BRASS: Budget-Aware RAW Sensor Sampling for Edge Vision via Co-Design

Kailash Talreja*
kailashntalreja@gmail.com

Saurabh Jha
saurabhjha.21@gmail.com

Abstract

Most cameras read every pixel at full precision, even in areas that do not matter—wasting a lot of energy. We propose **BRASS**, a budget-aware RAW sensing framework that treats *sensed bits* as a first-class resource. For each small patch of a mosaic RAW image, a lightweight policy jointly learns **(1) whether to read** (a mask) and **(2) if read, how many ADC bits to use** (mixed precision). A compact RAW backbone then processes the resulting sparse, mixed-precision tensor directly (no demosaicing). We train with a budget-aware objective to obtain controllable accuracy–efficiency operating points. On Imagenette-RAW, BRASS matches the accuracy of a small RGB baseline while using about $0.47\times$ the normalized sensing-bit proxy, and it produces better-calibrated confidence scores (lower ECE). GPU measurements on an NVIDIA A800 at batch 128 indicate higher *throughput* consistent with reduced sensed bits; these numbers are not single-image latencies and exclude ISP/demosaic and I/O. These results support the L2S goal of co-designing sensors and models under explicit measurement budgets.

1 Introduction

Most camera *systems* follow a pipeline: read **every** RAW pixel, run a fixed image signal processor (ISP), then feed a model that treats all pixels with the **same precision**. This can speed up computation **after** capture, but it ignores *what* the sensor should read in the first place. As a result, effort is spent on uninformative regions, and when resources are tight the model's confidence can become poorly calibrated. Prior work that learns directly from RAW often keeps fidelity uniform [1; 2; 3], and most speed-up methods operate only on the model after capture [4; 5; 6; 7]. Sensing–inference co-design, including event-inspired approaches, shows that **what we measure** can make systems much more efficient [8; 9].

Goal: Co-design sensing and inference so the camera reads only what matters, at only the **precision it needs** without requiring demosaicing or uniform precision.

BRASS: A tiny controller decides, per small RAW mosaic patch, **whether to read** (mask) and **bit-depth** (mixed precision); a lightweight RAW model consumes this sparse, mixed-precision input directly (no demosaicing). Training optimizes task accuracy under a sensed-bit target.

Contributions: (i) A differentiable sensing policy that learns both the mask and the bit-depth; (ii) compact RAW backbones tailored to masked inputs; (iii) a budget-aware objective that exposes tunable accuracy–efficiency trade-offs.

Scope and limitations: Our study is software-based. We report a sensing-bit *proxy* (not absolute energy) and use A800 batch-128 GPU throughput as an indicator rather than single-image timings. Deployment requires sensor/driver support such as region-of-interest (ROI) readout and configurable ADC bit-depth [10; 11; 12].

*Appears in the NeurIPS 2025 Workshop on Learning to Sense (L2S), non-archival track.

2 Methodology

Overview: We split the RAW *RGGB* mosaic into small square patches and, for each patch, decide (i) whether to read it and (ii) how many analog-to-digital converter (ADC) bits to use. Let $H, W \in \mathbb{N}$ be image height and width, and $x \in \mathbb{R}^{H\times W\times 4}$ the four-plane RAW mosaic (R, G_1, G_2, B). We tile x into non-overlapping $p\times p$ patches (index (i, j)) forming an $H_p\times W_p$ grid. A policy network π_θ outputs a binary mask $M \in \{0, 1\}^{H_p\times W_p}$ (read=1/skip=0) and a per-patch bit-depth map $B \in \mathcal{B}^{H_p\times W_p}$ with $B_{ij} \in \mathcal{B} \subset \{4, 5, 6, 7, 8\}$. A sampler $\mathcal{S}$ constructs $\tilde{x} = \mathcal{S}(x; M, B, p)$ by zeroing skipped patches and quantizing read patches to B_{ij} bits; a compact RAW backbone f_ϕ predicts labels from $\tilde{x}$.

Quantization: We assume RAW values are normalized to $[0, 1]$ and use uniform mid-tread quantization for a patch assigned $b \in \mathcal{B}$:

$$q_b(z) = \Delta_b \cdot \mathrm{clip}\left(\left\lfloor \frac{z}{\Delta_b} \right\rceil, 0, 2^b{-}1\right), \qquad \Delta_b = \frac{1}{2^b - 1}.$$

Here $\lfloor\cdot\rceil$ is round-to-nearest and $\mathrm{clip}(u, a, b) = \min(\max(u, a), b)$. The sampler applies $q_{B_{ij}}(\cdot)$ to all pixels in patch (i, j) if $M_{ij}{=}1$ and writes zeros otherwise.

Learning discrete choices: Because read/skip and bit-depth are discrete, we train with relaxations. The mask uses a Binary-Concrete (Gumbel–Sigmoid) relaxation $\tilde{M}_{ij} \in (0, 1)$; bit-depth uses a softmax distribution $\tilde{\pi}_{ij}(b)$, with expected bit-depth $\bar{B}_{ij} = \sum_{b\in\mathcal{B}} b\,\tilde{\pi}_{ij}(b)$. At inference we take hard decisions: $M_{ij} = \mathbb{1}[\tilde{M}_{ij} \geq \tau_m]$ (we use $\tau_m = 0.5$) and $B_{ij} = \arg\max_b \tilde{\pi}_{ij}(b)$ [13; 14]. We use a straight-through estimator so gradients pass through these choices [15].

Budget-aware objective: We optimize $\{\theta, \phi\}$ with

$$\mathcal{L} = \underbrace{\mathcal{L}_{\text{task}}(f_\phi(\tilde{x}), y)}_{\text{task loss}} + \lambda\, C_{\text{bits}} + \lambda_\tau\, C_{\text{sp}},$$

where y is the label, $\lambda, \lambda_\tau \geq 0$ are trade-off weights, and

$$C_{\text{bits}} = \frac{1}{H_p W_p} \sum_{i,j} \mathbb{E}[M_{ij}]\,\mathbb{E}[\bar{B}_{ij}], \qquad C_{\text{sp}} = \left|\bar{M} - \tau\right|, \qquad \bar{M} = \frac{1}{H_p W_p} \sum_{i,j} M_{ij}.$$

C_{bits} penalizes expected sensed bits per patch; C_{sp} nudges the read fraction toward a target τ.

Reporting the budget (normalized proxy): At evaluation we report the *normalized* hard budget that matches our tables/figures,

$$\widehat{\text{Proxy}}_{\text{hard}} = \frac{1}{8\, H_p W_p} \sum_{i,j} M_{ij}\, B_{ij},$$

which scales the per-patch average by the 8-bit full-frame baseline (set to 1.0). The same normalization $(1/8)$ is applied when summarizing operating points as $(sb/8)$.

Backbones and training: We use two lightweight RAW backbones (no demosaicing): (i) patch-embedding $\rightarrow$ biGRU [16] $\rightarrow$ mean-pooling $\rightarrow$ linear head, following mobile design guidance [17; 18]; (ii) patch-embedding $\rightarrow$ shallow CNN mixer $\rightarrow$ adaptive-pooling $\rightarrow$ linear head. We jointly optimize $\{\theta, \phi\}$ with AdamW [19]; mask/bit temperatures are annealed; ONNX exports enable CPU/GPU timing [20]. Optional conditional compute can be added [21].

3 Experiments

Data and preprocessing: We use *Imagenette-RAW* at 160×160 pixels [22]. Each image is represented as an RGGB mosaic (four planes: R, G_1, G_2, B) and normalized to $[0, 1]$. When starting from sRGB images, we convert to realistic RAW using the "unprocessing" procedure of Brooks et al. [1]. We follow a standard train/validation split. Unless noted, batch size is 128 and all timing numbers are measured with CUDA synchronization enabled.

Method	Backbone	Read s	Bits b	Proxy $(sb/8)$	Val Top-1 (%)	Time† (ms)	Speedup
RGB	small CNN (RGB)	1.00	8.00	1.000	62.3	4.999	1.00×
Uniform	RAW (s=0.50, b=6)	0.50	6.00	0.375	41.6	1.313	3.81×
Uniform	RAW (s=0.75, b=6)	0.75	6.00	0.562	46.4	1.293	3.87×
BRASS	RAW CNN	**0.63**	**5.95**	**0.466**	**62.7**	**0.875**	**5.71**×
BRASS	RAW biGRU	0.62	6.00	0.468	52.6	—	—

Table 1: **Fairness table:** Best validation accuracy with corresponding read fraction s and bit-depth b. Proxy is normalized to a full-frame 8-bit read ($= 1.0$). †*GPU throughput indicator*: batch=128, forward-only on an NVIDIA A800; excludes ISP/demosaic (RGB) and host I/O; policy cost included.

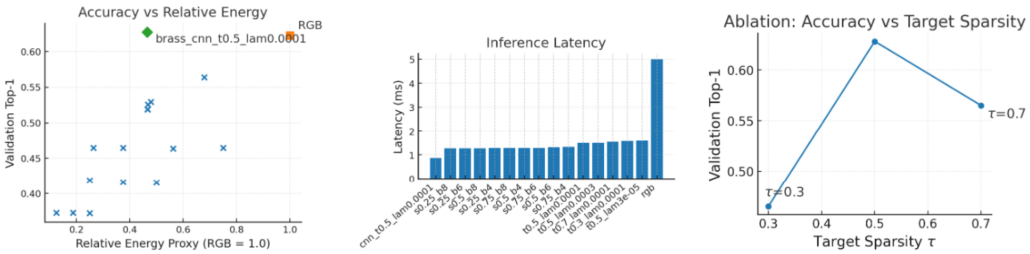

Figure 1: **Summary of results.** (a) Accuracy vs. normalized sensing-bit proxy: BRASS reaches RGB-level accuracy at about $0.47\times$ the proxy. (b) *GPU throughput indicator* on an NVIDIA A800 (batch=128; forward-only; CUDA-synchronized; ISP/demosaic for RGB excluded; policy cost included; host I/O excluded.). (c) Accuracy remains stable as we vary the target sparsity τ in the loss.

3.1 Setup, baselines, and metrics

Backbones. Two compact RAW backbones: (i) patch-embed + biGRU, and (ii) patch-embed + shallow CNN mixer (no demosaicing).

Baselines. (a) **RGB:** a small convolutional network trained on demosaiced RGB; (b) **Uniform:** RAW patches sampled uniformly with fixed sparsity s and fixed bit-depth b (see Fig. 1a for matched-proxy comparisons).

Metrics. *Val Top-1* accuracy; *Normalized sensing-bit proxy* $(sb/8)$ (below; x-axis in Fig. 1a); *GPU throughput indicator* on an NVIDIA A800 at batch size 128 (forward-only, CUDA-synchronized; Fig. 1b); *Calibration* via Expected Calibration Error (ECE; 15 bins) [23; 24] (Fig. 2a).

Normalized sensing-bit proxy: We summarize sensed work as the normalized proxy

$$\widehat{\text{Proxy}} = \frac{1}{8\,H_p W_p} \sum_{i,j} M_{ij}\, B_{ij},$$

the per-patch average of "whether we read" times "how many ADC bits we used," scaled by the 8-bit full-frame baseline (set to 1.0). This correlates with readout/quantization effort, but it is *not* a full energy model (it omits analog front-end and row/column overheads).

Main trade-off and calibration: Figure 1a shows the key trade-off: BRASS matches (or slightly exceeds) the RGB baseline in accuracy while using roughly half the normalized sensing-bit proxy. The **Uniform** baseline at the same proxy lags behind, highlighting the benefit of learning both *where* to read and *how many bits* to use. For calibration (Fig. 2a) we compute ECE with 15 equal-width bins and do *not* apply temperature scaling; BRASS consistently yields lower ECE than the RGB baseline.

Throughput (indicator): We measure synchronized forward-pass times on an NVIDIA A800 with batch size 128. As shown in Fig. 1b, BRASS achieves substantially higher GPU throughput than the RGB model, consistent with reading fewer bits and running lighter computation on masked RAW. These are not single-image latencies; measuring on-device/on-sensor timings is left for future work.

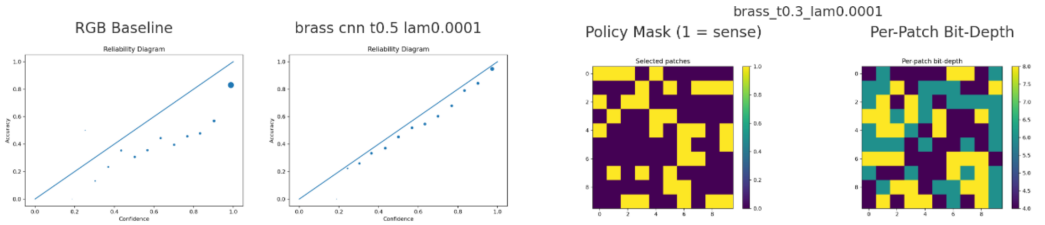

Figure 2: **Calibration and policy behavior.** (a) Reliability diagrams (ECE with 15 bins; no post-hoc temperature scaling) show lower ECE for BRASS than the RGB baseline. **RGB 0.206, BRASS-CNN 0.054** (b) Visualization of the learned policy: reads concentrate on informative regions, with higher bit-depth near edges.

Robustness: We evaluate RAW-aware corruptions (shot/read noise, RAW-domain blur, exposure darkening). At matched proxy budgets, BRASS retains a higher fraction of clean accuracy than uniform sampling. Full per-corruption results are provided in `robustRAW_*.csv`. These tests avoid RGB-only artifacts (e.g., JPEG-on-RAW) and better reflect on-sensor perturbations.

Policy behavior: Figure 2b visualizes masks and bit-depth maps. The policy focuses sensing on textured/high-gradient regions and increases precision near edges. Failure-case grids (supplement) show remaining confusions are often tied to heavy downweighting of large smooth regions—the intended budget trade-off.

Ablations and stability: We sweep target sparsity τ, budget weight λ, and backbone family. Across settings (Fig. 1c for the τ sweep), BRASS dominates uniform sampling at the same proxy and matches RGB accuracy at substantially lower proxy. A small 3-seed study (supplement) shows low variance and consistent ranking.

4 Threats to validity

- **Proxy vs. energy:** The normalized sensing-bit proxy correlates with readout/quantization work but omits analog front-end and row/column overheads.
- **Hardware realism:** Per-patch ADC control and ROI readout are emulated; commercial parts expose ROI/windowing and selectable ADC precision—typically at frame/global scope—via registers [12; 25].
- **Task/domain scope:** Results are on Imagenette-RAW classification; extensions to other RAW domains and tasks (detection/segmentation) are future work.
- **Throughput measurement:** A800 batch-128 throughput is an indicator, not single-image latency; on-device/on-sensor timings will differ.

5 Conclusion

BRASS treats *sensed bits* as a first-class, controllable budget. A tiny controller jointly decides *where* to read and *with how many ADC bits* on mosaic RAW, and a compact RAW backbone consumes the resulting sparse, mixed-precision tensor end-to-end. On Imagenette-RAW, BRASS matches a small RGB baseline at roughly $\sim 0.47\times$ the normalized proxy while delivering lower ECE and higher GPU throughput (measured as a synchronized, batch-128 forward-pass indicator). At matched budgets it also retains competitive accuracy under RAW-aware corruptions.

Limitations and Future Work: Our proxy is not a full energy model and our timings are throughput indicators; deployment requires ROI readout and per-region ADC control. Future work includes hardware-in-the-loop evaluation with sensor/driver support, single-image/on-device latency, richer RAW noise/exposure models, adaptive budgets at test time, broader tasks (detection/segmentation) and datasets, coupling with learned ISPs and optical co-design, and establishing standardized "sensed-work" metrics.

References

[1] Tim Brooks, Ben Mildenhall, Tianfan Xue, Jiawen Chen, Dillon Sharlet, and Jonathan T. Barron. Unprocessing images for learned raw denoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. URL https://openaccess.thecvf.com/content_CVPR_2019/papers/Brooks_Unprocessing_Images_for_Learned_Raw_Denoising_CVPR_2019_paper.pdf.

[2] Eli Schwartz, Raja Giryes, and Alex M. Bronstein. Deepisp: Toward learning an end-to-end image processing pipeline. *IEEE Transactions on Image Processing*, 27(10):4935–4949, 2018. doi: 10.1109/TIP.2018.2837019. URL https://doi.org/10.1109/TIP.2018.2837019.

[3] Chen Chen, Qifeng Chen, Jia Xu, and Vladlen Koltun. Learning to see in the dark. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. doi: 10.1109/CVPR.2018.00347. URL https://openaccess.thecvf.com/content_cvpr_2018/papers/Chen_Learning_to_See_CVPR_2018_paper.pdf.

[4] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2704–2713, 2018. URL https://openaccess.thecvf.com/content_cvpr_2018/html/Jacob_Quantization_and_Training_CVPR_2018_paper.html.

[5] Moran Shkolnik, Brian Chmiel, Ron Banner, Gil Shomron, Yury Nahshan, Alex M. Bronstein, and Uri Weiser. Robust quantization: One model to rule them all. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. URL https://papers.nips.cc/paper/2020/hash/3948ead63a9f2944218de038d8934305-Abstract.html.

[6] 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 (NeurIPS)*, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/747d3443e319a22747fbb873e8b2f9f2-Abstract.html.

[7] Daniel Bolya and Judy Hoffman. Token merging: Your vit but faster. *arXiv preprint arXiv:2210.09461*, 2023. URL https://arxiv.org/abs/2210.09461.

[8] Felix Heide, James Gregson, Matthias B. Hullin, and Wolfgang Heidrich. Deep optics: Learning deformable optical elements for task-specific imaging. In *ACM SIGGRAPH 2016*, 2016. doi: 10.1145/2897824.2925895. URL https://doi.org/10.1145/2897824.2925895.

[9] Guillermo Gallego, Tobi Delbrück, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew J. Davison, Jörg Conradt, Kostas Daniilidis, and Davide Scaramuzza. Event-based vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(1):154–180, 2022. doi: 10.1109/TPAMI.2020.3008413. URL https://doi.org/10.1109/TPAMI.2020.3008413.

[10] Mohit Jain, Nitin Choubey, Parth Bhatia, Sudhir Raut, Aditya Bohara, and Abhilasha Kasana. A review of recent advances in high-dynamic-range CMOS image sensors. *Imaging*, 4(1):8, 2025. URL https://www.mdpi.com/2674-0729/4/1/8.

[11] Sony Semiconductor Solutions Corporation. Sony develops a stacked CMOS image sensor technology with 2-layer transistor pixel, 2019. URL https://www.sony-semicon.com/en/technology/technology/library/stacked-cmos-image-sensor-2-layer-transistor-pixel.html. Press release; describes on-sensor functions including *smart ROI* / windowing.

[12] onsemi. Python 2000 and python 5000: High-performance global shutter CMOS image sensors, 2017. URL https://www.onsemi.com/pdf/datasheet/python-2000-d.pdf. Datasheet; documents random programmable ROI readout and selectable ADC resolution (8/10-bit).

[13] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations (ICLR)*, 2017. URL https://arxiv.org/abs/1611.01144.

[14] Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations (ICLR)*, 2017. URL https://arxiv.org/abs/1611.00712.

[15] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1305.2982*, 2013. URL https://arxiv.org/abs/1305.2982.

[16] Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, 2014. URL https://aclanthology.org/D14-1179/.

[17] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. URL https://arxiv.org/abs/1704.04861.

[18] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 122–138, 2018. doi: 10.1007/978-3-030-01264-9_8. URL https://arxiv.org/abs/1807.11164.

[19] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

[20] Microsoft. ONNX Runtime: High-performance machine learning inference and training, 2019. URL https://onnxruntime.ai/.

[21] Brandon Yang, Gabriel Bender, Quoc V. Le, and Jiquan Ngiam. Condconv: Conditionally parameterized convolutions for efficient inference. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/f2201f5191c4e92cc5af043eebfd0946-Abstract.html.

[22] Jeremy Howard. Imagenette dataset (fast.ai), 2019. URL https://github.com/fastai/imagenette/blob/master/README.md.

[23] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330, 2017. URL https://proceedings.mlr.press/v70/guo17a.html.

[24] Matthias Minderer, Josip Djolonga, Rob Romijnders, Filip Hubis, Xiaohua Zhai, Neil Houlsby, Dustin Tran, and Mario Lucic. Revisiting the calibration of modern neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/8420d359404024567b5aefda1231af24-Abstract.html.

[25] Teledyne FLIR. Blackfly s camera features: Image format control, 2020. URL https://www.flir.com/support/products/blackfly-s/. Product documentation; shows Multiple ROI and ADC Bit Depth settings exposed to users.

Appendix

.1 Supplementary Figures

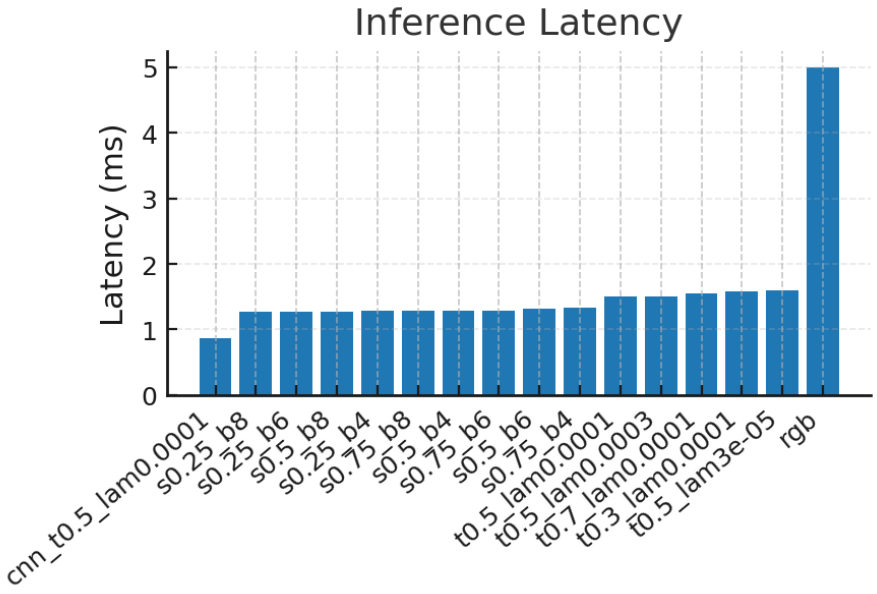

Figure 3: **GPU throughput indicator** (A800, batch=128, forward-only; excludes ISP/demosaic and I/O).

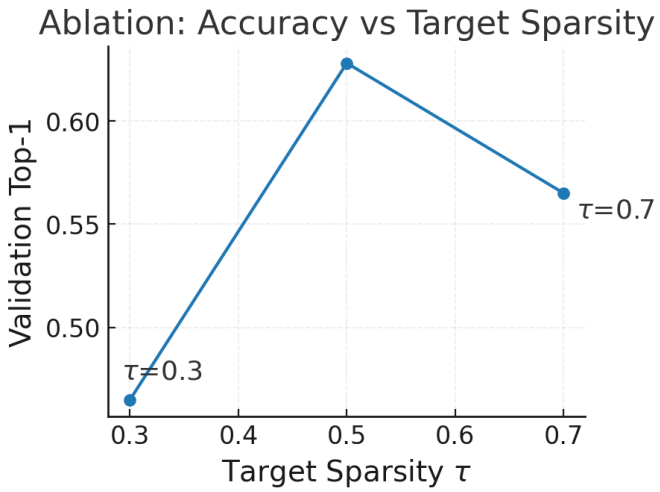

Figure 4: **Ablation: accuracy vs. target sparsity τ.**

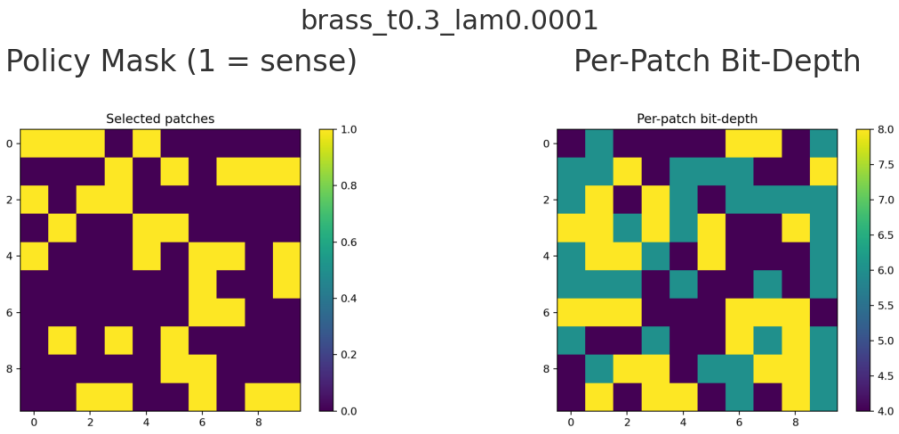

Figure 5: **Policy overlay: mask (left) and per-patch bit-depth (right)** for a representative operating point.