3.93	76.60 ± 0.92	80.17 ± 0.29
ACT	Ours: 512/3B	3	69.80 ± 1.56	94.47 ± 0.99	88.40 ± 2.36	78.60 ± 1.31	82.82 ± 1.09
ACT	Δ (Ours $-$ Base)	—	+3.53	+1.94	+3.13	+2.00	+2.65
DP	Base20K	3	90.07 ± 0.23	90.47 ± 0.76	81.87 ± 0.92	67.40 ± 1.51	82.45 ± 0.23
DP	Ours: Fine3K	3	90.20 ± 0.92	92.00 ± 1.51	83.07 ± 0.42	66.27 ± 1.01	82.88 ± 0.19
DP	Δ (Ours $-$ Base)	—	+0.13	+1.53	+1.20	-1.13	+0.43
π_0	Base (LIBERO)	3	97.33 ± 0.23	97.87 ± 0.58	93.67 ± 0.99	80.93 ± 1.14	92.45 ± 0.67
π_0	Ours: Fine	3	97.20 ± 0.35	97.60 ± 0.40	93.07 ± 1.86	82.40 ± 1.25	92.57 ± 0.65
π_0	Δ (Ours $-$ Base)	—	-0.13	-0.27	-0.60	+1.47	+0.12

Note: Success rates are means $\pm$ sample standard deviations over three seeds; Macro weights the four suites equally. Each method uses its matched Base. Δ uses the displayed two-decimal values. Appendix C gives training and statistical settings.

5.3 Is Learned Location–Precision Allocation Effective?

At FULL512/3B, we replace ACT's learned routing with random selection of precisions and corresponding locations, holding the candidate space, budget, and refinement pathway fixed. Learned allocation improves all four suites and raises Macro from 81.77% to 82.82% (+1.05 pp; Table 3), showing a benefit beyond merely adding Fine visual tokens.

Table 3: Closed-loop comparison of learned and random location–precision allocation.

Variant	Budget	Spatial	Object	Goal	LONG	Macro
Base	—	66.27 ± 0.31	92.53 ± 1.86	85.27 ± 3.93	76.60 ± 0.92	80.17 ± 0.29
Ours	3B	69.80 ± 1.56	94.47 ± 0.99	88.40 ± 2.36	78.60 ± 1.31	82.82 ± 1.09
Random	3B	69.04 ± 2.34	93.24 ± 0.61	87.64 ± 3.11	77.17 ± 1.85	81.77 ± 1.77
Δ (Random $-$ Ours)	—	-0.76	-1.23	-0.76	-1.43	-1.05

Note: Both variants use FULL512, 3B, and the same Fine pathway. Random samples precision levels, then locations from their complete grids. Results are three-seed means $\pm$ sample standard deviations; Δ uses the displayed two-decimal values.

The Router scores locations and precisions jointly, so this control tests the full allocation and cannot separate their contributions.

6 Discussion and Conclusion

Action sensitivity offers a perspective on visual resource value: additional observations should distinguish state differences that affect the current action. Conditional allocation under fixed budgets, nonmonotonic gains from more resources, and task-specific preferred configurations together support matching precision and capacity to action requirements.

With K_{res} fixed and Base frozen, Action–Visual Precision Matching learns action-conditioned location–precision allocation and refines native action generation through base-aligned visual residuals. This retains the base policy's existing capabilities while examining the effects of additional allocation and refinement.

The current method does not learn the total additional visual capacity. Future work could dynamically truncate candidate rankings with action-conditioned Top-K to obtain a budget $K_t = f(H_t)$, then explore joint Base–Fine optimization. Dynamic capacity selection and its control benefits remain to be evaluated.

For pretrained VLAs, location, precision, and additional capacity selection could become capabilities acquired during pretraining or post-training, allowing the policy to use its current action state to determine which differences warrant finer observation and how many visual resources to allocate.

AI Use Statement

AI assisted with material retrieval and checking, organization of code and experimental records, Chinese drafting, English translation, figure–text integration, and language revision. The authors proposed and determined the research question and core method, and are responsible for experimental design, evidence review, and the final conclusions. AI also assisted with some syntax corrections during code checking.

References

Ruzena Bajcsy. Active Perception. In *Proceedings of the IEEE, 76(8):996–1005*, 1988. URL https://people.eecs.berkeley.edu/~yang/courses/cs294-6/papers/Bajcsy.active%20Perception.pdf.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation. arXiv:1308.3432, 2013. URL https://arxiv.org/abs/1308.3432.

Homanga Bharadhwaj, Jay Vakil, Mohit Sharma, Abhinav Gupta, Shubham Tulsiani, and Vikash Kumar. RoboAgent: Generalization and Efficiency in Robot Manipulation via Semantic Augmentations and Action Chunking. arXiv preprint arXiv:2309.01918, 2023. URL https://arxiv.org/html/2309.01918v1.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A Vision-Language-Action Flow Model for General Robot Control. arXiv preprint arXiv:2410.24164, 2024. URL https://arxiv.org/html/2410.24164v1.

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token Merging: Your ViT But Faster. arXiv:2210.09461v2, 2023. URL https://arxiv.org/abs/2210.09461v2.

Anthony Brohan et al. RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control. arXiv:2307.15818, 2023. URL https://arxiv.org/abs/2307.15818.

Mengzhao Chen, Mingbao Lin, Ke Li, Yunhang Shen, Yongjian Wu, Fei Chao, and Rongrong Ji. CF-ViT: A General Coarse-to-Fine Method for Vision Transformer. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(6):7042–7052, 2023. doi: 10.1609/aaai.v37i6.25860. URL https://ojs.aaai.org/index.php/AAAI/article/view/25860.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. In *Robotics: Science and Systems (RSS)*, 2023. URL https://roboticsproceedings.org/rss19/p026.pdf.

Jean-Baptiste Cordonnier, Aravindh Mahendran, Alexey Dosovitskiy, Dirk Weissenborn, Jakob Uszkoreit, and Thomas Unterthiner. Differentiable Patch Selection for Image Recognition. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2351–2360. IEEE, 2021. doi: 10.1109/CVPR46437.2021.00238. URL https://openaccess.thecvf.com/content/CVPR2021/html/Cordonnier_Differentiable_Patch_Selection_for_Image_Recognition_CVPR_2021_paper.html.

Jin Cui, Yanbin Hu, Xinyue Long, Linkai Li, Boran Zhao, and Pengju Ren. Look Where It Matters: Adaptive Visual Refinement for Vision-Language-Action Models. arXiv preprint arXiv:2608.02197, 2026. URL https://arxiv.org/html/2608.02197v1.

Chenxin Fang, Tao Chen, JunChao You, Jun Peng, Yiyi Zhou, and Rongrong Ji. Visual token coding for video multimodal large language models, 2026. URL https://arxiv.org/abs/2608.28008v1. arXiv version 1.

Juntao Gao, Feiyang Ye, Jing Zhang, and Wenjing Qian. Compressor-VLA: Instruction-Guided Visual Token Compression for Efficient Robotic Manipulation. arXiv preprint arXiv:2511.18950, 2025. URL https://arxiv.org/html/2511.18950v1.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. URL https://www.deeplearningbook.org/.

Siddhant Haldar, Zhuoran Peng, and Lerrel Pinto. BAKU: An Efficient Transformer for Multi-Task Policy Learning. In *Advances in Neural Information Processing Systems 37*, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/ff887781480973bd3cb6026feb378d1e-Paper-Conference.pdf.

Jakob Drachmann Havtorn, Amélie Royer, Tijmen Blankevoort, and Babak Ehteshami Bejnordi. MSViT: Dynamic Mixed-Scale Tokenization for Vision Transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops, pp. 838–848*, 2023. URL https://arxiv.org/html/2307.02321v2.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html.

Gao Huang, Yulin Wang, Kangchen Lv, Haojun Jiang, Wenhui Huang, Pengfei Qi, and Shiji Song. Glance and Focus Networks for Dynamic Visual Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4):4605–4621, 2023. doi: 10.1109/TPAMI.2022.3196959. URL https://ieeexplore.ieee.org/document/9851927/.

Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial Transformer Networks. In *Advances in Neural Information Processing Systems*, 2015. URL https://proceedings.neurips.cc/paper/2015/file/33ceb07bf4eeb3da587e268d663aba1a-Paper.pdf.

Dong Jing, Gang Wang, Jiaqi Liu, Weiliang Tang, Zelong Sun, Yunchao Yao, Zhenyu Wei, Yunhui Liu, Zhiwu Lu, and Mingyu Ding. Mixture of Horizons in Action Chunking. In *International Conference on Machine Learning (ICML), accepted version*, 2026. URL https://arxiv.org/html/2511.19433v2.

Angelos Katharopoulos and Francois Fleuret. Processing Megapixel Images with Deep Attention-Sampling Models. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3282–3291. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/katharopoulos19a.html.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag 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. arXiv:2406.09246, 2024. URL https://arxiv.org/abs/2406.09246.

Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Deep Laplacian Pyramid Networks for Fast and Accurate Super-Resolution. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5835–5843. IEEE, 2017. doi: 10.1109/CVPR.2017.618. URL https://doi.org/10.1109/CVPR.2017.618.

Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature Pyramid Networks for Object Detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. URL https://arxiv.org/abs/1612.03144.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for Generative Modeling. arXiv:2210.02747v2, 2023. URL https://arxiv.org/abs/2210.02747v2.

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 36, Datasets and Benchmarks Track*, 2023. URL https://papers.nips.cc/paper_files/paper/2023/file/8c3c666820ea055a77726d66fc7d447f-Paper-Datasets_and_Benchmarks.pdf.

Zhenyang Liu, Yongchong Gu, Yikai Wang, Xiangyang Xue, and Yanwei Fu. ActiveVLA: Injecting Active Perception into Vision-Language-Action Models for Precise 3D Robotic Manipulation. arXiv preprint arXiv:2601.08325, 2026. URL https://arxiv.org/html/2601.08325v1.

Volodymyr Mnih, Nicolas Heess, Alex Graves, and Koray Kavukcuoglu. Recurrent Models of Visual Attention. In *Advances in Neural Information Processing Systems*, 2014. URL https://arxiv.org/abs/1406.6247.

Athanasios Papadopoulos, Pawel Korus, and Nasir Memon. Hard-Attention for Scalable Image Classification. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 14694–14707. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/7b7916dd2de56297aa29cccb2bbf48d4-Paper.pdf.

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. URL https://arxiv.org/html/2509.22093v1.

Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. DynamicViT: Efficient Vision Transformers with Dynamic Token Sparsification. In *Advances in Neural Information Processing Systems 34*, 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/747d3443e319a22747fbb873e8b2f9f2-Paper.pdf.

Byungseok Roh, JaeWoong Shin, Wuhyun Shin, and Saehoon Kim. Sparse DETR: Efficient End-to-End Object Detection with Learnable Sparsity. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=RRGVCN8kjim.

Tomer Ronen, Omer Levy, and Avram Golbert. Vision Transformers with Mixed-Resolution Tokenization. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 4613–4622. IEEE, 2023. doi: 10.1109/CVPRW59228.2023.00486. URL https://openaccess.thecvf.com/content/CVPR2023W/ECV/html/Ronen_Vision_Transformers_With_Mixed-Resolution_Tokenization_CVPRW_2023_paper.html.

Michael S. Ryoo, AJ Piergiovanni, Anurag Arnab, Mostafa Dehghani, and Anelia Angelova. TokenLearner: What Can 8 Learned Tokens Do for Images and Videos? Advances in Neural Information Processing Systems; extended version arXiv:2106.11297v4, 2021. URL https://arxiv.org/abs/2106.11297v4.

Saumya Saxena, Mohit Sharma, and Oliver Kroemer. Multi-Resolution Sensing for Real-Time Control with Vision-Language Models. In *Conference on Robot Learning; Proceedings of Machine Learning Research 229:2210–2228*, 2023. URL https://proceedings.mlr.press/v229/saxena23a/saxena23a.pdf.

Chajin Shin, Yonghwan Kim, KwangPyo Choi, and Sangyoun Lee. Masked Feature Residual Coding for Neural Video Compression. *Sensors*, 25(14):4460, 2025. doi: 10.3390/s25144460. URL https://doi.org/10.3390/s25144460.

Shitao Tang, Jiahui Zhang, Siyu Zhu, and Ping Tan. QuadTree Attention for Vision Transformers. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=fR-EnKWL_Zb.

Burak Uzkent and Stefano Ermon. Learning When and Where to Zoom With Deep Reinforcement Learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12342–12351. IEEE, 2020. doi: 10.1109/CVPR42600.2020.01236. URL https://doi.org/10.1109/CVPR42600.2020.01236.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In *Advances in Neural Information Processing Systems*, 2017. URL https://arxiv.org/abs/1706.03762.

Yixuan Wang, Yunzhu Li, Katherine Driggs-Campbell, Li Fei-Fei, and Jiajun Wu. Dynamic-Resolution Model Learning for Object Pile Manipulation. In *Robotics: Science and Systems (RSS)*, 2023. URL https://roboticsproceedings.org/rss19/p047.pdf.

Zhenbin Wang, Lei Zhang, Lituan Wang, Wei Huang, Yan Wang, and Zhenwei Zhang. StackTok: Accelerating VLMs Inference with Budget-Adaptive Visual Token Selection. arXiv:2609.16841, 2026. URL https://arxiv.org/abs/2609.16841v1. Preprint; version 1.

Lei Xiao, Jifeng Li, Juntao Gao, Feiyang Ye, Yan Jin, Jingjing Qian, Jing Zhang, Yong Wu, and Xiaoyuan Yu. AVA-VLA: Improving Vision-Language-Action models with Active Visual Attention. arXiv preprint arXiv:2511.18960, 2025. URL https://arxiv.org/html/2511.18960v1.

Long Xu, Shanghong Li, Yongquan Chen, and Jun Luo. MST: Adaptive multi-scale tokens guided interactive segmentation, 2024. URL https://arxiv.org/abs/2401.04403v1. arXiv version 1; implementation comparison uses this version.

Jeffrey O. Zhang, Alexander Sax, Amir Zamir, Leonidas Guibas, and Jitendra Malik. Side-Tuning: A Baseline for Network Adaptation via Additive Side Networks. In *Computer Vision – ECCV 2020*, pp. 698–714. Springer International Publishing, 2020. ISBN 9783030585792. doi: 10.1007/978-3-030-58580-8_41. URL https://link.springer.com/chapter/10.1007/978-3-030-58580-8_41.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware. In *Robotics: Science and Systems (RSS)*, 2023. URL https://roboticsproceedings.org/rss19/p016.pdf.

Dehua Zheng, Wenhui Dong, Hailin Hu, Xinghao Chen, and Yunhe Wang. Less is More: Focus Attention for Efficient DETR. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6674–6683, October 2023. URL https://openaccess.thecvf.com/content/ICCV2023/html/Zheng_Less_is_More_Focus_Attention_for_Efficient_DETR_ICCV_2023_paper.html.

Gaoyue Zhou, Zichen Jeff Cui, Ada Langford, Bowen Tan, Yann LeCun, and Lerrel Pinto. Patch Policy: Efficient Embodied Control via Dense Visual Representations. arXiv preprint arXiv:2607.18236, 2026. URL https://arxiv.org/html/2607.18236v1.

Lei Zhu, Xinjiang Wang, Zhanghan Ke, Wayne Zhang, and Rynson W.H. Lau. BiFormer: Vision Transformer With Bi-Level Routing Attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10323–10333, June 2023. URL https://openaccess.thecvf.com/content/CVPR2023/html/Zhu_BiFormer_Vision_Transformer_With_Bi-Level_Routing_Attention_CVPR_2023_paper.html.

Mingjian Zhu, Kai Han, Enhua Wu, Qiulin Zhang, Ying Nie, Zhenzhong Lan, and Yunhe Wang. Dynamic Resolution Network. In *Advances in Neural Information Processing Systems*, volume 34, pp. 27319–27330, 2021a. URL https://proceedings.neurips.cc/paper/2021/hash/e56954b4f6347e897f954495eab16a88-Abstract.html.

Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable DETR: Deformable Transformers for End-to-End Object Detection. In *International Conference on Learning Representations*, 2021b. URL https://openreview.net/forum?id=koX9F3hvCG.

A Implementation Details

A.1 Base Context and the Additional Visual Pathway

During the Fine-only stage, the parameters for Base visual encoding, context generation, and native action generation are frozen. We train the added scoring, geometry projection, Fine encoding, and action refinement modules. The intermediate Base representations still vary with the input; input normalization and action scaling match the Base checkpoint.

ACT produces a 50-step sequence of action hidden states and Base actions. It ranks candidates using action, visual, and local features, and reads Fine memory using Q_b queries. DP and π_0 use representations from their native denoising and flow generation processes, respectively, to condition visual allocation and residual readout, while retaining their respective training objectives.

A.2 Joint Parent–Precision–Child Scoring

The two-camera Base64 input in ACT yields eight spatial parents. Each candidate $c = (v, p, r, j)$ contains camera, parent, precision, and child indices; location and precision jointly determine the spatial support of a patch. The Parent, Precision, and Child modules produce $P_P(p)$, $P_R(r \mid p)$, and $P_C(j \mid p, r)$, respectively. Let N_r denote the number of children within a parent at precision r. Candidates are ranked using

$$S(c) = \log P_P(p) + \log P_R(r \mid p) + \log P_C(j \mid p, r) + \log N_r.$$

Here, $S(c)$ is the count-corrected log ranking score. All candidates participate in a single Global Top-K selection, with no predefined quotas for parents, precisions, or cameras. Unselected candidates are not encoded at high precision.

Candidate geometry consists of the normalized center, width, and height, $g_c = (x_c, y_c, w_c, h_c)$. For grid cell (i, j) in the full image, with $G = r/32$, the center is $((j + 0.5)/G, (i + 0.5)/G)$ and both width and height are $1/G$. Different children within the same parent have different centers. The dimensions of the geometry projection and temporal mixing modules are listed in Table C.1.

A.3 Actual Observations and Base-Aligned Residuals

For each selected candidate, we crop the actual High patch from the image at the corresponding precision. We bilinearly upsample the Base image from the same camera to that precision and crop a Reference patch at exactly the same location and with the same support. Both patches pass through a shared Fine encoder:

$$\delta z_c = E_F(I_c^{\text{high}}) - E_F(I_c^{\text{ref}}).$$

Each 32×32 patch produces one token. Fine Attention uses Q_b (Qbase) as its query. The Key is the sum of the Base parent representation and the geometry projection, and the Value is the visual residual. Action context H conditions candidate scoring for Global Top-K selection. When replacing a candidate, its actual patch, Reference, geometry, and Key must all be updated together.

A.4 Representations over Multiple Temporal Ranges and Coefficient-Based ACT Refinement

Inspired by the use of multiple horizons in MoH (Jing et al., 2026), ACT uses $H = [h_1, \ldots, h_{50}]$ to construct $D_L = L^{-1} \sum_{k=1}^{L} h_k$, where $L \in \{10, 20, 30, 40, 50\}$. Each candidate is assigned a compatibility score with each D_L, denoted by $s_{c,L}$. A shared 5→32→5 MLP then produces candidate-specific weights:

$$\alpha_{c,L} = \text{softmax}_L(f_\alpha(s_c)), \qquad u_c = \sum_L \alpha_{c,L} s_{c,L}.$$

The resulting u_c is used for Child scoring. The cumulative representations include near-term actions in more temporal ranges while retaining the full 50-step context. Base is frozen, and the added temporal mixing and Fine modules use the final action objective.

T0_CURRENT_PREFIX also includes five coefficient experts with nonzero support over the first 10, 20, 30, 40, and 50 steps, respectively. An independent gate combines their outputs to refine actions according to $a_t^{\text{final}} = (1 + c_t) \odot a_t^{\text{base}}$. This gate uses weights distinct from those used for candidate temporal mixing.

A.5 Action Supervision and the Straight-Through Surrogate

The added pathway uses the final action objective of the corresponding policy, $\mathcal{L}_{\text{action}} = \ell(a^{\text{final}}, a^*)$, with no additional location, precision, or Top-K labels. Hard Top-K indices are not directly differentiable. For selected candidates, ACT uses the joint log probability without count correction, $\ell_c = \log P_P + \log P_R + \log P_C$, to compute

$$\gamma_c^{\text{ST}} = 1 + \ell_c - \text{sg}(\ell_c), \qquad V_c = \gamma_c^{\text{ST}} W_V \delta z_c.$$

The gate is always 1 in the forward pass and supplies surrogate gradients along selected paths in the backward pass. This is not an exact derivative of the discrete Top-K indices, and it does not require encoding unselected candidates.

B Visual Precision and Budget Instantiations

B.1 Budget Units

Let B denote the number of Base spatial tokens for the specified camera and temporal inputs, and K_{res} the number of added residuals. The relative budget is $\rho_{\text{res}} = K_{\text{res}}/B$, and the total token count is $B + K_{\text{res}}$. Each configuration has a fixed budget; equal ratios do not imply equal FLOPs or token widths across models.

Table B.1: Visual resource configurations for each policy

Configuration	Base input	B	Fine precision	Total candidates	Residual budget
ACT MAX256	Two cameras, 64×64	8	128 / 256	160	3B: K24; 6B: K48
ACT FULL512	Two cameras, 64×64	8	128 / 256 / 512	672	3B: K24; 6B: K48
DP	Two frames, 96×96	18	192 / 384	360	3B: global K54 across frames
π_0	Two cameras, 224×224	512	448 / 896	10,240	3B: global K1536

For π_0, Base representations are counted jointly across the two actual camera inputs, giving $B = 2 \times 16 \times 16 = 512$. The added residual budget is three times the total Base count, i.e., $K_{\text{res}} = 3B = 1536$. Candidates from both cameras and both precisions compete for this budget, with no fixed camera quotas.

B.2 Sampling Density and Spatial Support in ACT

ACT uses a fixed local encoder input size of 32×32. Higher source precision corresponds to a denser grid and smaller spatial support. The image pyramid maintains spatial alignment across 512→256→128→64.

Table B.2: Spatial correspondence of ACT candidate patches

Source precision	Grid per camera	Candidates across two cameras	Children per parent	Normalized side length	Side length on the 512 reference image
128	4×4	32	4	1/4	128 px
256	8×8	128	16	1/8	64 px
512	16×16	512	64	1/16	32 px

Spatial supports at different precisions may overlap; token proportions do not represent pixel coverage.

C Training Hyperparameters and Evaluation Protocol

C.1 ACT Configuration

We construct ACT demonstration data following the LIBERO data preparation procedure in BAKU MT-ACT. Using the states and actions from the original LIBERO demonstrations, we restore simulation observations state by state to obtain external and wrist camera images and proprioceptive states, while retaining the original action labels. We acquire observations at precision 512 for the same demonstration states and construct a 512→256→128→64 image pyramid, so that all precisions share the same task, demonstration, frame, and action supervision.

Training follows the task→demonstration→start-time random sampling procedure of MT-ACT BC-Dataset, with a lossless HDF5 adapter for data loading. ACT is trained separately for each suite. Fine configurations for the same suite and seed use a matched Base64/20E checkpoint, with Base frozen.

Table C.1: ACT training and module hyperparameters

Parameter	Setting
Training seeds	2, 3, 4
Base / Fine training budget	20E / 3E; see Table C.2 for update counts
Demonstrations per task	Base: 50; Fine: 49 after excluding one fixed demonstration
Data construction and sampling	Following BAKU MT-ACT, with random sampling of task→demonstration→time step
Batch size	64
Fine optimizer	AdamW
Fine encoder learning rate	10^{-4}
Learning rate for other added modules	10^{-5}
Weight decay	10^{-4}
AdamW betas / epsilon	(0.9, 0.999) / 10^{-8}
Fine learning-rate schedule	Constant learning rate, no scheduler
Action prediction / closed-loop execution steps	50 / 10
Temporal aggregation	Disabled
Fine patch input	32×32, one token per patch
Geometry projection	Linear(4,128) → GELU → Linear(128,512)
Camera embedding dimension	512
Cumulative action ranges	10, 20, 30, 40, 50 steps
Candidate temporal mixing MLP	5→32→5
ACT refinement backend	Five prefix coefficient experts with support over 10 / 20 / 30 / 40 / 50 steps

Table C.2: Optimizer update counts for each suite

Suite	Base20E	Fine3E
Spatial	19,300	2,838
Object	23,140	3,402
Goal	19,760	2,910
LONG	43,000	6,327

The update counts corresponding to 20E and 3E are listed in Table C.2. Under random sampling, E denotes a training budget set according to dataset size, not the number of passes through the data without replacement.

C.2 Diffusion Policy Configuration

DP is trained separately for each task. For each training seed, Fine is initialized from the Base20K EMA checkpoint for the same task and seed. The added visual memory is connected to the native denoising hidden-state pathway, retaining the native MSE objective and sampling procedure.

Table C.3: Diffusion Policy training settings

Parameter	Setting
Training seeds	42, 43, 44
Base / Fine training checkpoints	20,000 / 3,000 optimizer updates
Fine initialization	Base20K EMA for the same task and training seed
Visual input	Two frames with separate temporal encoding
Base parents / total candidates	18 / 360
Residual budget	Global K54 across both frames
Microbatch size	64
Gradient accumulation / synchronized devices	1 / 1
Data sampling	Windows shuffled without replacement; smaller final batch retained
Training objective	Mean native MSE
Gradient clipping	None
Optimization update frequency	One optimizer, scheduler, and EMA update per batch
Fine learning-rate schedule	Cosine, retaining the 5,000-step cycle
Warmup	1,000 steps
Reported evaluation checkpoint	Fine at 3,000 steps

C.3 π_0 Configuration

π_0 conditions visual precision allocation on action latent representations from its native flow generation process and applies a multiplicative coefficient correction to these representations. The Base backbone and native action output mapping remain frozen, and training retains the native flow-matching objective. Table C.4 describes the current 3B configuration, with the visual budget computed from the total Base count across both cameras.

Table C.4: π_0 training and generation hyperparameters

Parameter	Setting
Base / Fine precision	Two cameras, 224×224 / 448, 896
Base representation count / total candidates	512 / 10,240

Table C.4 (continued)

Parameter	Setting
Residual budget	3B: global K1536 across both cameras and precisions
Trainable components	Base and the native action output mapping are frozen; only added modules are trained
LoRA	Not used
Fine training steps	3,000 optimizer updates
Optimizer	AdamW
Learning rate	10^{-4}, constant learning rate
Weight decay	0
Global gradient-norm clipping	1.0
Microbatch / gradient accumulation	4 / 8
Effective batch size	32
Action prediction / closed-loop execution steps	50 / 5
Flow sampling steps	10
Training objective	Flow matching on the full native 50×32 action tensor
Action correction	Multiplicative coefficient correction of native action latent representations, followed by the frozen output mapping

Training duration is measured in optimizer updates. With 3,000 steps and an effective batch size of 32, training comprises 96,000 sample presentations and is not directly described as one complete epoch.

C.4 Closed-Loop Evaluation and Statistics

Table C.5: Evaluation scale and aggregation protocol

Item	Setting
Task suites	LIBERO Spatial, Object, Goal, LONG
Tasks per suite	10
Episodes per task per model	50, resets 0–49
Episodes per suite per model	500
Training seeds per ACT/DP configuration	3
ACT diagnostic coverage	4 suites $\times$ 3 seeds $\times$ 4 configurations, totaling 24,000 episodes
DP Base / Fine comparison coverage	4 suites $\times$ 10 tasks $\times$ 3 seeds $\times$ 2 models, totaling 12,000 episodes
π_0 Base / Fine comparison coverage	3 inference seeds $\times$ 2 models $\times$ 4 suites $\times$ 500 episodes, totaling 12,000 episodes
ACT original model / independent random control	4 suites $\times$ 3 seeds $\times$ 2 conditions $\times$ 500 episodes, totaling 12,000 episodes
Suite success rate	Equal weighting of tasks, with valid episodes as the denominator
Macro	Equal-weight average over the four suites within each seed first
Mean and standard deviation	Mean $\pm$ sample standard deviation over three seeds: training seeds for ACT/DP and inference seeds for π_0

Training seeds, inference seeds, and resets are recorded separately. ACT/DP results are aggregated over three training seeds, whereas π_0 results are aggregated over three inference seeds. Success-rate tables report the mean $\pm$ sample standard deviation over three seeds. For Macro, we first take the equal-weight average over the four suites within each seed, then compute the mean and sample standard deviation across seeds. Error bars in Figure 3 show sample standard deviations.

C.5 Per-Task Configuration Selection

Per-task selection uses the mean success rate over three seeds. Ties are broken first by the smaller budget, then by the lower precision ceiling. The same selected configuration is used for all seeds, and the error bars do not include uncertainty from configuration selection.

C.6 Learned versus Random Location–Precision Allocation

The configuration is Base64/FULL512/3B, with $K_{\text{res}} = 24$. At each replanning step, the random branch independently draws 24 precision labels (128, 256, 512), each with equal probability. For each precision, it then samples the corresponding number of locations uniformly without replacement from the full grids across both cameras. The total count is fixed, while counts per precision and camera result from sampling.

Random sampling bypasses joint routing, temporal mixing, and Top-K. After candidate selection, the actual patches are read again, and the geometry, Base references, and feature residuals are updated together before entering the existing Fine readout and refinement pathway. All parameters remain unchanged.

This control uses a fixed P64/FULL512/K24/Fine3E configuration with training seeds 2, 3, and 4. Each suite, seed, and condition is evaluated on 500 episodes.

D Additional Figures for ACT Precision Allocation

For the precision compositions in Figures D.1 and D.2, we first average over actual planning calls within each episode, then aggregate with equal weighting across episodes and training seeds, including both successful and failed episodes. Both sets of figures use Fine3E with training seeds 2 / 3 / 4; K24 and K48 correspond to 3B and 6B, respectively. Figure D.3 shows states from successful trajectories under Seed2 and K24. The proportions in these figures represent token allocation, not pixel coverage. The qualitative example in Figure 2 of the main text uses P64/FULL512/K24/Fine3E and comes from the same successful trajectory for Goal T03, Seed2, reset33.

D.1 FULL512 across Tasks, Budgets, and Execution Progress

Figures D.1(a–c) show the FULL512 task-level precision heatmaps, a comparison of K24 and K48 compositions, and heatmaps over execution progress, respectively. Increasing the budget does not change the shares of all precisions proportionally. Within a fixed budget, the composition also changes with execution progress.

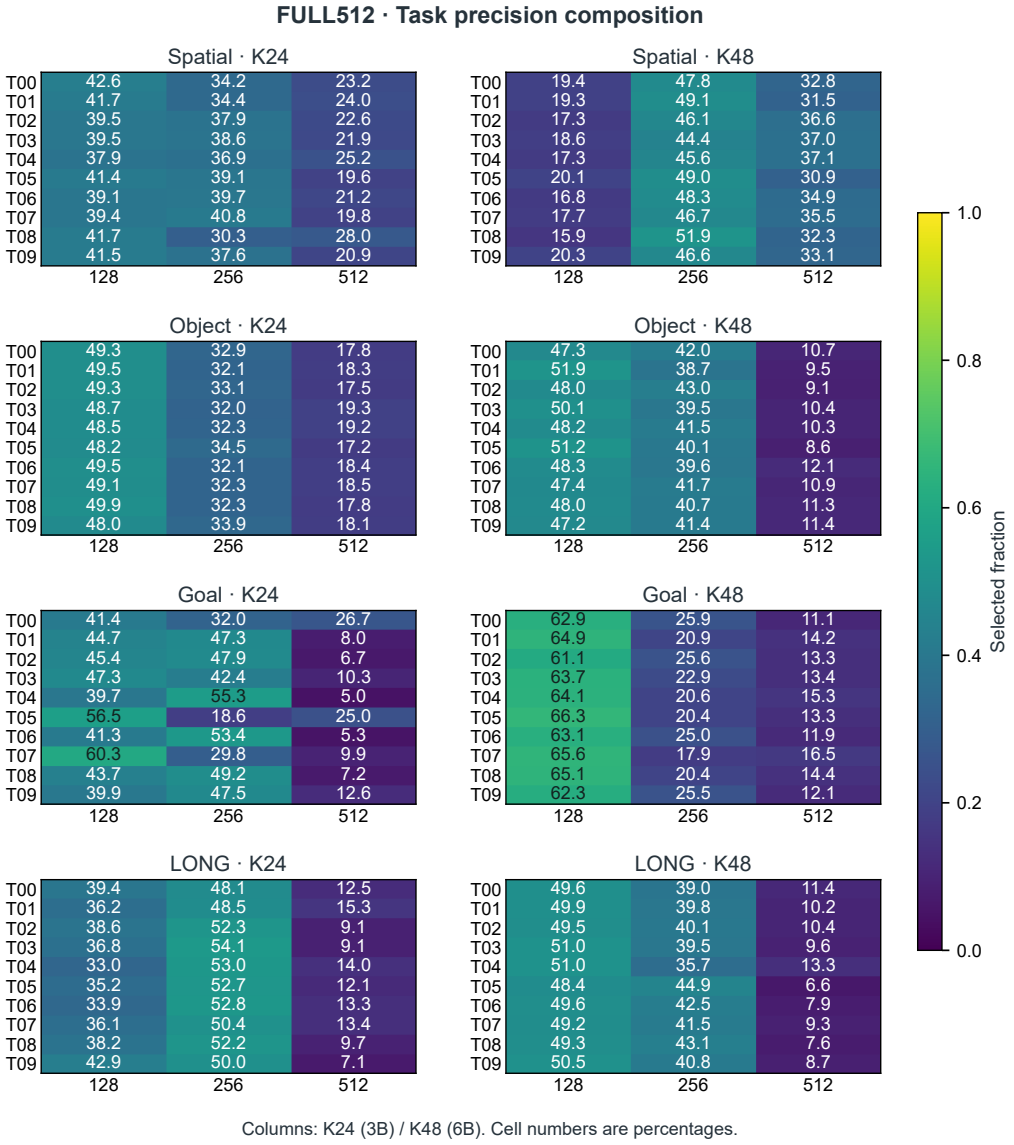

Figure D.1(a): Task-level precision composition heatmaps for FULL512. The four rows correspond to Spatial, Object, Goal, and LONG. The left column shows K24, and the right column shows K48; each panel includes T00–T09. Cell values are the percentages of selected tokens at each precision, and colors indicate proportions on a 0–1 scale.

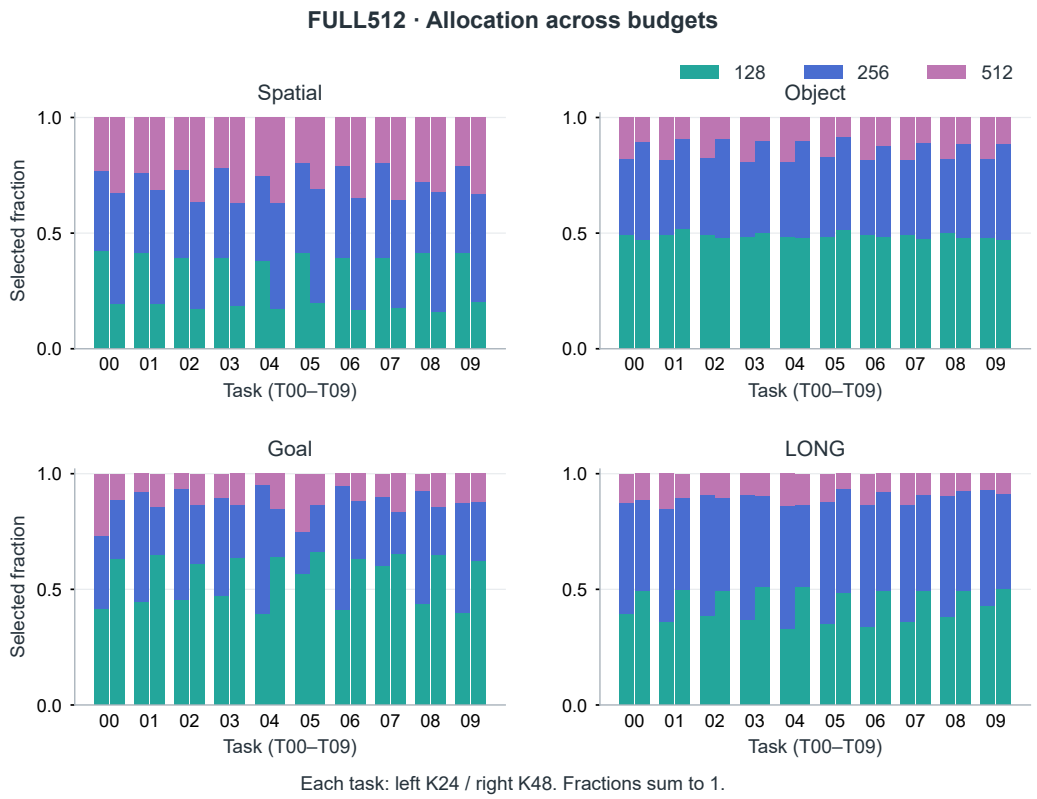

Figure D.1(b): Budget comparison for FULL512. For each task, the left bar shows K24 and the right bar shows K48; colors correspond to 128, 256, and 512. Each bar is the mean precision composition over three training seeds and sums to 1.

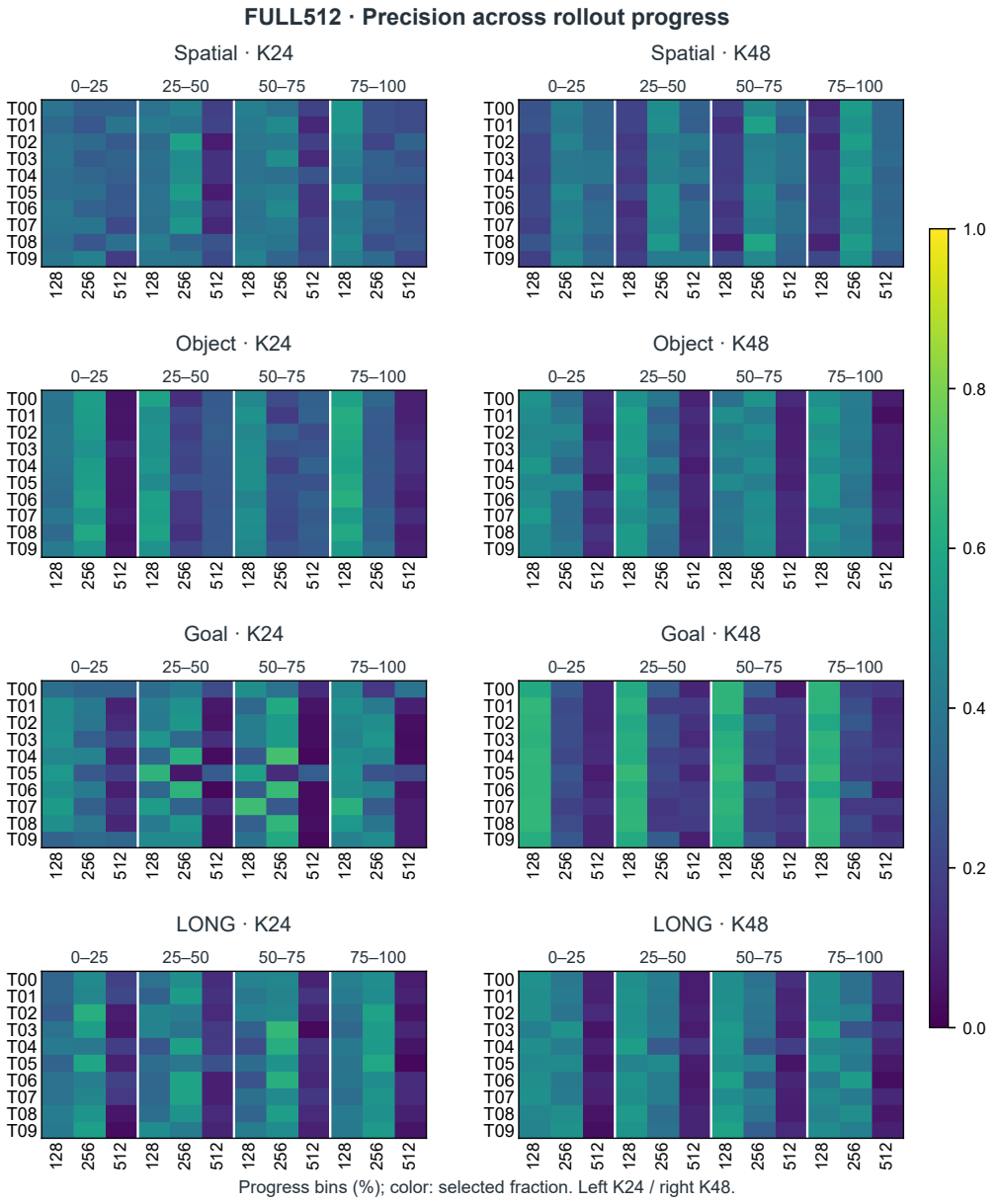

Figure D.1(c): Execution progress and precision composition for FULL512. The left column shows K24, and the right column shows K48. Each panel divides normalized control progress into four intervals and shows the selected proportions at 128, 256, and 512, in that order, within each interval. Statistics include episodes with planning calls in the corresponding interval and are averaged with equal weighting across training seeds.

D.2 MAX256 ACROSS TASKS, BUDGETS, AND EXECUTION PROGRESS

Figures D.2(a–c) present the full MAX256 results in the same format. Under K24, tasks differ in their precision compositions and how these compositions change with progress. Under K48, compositions in several suites are closer to a common ratio across tasks and progress intervals.

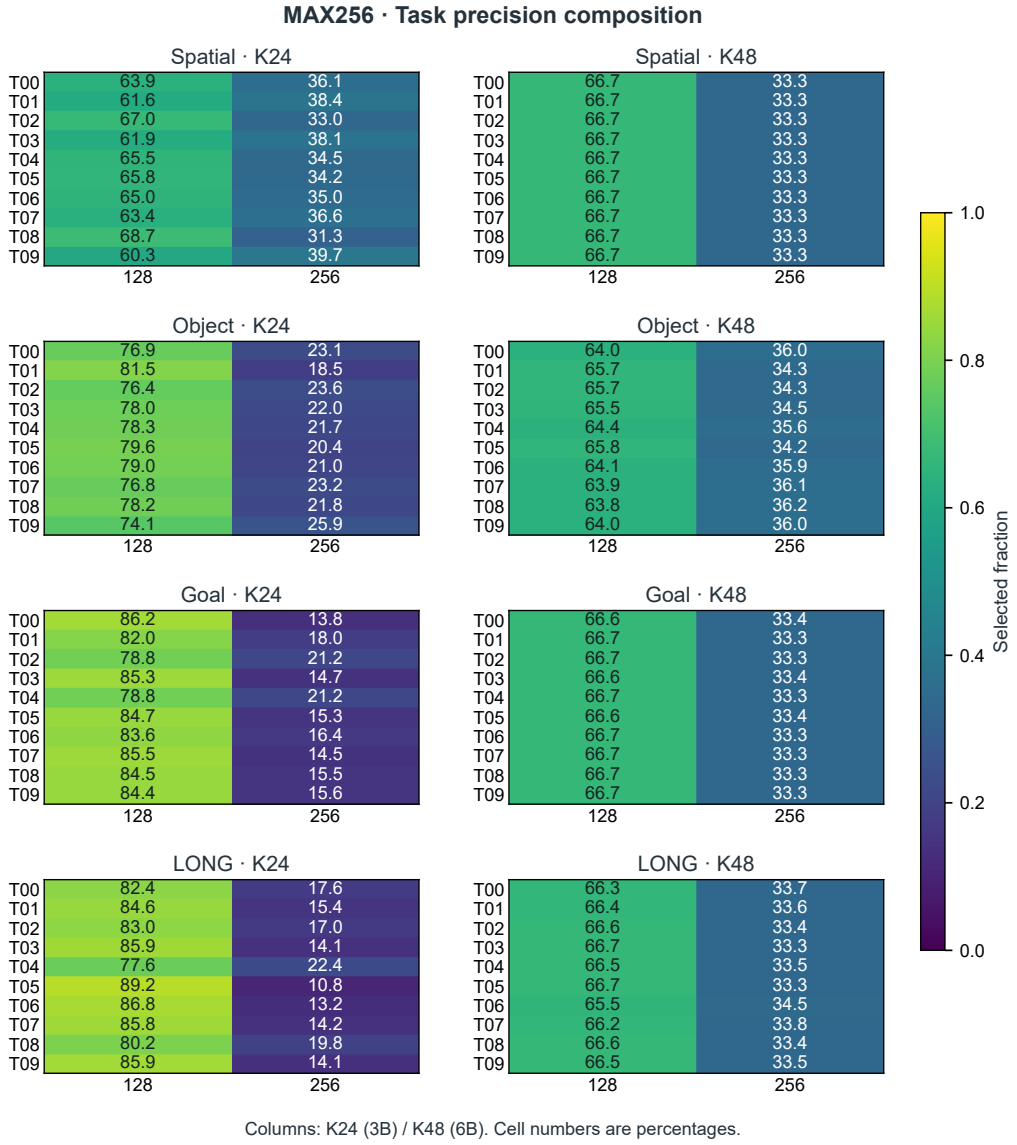

Figure D.2(a): Task-level precision composition heatmaps for MAX256. The four rows correspond to Spatial, Object, Goal, and LONG. The left column shows K24, and the right column shows K48; each panel includes T00–T09. Cell values are the percentages of selected tokens at each precision, and colors indicate proportions on a 0–1 scale.

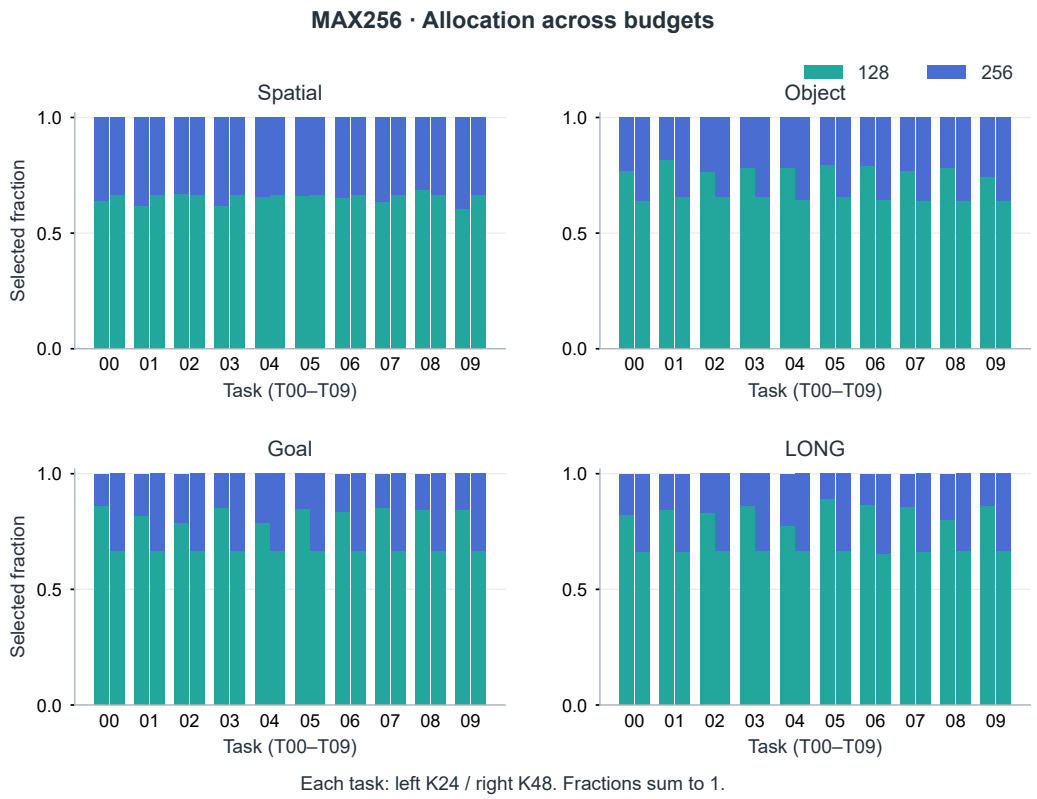

Figure D.2(b): Budget comparison for MAX256. For each task, the left bar shows K24 and the right bar shows K48; colors correspond to 128 and 256. Each bar is the mean precision composition over three training seeds and sums to 1.

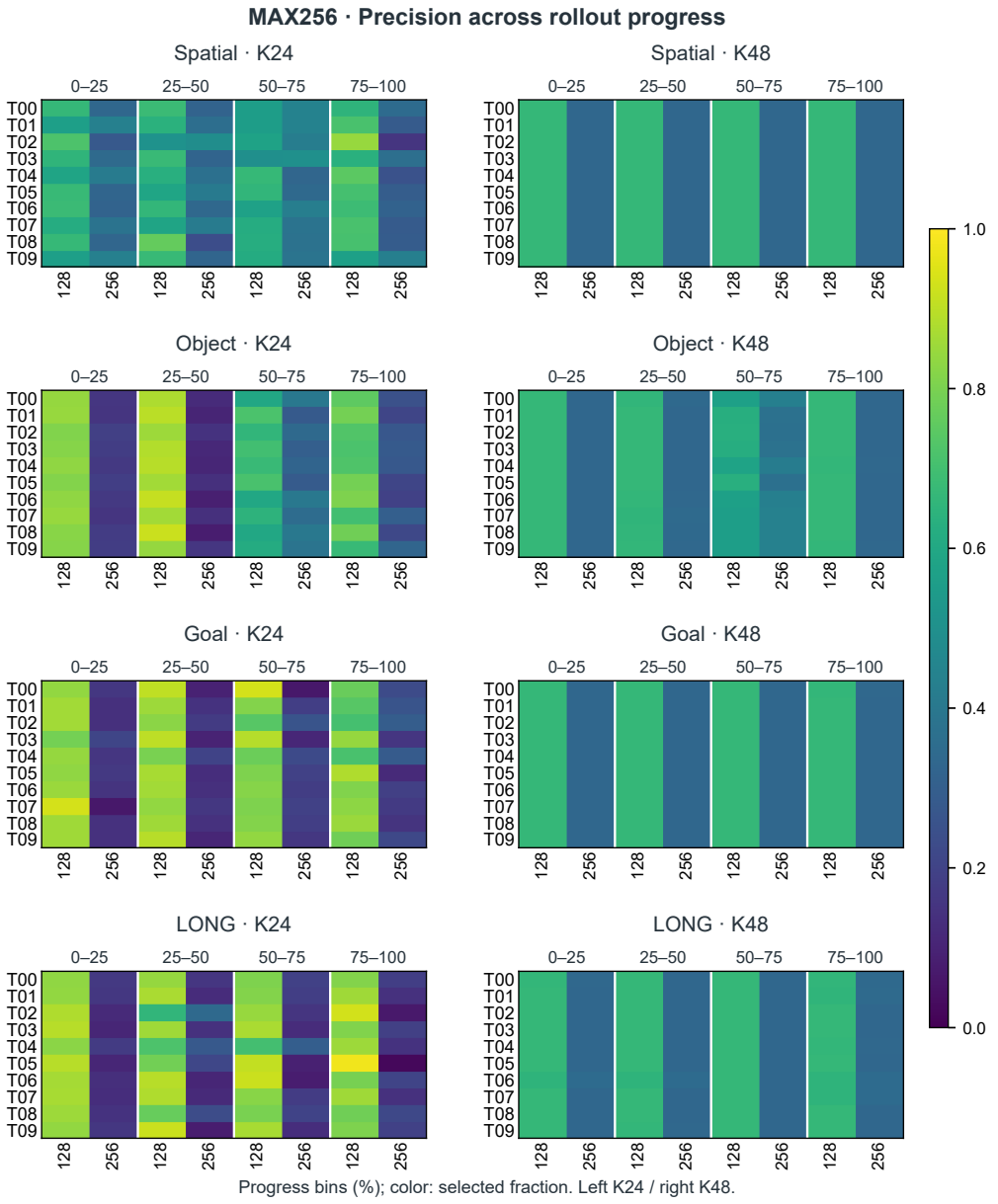

Figure D.2(c): Execution progress and precision composition for MAX256. The left column shows K24, and the right column shows K48. Each panel divides normalized control progress into four intervals and shows the selected proportions at 128 and 256, in that order, within each interval. Statistics include episodes with planning calls in the corresponding interval and are averaged with equal weighting across training seeds.

D.3 Precision Distributions across States

Figure D.3 shows state-level precision allocation for four tasks: Spatial T06, Object T01, Goal T07, and Spatial T03. All use P64/FULL512/K24/Fine3E/Seed2/Coefficient/T0 Near. Each figure shows four actual planning states from the same successful trajectory, together with the selected supports in both camera views, all 24 patches, and the corresponding precision composition.

The two Spatial examples favor 256 more in intermediate states and 512 more in earlier and later states. In Object T01, the shares shift between 256 and 128. Goal T07 changes camera allocation,

locations, and precision composition together. The checkpoint and K24 budget remain fixed within each trajectory.

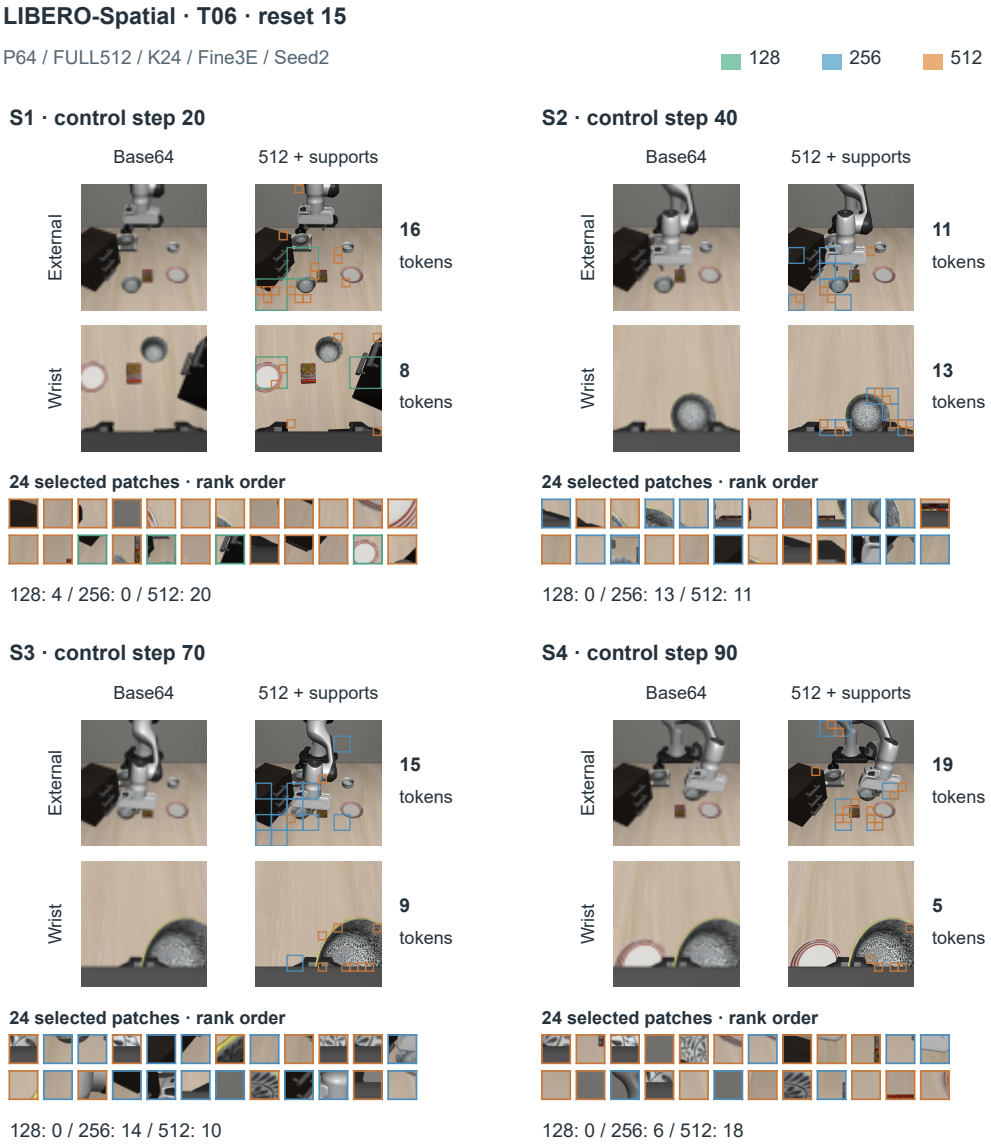

Figure D.3(a): Spatial T06, reset15, at control steps 20, 40, 70, and 90. Green, blue, and orange indicate 128, 256, and 512, respectively. Counts are the actual numbers of selected tokens at the displayed states, with a fixed total of 24.

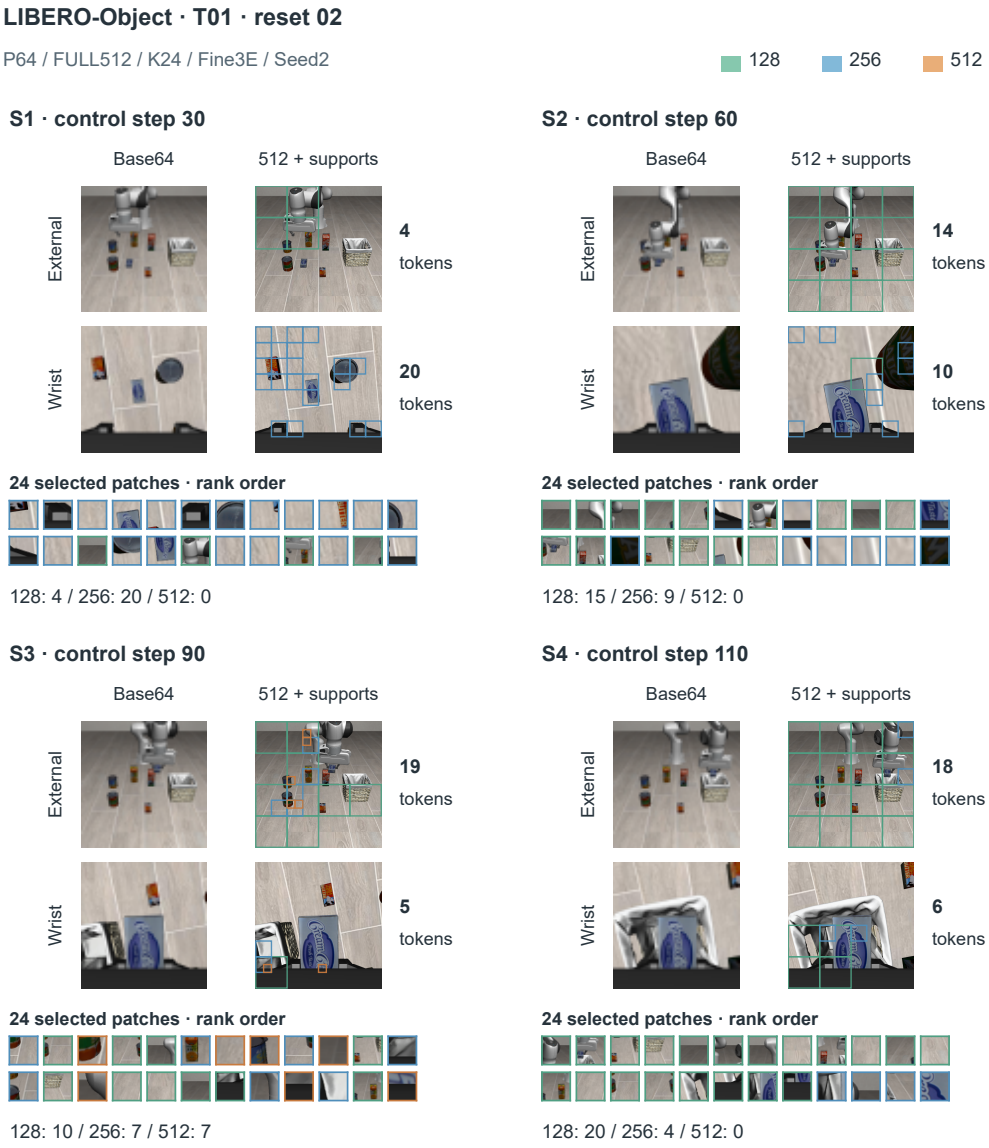

Figure D.3(b): Object T01, reset2, at control steps 30, 60, 90, and 110. Green, blue, and orange indicate 128, 256, and 512, respectively. Counts are the actual numbers of selected tokens at the displayed states, with a fixed total of 24.

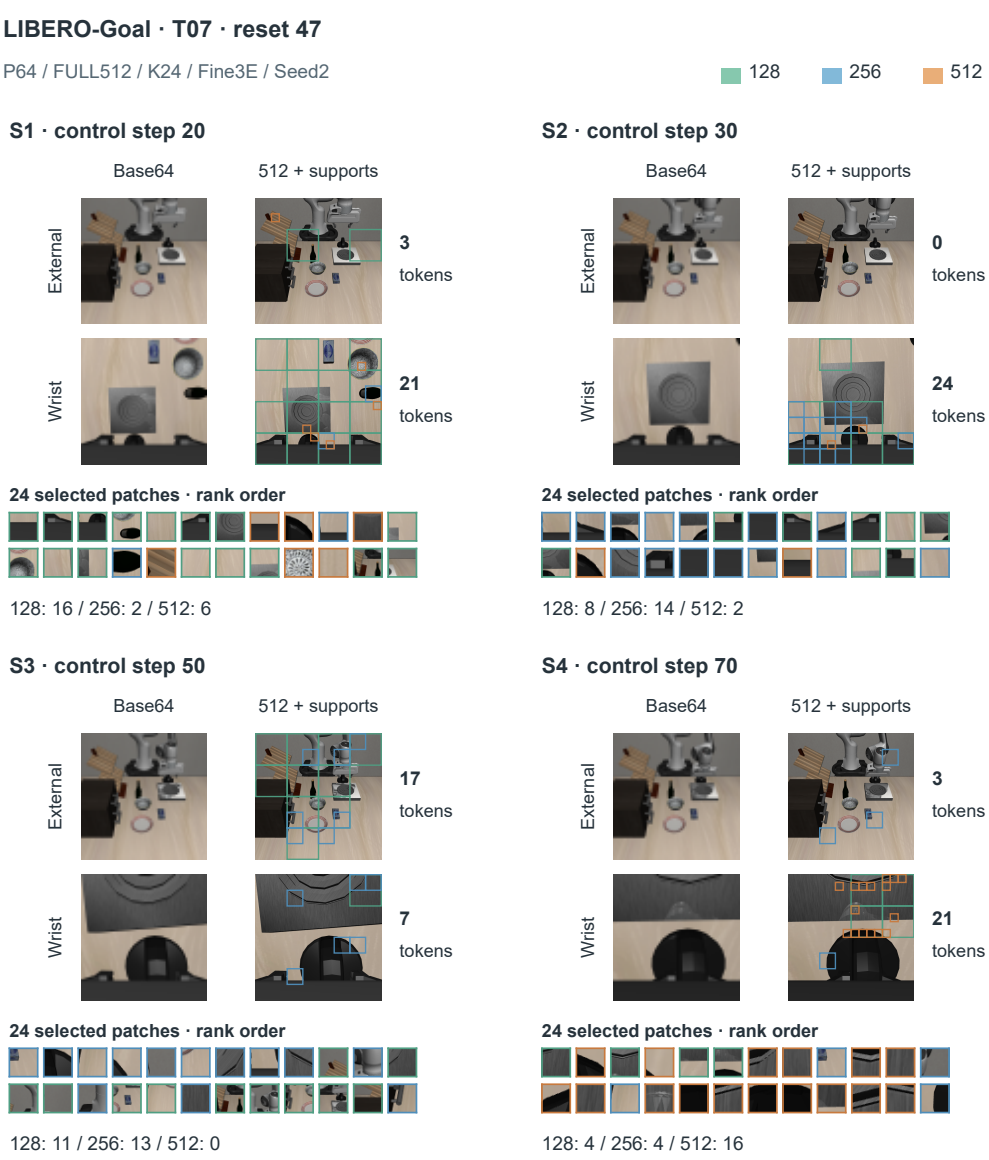

Figure D.3(c): Goal T07, reset47, at control steps 20, 30, 50, and 70. Green, blue, and orange indicate 128, 256, and 512, respectively. Counts are the actual numbers of selected tokens at the displayed states, with a fixed total of 24.

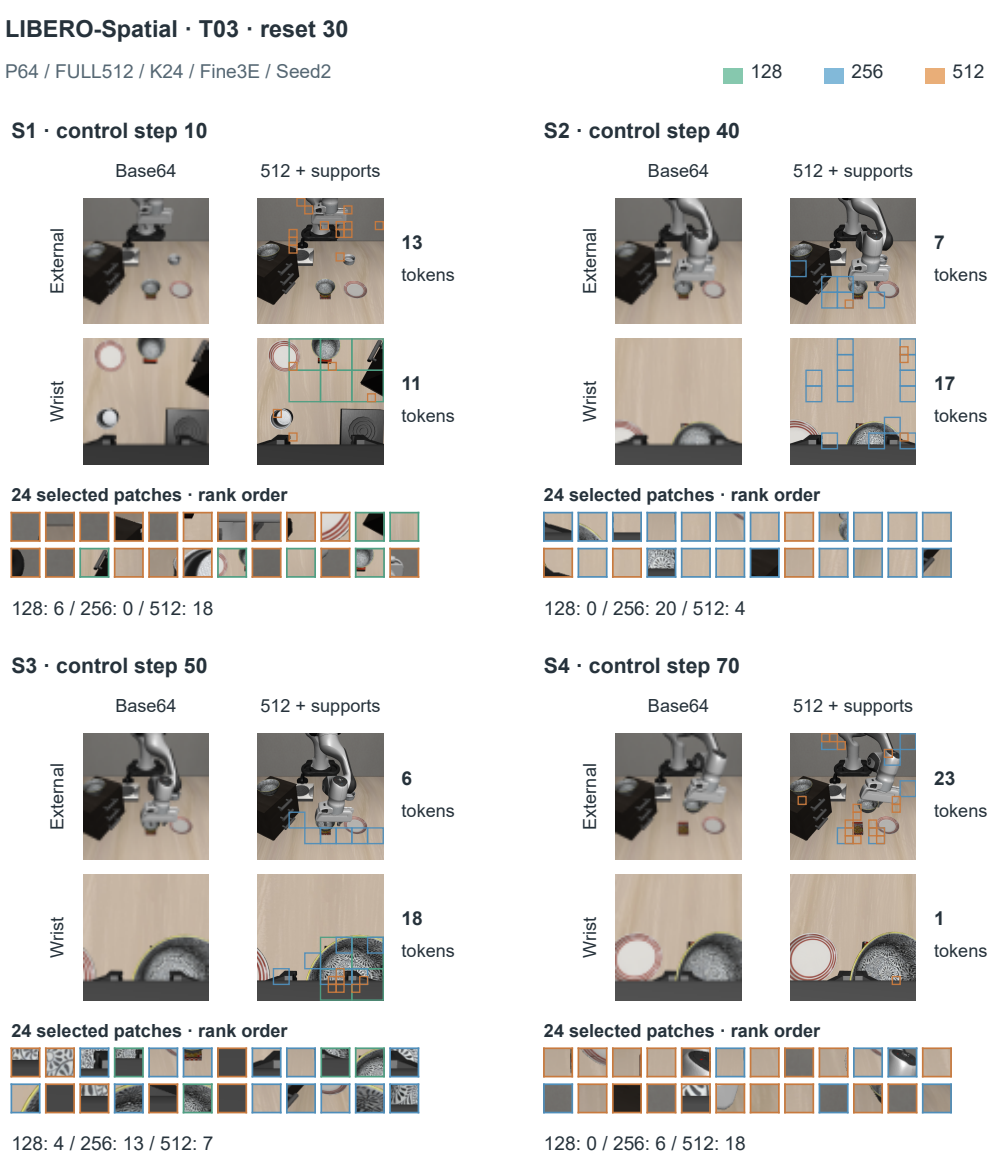

Figure D.3(d): Spatial T03, reset30, at control steps 10, 40, 50, and 70. Green, blue, and orange indicate 128, 256, and 512, respectively. Counts are the actual numbers of selected tokens at the displayed states, with a fixed total of 24.

We divide normalized control progress into four intervals and select the planning call whose precision composition is closest to the interval mean, breaking ties in favor of the earlier call. The counts shown are the actual values at that state.

E FORWARD PROCEDURE

E.1 CANDIDATE CONSTRUCTION AND LEARNED SELECTION

The inputs are observations at multiple precisions, a Base checkpoint, and a fixed K_{res}.

1. Run Base to obtain the native action representations, visual context, and Base actions.
2. Construct valid child grids for each camera, parent, and allowed precision, and compute candidate centers, supports, and geometry encodings.
3. Use Base context and geometry to compute Parent, Precision, and Child probabilities and form count-corrected ranking scores.
4. Combine all candidates and perform a single Global Top-K selection to obtain exactly K_{res} candidate identities.
5. Read the actual High patches and References with matching support for selected candidates, and construct feature residuals using the shared Fine encoder.
6. Read residual memory using the native representations for action generation, and produce final actions through the Fine interface of the corresponding policy.
7. During training, use the native action objective. The ST gates on selected Values provide surrogate gradients to the added routing modules; Base is not updated during the Fine-only stage.
8. At inference time, execute actions over the policy's execution horizon, then replan from new observations.