DEYES: DENOISE EVERYTHING YOU EVER SAW PRETRAINED VISUAL PRIORS FOR SPATIOTEMPORAL MONTE CARLO DENOISING

Anonymous authors
Paper under double-blind review

ABSTRACT

Physically based rendering produces much of the synthetic data that vision models train on, and Monte Carlo sampling sets its cost. A denoiser must reconstruct unbounded linear HDR radiance corrupted by heavy-tailed noise, a problem unlike photographic restoration. We ask whether visual representations learned from photographs transfer to Monte Carlo denoising. DEYES reads the features of a pretrained DINOv3 ConvNeXt encoder out as a multi-scale kernel pyramid that filters linear radiance with non-negative weights. Fine-tuned only on the open Noisebase corpus, it beats OIDN 2.5 and OptiX 9 on PSNR at every sample count from 1 to 512 on 39 scenes from an unseen renderer (Mitsuba 3), and beats OIDN, OptiX and NPPD on every temporal metric on Noisebase video. With its trunk frozen, DEYES beats every baseline on mean PSNR from one eighth of the training data and 9 A40-hours. Controls attribute the gain to the prior: the identical network trained from scratch peaks 1.04 dB lower and never reaches the best production denoiser on mean PSNR; a ConvNeXt pretrained by masked reconstruction on ImageNet-1k recovers most of DINOv3's gain, while DINOv3's own weights with their structure shuffled recover little. Measured end to end, DEYES reaches the best baseline's PSNR with 27–38% less render-plus-denoise time across quality targets from 32 to 512 samples per pixel. Two variants run at about 3 and 10 frames per second at 1920×1024 on one A40. Code, trained models and the MITSUBA evaluation set will be released.

1 INTRODUCTION

Much of modern machine learning learns from rendered images. About half of the seventeen training datasets of VGGT are rendered (Wang et al., 2025); the Depth Anything V2 teacher is trained on 595K synthetic images and no labeled real ones (Yang et al., 2024); Marigold trains only on Hypersim and Virtual KITTI (Ke et al., 2024; Roberts et al., 2021; Cabon et al., 2020) and CoTracker on Kubric (Karaev et al., 2024; Greff et al., 2022); rendered supervision made learned optical flow practical (Dosovitskiy et al., 2015; Mayer et al., 2016) and underlies embodied AI (Savva et al., 2019; Deitke et al., 2022). Rendered imagery supplies these models' training inputs, and for image-to-image tasks their targets, so the fidelity and cost of physically based rendering shape what they learn from.

Monte Carlo (MC) rendering is also a reconstruction problem. A renderer evaluates a finite number of stochastic light paths per pixel (samples per pixel, spp) and a denoiser turns the noisy estimate into the frame. Production path tracers ship learned denoisers (Áfra, 2026; NVIDIA, 2025; Pixar Animation Studios, 2023; Dahlberg et al., 2019), open renderers recommend rendering to a noise threshold and denoising the rest (Blender Foundation, 2021), and real-time ray tracing at one or two samples per pixel depends on denoising (Schied et al., 2017). Samples are the dominant cost of such data: Hypersim's 77,400 images took 231 vCPU-years to render (Roberts et al., 2021), and a denoiser that reaches a given quality from fewer samples lowers the sampling part of that cost. DEYES targets this offline setting and interactive previews, not real time.

Render noise is not the noise the image-restoration literature models. A one-sample pixel is a single stochastic path: its distribution is heavy-tailed (a rare path through a small bright light produces a *firefly* thousands of times brighter than its neighbors), a frame spans five decades of linear radiance,

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

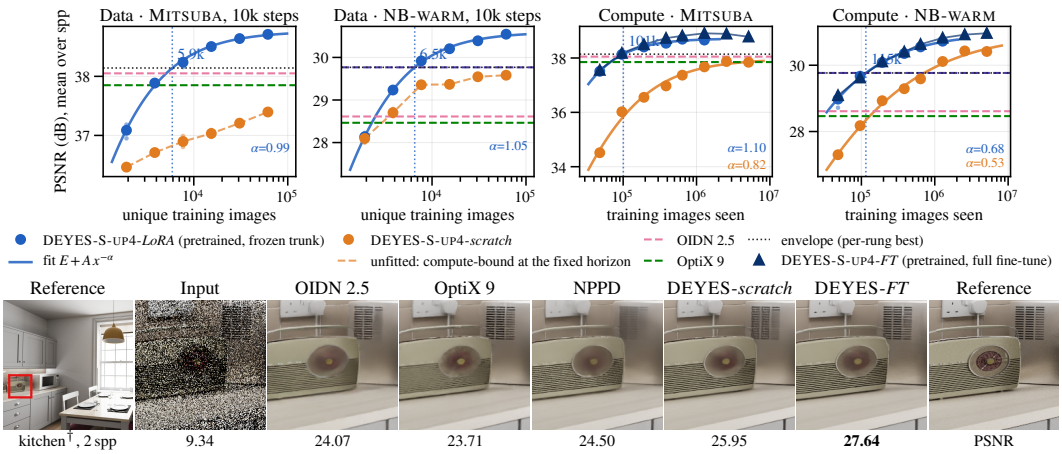

Figure 1: **Top: the pretrained prior raises the held-out plateau.** Mean PSNR over sample-count rungs of the same architecture trained from the pretrained DINOv3 trunk (blue dots: DEYES-S-UP4-*LoRA*, three seeds; navy triangles: DEYES-S-UP4-*FT*) and from scratch (orange, DEYES-S-UP4-*scratch*), against unique training images at a fixed 10,000-step schedule (left pair) and against training images seen over complete schedules of 750–80,000 steps (right pair), on the held-out MITSUBA renderer and on the NB-WARM protocol. Dashed: production baselines; dotted black: their per-rung envelope; dotted verticals: where the fitted pretrained curve crosses it. **Bottom:** a NB-WARM kitchen frame at 2 spp (temporal history); the reference frame (crop in red), then crops of the noisy input, OIDN 2.5, OptiX 9, NPPD, the same architecture trained from scratch (DEYES-*scratch* = DEYES-S-UP4-*scratch*) and fine-tuned from DINOv3 (DEYES-*FT* = DEYES-S-UP4-*FT*), and the reference, with crop PSNR (best method in bold).

the noise level is *known* per pixel (the sample count), and the renderer supplies noise-free geometry (depth, normals, albedo) beside the noisy radiance. Kernel-predicting denoisers (Bako et al., 2017) therefore do not regress colors; they predict per-pixel filter weights over neighboring input values, so the output is assembled from rendered radiance, and the network's job is to decide *where* to average. Video adds a second difficulty: at a few samples per frame the only route to a stable image is to reuse previous frames, which must be reprojected through motion vectors, discarded where surfaces are newly revealed, and blended without ghosting (Schied et al., 2017; Bálint et al., 2023).

This paper asks a representation-learning question inside that loop: *can representations learned from photographs transfer into the reconstruction of noisy HDR light transport?* The transfer crosses domain (sRGB photographs to linear HDR radiance with heavy-tailed noise) and task (recognition to reconstruction). DINO-family encoders (Caron et al., 2021; Oquab et al., 2024; Siméoni et al., 2026) are trained without labels on real images; the DINOv3 ConvNeXt encoders are distilled from a 7B-parameter teacher trained on 1.7 billion curated images. We answer with controlled data and compute ladders: at a fixed step budget they show whether the prior changes how well added data is used, and trained to their plateaus, whether it changes where learning *ends*.

DEYES (*Denoise Everything You Ever Saw*) takes a DINOv3 ConvNeXt encoder, feeds it at four times the input resolution so that its finest features land on the pixel grid, adapts it together with a small tower for the renderer's geometry buffers, and reads the features out through NPPD's multi-scale kernel pyramid (Bálint et al., 2023), which filters linear radiance with non-negative weights. One model, conditioned on sample count and trained with buffer dropout, serves 1–512 spp with or without any buffer, single frame or with temporal history. The name pays homage to REYES (*Renders Everything You Ever Saw*), Pixar's rendering architecture (Cook et al., 1987). Models are named DEYES-*size*-up*k*-*trunk*: encoder T/S/B/L, input upscale k, trunk fully fine-tuned (*FT*), frozen with LoRA (*LoRA*) or not pretrained (*scratch*). Our contributions:

- **DEYES beats production and published MC denoisers at matched sample count.** Trained only on Falcor frames, the default model, DEYES-S-UP4-*FT* (three seeds), beats the best of OIDN 2.5 and OptiX 9 at all ten rungs from 1 to 512 spp on 39 cross-renderer MITSUBA scenes; **given only the noisy radiance, without the albedo and normals OIDN and OptiX receive, it still exceeds their best on mean PSNR and has lower $\mathcal{L}_1$ at every rung up to 256 spp, tying it**

at 512 (Table 5). It beats OIDN, OptiX and NPPD on the Falcor test set per frame (NB-COLD) and with temporal history (NB-WARM), on PSNR, $\mathcal{L}_1$, temporal PSNR and FVVDP (Mantiuk et al., 2021) (Table 1). Measured end to end on MITSUBA, it reaches the best baseline's PSNR with 27–38% less render-plus-denoise time from 32 to 512 spp, and its $\mathcal{L}_1$ and LPIPS with 37–55% and 63–83% less (Figure 3).

- **It is data- and compute-efficient.** Adapted with its trunk frozen on one eighth of the corpus (120 sequences) for the full 10,000-step schedule (9 A40-hours), it already exceeds the OIDN/OptiX/NPPD envelope on mean PSNR and $\mathcal{L}_1$ on all evaluation datasets. With the full corpus, it matches that envelope on MITSUBA after 1.45 A40-hours of training (Figure 1).
- **Controlled data and compute ladders show that cross-renderer robustness comes from the prior.** Trained on Noisebase (Falcor renderer), DEYES is evaluated on MITSUBA at 1–512 spp, with buffers withheld and in single-frame mode (history reset every frame), and degrades gracefully in every case (Section 4.4). The same architecture trained from scratch plateaus 1.04 dB lower on the held-out renderer, where it never reaches the production envelope (Section 4.2).

2 RELATED WORK

Monte Carlo denoising. Feature-guided filters that use the renderer's auxiliary buffers (Rousselle et al., 2013; Bitterli et al., 2016), some with learned filter parameters (Kalantari et al., 2015), gave way to learned kernel prediction: KPCN (Bako et al., 2017) and its production successors (Vogels et al., 2018; Dahlberg et al., 2019) predict per-pixel weights over neighboring pixel values instead of regressing colors. NPPD (Bálint et al., 2023) extends this to a multi-scale partitioning pyramid with learned upsampling and temporal reprojection, and released the Noisebase corpus we train on; we adopt its pyramid with a wider finest kernel and a learned strength gate, and replace its from-scratch U-Net with a pretrained encoder. The production baselines are OIDN (Áfra, 2026) and the OptiX denoiser (NVIDIA, 2025), the latter based on Chaitanya et al. (2017); NRD (NVIDIA, 2020) targets real-time budgets and SVGF (Schied et al., 2017) introduced the variance-guided spatiotemporal filtering that real-time denoisers build on. AFGSA (Yu et al., 2021) uses buffer-guided attention and Denoising Mamba (Zhu et al., 2025) a lower-memory state-space scan; StatER (Sakai et al., 2025) and its predecessor StatMC (Sakai et al., 2024) are training-free estimators on per-pixel sample statistics, and DSCombiner (Zhou et al., 2025) shrinks paired biased and unbiased renders toward a prior. Noise2Noise training (Lehtinen et al., 2018; Tinits & Mann, 2025) removes the clean reference and learned quadrature (Lu et al., 2026) replaces the per-pixel average itself; both are orthogonal to the denoiser. DMRD (Vavilala et al., 2025) conditions a pretrained pixel-space diffusion model on the noisy render and is the closest prior use of a large image prior, at about 55× our inference cost. We compare against OIDN, OptiX, NPPD, NRD, StatER and DMRD.

Pretrained encoders and parameter-efficient adaptation. DINO (Caron et al., 2021), DINOv2 (Oquab et al., 2024) and DINOv3 (Siméoni et al., 2026) learn general visual features without labels; DINOv3's ConvNeXt (Liu et al., 2022) distillations are fully convolutional, which lets a model trained on crops run on full frames. LoRA (Hu et al., 2022), DoRA (Liu et al., 2024) and rsLoRA (Kalajdzievski, 2023) adapt frozen weights through low-rank updates; Marigold V2 (Pavlovic et al., 2026) trains low-rank adapters on a frozen generative transformer with rendered data only. We adapt a *recognition* encoder with no restoration objective and measure what its prior adds.

Scaling laws. Power-law dependence of loss on data and compute is established for language and vision (Hestness et al., 2017; Kaplan et al., 2020; Hoffmann et al., 2022; Zhai et al., 2022), and Abnar et al. (2022) study how pretraining changes downstream scaling. We use the same $E + Ax^{-\alpha}$ form, with every compute point a complete schedule, on a reconstruction task evaluated on a held-out renderer. Rendered and scan-derived datasets (Roberts et al., 2021; Greff et al., 2022; Zamir et al., 2018; Eftekhar et al., 2021) supply dense supervision to downstream models; the denoiser sits between the renderer and those models.

3 METHOD

DEYES is a kernel-predicting denoiser whose feature extractor is a pretrained image encoder (Figure 2). Every operator is local except one per-frame scalar, the mean used to normalize the input

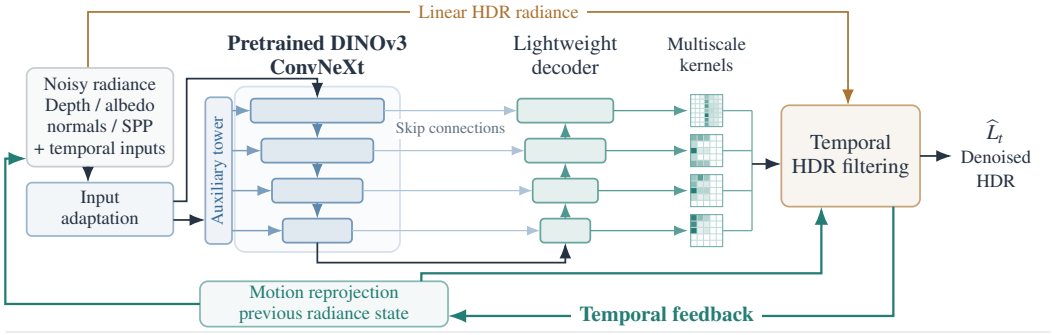

Figure 2: **DEYES overview.** A learned $4\times$ upsampler (identity at initialization) maps the noisy radiance, auxiliary buffers, sample-count plane and temporal inputs to three channels for the pretrained DINOv3 ConvNeXt stem and fifteen for an auxiliary tower that conditions the encoder stages; a lightweight decoder predicts multi-scale kernels that a partitioning pyramid applies to the *linear* radiance, blended with the motion-reprojected previous output.

radiance, so a model trained on 256^2 windows runs on a 1024×1920 frame in a single pass without tiling. The output is produced by filtering linear radiance with predicted non-negative kernels.

Input. The network sees 18 channels at native resolution: the noisy linear radiance x_t of frame t, normalized by a mean $\bar{x}_t$ over pixels and channels and compressed by a μ-law, $\tilde{x}_t = \mathrm{clip}\big(2\,\ln(1{+}\mu x_t/\bar{x}_t)/\ln(1{+}\mu) - 1,\, -1,\, 3\big)$ with $\mu{=}16$ (3 channels); depth as $\log(1{+}1/z)$ (1); shading normal (3); diffuse albedo (3); an SVGF-style (Schied et al., 2017) exponential accumulation of motion-warped noisy radiance, $A_t = \rho\, m\, \mathcal{M}(A_{t-1}) + (1-\rho\, m)\, x_t$ with $\rho{=}0.5$, reprojection $\mathcal{M}$ and validity mask m, the fraction of each pixel that reprojects inside the previous frame (3); the previous frame's blended radiance $\hat{x}_{t-1}$ (below), reprojected (3); its validity mask (1); and a constant sample-count plane $2\log_2(\min(s,512))/\log_2 512 - 1$ for s samples per pixel (1). The mean $\bar{x}_t$ is taken jointly over x_t and the two temporal radiance buffers (over x_t alone when there is no history).

The encoder octave. The encoder is the ConvNeXt trunk of DINOv3 (Siméoni et al., 2026) at one of the four sizes DINOv3 distills (tiny, small, base, large), with its global pooling and final normalization removed. Its stem has stride 4, which would place the finest feature map two octaves below the pixel grid on which a denoising kernel must be predicted. We instead feed the stem a $4\times$ upsampled input produced by a 3×3 convolution and pixel shuffle (Shi et al., 2016), initialized to nearest-neighbor upsampling of the 18 channels; its first three output channels enter the stem (the stem's first convolution absorbs DINOv3's per-channel input normalization) and the other fifteen enter an auxiliary tower (195K parameters) whose outputs are added to the encoder stages. The four stage outputs then land at native strides $/1$, $/2$, $/4$ and $/8$; the decoder synthesizes $/16$ and $/32$ with two strided convolutions. We call this *up4*; *up2* and *up1* feed the stem at $2\times$ and $1\times$. For the small encoder, up2 costs $2.4\times$ and up4 $7.9\times$ the multiply-accumulates of up1 (26, 62 and 204 GMACs per 256^2 input), and each octave adds 0.3–0.7 dB at every MITSUBA rung (Appendix F). Tiny at up2, DEYES-T-UP2-*FT*, runs in 103 ms against DEYES-S-UP4-*FT*'s 341 ms (Appendix B) and trails it by 0.5 dB on the MITSUBA mean; tiny at up1 trails by 1.0 dB. It is our interactive model: at about 10 frames per second at 1920×1024 it suits interactive preview, not real-time rendering.

Adaptation. The auxiliary tower, a decoder (1.79M parameters), the kernel heads (1.96M) and the upsampler (47K) are always trained. The default model, DEYES-S-UP4-*FT*, *fully fine-tunes* the trunk (*FT*): every trunk weight is trained, at a tenth of the learning rate of the rest. The cheaper alternative (*LoRA*) freezes the trunk and trains rank-64 LoRA (Hu et al., 2022) factors on both pointwise convolutions of every ConvNeXt block (unit scaling, one factor zero-initialized) plus the LayerNorm affines, 12.7M of 62.1M parameters during training for the small model (53.4M once the factors are folded); we use it for the data-scaling ladder. The two routes deploy the *same* network, because a LoRA update folds into the weight it modifies, so parameter count and latency are identical. Full fine-tuning is worth +0.100 dB on the MITSUBA mean at 10,000 steps and +0.22 dB at 20,000 steps, where the *LoRA* recipe saturates.

Decoder and heads. A six-level top-down decoder (widths 32–192, ConvNeXt blocks with per-pixel LayerNorm) fuses the encoder taps by addition through 1×1 laterals, so it is independent of trunk width and the four sizes differ only in the lateral shapes. Levels $/1$ and $/2$ also receive the raw 18-channel input through two small bypass blocks. Each level ends in a three-layer 3×3 head.

Kernel pyramid. Following NPPD (Bálint et al., 2023), the heads' outputs parameterize a partitioning pyramid on linear radiance. Let $\hat{x}_t = \lambda \odot \mathcal{M}(\hat{x}_{t-1}) + (1-\lambda) \odot x_t$ be the current frame's linear radiance blended with its own reprojected predecessor (NPPD's input accumulation), with a predicted blend $\lambda \in (0,1)$. A softmax over $K{=}6$ logits splits it into bands $\hat{x}_t^{(i)} = \pi_i \odot \hat{x}_t$ with $\sum_i \pi_i = 1$; band i is box-downsampled by 2^i to $b^{(i)}$ and filtered by a kernel predicted at each *source* pixel q,

$$d^{(i)}(p) = \sum_{q \in \mathcal{N}(p)} w_q^{(i)}(p-q)\, b^{(i)}(q), \qquad w_q^{(i)} = \mathrm{softmax}\big(z_q^{(i)}\big), \tag{1}$$

over a 7×7 neighborhood at level 0 and 5×5 below. The softmax normalizes the kernel predicted at each source pixel q over its taps, so Eq. 1 is a scatter: every source pixel distributes its value to its neighbors with non-negative weights that sum to one, which conserves radiance away from the frame border. The levels are recombined coarse to fine, $r^{(i)} = d^{(i)} + U^{(i)}\big(r^{(i+1)}\big)$, with learned $2\times$ upsampling $U^{(i)}$: each coarse pixel spreads its value over a 4×4 block of finer pixels with softmax weights scaled by four, so its radiance is conserved. The frame output is

$$y_t = \hat{x}_t + g \odot \Big(\beta \odot S\big(\mathcal{M}(y_{t-1})\big) + (1-\beta) \odot r^{(0)} - \hat{x}_t\Big), \tag{2}$$

where S splats the reprojected previous output with a further predicted 5×5 kernel, $\beta \in (0,1)$ is a temporal blend and $g \in (0,1)$ a strength gate. The blends λ and β, the gate g, the band logits and the level-0 kernels are per-pixel maps that the finest head predicts anew every frame; λ and β are zero where the history is invalid, and the coarser heads predict only their own kernels and upsampling weights. Every kernel weight is non-negative, and given the predicted weights the output is linear in radiance. Each kernel in Eq. 1 sums to one over the pixels a sample is spread *to*, so every sample's radiance is redistributed without loss or gain; the weights an output pixel receives need not sum to one, so an output is not a weighted average of its inputs and can exceed their local maximum. The constraint is that radiance is filtered, not synthesized by a regressor. The recurrence is detached (no backpropagation through time); a consecutive-frame sampler carries state across optimizer steps.

Sample-count conditioning and buffer dropout. The sample-count plane is the explicit noise-level condition. During training, with probability 0.30 a uniformly random non-empty subset of {depth, normal, albedo, sample-count plane} is removed (buffers zeroed, the plane replaced by a learned "unknown" value), which trains the model to be robust to any of them being absent (Section 4.4).

Synthetic high-spp data. Noisebase stores 32 samples per pixel, so higher sample counts are synthesized: a 32-spp row's residual is rescaled toward its reference R, $x_k = \max\big(0,\ R + (x_{32} - R)\sqrt{32/k}\big)$ for $k \in \{64, \ldots, 512\}$, which reproduces the variance of a k-spp estimate if R were exact and the variance fell as $1/k$. Sample counts follow a ramp that favors low counts: 18% of rows at 1 spp, 9% at 32 and 19% synthesized at 64–512. Without this augmentation the model is unchanged up to 16 spp and loses 1.16 dB at 512 spp, falling below the production envelope from 256 spp (Appendix L); the from-scratch baseline uses high-spp augmentation too.

Objective and training. The loss is NPPD's (Bálint et al., 2023) $8\,\mathrm{SMAPE} + 0.1\,\mathrm{Perc}$ on reference-mean-normalized linear radiance, where Perc compares frozen three-layer perceptual features of $\log(1{+}x)$; we train with detached history and no explicit temporal loss. We train at a global batch of 64 (16 consecutive-frame "warm" rows, 48 independent "cold" rows) with AdamW (Loshchilov & Hutter, 2019) ($\beta_1{=}0.9$, $\beta_2{=}0.95$, $\epsilon{=}10^{-6}$, weight decay 0.01), peak learning rate $1.4{\times}10^{-3}$, 150 warm-up steps and cosine decay to zero (Loshchilov & Hutter, 2017), in bf16 autocast with fp32 master weights and pyramid, and evaluate an exponential moving average (decay 0.9999) of the trainable weights (Polyak & Juditsky, 1992). The full fine-tune runs 20,000 steps with the trunk's learning rate multiplied by 0.1 and the gradient norm clipped at 4; the frozen-trunk recipe runs 10,000 steps with a clip of 1. Training the default model takes 7–18 hours on one A40. Every other hyperparameter is shared by all sizes.

Table 1: **Main comparison.** Denoiser compute time per 1024×1920 frame on one A40 (NPPD: its 2/4/8/32-spp checkpoints), PSNR averaged over each protocol's sample-count rungs (MITSUBA 1–512, NB-WARM and NB-COLD 2–32 spp), and temporal metrics on the lossless NB-WARM sequences at 2/8/32 spp. Bold best, underlined second. †NPPD: best of its four checkpoints per rung on MITSUBA (pixel-mean inputs), matched checkpoint on Noisebase. *The same network trained without pretraining for 40,000 steps.

	compute	PSNR ↑, mean over rungs			NB-WARM tPSNR ↑			FVVDP ↑			tRMAE ↓		
	ms	MITSUBA	NB-WARM	NB-COLD	2	8	32	2	8	32	2	8	32
OIDN 2.5	42	38.05	28.61	28.61	24.11	26.34	29.42	6.74	7.49	8.29	0.253	0.194	0.151
OptiX 9 (temporal)	15	37.85	28.46	28.23	24.54	26.54	29.25	6.97	7.65	8.33	0.222	0.184	0.149
NPPD†	71/76/331/386	36.16	29.77	28.73	25.59	27.31	29.85	7.19	7.81	8.44	0.215	0.185	0.157
DEYES-S-UP4-*scratch**	341	37.91	30.43	29.56	26.22	28.13	30.88	7.30	7.90	8.54	0.197	0.166	0.138
DEYES-T-UP2-*FT* (interactive)	103	38.41	30.24	29.51	25.95	27.90	30.71	7.25	7.88	8.54	0.204	0.170	0.139
DEYES-S-UP4-*FT* (default)	341	38.92	30.82	30.05	26.61	28.50	31.24	7.41	7.98	8.59	0.192	0.162	0.132
DEYES-B-UP4-*FT*	463	<u>39.04</u>	<u>30.95</u>	<u>30.15</u>	<u>26.78</u>	<u>28.66</u>	<u>31.41</u>	<u>7.44</u>	<u>8.00</u>	<u>8.63</u>	<u>0.189</u>	<u>0.160</u>	<u>0.131</u>
DEYES-L-UP4-*FT*	797	**39.12**	**31.13**	**30.28**	**26.96**	**28.85**	**31.60**	**7.47**	**8.03**	**8.64**	**0.185**	**0.157**	**0.130**

Scaling-law protocol. We train the same architecture from the pretrained trunk and from random initialization, with every parameter trainable (DEYES-S-UP4-*FT* and DEYES-S-UP4-*scratch*) and the same augmentation and regularization (Appendix D); the pretrained data ladder, and a second compute ladder, use the frozen-trunk *LoRA* recipe above. We vary one axis at a time: nested prefixes of the 960 training sequences (3.125–100%, six points, at the fixed 10,000-step schedule) for *data*, and complete cosine schedules of 750–80,000 steps on the full corpus for *compute*, so every point is a finished run. For each corpus we average PSNR over its sample-count rungs, convert it to the corresponding (geometric-mean) MSE, and fit $E + Ax^{-\alpha}$ in log space; α is the rate.

4 EXPERIMENTS

Training data. All models train on Noisebase (Bálint et al., 2023): 1024 Falcor (Kallweit et al., 2022) path-traced sequences of 64 frames, each frame stored as a 256^2 window of a full-HD render at 32 samples per pixel. We hold out 64 sequences for validation (NPPD's 51 plus 13 that fill one batch) and train on the remaining 960, all inside NPPD's own training split, i.e. 1.3% less data than NPPD. The test sequences are full 1024×1920 frames.

Test protocols and metrics. Each protocol is scored at a ladder of sample counts; one sample count of a ladder is a *rung*. NB-WARM: recurrent state carried through each Noisebase test sequence (1200 frames at 2/4/8 spp, 280 at 32); NB-COLD: state reset before every frame. MITSUBA: 39 stills from public Mitsuba 3 (Jakob et al., 2022) scenes that no DEYES model was trained on, rendered by us at 1–512 spp, with a reference averaged over 12–94 independent 1024-spp renders. PSNR, SSIM (Wang et al., 2004), MS-SSIM (Wang et al., 2003) and LPIPS (Zhang et al., 2018) are computed in floating point on ACES-tonemapped (Arrighetti, 2017) frames; SMAPE and $\mathcal{L}_1$ (mean absolute error) on linear radiance; temporal PSNR (tPSNR), FVVDP and temporal relative MAE (tRMAE) on lossless 8-bit exports of the warm sequences.

Baselines. Intel OIDN 2.5 (`rt_hdr` with albedo and normal prefiltering) and NVIDIA OptiX 9 (AOV model; temporal on Noisebase) are the production denoisers; their per-rung best, an oracle choice rather than one deployable method, is the *envelope* every MITSUBA claim is measured against. NPPD uses the authors' code and checkpoints for 2/4/8/32 spp; on MITSUBA, whose per-sample buffers we do not store, it receives pixel-mean buffers repeated to its checkpoint's sample count. DMRD is an author-supplied checkpoint trained on Noisebase, NRD 4.17 ReLAX runs with matched inputs, and StatER at its defaults. The default model is DEYES-S-UP4-*FT* (20,000 steps) and the interactive model DEYES-T-UP2-*FT*; DEYES-B-UP4-*FT* and DEYES-L-UP4-*FT* give the highest quality.

4.1 MAIN COMPARISON

MITSUBA. Across the ten rungs the default model averages 38.92 dB against the envelope's 38.14, DMRD's 34.16 and NRD's 27.35, and wins every rung, by +0.32 to $+1.08$ dB; every up4 size and seed of the full fine-tune wins every rung, and DEYES-B-UP4-*FT* and DEYES-L-UP4-*FT* lead

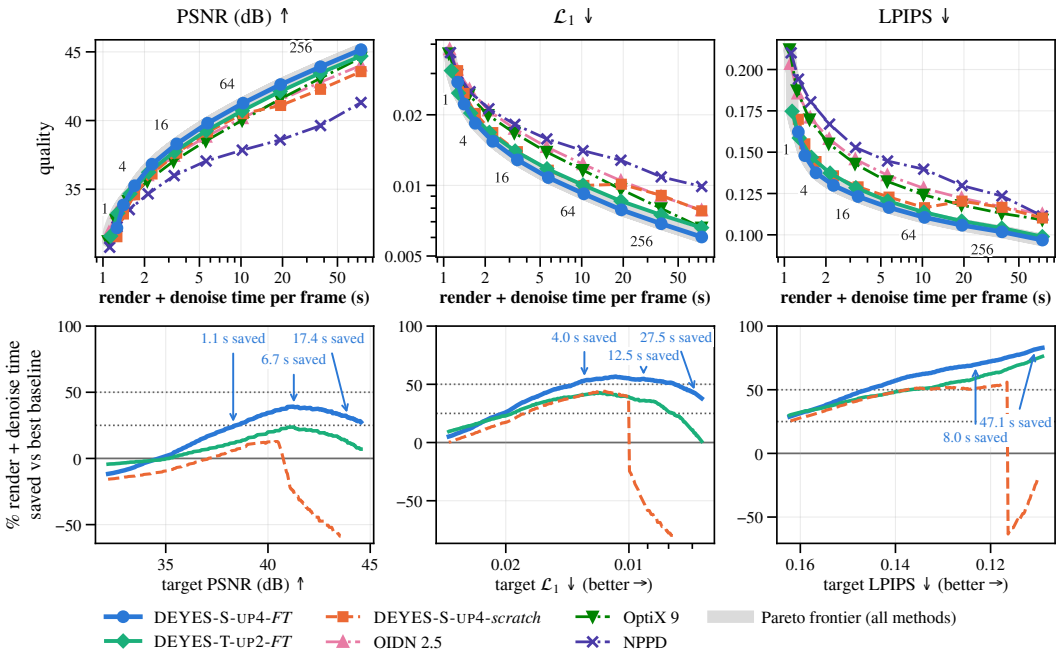

Figure 3: **Time to quality on the held-out renderer.** Top: quality against measured **render + denoise time** per frame on the 39 MITSUBA scenes, each method's 1–512 spp ladder, with the all-method Pareto frontier in gray. Bottom: render + denoise time each DEYES model saves against the cheapest baseline (OIDN 2.5, OptiX 9 or NPPD) that reaches the same quality, per quality target (DEYES-S-UP4-*FT* and DEYES-T-UP2-*FT* solid, DEYES-S-UP4-*scratch* dashed); below -150% clipped; arrows give the seconds per frame DEYES-S-UP4-*FT* saves on average over the 39 scenes to reach the quality it has at 16, 64 and 256 spp. Render time is each scene's measured path-tracing time at that sample count plus its scene load, which a single still image pays in full. Denoise time is compute time at each scene's own resolution. Appendix Figure 16 counts only the marginal sampling cost, more representative of video sequences that amortize fixed costs.

every column of Table 1 (Section 4.3). Against the per-cell best of OIDN, OptiX and NPPD it wins 105 of 108 (protocol, metric, rung) cells (Appendix N). The data favor the baselines: DEYES is trained only on the 960 open Noisebase sequences from a single renderer, while the training sets of OIDN and OptiX are not public and are presumably much larger, drawn from multiple renderers, and may include public scenes like ours. **Noisebase.** DEYES averages 30.82 dB on NB-WARM and 30.05 dB on NB-COLD against envelopes of 29.77 and 28.74, and leads NPPD, the best baseline on most temporal metrics (Appendix Figure 10), by 1.0–1.4 dB tPSNR and 0.15–0.22 JOD FVVDP at every rung (Table 1). The interactive model, at 103 ms, beats NPPD on every temporal metric at every rung as well. The from-scratch network shares most of the temporal lead: on the training renderer the architecture carries it, and pretraining adds a further 0.36–0.39 dB tPSNR and 0.39/0.49 dB PSNR on NB-WARM/NB-COLD; on the held-out renderer it adds 1.01 dB.

Cost. Because a user controls the sample budget, Figure 3 asks what that quality costs in wall clock, with render and denoise times both measured per scene. The default model alone reaches the best baseline's quality with 27–38% less time per frame from 32 to 512 spp on PSNR, 37–55% on $\mathcal{L}_1$ and 63–83% on LPIPS; each figure is metric-specific. One scene, *banner_06*, accounts for 77% of the mean per-sample render cost; without it the savings are 13–32%, 27–46% and 37–80% (Appendix L). Below 8 spp on PSNR, and below 4 spp on $\mathcal{L}_1$ and LPIPS, the production denoisers' 11–42 ms inference is worth more than the samples DEYES would save: they are equal or up to 16% faster than the default model (5% than the faster of our two models). Noisebase does not report per-frame timings, so there we compare at equal sample count against denoiser compute (Appendix E): the default model removes 18–28% of NPPD's mean-squared error at every rung of both protocols, at about NPPD's own compute from 8 spp.

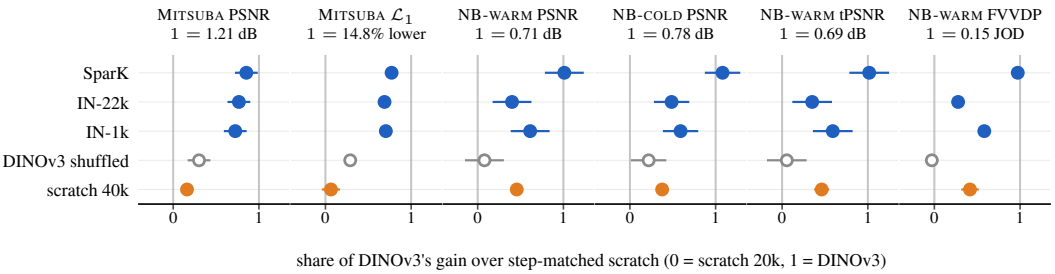

Figure 4: **The gain is not specific to DINOv3.** Share of DINOv3's gain over scratch that each ConvNeXt-S initialization recovers under DEYES-S-UP4-*FT*'s recipe (full fine-tune, 20,000 steps), per evaluation set and metric: 0 is the from-scratch network at the same 20,000 steps, 1 is DINOv3; what one unit is appears under each title ($\mathcal{L}_1$: geometric mean over rungs, share taken on its logarithm; FVVDP in JOD). Rows: SparK, masked reconstruction on ImageNet-1k without labels; IN-22k and IN-1k, supervised ImageNet classification; DINOv3 with every tensor's weights shuffled (statistics kept, structure destroyed), which recovers little; the from-scratch network trained twice as long. Whiskers on the dB panels: ± 0.16 dB, the one-seed floor (Appendix B).

4.2 SCALING: WHAT THE PRIOR ADDS

Figure 1 and Appendix O show the data and compute ladders on all three protocols. **A persistent held-out advantage.** On the compute axis both initializations are trained to completion at 750–80,000 steps, and both plateau inside that range on MITSUBA. The full fine-tune, which trains every weight as the scratch model does, peaks at 20,000 steps (38.91 dB, three seeds) and changes by -0.13 dB by 80,000; the scratch model peaks at 40,000 (37.87 dB, three seeds) and changes by -0.03 dB by 80,000 while its training loss keeps falling, a change within the spread of repeated runs (Appendix B). These horizons are the best points of each MITSUBA ladder, and the choice does not affect any comparison with the baselines: every full fine-tune from 20,000 to 80,000 steps beats the envelope at all ten rungs, and no scratch horizon reaches it. The best points differ by 1.04 dB with raw weights, the compute axis's convention (with the EMA weights of Table 1, 38.92 against 37.91 dB, a gap of 1.01 dB). Part of it is robustness: on three dark interiors the scratch network's error grows with sample count on every seed, and without them the gap is 0.87 dB (Appendix N). We report this measured separation, showing that DINOv3 pretraining improves the best held-out quality reached within every budget we trained. DINOv3 is the strongest initialization we tried but not the only effective one: at an identical recipe, a ConvNeXt pretrained by masked reconstruction on ImageNet-1k without labels (SparK; Tian et al., 2023) matches it on Noisebase and recovers 85% of its gain over scratch on MITSUBA, supervised ImageNet initializations recover 72–77%, and DINOv3's weights with their structure shuffled away recover at most 30% (Figure 4, Appendix C).

At equal training steps, the gap grows with data. The data panels compare the frozen-trunk model and scratch at the same 10,000-step schedule on nested prefixes of the corpus. The pretrained model follows a power law with $\alpha = 0.99$ on MITSUBA (1.05 on NB-WARM); scratch is still improving at this horizon, so its points are drawn, not fitted, and the gap widens from 0.62 dB at 3% of the corpus to 1.31 dB at 100%: at a fixed step budget the pretrained model makes better use of additional data. **Crossing points.** With the full step budget, the fitted pretrained curve reaches the production envelope (averaged over rungs) with about 9% (MITSUBA), 10% (NB-WARM) and 7% (NB-COLD) of NPPD's training data; with the full corpus, at about 1571 steps or 1.45 A40-hours. The scratch curve never reaches the MITSUBA envelope; compared with DEYES-S-UP4-*LoRA*, DEYES-S-UP4-*scratch* needs 6× the training steps to reach the envelope on NB-WARM and 10× on NB-COLD.

4.3 CAPACITY

Appendix Figure 7 gives the four encoder sizes at an identical recipe. The mean margin over the MITSUBA envelope rises $+0.61 \rightarrow +0.78 \rightarrow +0.90 \rightarrow +0.98$ dB from tiny to large, and the margin at the worst rung (1 spp) from $+0.21$ to $+0.54$ dB; every size wins every rung. The increments ($+0.16$, $+0.12$, $+0.08$) shrink while compute time grows from 231 to 341, 463 and 797 ms per frame. At every size and on all three evaluation sets, the full fine-tune (*FT*) beats the best frozen-trunk (*LoRA*) model by 0.20–0.24 dB on MITSUBA and 0.07–0.22 dB on Noisebase.

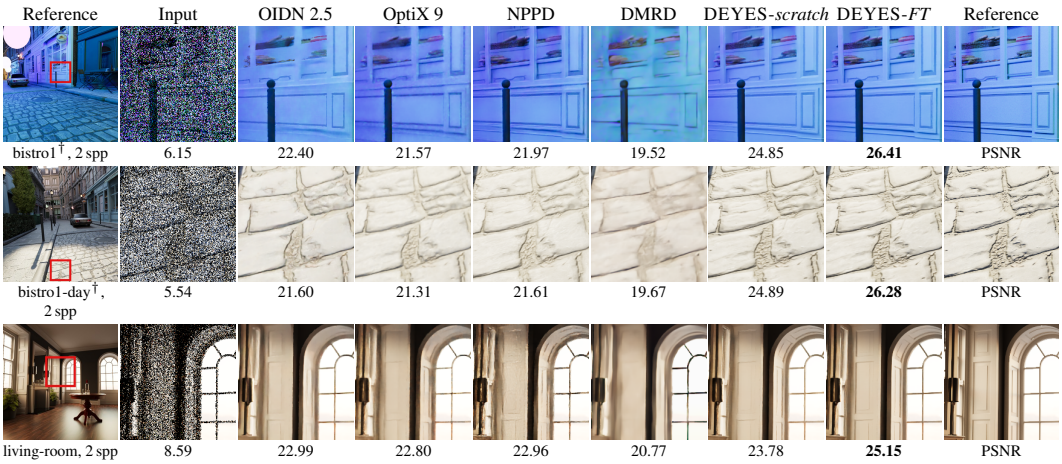

Figure 5: **Qualitative comparison.** The pretrained model (DEYES-*FT*) keeps highlights and fine texture where OIDN and OptiX over-smooth. Rows: two NB-WARM frames with temporal history and one MITSUBA scene, all at 2 spp. Columns: the reference frame (crop in red), noisy input, OIDN 2.5, OptiX 9, NPPD (on MITSUBA, its best checkpoint per crop), DMRD (single frame), the same network without pretraining (DEYES-*scratch*) and with it, and the reference, as in Figure 1; crop PSNR under each crop, best in bold.

4.4 AUXILIARY-BUFFER ROBUSTNESS

We remove each input at test time (Table 5, Appendix G). With exactly the inputs OIDN and OptiX receive (albedo and normal; depth withheld, sample count unknown) the default model changes by -0.03 dB over three seeds and still beats the envelope on every metric at every rung. With radiance alone it loses 0.69 dB of PSNR and 10% of $\mathcal{L}_1$ against all inputs, and still exceeds the envelope, whose methods receive albedo and normals: on mean PSNR ($+0.08$ dB; at every rung from 8 to 256 spp) and on $\mathcal{L}_1$ at every rung up to 256 spp (12% lower on average), and ties it on both at 512 spp. Depth changes nothing, the sample-count plane matters only at the clean end (-0.16 dB at 512 spp), normals cost -0.21 dB and albedo -0.41 dB, growing to -0.60 at 512 spp. On NB-COLD, every buffer matters about twice as much. Trained with buffer dropout, the model shows robustness here, not necessarily what each buffer contributes.

5 DISCUSSION AND LIMITATIONS

What the scaling studies show. Within every data and training budget we ran, the pretrained initialization reaches better held-out quality than the same architecture from scratch, and at a fixed step budget it makes better use of additional data. With the trunk fine-tuned, both models have the same function class, so the difference lies in initialization and inductive bias; the studies show a persistent advantage within tested budgets, not an intrinsic ceiling on what scratch training could reach under another recipe. The advantage is largest on the held-out renderer (1.04 dB) and small on the training renderer's temporal metrics, where the architecture and hyperparameter tuning carry most of the lead over NPPD. The gain shows up as lower variance between renders at high sample counts and lower bias at low ones (Appendix I). **Scope.** Every scaling law is on the small encoder, and the from-scratch baseline's recipe is documented in Appendix D. Mitsuba and Falcor differ in both renderer and scene content. We compare four pretraining sources at one recipe, with one seed each except DINOv3 (Figure 4). **Cost.** The default model takes 341 ms per 1920×1024 frame and the interactive DEYES-T-UP2-*FT* 103 ms (Table 2); in measured wall clock on MITSUBA the default model pays for itself from 8 spp.

Conclusion. A pretrained recognition encoder, given the pixel grid and a kernel-predicting readout on linear radiance, beats the production denoisers on a renderer it never saw at every sample count, and reaches their quality with less render + denoise time over a wide range of quality targets. Controlled ladders attribute the gain to the prior, and the gain is not specific to DINOv3: other pretrained ConvNeXt encoders recover much of it, while a shuffled DINOv3 trunk does not (Figure 4).

Ethics statement

This work uses publicly available scene assets and datasets and involves no human subjects or personal data. Better denoisers lower the cost of rendered imagery, including synthetic training data, which is the intended benefit; the same models could make synthetic imagery cheaper to produce for any purpose.

Reproducibility statement

Section 3 gives every architectural and training hyperparameter; Appendix B specifies the metric pipeline, the reference policy, package versions and the timing protocol. The training corpus and the baselines are public. We will release the code and the MITSUBA evaluation set.

The use of large language models

Large language models were used to draft and edit text, to write figure and table scripts, and to search the literature. All experiments, numbers, figures and claims were produced from the authors' data and verified by the authors, who take full responsibility for the content.

References

Samira Abnar, Mostafa Dehghani, Behnam Neyshabur, and Hanie Sedghi. Exploring the limits of large scale pre-training. In *International Conference on Learning Representations (ICLR)*, 2022.

Attila T. Áfra. Intel Open Image Denoise. `https://www.openimagedenoise.org`, 2026. Version 2.5.

Walter Arrighetti. The Academy Color Encoding System (ACES): A professional color-management framework for production, post-production and archival of still and motion pictures. *Journal of Imaging*, 3(4):40, 2017. doi: 10.3390/jimaging3040040.

Steve Bako, Thijs Vogels, Brian McWilliams, Mark Meyer, Jan Novák, Alex Harvill, Pradeep Sen, Tony DeRose, and Fabrice Rousselle. Kernel-predicting convolutional networks for denoising Monte Carlo renderings. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 36(4):97:1–97:14, 2017. doi: 10.1145/3072959.3073708.

Martin Bálint, Krzysztof Wolski, Karol Myszkowski, Hans-Peter Seidel, and Rafał Mantiuk. Neural partitioning pyramids for denoising Monte Carlo renderings. In *ACM SIGGRAPH 2023 Conference Proceedings*, pp. 60:1–60:11, 2023. doi: 10.1145/3588432.3591562.

Benedikt Bitterli, Fabrice Rousselle, Bochang Moon, José A. Iglesias-Guitián, David Adler, Kenny Mitchell, Wojciech Jarosz, and Jan Novák. Nonlinearly weighted first-order regression for denoising monte carlo renderings. *Computer Graphics Forum (Proc. EGSR)*, 35(4):107–117, 2016. doi: 10.1111/cgf.12954.

Blender Foundation. Blender 3.0 release notes: Cycles. `https://developer.blender.org/docs/release_notes/3.0/cycles/`, 2021.

Yohann Cabon, Naila Murray, and Martin Humenberger. Virtual KITTI 2, 2020. arXiv:2001.10773.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

Chakravarty R. Alla Chaitanya, Anton S. Kaplanyan, Christoph Schied, Marco Salvi, Aaron Lefohn, Derek Nowrouzezahrai, and Timo Aila. Interactive reconstruction of Monte Carlo image sequences using a recurrent denoising autoencoder. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 36 (4):98:1–98:12, 2017. doi: 10.1145/3072959.3073601.

Robert L. Cook, Loren Carpenter, and Edwin Catmull. The Reyes image rendering architecture. In *Computer Graphics (Proceedings of SIGGRAPH)*, volume 21, pp. 95–102, 1987. doi: 10.1145/37402.37414.

Henrik Dahlberg, David Adler, and Jeremy Newlin. Machine-learning denoising in feature film production. In *ACM SIGGRAPH 2019 Talks*, 2019. doi: 10.1145/3306307.3328150.

Matt Deitke, Eli VanderBilt, Alvaro Herrasti, Luca Weihs, Kiana Ehsani, Jordi Salvador, Winson Han, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. ProcTHOR: Large-scale embodied AI using procedural generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 5982–5994, 2022.

Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Häusser, Caner Hazırbaş, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. FlowNet: Learning optical flow with convolutional networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2015.

Ainaz Eftekhar, Alexander Sax, Jitendra Malik, and Amir Zamir. Omnidata: A scalable pipeline for making multi-task mid-level vision datasets from 3D scans. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J. Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann, Thomas Kipf, Abhijit Kundu, Dmitry Lagun, Issam Laradji, Hsueh-Ti (Derek) Liu, Henning Meyer, Yishu Miao, Derek Nowrouzezahrai, Cengiz Oztireli, Etienne Pot, Noha Radwan, Daniel Rebain, Sara Sabour, Mehdi S. M. Sajjadi, Matan Sela, Vincent Sitzmann, Austin Stone, Deqing Sun, Suhani Vora, Ziyu Wang, Tianhao Wu, Kwang Moo Yi, Fangcheng Zhong, and Andrea Tagliasacchi. Kubric: A scalable dataset generator. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md. Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou. Deep learning scaling is predictable, empirically, 2017. arXiv:1712.00409.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Oriol Vinyals, Jack W. Rae, and Laurent Sifre. Training compute-optimal large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 30016–30030, 2022.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022.

Wenzel Jakob, Sébastien Speierer, Nicolas Roussel, Merlin Nimier-David, Delio Vicini, Tizian Zeltner, Baptiste Nicolet, Miguel Crespo, Vincent Leroy, and Ziyi Zhang. Mitsuba 3 renderer. `https://mitsuba-renderer.org`, 2022.

Damjan Kalajdzievski. A rank stabilization scaling factor for fine-tuning with LoRA, 2023. arXiv:2312.03732.

Nima Khademi Kalantari, Steve Bako, and Pradeep Sen. A machine learning approach for filtering Monte Carlo noise. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 34(4):122:1–122:12, 2015. doi: 10.1145/2766977.

Simon Kallweit, Petrik Clarberg, Craig Kolb, Tomáš Davidovič, Kai-Hwa Yao, Theresa Foley, Yong He, Lifan Wu, Lucy Chen, Tomas Akenine-Möller, Chris Wyman, Cyril Crassin, and Nir Benty. The Falcor rendering framework. `https://github.com/NVIDIAGameWorks/Falcor`, 2022.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020. arXiv:2001.08361.

Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. CoTracker: It is better to track together. In *European Conference on Computer Vision (ECCV)*, 2024.

Bingxin Ke, Anton Obukhov, Shengyu Huang, Nando Metzger, Rodrigo Caye Daudt, and Konrad Schindler. Repurposing diffusion-based image generators for monocular depth estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Jaakko Lehtinen, Jacob Munkberg, Jon Hasselgren, Samuli Laine, Tero Karras, Miika Aittala, and Timo Aila. Noise2Noise: Learning image restoration without clean data. In *International Conference on Machine Learning (ICML)*, volume 80 of *PMLR*, pp. 2965–2974, 2018.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. DoRA: Weight-decomposed low-rank adaptation. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *PMLR*, pp. 32100–32121, 2024.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A ConvNet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11976–11986, 2022.

Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations (ICLR)*, 2017.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

Haolin Lu, Liwen Wu, Zimo Wang, Tzu-Mao Li, and Ravi Ramamoorthi. Neural quadrature rule and autoregressive adaptive sampling. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 45(4), 2026. doi: 10.1145/3811318. Article 528.

Rafał K. Mantiuk, Gyorgy Denes, Alexandre Chapiro, Anton Kaplanyan, Gizem Rufo, Romain Bachy, Trisha Lian, and Anjul Patney. FovVideoVDP: A visible difference predictor for wide field-of-view video. *ACM Transactions on Graphics*, 40(4):49:1–49:19, 2021. doi: 10.1145/3450626.3459831.

Nikolaus Mayer, Eddy Ilg, Philip Häusser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

NVIDIA. NVIDIA real-time denoisers (NRD). `https://github.com/NVIDIA-RTX/NRD`, 2020. Version 4.17.

NVIDIA. NVIDIA OptiX AI-accelerated denoiser. `https://developer.nvidia.com/optix-denoiser`, 2025. OptiX 9.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. arXiv:2304.07193, 2023.

Igor Pavlovic, Thiemo Wandel, Anton Obukhov, Luca Bartolomei, Andrey Davydov, Fabio Tosi, Matteo Poggi, Sabine Süsstrunk, and Dengxin Dai. Marigold V2: Revisiting diffusion transformers for monocular depth estimation. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, 45(6), 2026. To appear; arXiv:2609.08084.

Pixar Animation Studios. RenderMan denoiser. `https://renderman.atlassian.net/wiki/spaces/REN26/pages/19662375/Denoiser`, 2023. Machine-learning denoiser from Disney Research, shipped since RenderMan 25.

Boris T. Polyak and Anatoli B. Juditsky. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, 1992. doi: 10.1137/0330046.

Mike Roberts, Jason Ramapuram, Anurag Ranjan, Atulit Kumar, Miguel Angel Bautista, Nathan Paczan, Russ Webb, and Joshua M. Susskind. Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

Fabrice Rousselle, Marco Manzi, and Matthias Zwicker. Robust denoising using feature and color information. *Computer Graphics Forum (Proc. Pacific Graphics)*, 32(7):121–130, 2013. doi: 10.1111/cgf.12219.

Hiroyuki Sakai, Christian Freude, Thomas Auzinger, David Hahn, and Michael Wimmer. A statistical approach to Monte Carlo denoising. In *SIGGRAPH Asia 2024 Conference Papers*, pp. 68:1–68:11, 2024. doi: 10.1145/3680528.3687591.

Hiroyuki Sakai, Christian Freude, Michael Wimmer, and David Hahn. Statistical error reduction for Monte Carlo rendering. In *SIGGRAPH Asia 2025 Conference Papers*, pp. 50:1–50:12, 2025. doi: 10.1145/3757377.3763995.

Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A platform for embodied AI research. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

Christoph Schied, Anton Kaplanyan, Chris Wyman, Anjul Patney, Chakravarty R. Alla Chaitanya, John Burgess, Shiqiu Liu, Carsten Dachsbacher, Aaron Lefohn, and Marco Salvi. Spatiotemporal variance-guided filtering: Real-time reconstruction for path-traced global illumination. In *Proceedings of High Performance Graphics (HPG)*, pp. 2:1–2:12, 2017. doi: 10.1145/3105762.3105770.

Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1874–1883, 2016. doi: 10.1109/CVPR.2016.207.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. DINOv3. *Transactions on Machine Learning Research*, 2026. arXiv:2508.10104.

Keyu Tian, Yi Jiang, Qishuo Diao, Chen Lin, Liwei Wang, and Zehuan Yuan. Designing BERT for convolutional networks: Sparse and hierarchical masked modeling. In *International Conference on Learning Representations (ICLR)*, 2023.

Andrew Tinits and Stephen Mann. Nonlinear Noise2Noise for efficient Monte Carlo denoiser training. In *SIGGRAPH Asia 2025 Conference Papers*, 2025. doi: 10.1145/3757377.3763931. Article 49.

Vaibhav Vavilala, Rahul Vasanth, and David Forsyth. Denoising Monte Carlo renders with diffusion models. In *International Conference on 3D Vision (3DV)*, pp. 835–844, 2025.

Thijs Vogels, Fabrice Rousselle, Brian McWilliams, Gerhard Röthlin, Alex Harvill, David Adler, Mark Meyer, and Jan Novák. Denoising with kernel prediction and asymmetric loss functions. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 37(4):124:1–124:15, 2018. doi: 10.1145/3197517.3201388.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. VGGT: Visual geometry grounded transformer. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Zhou Wang, Eero P. Simoncelli, and Alan C. Bovik. Multiscale structural similarity for image quality assessment. In *Thirty-Seventh Asilomar Conference on Signals, Systems & Computers*, volume 2, pp. 1398–1402, 2003. doi: 10.1109/ACSSC.2003.1292216.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612, 2004. doi: 10.1109/TIP.2003.819861.

Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth Anything V2. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Jiaqi Yu, Yongwei Nie, Chengjiang Long, Wenju Xu, Qing Zhang, and Guiqing Li. Monte Carlo denoising via auxiliary feature guided self-attention. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, 40(6), 2021. doi: 10.1145/3478513.3480565. Article 273.

Amir R. Zamir, Alexander Sax, William Shen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 586–595, 2018. doi: 10.1109/CVPR.2018.00068.

Chenxi Zhou, Keheng Xu, Mufan Guo, Xianhao Yu, Zhimin Fan, Guihuan Feng, Yanwen Guo, and Jie Guo. DSCombiner: Double shrinkage for combining biased and unbiased Monte Carlo renderings. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, 44(6), 2025. doi: 10.1145/3763315. Article 203.

Rui Zhu, Jiajing Yu, and Tongzhou Zhao. Efficient Monte Carlo denoising via state-space model with low GPU memory overhead. *The Visual Computer*, 41(14):11909–11925, 2025. doi: 10.1007/s00371-025-04134-x.

A KERNEL FOOTPRINTS AND THE SIZE LADDER

Figure 6 shows the effective kernel of DEYES-S-UP4-*FT*.

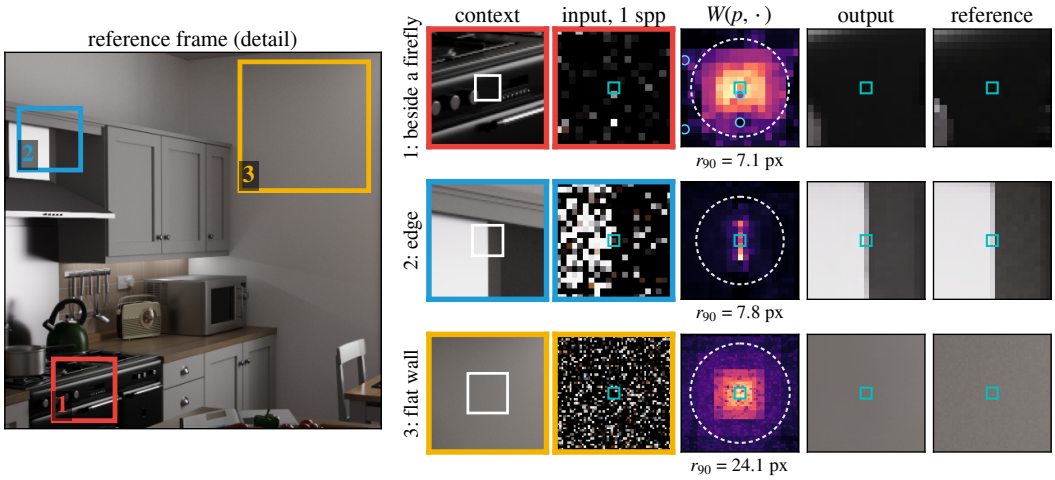

Figure 6: **Kernel footprints of DEYES-S-UP4-*FT*** on *kitchen* at 1 spp. Left: a crop of the reference frame with three regions boxed. Each row follows one output pixel p (cyan square) in one region: beside a firefly, on an edge, and on a flat wall. Columns: the region in the reference (white box: the window shown to its right), then the 1-spp input, $W(p, \cdot)$, the output and the reference over a window of 17, 21 and 57 pixels. $W(p, \cdot)$, drawn on a square-root color scale, is row p of the effective filter: the weight each input pixel receives in output p through the band split, per-level kernels and resampling of all six levels, with the predicted weights held fixed. No single predicted kernel shows it: each level filters a downsampled band, its kernels scatter every source pixel's value to its neighbors, and the learned upsampling spreads each coarse pixel over finer ones, so the weight p receives is gathered from many kernels across levels. With the weights fixed the pyramid is linear in radiance, $y(p) = \sum_q W(p, q)\, x(q)$, so one backward pass gives $W(p, \cdot) = \partial y(p)/\partial x$ exactly. r_{90} (dashed circle) is the radius of the disc around p that holds 90% of the weight. **Beside the firefly** the kernel stays compact ($r_{90} = 7.1$ px) and gives the brightest sample in the window, 940× the local reference, 0.1% of its peak weight (blue circles: samples more than 10× the local reference). **On the edge**, 76% of the weight lies in a 3-pixel strip along the edge. **On the flat wall** it spreads to $r_{90} = 24.1$ px.

Figure 7 is the size ladder of Section 4.3.

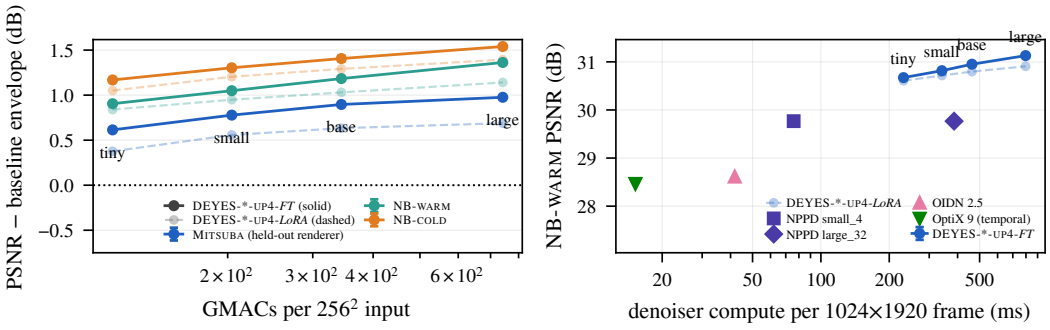

Figure 7: **Size ladder.** Left: mean PSNR margin over the external envelope (MITSUBA: best of OIDN and OptiX per rung; Noisebase: also NPPD) for the four DINOv3 encoder sizes against multiply-accumulates per 256^2 input. Solid: DEYES-*-UP4-*FT* (full fine-tune), 20,000 steps, with the seed range as error bars where a size has several seeds; NB-COLD on every frame. Dashed, light: DEYES-*-UP4-*LoRA* (frozen trunk), trained for the same 20,000 steps. Right: NB-WARM PSNR against denoiser compute time per 1024×1920 frame (Appendix B); both adaptation routes deploy the same network, so they share an x value.

B METRICS, REFERENCES AND LATENCY

Metrics. PSNR, SSIM (Wang et al., 2004) and MS-SSIM (Wang et al., 2003) (`piq` 0.8.0, `data_range`=1) and LPIPS (Zhang et al., 2018) (VGG, `torchmetrics` 1.8.2) are computed in floating point on frames tonemapped with the ACES curve (Arrighetti, 2017) (Hill's RRT+ODT fit), gamma 2.2 and clamped to $[0, 1]$. $\mathcal{L}_1$ is the mean absolute error of linear radiance, and SMAPE is the mean of $|y - R|/(|y| + |R| + 10^{-2})$ over pixels and channels, for output y and reference R; SMAPE is scale-relative and therefore usable on unbounded HDR values, but it is not a perceptual metric. Temporal metrics are computed on lossless 8-bit WebP exports of the tonemapped warm sequences, scaled to $[0, 1]$, with Noisebase's implementation. With $\Delta y_t = y_t - y_{t-1}$ and $\Delta R_t = R_t - R_{t-1}$, tPSNR is the mean over $t \geq 1$ of $10 \log_{10}\big(1/\mathrm{MSE}(\Delta y_t, \Delta R_t)\big)$, and tRMAE the mean over $t \geq 1$ of $\frac{1}{3}\,\mathrm{mean}\big(|\Delta y_t - \Delta R_t|/(|\Delta R_t| + 10^{-2})\big)$, the inner mean over pixels and channels (the factor $\frac{1}{3}$ is Noisebase's). FVVDP is FovVideoVDP v1.2 (`pyfvvdp` 1.2.0), non-foveated, on each whole sequence at 30 frames per second with the `standard_fhd` display model: a 24-inch 1920×1080 sRGB display with 200 cd/m^2 peak luminance and 1000:1 contrast, under 250 lux ambient light, viewed from 0.6 m. Per-sequence scores are averaged with weights proportional to frame count.

OptiX and the linear-radiance metrics. OptiX 9 attenuates extreme HDR peaks. We verified the integration: the AOV model receives float32 linear radiance, albedo and normals; `hdrAverageColor` is computed with `optixDenoiserComputeAverageColor`; `hdrIntensity` is left null, for which OptiX computes the exposure of the input image itself (the per-tile caveat does not apply, as frames are denoised whole); and our Python path matches, bit for bit, an independent C++ harness that calls the OptiX library directly. On *bistro1-day*, whose opening frames contain the sun disc, the reference and OIDN both peak near 9.7×10^5 while OptiX's output peaks near 2×10^4, giving a frame $\mathcal{L}_1$ of 21.8 against OIDN's 0.21 on the same frame; both OptiX model families behave alike. $\mathcal{L}_1$ and SMAPE, computed on linear radiance, therefore penalize OptiX heavily on the few frames with very bright emitters, and its Noisebase $\mathcal{L}_1$ is discontinuous between rungs scored on different frame subsets (280 frames over seven scenes at 32 spp, 1200 over nine otherwise). The tonemapped metrics barely move. Our comparisons against OIDN and NPPD are unaffected.

DMRD's HDR clamp. DMRD clamps linear radiance at $e^{2.2}-1 \approx 8.02$ on input and output; like every method, it is scored against the unclamped reference. The clamp dominates its $\mathcal{L}_1$: on the six interior scenes with more than 1% of reference pixels above 8.02 its $\mathcal{L}_1$ is 8–27$\times$ OIDN's, against 1.5–2.5$\times$ on the 29 scenes with under 0.1%.

Aggregation. MITSUBA numbers are means over the 39 scenes per rung. Means over rungs are arithmetic for PSNR, which corresponds to a geometric mean of MSE, and geometric means of the error ($1-$SSIM, $1-$MS-SSIM, LPIPS, SMAPE, $\mathcal{L}_1$) otherwise. Noisebase numbers are frame-pooled means over the test sequences. Aggregates are recomputed from per-scene files and asserted bit-exact against each baseline's own summary.

Reference policy. Each MITSUBA scene's reference is the mean of n independent 1024-spp renders, with n between 12 and 94 chosen per scene by render cost; for the three scenes whose reference is firefly-dominated (*veach-mis*, *dining-room*, *volumetric-caustic*) the per-pixel median over seeds is used. *volumetric-caustic* is excluded from every aggregate because the unidirectional path tracer we render with cannot sample its caustic beam, so its reference is wrong. Normals use the un-flipped world-space convention that Mitsuba and Falcor share.

Single-frame input on MITSUBA. MITSUBA stills carry no motion vectors, so DEYES scores each as the first frame of a sequence: the accumulation A_t equals x_t, the reprojected previous radiance is zero, and λ and β are masked.

Run-to-run spread. Across our experiments, repeated runs of an identical recipe differed by up to 0.16 dB on the MITSUBA mean; we call this the one-seed floor. The default model's three seeds score 38.917, 38.918 and 38.924 dB on the MITSUBA mean, and the from-scratch network's three seeds at 40,000 steps span 0.12 dB (37.84–37.97).

Envelope. A cell is (protocol, metric, rung). The external value is the best of OIDN 2.5, OptiX 9 and NPPD (best of its four checkpoints on MITSUBA, matched checkpoint on Noisebase); adding DMRD, NRD and StatER changes no cell, as all three sit below that envelope everywhere. The envelope selects a different method per cell and is therefore an oracle comparator, not a single deployable baseline. Production denoisers are evaluated as released; we cannot rule out that their training data include Noisebase or MITSUBA scenes. Figure 10 shows which method owns the envelope in each cell.

Compute time (main text and figures). Every latency in the main text, Table 1 and the x-axes of Figures 3, 8 and 9 is *compute time*: every device-side operation of the denoiser, host–device transfers excluded, for every method, on one A40 (median over the timed frames; Table 2 lists the timer used for each method). For DEYES it includes the temporal preparation (accumulation and motion reprojection of the recurrent state, 7.6 ms warm; padding and state initialization, 4.6 ms cold), because NPPD's step contains its own reprojection. NPPD is its model step on the per-sample buffers at its checkpoint's training spp on Noisebase, and on the pixel-mean buffers broadcast to that count on MITSUBA, whose per-sample buffers are not stored; OIDN 2.5 is its three prefiltered networks; OptiX 9 its execute call (the temporal AOV model on the warm protocol); NRD its ReLAX dispatches, at the GPU's boost clock; DMRD its whole sampler (fp16, compiled, as in its scored run; eager it takes 31.1 s); StatER its filter kernels, separated from its uploads. On MITSUBA every method is timed at each of the ten scene resolutions. At the Noisebase frame size, 1024×1920, the compute times are 341/338 ms (warm/cold) for DEYES-S-UP4-*FT*, 103/100 ms for DEYES-T-UP2-*FT*, 231, 463 and 797 ms for DEYES-T-UP4-*FT*, DEYES-B-UP4-*FT* and DEYES-L-UP4-*FT*, 71, 76, 331 and 386 ms for NPPD's 2-, 4-, 8- and 32-spp checkpoints, 42 ms for OIDN, 11/15 ms for OptiX (AOV/temporal), 3 ms for NRD, 340 ms for StatER and 18.6 s for DMRD. The cost tables of Appendix L use these compute times too. Table 2 gives every method's compute time, host–device transfer time and end-to-end total.

Table 2: **Inference cost** per 1024×1920 frame (one A40, batch 1, no tiling; DEYES and NPPD in bf16, DMRD in fp16, OIDN and OptiX at their own precision), on preloaded frames, median of 50 frames (DMRD 10) after warm-up. Timers: CUDA events for the PyTorch rows (DEYES, NPPD, DMRD), a synchronized host clock around the blocking calls for OIDN and OptiX, Vulkan timestamps for NRD and StatER's own timer. "Warm" and "cold" are compute time as defined above (DEYES includes its temporal preparation); "p90" is the 90th percentile of the warm compute time. "Transfers" are the host–device copies (DEYES: input assembly and readback; NPPD: its per-sample buffers, 32 spp for `large_32`; OIDN: upload and readback; OptiX: host-side packing and copies; NRD: upload, readback and unpacking, excluding the G-buffer encoding; DMRD: its CPU pre- and post-processing and copies), timed with the process pinned to the CPU socket attached to the GPU. "Total" is compute plus transfers. "Peak GB" is the PyTorch allocator's peak during the timed frames (DEYES: the warm path; NPPD: the timed variant; CUDA-graph and caching reserves excluded); OIDN and the OptiX AOV model allocate outside PyTorch, so for them it is the process's device memory during the timed frames minus its CUDA context (the same measurement reproduces OptiX temporal's own figure); OptiX temporal: the device memory OptiX reports; NRD: its resource pool. DEYES rows are the deployed networks: trunk, kernel pyramid and temporal warp compiled, weights in NHWC. ¶A frozen-trunk (*LoRA*) model folds into the same network, so its timings and parameter count are identical and only the trainable count differs (12.7 M for DEYES-S-UP4-*LoRA*). GMACs are at 1024×1920. •OptiX is a closed binary, and NRD and StatER are filters without learned weights, so none has a parameter or MAC count. Cold column: §OIDN is a single-frame method, warm = cold; ‡NPPD's network cost does not depend on history (a cold frame runs the same graph with zeroed state); OptiX AOV is single-frame, and OptiX temporal's cold value is its first-frame path; DMRD carries no temporal state (it trains a 220.6 M ControlNet over a frozen 461.1 M U-Net). ‖NRD is a temporal accumulator with no single-frame mode. It runs through Vulkan with about 100 ms of host work per frame, during which the driver lowers the GPU clock; it is timed with the clock held at its boost value, as in a renderer that keeps the GPU busy, in its default diffuse-and-specular configuration (the diffuse-only configuration we score takes 1.6 ms). °StatER was built and timed on the evaluation host; its own timer spans the upload of its per-pixel statistics, the filter and the download, and a sweep over filter radii separates the filter's compute time from the transfers. Its peak subtracts an estimated CUDA context.

method	warm ms	p90	cold ms	transfers ms	total ms	peak GB	GMACs	params M	trainable M
DEYES-T-UP4-*FT*¶	231.1	231.6	227.3	7.0	238.1	3.4	3468	31.8	31.8
DEYES-S-UP4-*FT*¶ (default)	341.1	342.2	337.7	7.0	348.1	3.5	6119	53.4	53.4
DEYES-B-UP4-*FT*¶	462.7	463.3	457.4	7.1	469.8	4.5	10315	91.6	91.6
DEYES-L-UP4-*FT*¶	797.0	799.0	790.6	7.2	804.2	6.3	22258	200.5	200.5
DEYES-T-UP2-*FT* (interactive)	103.3	103.5	100.3	7.1	110.4	1.6	1190	31.7	31.7
NPPD small_2 (compiled)‡	70.5	70.7	70.5	8.5	79.1	3.3	1366	15.6	15.6
NPPD small_4 (compiled)‡	75.9	76.0	75.9	15.5	91.4	3.7	1375	15.6	15.6
NPPD large_8 (compiled)‡	331.1	332.1	331.1	29.8	360.9	8.5	4020	33.4	33.4
NPPD large_32 (compiled)‡	386.0	387.2	386.0	114.6	500.6	11.8	4131	33.4	33.4
OIDN 2.5 (prefiltered)§	41.7	41.9	41.7	10.6	52.4	0.7	719	2.7	2.7
OptiX 9 (AOV)	11.4	11.4	11.4	52.7	64.1	0.6	n/a•	n/a•	n/a•
OptiX 9 (temporal AOV)	15.2	15.3	14.8	54.8	70.0	0.9	n/a•	n/a•	n/a•
NRD 4.17 (ReLAX)‖	3.2	3.2	–	99.8	102.9	0.1	n/a•	0.0	0.0
DMRD (author checkpoint, fp16, compiled)	–	–	18618.8	117.9	18736.8	9.9	557968	681.7	220.6
StatER (MCL)°	–	–	340.3	16.7	357.0	1.5	n/a•	0.0	0.0

C WHICH PRETRAINING? ENCODER INITIALIZATION

Table 3 and Figure 4 replace the DINOv3 trunk of DEYES-S-UP4-*FT* by other ConvNeXt-S initializations and keep everything else: full fine-tune, 20,000 steps, the same data and augmentation. The recipe, including the 20,000-step horizon, is the one tuned for DINOv3. Two controls are part of the comparison: the network from scratch at the same 20,000 steps, and a DINOv3 trunk with every tensor's elements randomly permuted, which keeps the pretrained weight statistics but destroys their learned structure. The shuffled trunk tests whether that recipe, rather than the prior, carries the gain: it is trained with exactly the fine-tuning settings (gradient clip 4, 20,000 steps, trunk learning rate $\times 0.1$ or $\times 1$; the from-scratch model uses clip 1) and recovers at most 30% of DINOv3's gain on MITSUBA.

SparK (Tian et al., 2023), a masked-reconstruction model trained on ImageNet-1k without labels, matches DINOv3 on both Noisebase protocols (+0.01 and +0.06 dB PSNR) and on the temporal metrics, and trails it by 0.18 dB on MITSUBA, recovering 85% of DINOv3's gain over scratch there. Supervised ImageNet initializations recover 72–77% on MITSUBA but only 40–61% on Noisebase, where they are essentially within the one-seed floor of the from-scratch network (-0.04 to +0.17 dB). On the same 1.28M ImageNet-1k images, the objective matters more than the labels: masked reconstruction recovers 85% on MITSUBA and 101–108% on Noisebase, classification 72% and 59–61%. The shuffled trunk recovers at most 30% on MITSUBA. The gain is therefore mostly a learned natural-image prior rather than a property of DINOv3 or of well-scaled weights; DINOv3's remaining edge shows on the held-out renderer.

Table 3: **Encoder initialization**, DEYES-S-UP4-*FT*'s recipe with only the trunk initialization (and its learning-rate multiplier) changed. MITSUBA: 39 scenes, mean over the ten rungs; NB-WARM and NB-COLD (scored on every frame): mean over 2/4/8/32 spp; $\mathcal{L}_1$ and LPIPS are geometric means over rungs. "Gain recovered" $= (X - \text{scratch 20k})/(\text{DINOv3} - \text{scratch 20k})$ on PSNR, against the step-matched control. Gray beneath a value: half the seed range, for rows with several seeds. Bold best, underline second among the pretrained rows. †Every trunk tensor's elements randomly permuted: weight statistics kept, learned structure destroyed. *Twice the steps of every other row, the best horizon of its compute ladder. Every other row uses the 20,000-step recipe tuned for DINOv3.

		MITSUBA			NB-WARM		NB-COLD	temporal (NB-WARM)		gain recovered		
initialization	trunk LR	PSNR↑	$\mathcal{L}_1\times10^2$ ↓	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	tPSNR↑	FVVDP↑	MITSUBA	NB-WARM	NB-COLD
DINOv3 (LVD-1689M, self-sup.)	×0.1	**38.92** ±0.00	**1.23** ±0.01	**0.122** ±0.001	**30.82** ±0.01	**0.254** ±0.000	30.05 ±0.02	28.47 ±0.01	**7.925** ±0.005	1.00	1.00	1.00
SparK (IN-1k, masked rec.)	×0.1	38.74	1.27	**0.122**	**30.82**	0.255	**30.11**	**28.49**	7.921	0.85	1.01	1.08
IN-22k, labels	×0.1	38.64	1.29	0.125	30.39	0.262	29.65	28.03	7.820	0.77	0.40	0.49
IN-1k, labels	×0.3	38.59	1.29	0.125	30.54	0.259	29.73	28.19	7.864	0.72	0.61	0.59
DINOv3, shuffled†	×1.0	38.08	1.37	0.133	30.17	0.266	29.44	27.82	7.775	0.30	0.08	0.22
DINOv3, shuffled†	×0.1	37.99	1.42	0.130	29.97	0.268	29.20	27.61	7.736	0.23	−0.20	−0.09
scratch, 20k steps	–	37.71	1.44	0.134	30.11	0.262	29.27	27.78	7.779	–	–	–
scratch, 40k steps*	–	37.91 ±0.06	1.43 ±0.02	0.132 ±0.001	30.43 ±0.04	0.257 ±0.001	29.56 ±0.03	28.10 ±0.06	7.840 ±0.015	0.16	0.46	0.38

D TRAINING RECIPE: PRETRAINED AND FROM SCRATCH

The pretraining comparison is only as strong as the from-scratch baseline, so Table 4 lists every setting of the two runs behind it. They share every data augmentation and regularizer; DEYES-S-UP4-*scratch* had its learning rate swept and its training horizon extended to four times the pretrained schedule, where its held-out quality plateaus while its training loss keeps falling (Appendix O).

Table 4: **Training recipe**, DEYES-S-UP4-*FT* against DEYES-S-UP4-*scratch*, the same network with the DINOv3 initialization removed.

	DEYES-S-UP4-*FT*	DEYES-S-UP4-*scratch*
initialization	DINOv3 ConvNeXt-S trunk	random (ConvNeXt default)
trainable	every parameter	every parameter
peak learning rate	1.4×10^{-3}; trunk $\times0.1$	1.4×10^{-3}, every parameter
schedule	cosine, 150 warm-up steps	same
horizon	750–80,000 steps; peak at 20,000 is reported	750–80,000 steps; peak at 40,000 is reported
optimizer	AdamW, β=(0.9, 0.95), ϵ=10^{-6}, weight decay 0.01	same
gradient clip	4 (at 1 it clips 99% of steps)	1 (clips under 1% of steps)
weight averaging	EMA, decay 0.9999	same
batch	64: 16 warm rows, 48 cold	same
loss	8 SMAPE + 0.1 Perc	same
geometric augmentation	one of the 8 flips/rotations per training row, at random	same
data order	sequences shuffled every epoch; cold frames drawn at random	same
sample-count mix	1–32 spp ramp; 19% of rows synthesized at 64–512 spp	same
buffer dropout	p=0.30, uniform over the 15 subsets of depth, normal, albedo, sample-count plane	same
seeds	3	3

E NOISEBASE AT EQUAL SAMPLE COUNT

Noisebase does not ship with render times, so on Noisebase every method is compared at the same sample count, where the render cost is common, against the denoiser's own compute time. Figures 8 and 9 show all four full-fine-tune sizes, the interactive model and the from-scratch network; Figure 10 sets every full-fine-tune model's error against the per-cell envelope on all three protocols and shows which baseline owns each cell.

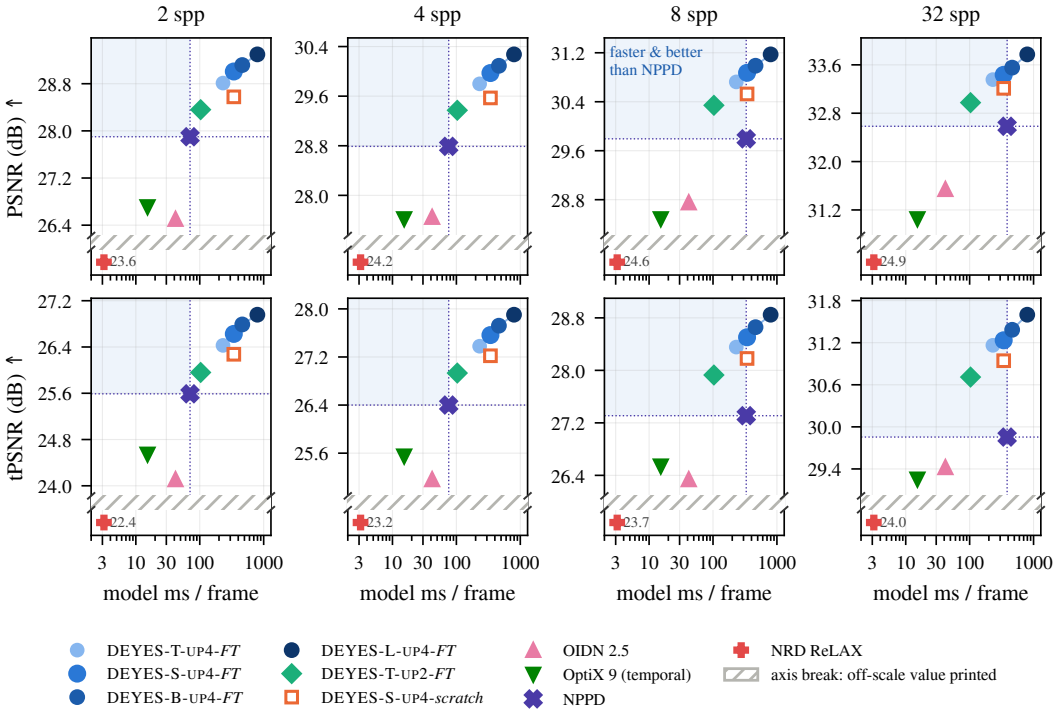

Figure 8: **Quality against denoiser compute, NB-WARM** (recurrent state carried; 1200 frames at 2/4/8 spp, 280 at 32). x: denoiser compute time per 1024×1920 frame on one A40 (Appendix B). Shaded: faster and better than NPPD at that rung. NPPD uses the checkpoint trained for each rung (`small_2`, `small_4`, `large_8`, `large_32`; 71, 76, 331 and 386 ms). DEYES-{T,S,B,L}-UP4-*FT*, DEYES-T-UP2-*FT* and DEYES-S-UP4-*scratch*; methods far off the quality range are printed beyond the axis break. Every DEYES model is better than every baseline at every rung, and the tiny models also run faster than NPPD at 8 and 32 spp, the small model at 32 spp.

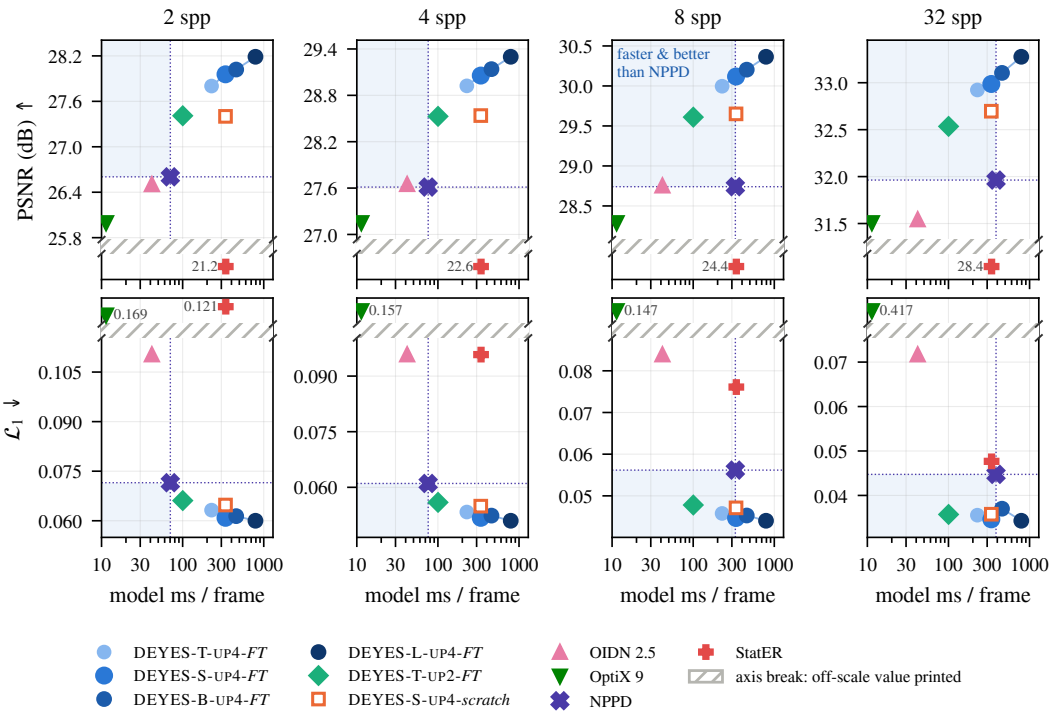

Figure 9: **Quality against denoiser compute, NB-COLD**: as Figure 8 with the state reset before every frame, PSNR and $\mathcal{L}_1$. Every DEYES model is better than every baseline at every rung, and the tiny models also run faster than NPPD at 8 and 32 spp, the small model at 32 spp.

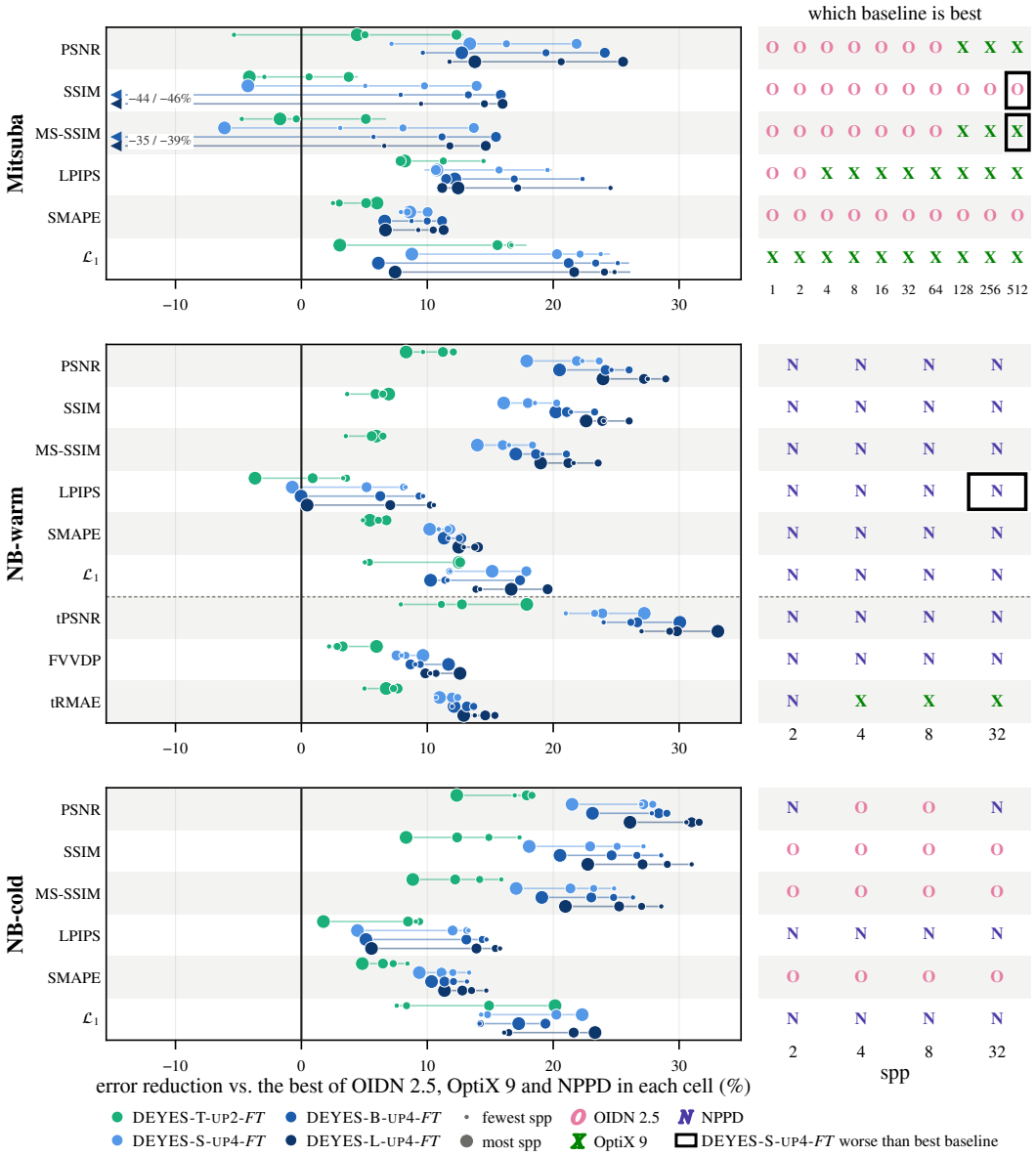

Figure 10: **DEYES beats the best baseline in almost every cell.** Left: error reduction $1 - e_{DEYES}/e_{\text{env}}$ against the envelope, the best of OIDN 2.5, OptiX 9 and NPPD in each (protocol, metric, sample count) cell; positive means DEYES is better. Lanes, top to bottom: the interactive model (tiny, up2) and the small (default), base and large up4 models, all fully fine-tuned. Marker size grows with the sample count (MITSUBA 1/8/64/512 spp, Noisebase 2/4/8/32); arrowheads mark off-scale values. The default model wins 105 of the 108 spatial cells and all 12 temporal cells; its three losses are at the clean end and narrow (MITSUBA SSIM 0.9967 vs. 0.9968 and MS-SSIM 0.9958 vs. 0.9961 at 512 spp, NB-WARM LPIPS 0.254 vs. 0.253 at 32 spp). Base and large lose five and four cells, the interactive model eleven, mostly at 1–8 spp; every model wins every temporal and every NB-COLD cell. Right: the baseline that sets the envelope (O OIDN, X OptiX, N NPPD); a black frame marks a cell the default model loses. No single baseline is the envelope: OIDN owns most MITSUBA cells, NPPD most Noisebase cells. e is the mean-squared error implied by PSNR and tPSNR, $1-$SSIM, $1-$MS-SSIM, $10-$JOD for FVVDP and the metric itself otherwise, so values are reductions in that quantity, not perceptual percentages.

F ENCODER UPSCALE AT A FIXED ENCODER SIZE

Figure 11 isolates the encoder's input upscale. Every model is the default recipe (full fine-tune, 20,000 steps) with only the encoder size and upscale changed. For small, each octave of upscale adds 0.3–0.7 dB at every MITSUBA rung and lowers $\mathcal{L}_1$ at every rung; for tiny the step from up2 to up4 shrinks at the clean end ($+0.44$ dB at 1 spp, $+0.12$ at 512, where $\mathcal{L}_1$ is unchanged); at up1, tiny and small are equal (37.91 and 37.89 dB), so at the cheap end the upscale, not the encoder size, sets quality. On NB-WARM the same ordering holds (tiny: 29.95, 30.24, 30.67 dB for up1, up2, up4; small: 30.20 and 30.82 for up2 and up4). Compute time at 1024×1920: DEYES-T-UP2-*FT* 103 ms, DEYES-T-UP4-*FT* 231 ms, DEYES-S-UP4-*FT* 341 ms. The alternative to upsampling the input is to restride the pretrained stem; in an earlier ablation on the frozen-trunk recipe, a stride-2 stem lost 0.47 dB at 1 spp against up4.

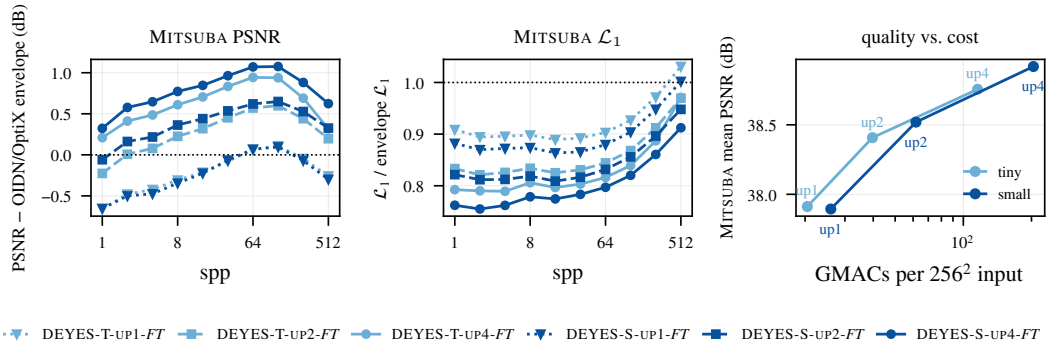

Figure 11: **Encoder upscale at a fixed size**, MITSUBA, full fine-tune at 20,000 steps. Left and center: PSNR margin over and $\mathcal{L}_1$ relative to the OIDN/OptiX envelope per rung. Right: mean PSNR against multiply-accumulates per 256^2 input (tiny: 20, 40, 116 GMACs; small: 26, 62, 204). Seeds: tiny up1/up2/up4 2/3/1, small 2/2/3.

G BUFFER REMOVAL

Table 5 removes each input of the default model at test time at every MITSUBA rung, beside OIDN 2.5, OptiX 9 and NPPD with the inputs they always receive. Over the default model's three seeds, albedo + normal only costs -0.03 dB and stays above the envelope in all 60 MITSUBA (metric, rung) cells; with radiance alone the model still exceeds the envelope on mean PSNR (at every rung from 8 to 256 spp) and on $\mathcal{L}_1$ at every rung up to 256 spp, and at 512 spp ties it on both (-0.0002 dB; $+0.06\%$ error). Table 6 repeats the removal on NB-COLD for the frozen-trunk model: every geometry buffer matters about twice as much on the training renderer, and without all three the model falls below the NB-COLD envelope.

Table 5: **Test-time input removal**, DEYES-S-UP4-*FT*, MITSUBA, mean of three seeds. Top: absolute PSNR of the baselines with their usual inputs (OIDN, OptiX: albedo and normal; NPPD†: best of its four checkpoints per rung, pixel-mean buffers) and their per-rung envelope. Middle: DEYES-S-UP4-*FT* with all inputs, then ΔPSNR per removed input. "Albedo+normal only" withholds depth and replaces the sample-count plane by the learned "unknown" value: the input OIDN and OptiX receive. Δenv: the row's ten-rung mean minus the envelope's. Then the margin over the envelope per rung. Lower block: the same conditions on $\mathcal{L}_1$; baselines, envelope and the all-inputs row are absolute ($\times 10^2$), condition rows the relative change in error against all inputs (%, negative = more error), Δenv and margins the error reduction against the envelope; means over rungs are geometric.

MITSUBA PSNR ↑	1	2	4	8	16	32	64	128	256	512	mean_{10}	Δenv
OIDN 2.5	31.87	33.28	34.62	36.02	37.42	38.83	40.18	41.49	42.79	44.01	38.05	-0.09
OptiX 9	31.21	32.78	34.15	35.60	37.04	38.51	40.01	41.56	43.07	44.57	37.85	-0.29
NPPD†	30.78	32.26	33.51	34.66	35.99	37.05	37.85	38.59	39.63	41.31	36.16	-1.98
envelope	31.87	33.28	34.62	36.02	37.42	38.83	40.18	41.56	43.07	44.57	38.14	
DEYES-S-UP4-*FT*, all inputs	32.19	33.86	35.26	36.79	38.27	39.79	41.25	42.64	43.95	45.19	38.92	+0.78
−depth	+0.00	+0.00	+0.01	+0.01	+0.01	+0.01	+0.00	+0.00	+0.00	-0.00	+0.004	+0.78
−normal	-0.22	-0.26	-0.26	-0.22	-0.21	-0.18	-0.17	-0.17	-0.19	-0.22	-0.211	+0.57
−albedo	-0.29	-0.32	-0.34	-0.35	-0.36	-0.40	-0.42	-0.46	-0.53	-0.60	-0.407	+0.37
−depth −normal −albedo	-0.72	-0.75	-0.75	-0.70	-0.67	-0.67	-0.65	-0.68	-0.77	-0.86	-0.723	+0.06
albedo+normal only	-0.03	-0.00	+0.02	+0.03	+0.03	+0.00	-0.04	-0.07	-0.08	-0.15	-0.029	+0.75
−sample-count plane	-0.03	-0.01	+0.02	+0.03	+0.03	+0.00	-0.04	-0.07	-0.08	-0.16	-0.030	+0.75
radiance only	-0.75	-0.76	-0.76	-0.72	-0.69	-0.70	-0.68	-0.65	-0.60	-0.62	-0.694	+0.08
margin: albedo+normal only − env.	+0.29	+0.58	+0.67	+0.81	+0.87	+0.97	+1.03	+1.01	+0.81	+0.47	+0.75	
margin: radiance only − env.	-0.42	-0.19	-0.11	+0.05	+0.15	+0.26	+0.39	+0.43	+0.28	-0.00	+0.08	
MITSUBA $\mathcal{L}_1$ ↓	1	2	4	8	16	32	64	128	256	512	mean_{10}	Δenv
OIDN 2.5 ($\times 10^2$)	3.81	3.12	2.57	2.09	1.74	1.46	1.23	1.05	0.90	0.78	1.65	-7%
OptiX 9 ($\times 10^2$)	3.65	2.97	2.43	1.99	1.67	1.39	1.16	0.96	0.80	0.66	1.53	+0%
NPPD† ($\times 10^2$)	3.68	2.99	2.51	2.12	1.82	1.58	1.41	1.28	1.09	0.99	1.78	-16%
envelope ($\times 10^2$)	3.65	2.97	2.43	1.99	1.67	1.39	1.16	0.96	0.80	0.66	1.53	
DEYES-S-UP4-*FT*, all inputs ($\times 10^2$)	2.78	2.25	1.85	1.55	1.30	1.09	0.93	0.79	0.69	0.60	1.23	+20%
−depth	+0.2	+0.1	+0.2	+0.3	+0.3	+0.4	+0.4	+0.3	+0.2	+0.1	+0.3	+20%
−normal	-0.9	-0.8	-0.8	-0.3	-0.5	-0.7	-0.7	-0.8	-1.1	-2.4	-0.9	+19%
−albedo	-6.6	-7.6	-7.9	-8.7	-9.6	-10.8	-11.0	-11.0	-10.5	-11.7	-9.5	+12%
−depth −normal −albedo	-9.2	-10.7	-10.6	-10.4	-11.1	-11.6	-11.0	-10.6	-10.5	-13.5	-10.9	+11%
albedo+normal only	+0.2	+0.4	+0.5	+0.7	+0.8	+0.6	+0.1	-0.6	-1.2	-2.2	-0.1	+20%
−sample-count plane	-0.0	+0.3	+0.4	+0.5	+0.5	+0.2	-0.3	-1.0	-1.6	-2.7	-0.4	+20%
radiance only	-9.1	-10.1	-10.1	-9.9	-10.5	-11.2	-11.2	-10.8	-10.2	-9.7	-10.3	+12%
margin: albedo+normal only vs. env.	+24%	+25%	+24%	+23%	+23%	+22%	+20%	+17%	+13%	+7%	+20%	
margin: radiance only vs. env.	+17%	+17%	+16%	+14%	+14%	+13%	+11%	+9%	+5%	-0%	+12%	

Table 6: Input removal on NB-COLD, DEYES-S-UP4-*LoRA* (state reset per frame), laid out as Table 5. NB-WARM is not scored under removal because the temporal state would carry the removed buffer's influence across frames.

NB-COLD PSNR ↑	2	4	8	32	mean_4	Δenv
OIDN 2.5	26.50	27.65	28.75	31.54	28.61	-0.13
OptiX 9	25.99	27.15	28.29	31.51	28.23	-0.51
NPPD†	26.60	27.61	28.74	31.96	28.73	-0.01
envelope	26.60	27.65	28.75	31.96	28.74	
DEYES-S-UP4-*LoRA*, all inputs	27.74	28.82	29.89	32.86	29.83	+1.09
−depth	-0.02	-0.02	-0.02	-0.02	-0.021	+1.06
−normal	-0.89	-0.86	-0.82	-0.63	-0.802	+0.28
−albedo	-0.85	-0.87	-0.88	-0.73	-0.831	+0.25
−depth −normal −albedo	-1.99	-1.89	-1.79	-1.35	-1.758	-0.67
−sample-count plane	-0.18	-0.08	-0.02	-0.00	-0.068	+1.02
radiance only	-2.00	-1.90	-1.78	-1.36	-1.761	-0.68
margin: −albedo − env.	+0.29	+0.30	+0.26	+0.17	+0.25	
margin: radiance only − env.	-0.86	-0.73	-0.64	-0.47	-0.68	

H KNOCK-OUT AND SPECTRA

Figure 12 reverts one trunk stage of the default model at a time to its DINOv3 weights at test time and re-scores ten MITSUBA scenes; the auxiliary tower, laterals and decoder keep their fine-tuned weights. Reverting stage 2 (27 of the 36 blocks, at native $/4$) costs 5.2–10.7 dB, stages 0 and 1 ($/1$, $/2$) cost 1.5–5.1 dB, and these costs grow with sample count, because at 1 spp the kernel mostly averages while at 512 spp its placement must be exact; stage 3 ($/8$) costs 1.1 dB at 1 spp and almost nothing at 512. The fine-tuning update is close to full rank in stages 0–2 and lower-rank in stage 3 (Figure 12, right); every update's effective rank, and the number of singular directions that hold 99% of its energy, exceed the 64 a rank-64 LoRA adapter could express. A knock-out measures how much the rest of the network depends on a stage's adaptation, not what that adaptation contributes to a model trained without it, and stage 2 holds most of the trunk's parameters. The frozen-trunk

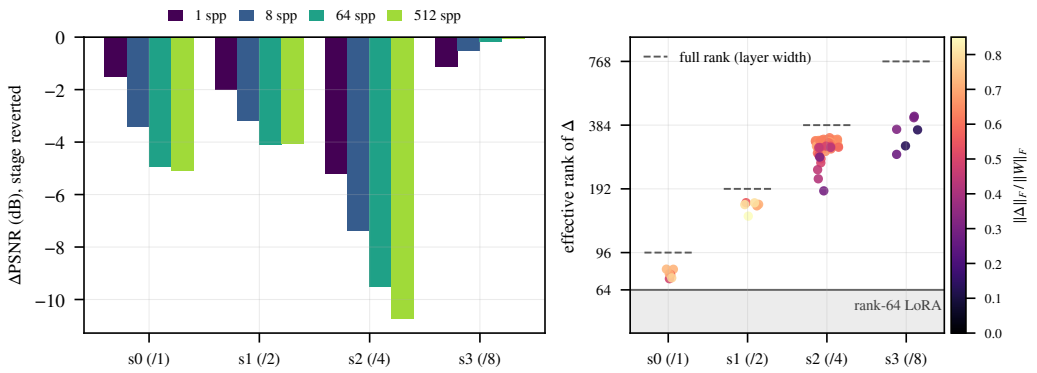

Figure 12: **Where DEYES-S-UP4-*FT* depends on its fine-tuning** (the default full fine-tune). Left: change in mean PSNR over ten MITSUBA scenes when one trunk stage (x-axis; native stride in parentheses) is reverted to its DINOv3 weights at test time, all other weights kept, at 1, 8, 64 and 512 spp (bars). Right: one marker per pointwise (1×1) layer of the trunk, 72 in all, at its stage on the x-axis; y is the effective rank of that layer's fine-tuning update $\Delta = W_{\mathrm{FT}} - W_{\mathrm{DINOv3}}$, the exponential of the entropy of its normalized singular values (log scale), and color is the update's size relative to the pretrained weight. Dashed: the largest rank the layer allows (its width); shaded: ranks a rank-64 LoRA update can reach. The network depends most on stage 2 and its dependence grows with sample count; every update lies above the shaded band, and the median update is at 79–84% of full rank in stages 0–2 and at 48% in stage 3.

model (DEYES-S-UP4-*LoRA*, rank-64 LoRA updates) shows the same ordering when each stage's LoRA update is zeroed at test time instead of reverted. On the same ten scenes, zeroing stage 2 costs 3.4/4.4/5.3/5.8 dB at 1/8/64/512 spp, stage 0 0.7/1.4/2.4/3.5, stage 1 1.2/1.9/1.8/2.4, and stage 3 is within 0.1 dB of zero; a second seed replicates the ordering and the growth with sample count (stage 2: 2.9/3.8/4.6/4.6 dB), and on NB-COLD at 32 spp (control 32.86 dB) the four stages cost -1.58, -1.49, -4.45 and -0.05 dB. The ordering is therefore not specific to full fine-tuning, and it holds on the training renderer as well.

I WHERE PRETRAINING LOWERS THE ERROR: VARIANCE AND BIAS

Figure 13 splits the error gap between DEYES-S-UP4-*FT* (FT below) and DEYES-S-UP4-*scratch* (scratch) into variance and bias. Each of six MITSUBA scenes (*lego*, *kitchen*, *veach-ajar*, *glass-of-water*, *curly-hair*, *staircase2*) is rendered 16 times at 4 and at 64 spp with disjoint sample seeds, radiance and auxiliary buffers alike, and both models denoise each render independently as a single frame without history. Per pixel, the variance V is the spread of a model's 16 outputs, that is, how much its result changes when only the sampling noise of its input changes, and the squared bias B^2 is the distance of their mean from the reference. For one model, a pixel-channel p and its $R = 16$

denoised outputs $y_{r,p}$,

$$m_p = \frac{1}{R}\sum_{r=1}^{R} T(y_{r,p}), \qquad V_p = \frac{1}{R-1}\sum_{r=1}^{R}\big(T(y_{r,p}) - m_p\big)^2, \qquad B_p^2 = \big(m_p - T(\mathrm{ref}_p)\big)^2 - \frac{V_p}{R}, \tag{3}$$

where T is the tone mapping in which PSNR is computed (ACES, $\gamma = 2.2$, clamped to $[0, 1]$). The mean m_p still carries V_p/R of residual noise, which $(m_p - T(\mathrm{ref}_p))^2$ alone would count as bias; the last term removes it. Scene-level values (Figure 13b,c) average V_p and B_p^2 over all pixels and channels, and subtract from B^2 the reference's own noise, estimated as $F = \mathrm{mean}_p \mathrm{Var}_i[T(\mathrm{ref}_p^{(i)})]/N$ from the N independent 1024-spp renders $\mathrm{ref}^{(i)}$ that the reference averages; F cancels in every scratch-minus-FT difference. $V + B^2 + F$ is then exactly the mean-squared error over the 16 renders, the quantity behind PSNR. The crops were fixed before either model was run; the two scenes shown follow the rule in the caption.

Pretraining lowers the mean-squared error in all 12 scene–sample-count pairs. At 64 spp the gap is mostly variance: scratch's outputs vary more between renders in all six scenes (1.40× on average, from 1.03× on *lego* to 1.80× on *veach-ajar*). At 4 spp it is mostly bias, in four of six scenes: scratch's average output is further from the reference. The extra error concentrates at edges and in fine texture, such as the slatted radio grille in the *kitchen* crop, and on *veach-ajar* in dark regions. The two components do not always move together: in four of the 12 pairs, scratch has less bias but more variance (*curly-hair* at both sample counts, *glass-of-water* at 64 spp) or the reverse (*glass-of-water* at 4 spp). These readings are in the display space in which PSNR is computed. In linear radiance relative to the reference, $y/(\mathrm{ref} + 0.01)$, which weights dark pixels far more heavily, the direction varies between scenes (FT has lower variance in four of six, lower bias in three).

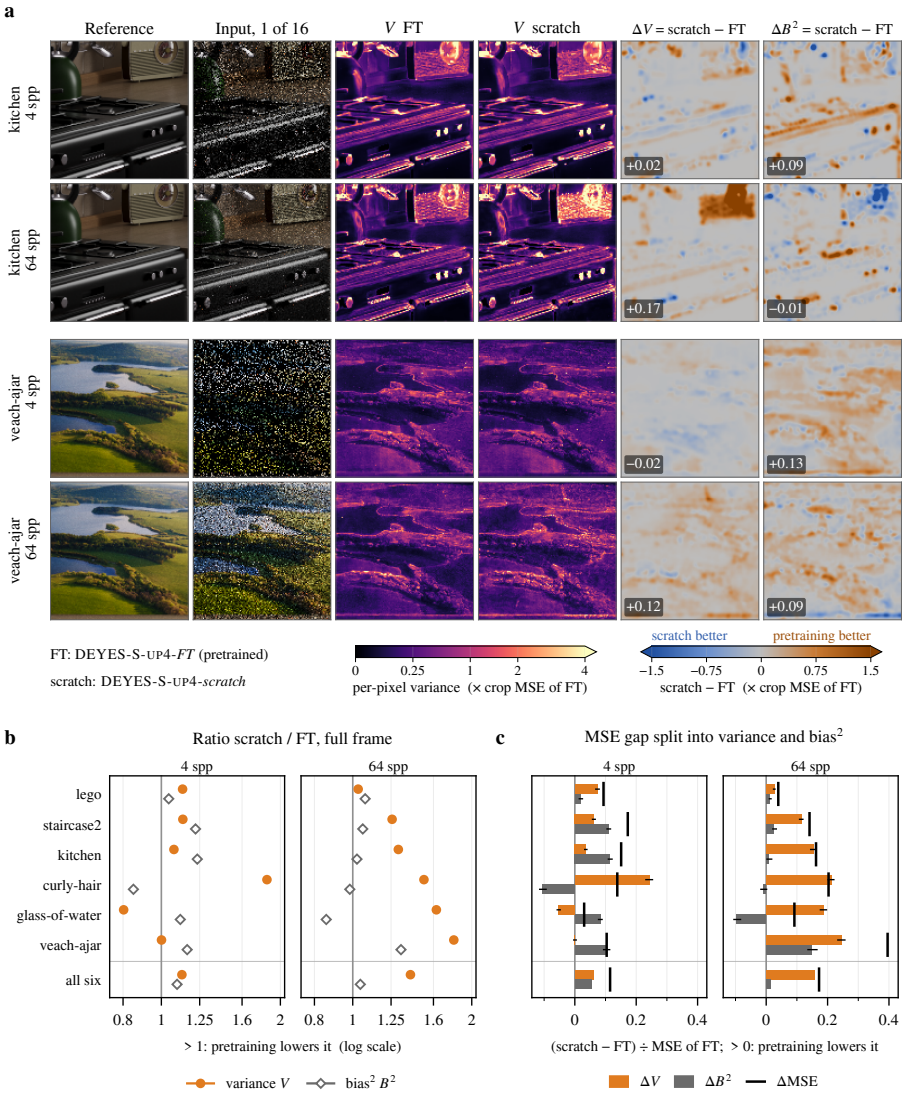

Figure 13: **Where the from-scratch network's extra error comes from.** Each scene is rendered 16 times with independent sample seeds and each render is denoised independently by DEYES-S-UP4-*FT* (FT, pretrained) and DEYES-S-UP4-*scratch* (scratch, the same network without pretraining). Both models denoise the same 16 renders, so the input noise is common to both. V: per-pixel variance of a model's 16 *denoised* outputs; B^2: squared bias of their mean against the reference; both in the tone-mapped PSNR space, as defined in Eq. 3. The input noise itself is much larger (on *kitchen* at 4 spp, a variance about 180 times FT's) and is shown only as one input render in column 2. Every difference is *scratch minus FT*, so a positive value, orange in the maps, means the from-scratch network has more error and pretraining is better; negative values, blue, mean the reverse. (a) *kitchen* (the scene whose 64-spp variance ratio is closest to the six-scene median) and *veach-ajar* (the largest ratio), at 4 and 64 spp; each crop is the 128^2 window of largest reference gradient, chosen before any model was run. Variance maps and differences are in units of the crop's FT mean-squared error; difference maps are smoothed ($\sigma = 1.5$ px) for display. The inset number is the crop mean of that difference: $+0.17$ under ΔV means the from-scratch network's variance exceeds FT's by 17% of FT's total error on that crop. (b) Full-frame ratio scratch/FT for all six scenes with 95% bootstrap intervals over the 16 renders (mostly narrower than the markers); a ratio above 1 means pretraining lowers that component; "all six" is the geometric mean: at 4 spp scratch has more variance ($\times 1.127$) and more bias^2 ($\times 1.096$), each in 5/6 scenes; at 64 spp more variance ($\times 1.399$, 6/6) and slightly more bias^2 ($\times 1.045$, 4/6). (c) The mean-squared-error gap split into its two parts, $\Delta V + \Delta B^2 = \Delta\text{MSE}$, each as (scratch $-$ FT)/FT's error; bars right of zero favor pretraining.

J FULL PER-METRIC TABLES

Tables 8–10 list every (protocol, metric, rung) cell for every method: PSNR, SSIM, MS-SSIM, LPIPS, SMAPE and $\mathcal{L}_1$ on the MITSUBA ladder, NB-WARM and NB-COLD, one table per protocol with a subtable per metric. DEYES rows are the full fine-tune at 20,000 steps at four sizes and DEYES-T-UP2-*FT*, with DEYES-S-UP4-*LoRA* and DEYES-S-UP4-*scratch* as unranked reference rows. Table 7 condenses them.

Table 7: **All models, condensed.** NB-WARM and NB-COLD at 2/8/32 spp and the 39-scene MITSUBA ladder (PSNR at 1/8/64/512 spp and the mean over ten rungs). Ranked block: *FT* rows, 20,000 steps (seeds as in Table 8); bold best, underlined second. †Best of NPPD's four checkpoints per column on MITSUBA (pixel-mean inputs), matched checkpoint on Noisebase; ‡StatER: no temporal state, and MITSUBA lacks the per-sample statistics it needs; §DMRD: author-supplied Noisebase-trained checkpoint, no temporal state. The *LoRA* rows (frozen trunk, 10,000 steps, three seeds) and *DEYES-S-UP4-*scratch* (40,000 steps, the best horizon of its compute ladder, three seeds) are not ranked; the distance from DEYES-S-UP4-*scratch* to DEYES-S-UP4-*FT* is what pretraining adds to the same network (1.01 dB on the MITSUBA mean here, with EMA weights; 1.04 dB between the best points of the two compute ladders, Section 4.2).

	NB-WARM (Falcor, 1024×1920, temporal)									NB-COLD		MITSUBA, 39 scenes, PSNR ↑				
	PSNR ↑			SSIM ↑			LPIPS ↓			PSNR ↑						
spp	2	8	32	2	8	32	2	8	32	2	32	1	8	64	512	mean_{10}
OIDN 2.5	26.50	28.75	31.54	0.908	0.950	0.976	0.346	0.297	0.288	26.50	31.54	31.87	36.02	40.18	44.01	38.05
OptiX 9	26.71	28.48	31.05	0.919	0.955	0.978	0.360	0.310	0.294	25.99	31.51	31.21	35.60	40.01	44.57	37.85
NPPD†	27.90	29.79	32.58	0.938	0.963	0.981	0.291	0.257	0.253	26.60	31.96	30.78	34.66	37.85	41.31	36.16
StatER‡	–	–	–	–	–	–	–	–	–	21.22	28.39	–	–	–	–	–
NRD (ReLAX)	23.62	24.56	24.85	0.879	0.903	0.914	0.419	0.382	0.386	22.05	23.26	25.16	27.43	27.94	28.07	27.35
DMRD§	–	–	–	–	–	–	–	–	–	23.84	29.31	28.81	33.50	36.07	37.09	34.16
DEYES-T-UP4-*FT*	28.81	30.72	33.36	0.947	0.968	0.984	0.269	0.244	0.255	27.80	32.92	32.08	36.63	41.12	44.88	38.76
DEYES-S-UP4-*FT*	29.00	30.87	33.44	0.950	0.969	0.984	0.267	0.243	0.254	27.97	33.01	32.19	36.79	41.25	45.19	38.92
DEYES-B-UP4-*FT*	29.13	31.00	33.58	0.951	0.970	0.985	0.262	0.241	0.253	28.06	33.14	32.31	36.96	41.38	45.16	39.04
DEYES-L-UP4-*FT*	**29.30**	**31.17**	**33.77**	**0.953**	**0.972**	**0.985**	**0.260**	**0.239**	**0.251**	**28.19**	**33.28**	**32.41**	**37.02**	**41.46**	**45.21**	**39.12**
DEYES-T-UP2-*FT* (interactive)	28.34	30.31	32.96	0.940	0.965	0.982	0.281	0.254	0.262	27.41	32.52	31.64	36.25	40.75	44.76	38.41
DEYES-T-UP4-*LoRA*	28.53	30.45	33.07	0.943	0.966	0.982	0.276	0.251	0.262	27.50	32.67	31.77	36.37	40.80	44.75	38.48
DEYES-S-UP4-*LoRA*	28.75	30.62	33.18	0.947	0.967	0.983	0.271	0.247	0.259	27.72	32.83	32.00	36.56	41.01	44.92	38.68
DEYES-B-UP4-*LoRA*	28.79	30.70	33.31	0.947	0.968	0.983	0.270	0.247	0.259	27.80	32.94	32.16	36.75	41.16	45.02	38.84
DEYES-L-UP4-*LoRA*	28.91	30.79	33.37	0.949	0.969	0.983	0.268	0.245	0.258	27.94	33.02	32.18	36.79	41.21	45.04	38.88
DEYES-S-UP4-*scratch**	28.54	30.49	33.16	0.946	0.967	0.983	0.271	0.246	0.254	27.39	32.67	31.49	36.06	40.35	43.63	37.91

Table 8: **Every MITSUBA cell** (39 scenes, single frame), per metric and rung. DEYES rows are the full fine-tune at 20,000 steps, the mean over their seeds (DEYES-T-UP4-*FT* one, DEYES-S-UP4-*FT* three, DEYES-B-UP4-*FT* two, DEYES-L-UP4-*FT* one, DEYES-T-UP2-*FT* three) with half the seed range in gray beneath (also for DEYES-S-UP4-*scratch*). Last column: mean over rungs for PSNR, geometric mean of the error ($1-$SSIM, $1-$MS-SSIM, LPIPS, SMAPE, $\mathcal{L}_1$) otherwise. Bold best, underline second among the ranked rows; DEYES-S-UP4-*LoRA* (10,000 steps, three seeds) and DEYES-S-UP4-*scratch** (40,000 steps, the peak of its compute ladder, three seeds) are unranked. †NPPD: best of its four checkpoints per rung on MITSUBA (pixel-mean inputs), matched checkpoint on Noisebase; ‡StatER: cold only; §DMRD: no temporal state, cold only.

MITSUBA PSNR ↑	1	2	4	8	16	32	64	128	256	512	mean
OIDN 2.5	31.87	33.28	34.62	36.02	37.42	38.83	40.18	41.49	42.79	44.01	38.05
OptiX 9	31.21	32.78	34.15	35.60	37.04	38.51	40.01	41.56	43.07	44.57	37.85
NPPD†	30.78	32.26	33.51	34.66	35.99	37.05	37.85	38.59	39.63	41.31	36.16
NRD (ReLAX)	25.16	26.29	26.99	27.43	27.69	27.85	27.94	28.01	28.05	28.07	27.35
DMRD§	28.81	30.60	32.16	33.50	34.58	35.46	36.07	36.53	36.86	37.09	34.16
DEYES-T-UP4-*FT*	32.08	33.69	35.10	36.63	38.13	39.66	41.12	42.50	43.76	44.88	38.76
DEYES-S-UP4-*FT*	32.19 ±0.01	33.86 ±0.01	35.26 ±0.01	36.79 ±0.03	38.27 ±0.02	39.79 ±0.02	41.25 ±0.01	42.64 ±0.01	43.95 ±0.04	45.19 ±0.04	38.92
DEYES-B-UP4-*FT*	32.31 ±0.03	34.00 ±0.02	35.42 ±0.03	36.96 ±0.02	38.44 ±0.02	39.94 ±0.02	41.38 ±0.00	42.76 ±0.00	44.01 ±0.05	45.16 ±0.05	39.04
DEYES-L-UP4-*FT*	**32.41**	**34.11**	**35.53**	**37.02**	**38.51**	**40.02**	**41.46**	**42.82**	**44.08**	**45.21**	**39.12**
DEYES-T-UP2-*FT*	31.64 ±0.04	33.29 ±0.03	34.69 ±0.02	36.25 ±0.02	37.74 ±0.02	39.28 ±0.03	40.75 ±0.03	42.16 ±0.04	43.51 ±0.06	44.76 ±0.08	38.41
DEYES-S-UP4-*LoRA*	32.00	33.64	35.04	36.56	38.03	39.55	41.01	42.39	43.69	44.92	38.68
DEYES-S-UP4-*scratch**	31.49 ±0.06	33.13 ±0.07	34.53 ±0.05	36.06 ±0.04	37.57 ±0.05	39.08 ±0.07	40.35 ±0.15	40.99 ±0.09	42.25 ±0.09	43.63 ±0.11	37.91

(a)

MITSUBA SSIM ↑	1	2	4	8	16	32	64	128	256	512	err.
OIDN 2.5	0.956	0.967	0.975	0.981	0.986	0.990	0.993	0.994	0.996	**0.997**	0.012
OptiX 9	0.952	0.964	0.973	0.980	0.985	0.989	0.992	0.994	0.996	0.997	0.013
NPPD†	0.945	0.958	0.968	0.975	0.980	0.985	0.988	0.990	0.992	0.993	0.018
NRD (ReLAX)	0.788	0.827	0.854	0.874	0.888	0.898	0.904	0.908	0.911	0.912	0.118
DMRD§	0.889	0.912	0.928	0.937	0.941	0.946	0.949	0.951	0.952	0.953	0.062
DEYES-T-UP4-*FT*	0.956	0.968	0.976	0.983	0.987	0.991	0.993	0.995	0.996	0.995	0.012
DEYES-S-UP4-*FT*	0.958 ±0.000	0.969 ±0.000	0.977 ±0.000	0.983 ±0.000	0.988 ±0.000	0.991 ±0.000	0.994 ±0.000	0.995 ±0.000	**0.996** ±0.000	0.997 ±0.000	**0.011**
DEYES-B-UP4-*FT*	0.959 ±0.000	0.970 ±0.000	0.978 ±0.000	0.984 ±0.000	0.988 ±0.000	0.991 ±0.000	0.994 ±0.000	**0.995** ±0.000	0.995 ±0.000	0.995 ±0.000	0.011
DEYES-L-UP4-*FT*	**0.960**	**0.971**	**0.978**	**0.984**	**0.988**	**0.992**	**0.994**	0.995	0.996	0.995	0.011
DEYES-T-UP2-*FT*	0.955 ±0.000	0.966 ±0.000	0.975 ±0.000	0.981 ±0.000	0.986 ±0.000	0.990 ±0.000	0.993 ±0.000	0.995 ±0.000	0.996 ±0.000	0.997 ±0.000	0.012
DEYES-S-UP4-*LoRA*	0.956	0.968	0.976	0.982	0.987	0.991	0.993	0.995	0.996	0.996	0.011
DEYES-S-UP4-*scratch**	0.952 ±0.001	0.964 ±0.001	0.973 ±0.000	0.980 ±0.000	0.986 ±0.000	0.990 ±0.000	0.991 ±0.002	0.982 ±0.002	0.985 ±0.001	0.989 ±0.001	0.018

(b)

MITSUBA MS-SSIM ↑	1	2	4	8	16	32	64	128	256	512	err.
OIDN 2.5	0.962	0.971	0.977	0.982	0.987	0.990	0.992	0.994	0.995	0.996	0.012
OptiX 9	0.958	0.968	0.976	0.981	0.986	0.989	0.992	0.994	0.995	**0.996**	0.012
NPPD†	0.953	0.964	0.971	0.977	0.982	0.985	0.988	0.990	0.991	0.993	0.017
NRD (ReLAX)	0.825	0.857	0.880	0.896	0.907	0.915	0.920	0.923	0.925	0.926	0.098
DMRD§	0.931	0.948	0.961	0.970	0.975	0.980	0.983	0.985	0.986	0.988	0.025
DEYES-T-UP4-*FT*	0.961	0.971	0.978	0.983	0.987	0.991	0.993	0.994	0.995	0.994	0.012
DEYES-S-UP4-*FT*	0.963 ±0.000	0.972 ±0.000	0.979 ±0.000	0.984 ±0.000	0.988 ±0.000	0.991 ±0.000	0.993 ±0.000	0.994 ±0.000	**0.995** ±0.000	0.996 ±0.000	**0.011**
DEYES-B-UP4-*FT*	0.964 ±0.000	0.973 ±0.000	0.980 ±0.000	0.984 ±0.000	0.988 ±0.000	**0.991** ±0.000	**0.993** ±0.000	**0.995** ±0.000	0.995 ±0.000	0.995 ±0.000	0.011
DEYES-L-UP4-*FT*	**0.964**	**0.973**	**0.980**	**0.984**	**0.988**	0.991	0.993	0.994	0.995	0.995	0.011
DEYES-T-UP2-*FT*	0.960 ±0.000	0.970 ±0.000	0.977 ±0.000	0.982 ±0.000	0.987 ±0.000	0.990 ±0.000	0.992 ±0.000	0.994 ±0.000	0.995 ±0.000	0.996 ±0.000	0.012
DEYES-S-UP4-*LoRA*	0.961	0.971	0.978	0.983	0.987	0.990	0.993	0.994	0.995	0.996	0.012
DEYES-S-UP4-*scratch**	0.958 ±0.001	0.968 ±0.001	0.976 ±0.000	0.982 ±0.000	0.986 ±0.000	0.990 ±0.000	0.990 ±0.002	0.983 ±0.001	0.985 ±0.001	0.989 ±0.001	0.017

(c)

Table 8: Every MITSUBA cell (continued).

MITSUBA LPIPS ↓	1	2	4	8	16	32	64	128	256	512	err.
OIDN 2.5	0.203	0.186	0.171	0.158	0.146	0.136	0.128	0.122	0.117	0.113	0.145
OptiX 9	0.212	0.187	0.170	0.155	0.143	0.132	0.124	0.118	0.113	0.109	0.143
NPPD†	0.210	0.194	0.180	0.167	0.153	0.145	0.140	0.130	0.123	0.111	0.152
NRD (ReLAX)	0.428	0.389	0.351	0.317	0.289	0.267	0.252	0.243	0.238	0.236	0.295
DMRD§	0.256	0.231	0.209	0.194	0.183	0.173	0.166	0.161	0.157	0.156	0.186
DEYES-T-UP4-*FT*	0.164	0.150	0.139	0.132	0.125	0.118	0.111	0.105	0.101	0.096	0.122
DEYES-S-UP4-*FT*	0.164 ±0.003	0.149 ±0.001	0.138 ±0.001	0.131 ±0.001	0.124 ±0.001	0.117 ±0.001	0.111 ±0.000	0.106 ±0.000	0.102 ±0.000	0.097 ±0.001	0.122
DEYES-B-UP4-*FT*	0.158 ±0.002	0.145 ±0.001	0.136 ±0.001	0.129 ±0.000	**0.122** ±0.000	**0.116** ±0.000	**0.110** ±0.000	0.105 ±0.000	**0.100** ±0.001	0.096 ±0.000	0.120
DEYES-L-UP4-*FT*	**0.153**	**0.142**	**0.135**	**0.129**	0.122	0.116	0.110	**0.105**	0.100	**0.096**	**0.119**
DEYES-T-UP2-*FT*	0.174 ±0.001	0.158 ±0.000	0.147 ±0.000	0.138 ±0.000	0.129 ±0.001	0.121 ±0.001	0.114 ±0.001	0.109 ±0.000	0.105 ±0.000	0.100 ±0.001	0.128
DEYES-S-UP4-*LoRA*	0.167	0.152	0.141	0.132	0.125	0.118	0.111	0.106	0.101	0.096	0.123
DEYES-S-UP4-*scratch**	0.172 ±0.002	0.157 ±0.001	0.146 ±0.001	0.137 ±0.001	0.130 ±0.000	0.123 ±0.000	0.117 ±0.001	0.121 ±0.001	0.117 ±0.001	0.109 ±0.002	0.132

(d)

MITSUBA SMAPE ↓	1	2	4	8	16	32	64	128	256	512	err.
OIDN 2.5	0.0379	0.0320	0.0274	0.0237	0.0206	0.0180	0.0159	0.0141	0.0126	0.0114	0.0198
OptiX 9	0.0421	0.0359	0.0309	0.0273	0.0240	0.0208	0.0180	0.0154	0.0134	0.0116	0.0221
NPPD†	0.0413	0.0353	0.0307	0.0272	0.0240	0.0213	0.0195	0.0181	0.0170	0.0156	0.0238
NRD (ReLAX)	0.0860	0.0722	0.0632	0.0575	0.0537	0.0510	0.0492	0.0480	0.0472	0.0468	0.0564
DMRD§	0.0960	0.0837	0.0737	0.0670	0.0634	0.0609	0.0593	0.0582	0.0575	0.0569	0.0666
DEYES-T-UP4-*FT*	0.0358	0.0301	0.0257	0.0222	0.0192	0.0166	0.0145	0.0129	0.0118	0.0110	0.0185
DEYES-S-UP4-*FT*	0.0349 ±0.0000	0.0295 ±0.0001	0.0252 ±0.0000	0.0217 ±0.0001	0.0188 ±0.0000	0.0163 ±0.0000	0.0143 ±0.0000	0.0126 ±0.0000	**0.0114** ±0.0001	**0.0104** ±0.0002	0.0181
DEYES-B-UP4-*FT*	0.0346 ±0.0002	0.0290 ±0.0001	0.0247 ±0.0001	0.0214 ±0.0000	0.0185 ±0.0000	0.0161 ±0.0000	0.0141 ±0.0000	**0.0125** ±0.0000	0.0115 ±0.0001	0.0107 ±0.0001	0.0179
DEYES-L-UP4-*FT*	**0.0344**	**0.0287**	**0.0245**	**0.0213**	**0.0185**	**0.0160**	**0.0141**	0.0125	0.0115	0.0107	**0.0179**
DEYES-T-UP2-*FT*	0.0370 ±0.0002	0.0312 ±0.0002	0.0267 ±0.0001	0.0230 ±0.0001	0.0199 ±0.0001	0.0172 ±0.0001	0.0151 ±0.0001	0.0132 ±0.0001	0.0119 ±0.0001	0.0107 ±0.0001	0.0190
DEYES-S-UP4-*LoRA*	0.0358	0.0302	0.0258	0.0223	0.0193	0.0167	0.0146	0.0129	0.0118	0.0107	0.0185
DEYES-S-UP4-*scratch**	0.0378 ±0.0002	0.0317 ±0.0002	0.0272 ±0.0002	0.0234 ±0.0001	0.0203 ±0.0001	0.0175 ±0.0001	0.0160 ±0.0005	0.0172 ±0.0006	0.0158 ±0.0005	0.0137 ±0.0004	0.0209

(e)

MITSUBA $\mathcal{L}_1$ ↓	1	2	4	8	16	32	64	128	256	512	err.
OIDN 2.5	0.0381	0.0312	0.0257	0.0209	0.0174	0.0146	0.0123	0.0105	0.0090	0.0078	0.0165
OptiX 9	0.0365	0.0297	0.0243	0.0199	0.0167	0.0139	0.0116	0.0096	0.0080	0.0066	0.0153
NPPD†	0.0368	0.0299	0.0251	0.0212	0.0182	0.0158	0.0141	0.0128	0.0109	0.0099	0.0178
NRD (ReLAX)	0.1067	0.0995	0.0942	0.0907	0.0882	0.0865	0.0854	0.0846	0.0842	0.0839	0.0901
DMRD§	0.2030	0.1874	0.1793	0.1739	0.1704	0.1679	0.1663	0.1652	0.1643	0.1637	0.1737
DEYES-T-UP4-*FT*	0.0289	0.0235	0.0192	0.0161	0.0133	0.0112	0.0095	0.0081	0.0071	0.0064	0.0127
DEYES-S-UP4-*FT*	0.0278 ±0.0003	0.0225 ±0.0002	0.0185 ±0.0001	0.0155 ±0.0001	0.0130 ±0.0001	0.0109 ±0.0001	0.0093 ±0.0001	0.0079 ±0.0000	0.0069 ±0.0000	**0.0060** ±0.0000	0.0123
DEYES-B-UP4-*FT*	**0.0273** ±0.0000	0.0220 ±0.0000	0.0182 ±0.0000	0.0153 ±0.0001	0.0128 ±0.0000	0.0108 ±0.0000	0.0092 ±0.0000	0.0078 ±0.0000	0.0069 ±0.0000	0.0062 ±0.0000	0.0122
DEYES-L-UP4-*FT*	0.0274	**0.0220**	**0.0180**	**0.0151**	**0.0127**	**0.0107**	**0.0091**	**0.0078**	**0.0068**	0.0061	**0.0121**
DEYES-T-UP2-*FT*	0.0304 ±0.0005	0.0244 ±0.0004	0.0201 ±0.0003	0.0166 ±0.0002	0.0138 ±0.0002	0.0115 ±0.0002	0.0098 ±0.0002	0.0084 ±0.0002	0.0073 ±0.0002	0.0064 ±0.0002	0.0131
DEYES-S-UP4-*LoRA*	0.0292	0.0235	0.0194	0.0162	0.0135	0.0114	0.0097	0.0083	0.0072	0.0063	0.0128
DEYES-S-UP4-*scratch**	0.0314 ±0.0006	0.0250 ±0.0004	0.0206 ±0.0003	0.0170 ±0.0002	0.0142 ±0.0002	0.0118 ±0.0001	0.0103 ±0.0003	0.0103 ±0.0002	0.0091 ±0.0002	0.0078 ±0.0001	0.0143

(f)

Table 9: **Every NB-WARM cell** (recurrent state carried; 1200 frames at 2/4/8 spp, 280 at 32), per metric and rung, as Table 8.

NB-WARM PSNR ↑	2	4	8	32	mean
OIDN 2.5	26.50	27.65	28.75	31.54	28.61
OptiX 9	26.71	27.62	28.48	31.05	28.46
NPPD†	27.90	28.79	29.79	32.58	29.77
NRD (ReLAX)	23.62	24.17	24.56	24.85	24.30
DEYES-T-UP4-*FT*	28.81	29.80	30.72	33.36	30.67
DEYES-S-UP4-*FT*	29.00 ±0.01	29.96 ±0.01	30.87 ±0.01	33.44 ±0.00	30.82
DEYES-B-UP4-*FT*	29.13 ±0.01	30.10 ±0.01	31.00 ±0.01	33.58 ±0.03	30.95
DEYES-L-UP4-*FT*	**29.30**	**30.27**	**31.17**	**33.77**	**31.13**
DEYES-T-UP2-*FT*	28.34 ±0.04	29.35 ±0.04	30.31 ±0.04	32.96 ±0.02	30.24
DEYES-S-UP4-*LoRA*	28.75	29.71	30.62	33.18	30.57
DEYES-S-UP4-*scratch**	28.54 ±0.05	29.54 ±0.04	30.49 ±0.04	33.16 ±0.05	30.43

(a)

NB-WARM SSIM ↑	2	4	8	32	err.
OIDN 2.5	0.908	0.932	0.950	0.976	0.052
OptiX 9	0.919	0.941	0.955	0.978	0.046
NPPD†	0.938	0.951	0.963	0.981	0.038
NRD (ReLAX)	0.879	0.893	0.903	0.914	0.102
DEYES-T-UP4-*FT*	0.947	0.959	0.968	0.984	0.033
DEYES-S-UP4-*FT*	0.950 ±0.000	0.961 ±0.000	0.969 ±0.000	0.984 ±0.000	0.031
DEYES-B-UP4-*FT*	0.951 ±0.000	0.962 ±0.000	0.970 ±0.000	0.985 ±0.000	0.030
DEYES-L-UP4-*FT*	**0.953**	**0.964**	**0.972**	**0.985**	**0.029**
DEYES-T-UP2-*FT*	0.940 ±0.000	0.954 ±0.000	0.965 ±0.000	0.982 ±0.000	0.036
DEYES-S-UP4-*LoRA*	0.947	0.958	0.967	0.983	0.033
DEYES-S-UP4-*scratch**	0.946 ±0.001	0.958 ±0.001	0.967 ±0.000	0.983 ±0.000	0.034

(b)

NB-WARM MS-SSIM ↑	2	4	8	32	err.
OIDN 2.5	0.908	0.928	0.944	0.968	0.059
OptiX 9	0.915	0.935	0.948	0.969	0.054
NPPD†	0.934	0.945	0.955	0.972	0.046
NRD (ReLAX)	0.880	0.893	0.902	0.912	0.102
DEYES-T-UP4-*FT*	0.942	0.953	0.961	0.976	0.040
DEYES-S-UP4-*FT*	0.945 ±0.000	0.955 ±0.000	0.963 ±0.000	0.976 ±0.000	0.039
DEYES-B-UP4-*FT*	0.947 ±0.000	0.956 ±0.000	0.964 ±0.000	0.977 ±0.000	0.037
DEYES-L-UP4-*FT*	**0.948**	**0.958**	**0.965**	**0.978**	**0.036**
DEYES-T-UP2-*FT*	0.936 ±0.000	0.948 ±0.000	0.958 ±0.000	0.974 ±0.000	0.044
DEYES-S-UP4-*LoRA*	0.942	0.952	0.961	0.975	0.041
DEYES-S-UP4-*scratch**	0.941 ±0.001	0.952 ±0.001	0.960 ±0.000	0.975 ±0.000	0.041

(c)

NB-WARM LPIPS ↓	2	4	8	32	err.
OIDN 2.5	0.346	0.320	0.297	0.288	0.312
OptiX 9	0.360	0.334	0.310	0.294	0.324
NPPD†	0.291	0.276	0.257	0.253	0.269
NRD (ReLAX)	0.419	0.397	0.382	0.386	0.396
DEYES-T-UP4-*FT*	0.269	0.255	0.244	0.255	0.256
DEYES-S-UP4-*FT*	0.267 ±0.000	0.254 ±0.000	0.243 ±0.000	0.254 ±0.001	0.254
DEYES-B-UP4-*FT*	0.262 ±0.000	0.251 ±0.000	0.241 ±0.000	0.253 ±0.000	0.251
DEYES-L-UP4-*FT*	**0.260**	**0.248**	**0.239**	**0.251**	**0.249**
DEYES-T-UP2-*FT*	0.281 ±0.001	0.267 ±0.001	0.254 ±0.001	0.262 ±0.001	0.266
DEYES-S-UP4-*LoRA*	0.271	0.258	0.247	0.259	0.259
DEYES-S-UP4-*scratch**	0.271 ±0.001	0.257 ±0.001	0.246 ±0.001	0.254 ±0.001	0.257

(d)

NB-WARM SMAPE ↓	2	4	8	32	err.
OIDN 2.5	0.0825	0.0723	0.0641	0.0506	0.0663
OptiX 9	0.0789	0.0715	0.0653	0.0530	0.0665
NPPD†	0.0721	0.0656	0.0600	0.0488	0.0610
NRD (ReLAX)	0.1038	0.0978	0.0936	0.0871	0.0954
DEYES-T-UP4-*FT*	0.0657	0.0591	0.0537	0.0442	0.0551
DEYES-S-UP4-*FT*	0.0642 ±0.0000	0.0580 ±0.0000	0.0529 ±0.0001	0.0438 ±0.0000	0.0542
DEYES-B-UP4-*FT*	0.0637 ±0.0003	0.0574 ±0.0002	0.0523 ±0.0001	0.0432 ±0.0000	0.0536
DEYES-L-UP4-*FT*	**0.0628**	**0.0566**	**0.0515**	**0.0427**	**0.0529**
DEYES-T-UP2-*FT*	0.0686 ±0.0002	0.0616 ±0.0002	0.0559 ±0.0002	0.0461 ±0.0002	0.0575
DEYES-S-UP4-*LoRA*	0.0657	0.0594	0.0542	0.0450	0.0555
DEYES-S-UP4-*scratch**	0.0672 ±0.0003	0.0605 ±0.0002	0.0549 ±0.0001	0.0451 ±0.0001	0.0563

(e)

NB-WARM $\mathcal{L}_1$ ↓	2	4	8	32	err.
OIDN 2.5	0.1103	0.0958	0.0839	0.0717	0.0893
OptiX 9	0.1683	0.1604	0.1527	0.4419	0.2066
NPPD†	0.0637	0.0550	0.0518	0.0401	0.0519
NRD (ReLAX)	0.2349	0.2289	0.2246	0.5563	0.2863
DEYES-T-UP4-*FT*	0.0579	0.0499	0.0436	0.0347	0.0457
DEYES-S-UP4-*FT*	0.0561 ±0.0004	0.0485 ±0.0003	0.0426 ±0.0002	0.0340 ±0.0005	0.0446
DEYES-B-UP4-*FT*	0.0563 ±0.0004	0.0488 ±0.0003	0.0428 ±0.0003	0.0360 ±0.0009	0.0453
DEYES-L-UP4-*FT*	**0.0546**	**0.0474**	**0.0417**	**0.0334**	**0.0436**
DEYES-T-UP2-*FT*	0.0604 ±0.0004	0.0520 ±0.0004	0.0453 ±0.0003	0.0351 ±0.0004	0.0473
DEYES-S-UP4-*LoRA*	0.0587	0.0508	0.0442	0.0350	0.0463
DEYES-S-UP4-*scratch**	0.0596 ±0.0012	0.0517 ±0.0013	0.0453 ±0.0014	0.0355 ±0.0016	0.0472

(f)

Table 10: **Every NB-COLD cell** (state reset before every frame), per metric and rung, as Table 8. DEYES and DEYES-S-UP4-*scratch* rows score every frame.

NB-COLD PSNR ↑	2	4	8	32	mean
OIDN 2.5	26.50	27.65	28.75	31.54	28.61
OptiX 9	25.99	27.15	28.29	31.51	28.23
NPPD†	26.60	27.61	28.74	31.96	28.73
StatER‡	21.22	22.56	24.39	28.39	24.14
NRD (ReLAX)	22.05	22.63	22.97	23.26	22.73
DMRD§	23.84	25.37	26.72	29.31	26.31
DEYES-T-UP4-*FT*	27.80	28.92	30.00	32.92	29.91
DEYES-S-UP4-*FT*	27.97 ±0.02	29.07 ±0.02	30.13 ±0.01	33.01 ±0.02	30.05
DEYES-B-UP4-*FT*	28.06 ±0.04	29.17 ±0.03	30.23 ±0.03	33.14 ±0.03	30.15
DEYES-L-UP4-*FT*	**28.19**	**29.30**	**30.36**	**33.28**	**30.28**
DEYES-T-UP2-*FT*	27.41 ±0.01	28.52 ±0.01	29.59 ±0.02	32.52 ±0.01	29.51
DEYES-S-UP4-*LoRA*	27.72	28.79	29.85	32.83	29.80
DEYES-S-UP4-*scratch**	27.39 ±0.04	28.54 ±0.04	29.65 ±0.04	32.67 ±0.03	29.56

(a)

NB-COLD SSIM ↑	2	4	8	32	err.
OIDN 2.5	0.908	0.932	0.950	0.976	0.052
OptiX 9	0.891	0.919	0.941	0.973	0.061
NPPD†	0.906	0.925	0.946	0.974	0.056
StatER‡	0.825	0.870	0.908	0.957	0.097
NRD (ReLAX)	0.784	0.801	0.812	0.839	0.190
DMRD§	0.831	0.873	0.906	0.954	0.098
DEYES-T-UP4-*FT*	0.930	0.947	0.960	0.980	0.041
DEYES-S-UP4-*FT*	0.933 ±0.000	0.949 ±0.000	0.962 ±0.000	0.981 ±0.000	0.040
DEYES-B-UP4-*FT*	0.935 ±0.000	0.951 ±0.000	0.963 ±0.000	0.981 ±0.000	0.039
DEYES-L-UP4-*FT*	**0.937**	**0.952**	**0.964**	**0.982**	**0.038**
DEYES-T-UP2-*FT*	0.924 ±0.000	0.942 ±0.000	0.956 ±0.000	0.978 ±0.000	0.045
DEYES-S-UP4-*LoRA*	0.929	0.945	0.958	0.978	0.043
DEYES-S-UP4-*scratch**	0.925 ±0.001	0.943 ±0.001	0.957 ±0.001	0.979 ±0.000	0.044

(b)

NB-COLD MS-SSIM ↑	2	4	8	32	err.
OIDN 2.5	0.908	0.928	0.944	0.968	0.059
OptiX 9	0.892	0.916	0.935	0.965	0.067
NPPD†	0.905	0.922	0.940	0.966	0.063
StatER‡	0.821	0.858	0.898	0.949	0.107
NRD (ReLAX)	0.799	0.816	0.826	0.849	0.177
DMRD§	0.849	0.884	0.911	0.950	0.094
DEYES-T-UP4-*FT*	0.928	0.943	0.954	0.973	0.048
DEYES-S-UP4-*FT*	0.931 ±0.000	0.945 ±0.000	0.956 ±0.000	0.973 ±0.000	0.046
DEYES-B-UP4-*FT*	0.932 ±0.000	0.946 ±0.000	0.957 ±0.000	0.974 ±0.000	0.045
DEYES-L-UP4-*FT*	**0.934**	**0.947**	**0.958**	**0.974**	**0.044**
DEYES-T-UP2-*FT*	0.922 ±0.000	0.938 ±0.000	0.950 ±0.000	0.970 ±0.000	0.052
DEYES-S-UP4-*LoRA*	0.926	0.941	0.952	0.970	0.050
DEYES-S-UP4-*scratch**	0.923 ±0.001	0.939 ±0.001	0.952 ±0.001	0.972 ±0.000	0.050

(c)

NB-COLD LPIPS ↓	2	4	8	32	err.
OIDN 2.5	0.346	0.320	0.297	0.288	0.312
OptiX 9	0.374	0.344	0.316	0.297	0.331
NPPD†	0.322	0.304	0.287	0.272	0.296
StatER‡	0.449	0.392	0.343	0.296	0.366
NRD (ReLAX)	0.487	0.471	0.460	0.451	0.467
DMRD§	0.447	0.414	0.383	0.345	0.395
DEYES-T-UP4-*FT*	0.281	0.266	0.254	0.261	0.265
DEYES-S-UP4-*FT*	0.279 ±0.000	0.264 ±0.000	0.252 ±0.000	0.260 ±0.000	0.264
DEYES-B-UP4-*FT*	0.274 ±0.000	0.261 ±0.000	0.249 ±0.000	0.258 ±0.000	0.260
DEYES-L-UP4-*FT*	**0.271**	**0.258**	**0.247**	**0.257**	**0.258**
DEYES-T-UP2-*FT*	0.293 ±0.001	0.277 ±0.001	0.263 ±0.001	0.268 ±0.001	0.275
DEYES-S-UP4-*LoRA*	0.286	0.271	0.258	0.273	0.272
DEYES-S-UP4-*scratch**	0.286 ±0.001	0.268 ±0.001	0.255 ±0.001	0.260 ±0.001	0.267

(d)

NB-COLD SMAPE ↓	2	4	8	32	err.
OIDN 2.5	0.0825	0.0723	0.0641	0.0506	0.0663
OptiX 9	0.0880	0.0774	0.0687	0.0525	0.0704
NPPD†	0.0833	0.0750	0.0668	0.0522	0.0683
StatER‡	0.1325	0.1134	0.0918	0.0627	0.0964
NRD (ReLAX)	0.1207	0.1108	0.1045	0.0931	0.1068
DMRD§	0.1363	0.1156	0.1005	0.0872	0.1084
DEYES-T-UP4-*FT*	0.0729	0.0647	0.0578	0.0464	0.0596
DEYES-S-UP4-*FT*	0.0715 ±0.0001	0.0636 ±0.0001	0.0570 ±0.0001	0.0459 ±0.0001	0.0587
DEYES-B-UP4-*FT*	0.0712 ±0.0005	0.0632 ±0.0003	0.0566 ±0.0002	0.0453 ±0.0001	0.0583
DEYES-L-UP4-*FT*	**0.0704**	**0.0625**	**0.0559**	**0.0449**	**0.0577**
DEYES-T-UP2-*FT*	0.0757 ±0.0001	0.0672 ±0.0002	0.0602 ±0.0002	0.0483 ±0.0001	0.0620
DEYES-S-UP4-*LoRA*	0.0735	0.0654	0.0586	0.0466	0.0602
DEYES-S-UP4-*scratch**	0.0759 ±0.0002	0.0672 ±0.0002	0.0598 ±0.0001	0.0474 ±0.0001	0.0617

(e)

NB-COLD $\mathcal{L}_1$ ↓	2	4	8	32	err.
OIDN 2.5	0.1103	0.0958	0.0839	0.0717	0.0893
OptiX 9	0.1687	0.1573	0.1466	0.4167	0.2007
NPPD†	0.0715	0.0610	0.0562	0.0447	0.0576
StatER‡	0.1209	0.0958	0.0761	0.0477	0.0805
NRD (ReLAX)	0.2910	0.2846	0.2806	0.6105	0.3451
DMRD§	0.8793	0.8479	0.8310	1.3740	0.9605
DEYES-T-UP4-*FT*	0.0632	0.0533	0.0458	0.0356	0.0484
DEYES-S-UP4-*FT*	0.0613 ±0.0003	0.0520 ±0.0002	0.0448 ±0.0001	0.0348 ±0.0002	0.0472
DEYES-B-UP4-*FT*	0.0614 ±0.0000	0.0523 ±0.0001	0.0451 ±0.0001	0.0369 ±0.0001	0.0481
DEYES-L-UP4-*FT*	**0.0600**	**0.0510**	**0.0440**	**0.0343**	**0.0464**
DEYES-T-UP2-*FT*	0.0662 ±0.0001	0.0562 ±0.0002	0.0482 ±0.0003	0.0363 ±0.0005	0.0505
DEYES-S-UP4-*LoRA*	0.0646	0.0552	0.0474	0.0384	0.0505
DEYES-S-UP4-*scratch**	0.0658 ±0.0008	0.0560 ±0.0008	0.0482 ±0.0009	0.0370 ±0.0010	0.0506

(f)

K TEMPORAL METRICS

Table 11 scores the NB-WARM protocol on the lossless frames of Appendix B with the corpus's temporal tooling: FVVDP (Mantiuk et al., 2021) on the frame sequence, temporal PSNR (PSNR of consecutive-frame differences against the reference's) and temporal relative MAE. Baselines are their archived lossless frame sets. Every DEYES row beats every baseline on every column; the full fine-tune leads the frozen trunk at every size, and larger encoders score higher. Temporal-difference metrics cancel an error that is constant over time, so they are read beside the spatial PSNR of Table 1. NRD and DMRD have no lossless warm frame set and are absent.

Table 11: Temporal metrics on NB-WARM (lossless frames): FVVDP ↑ (JOD), temporal PSNR ↑ (dB) and temporal relative MAE ↓ at 2/4/8/32 spp (280 frames at 32 spp, 1200 otherwise). †NPPD: the checkpoint matched to the rung. Ranked: *FT* rows, 20,000 steps (seeds as in Table 8); not ranked: the *LoRA* rows (10,000 steps, three seeds) and *DEYES-S-UP4-*scratch* (40,000 steps, three seeds).

	FVVDP ↑				tPSNR ↑				tRMAE ↓			
spp	2	4	8	32	2	4	8	32	2	4	8	32
OIDN 2.5	6.74	7.13	7.49	8.29	24.11	25.16	26.34	29.42	0.253	0.222	0.194	0.151
OptiX 9	6.97	7.35	7.65	8.33	24.54	25.55	26.54	29.25	0.222	0.201	0.184	0.149
NPPD†	7.19	7.51	7.81	8.44	25.59	26.40	27.31	29.85	0.215	0.202	0.185	0.157
DEYES-T-UP4-*FT*	7.36	7.67	7.94	8.59	26.43	27.38	28.35	31.16	0.197	0.181	0.165	0.133
DEYES-S-UP4-*FT*	7.41	7.72	7.98	8.59	26.61	27.55	28.50	31.24	0.192	0.176	0.162	0.132
DEYES-B-UP4-*FT*	7.44	7.75	8.00	8.63	26.78	27.72	28.66	31.41	0.189	0.174	0.160	0.131
DEYES-L-UP4-*FT*	**7.47**	**7.78**	**8.03**	**8.64**	**26.96**	**27.90**	**28.85**	**31.60**	**0.185**	**0.170**	**0.157**	**0.130**
DEYES-T-UP2-*FT*	7.25	7.58	7.88	8.54	25.95	26.91	27.90	30.71	0.204	0.187	0.170	0.139
DEYES-T-UP4-*LoRA*	7.30	7.61	7.90	8.53	26.15	27.09	28.07	30.88	0.199	0.184	0.168	0.137
DEYES-S-UP4-*LoRA*	7.37	7.68	7.94	8.56	26.36	27.28	28.24	30.96	0.196	0.181	0.166	0.135
DEYES-B-UP4-*LoRA*	7.36	7.67	7.95	8.57	26.43	27.37	28.34	31.12	0.195	0.179	0.164	0.134
DEYES-L-UP4-*LoRA*	7.40	7.70	7.97	8.58	26.52	27.46	28.42	31.17	0.194	0.179	0.164	0.134
DEYES-S-UP4-*scratch**	7.30	7.62	7.90	8.54	26.22	27.17	28.13	30.88	0.197	0.181	0.166	0.138

Recovery after a history reset. To test behavior at a camera cut, we clear each method's temporal state in mid-sequence and compare it with the same method's uninterrupted run. We use the four Noisebase test sequences with the fastest image motion (*bistro2*, *dining-room*, *measure7*, *bistro1-day*, ranked by the mean frame-to-frame change of the reference before any method was run) at 4 spp, frames 0–79, clearing the state before frame 16 or frame 40. The methods are DEYES-S-UP4-*FT* and DEYES-S-UP4-*scratch*, NPPD's 4-spp checkpoint and OptiX 9's temporal model; a reset is each method's own first-frame treatment. All metrics are computed from the saved 8-bit frames against the temporal scorer's references. A method has recovered once its per-frame PSNR stays within 0.1 dB of its uninterrupted run for three consecutive frames (median over the four sequences). DEYES-S-UP4-*FT* recovers in 2.5–3 frames against 3–4 for NPPD, and loses less at the reset frame (Table 12). DEYES-S-UP4-*scratch* recovers in 2.5–3.5 frames, so the fast recovery comes from the temporal design rather than from pretraining. OptiX takes 7.5–8 frames, but its deficit turns negative within two frames: clearing its accumulated history makes it better, so its history is costing it quality. On newly revealed surfaces, where no valid history exists, DEYES-S-UP4-*FT* has the highest PSNR both where history is reprojected from outside the frame (the model's validity mask) and where it is disoccluded inside the frame (pixels whose reprojected depth changes by more than 10%).

Table 12: **Recovery after a forced history reset**, NB-WARM at 4 spp, four fast-motion sequences, state cleared before frame 16 or 40 (values "16 / 40"). Frames to recover: median over sequences of the first frame after which PSNR stays within 0.1 dB of the uninterrupted run for three frames. PSNR deficit and tPSNR dip: uninterrupted minus reset run, at the reset frame. Last three columns: PSNR of the uninterrupted run on pixels with valid history, on pixels reprojected from outside the frame, and on pixels disoccluded inside it. All on the saved 8-bit frames.

	frames to recover	PSNR deficit at reset (dB)	tPSNR dip at reset (dB)	PSNR by region (dB)		
method				valid history	outside frame	disoccluded
DEYES-S-UP4-*FT*	3.0 / 2.5	1.03 / 0.87	0.65 / 0.54	**30.35**	**35.33**	**24.56**
DEYES-S-UP4-*scratch*	3.5 / 2.5	1.13 / 1.03	0.71 / 0.63	29.78	34.97	24.05
NPPD (4-spp checkpoint)	4.0 / 3.0	1.25 / 1.04	1.01 / 0.88	28.83	33.62	23.36
OptiX 9 (temporal)	8.0 / 7.5	0.56 / 0.43	0.77 / 0.62	27.01	33.01	21.75

L QUALITY AND COST AGAINST SAMPLE COUNT

Figure 14 places every method on one axis per protocol: the MITSUBA ladder over 1–512 spp and NB-WARM and NB-COLD over 2–32 spp, in PSNR and $\mathcal{L}_1$. The four encoder sizes lie in a narrow band at every rung and on every protocol; the from-scratch network at its best horizon approaches them in the middle of the MITSUBA ladder and falls behind at the clean end; and NPPD leads OIDN on Noisebase, its training distribution, but trails it on MITSUBA (with pixel-mean inputs). Every model except DEYES-S-UP4-*scratch* improves monotonically with sample count over 1–512 spp; DEYES-S-UP4-*scratch* fails on three scenes from 128 spp (Appendix N).

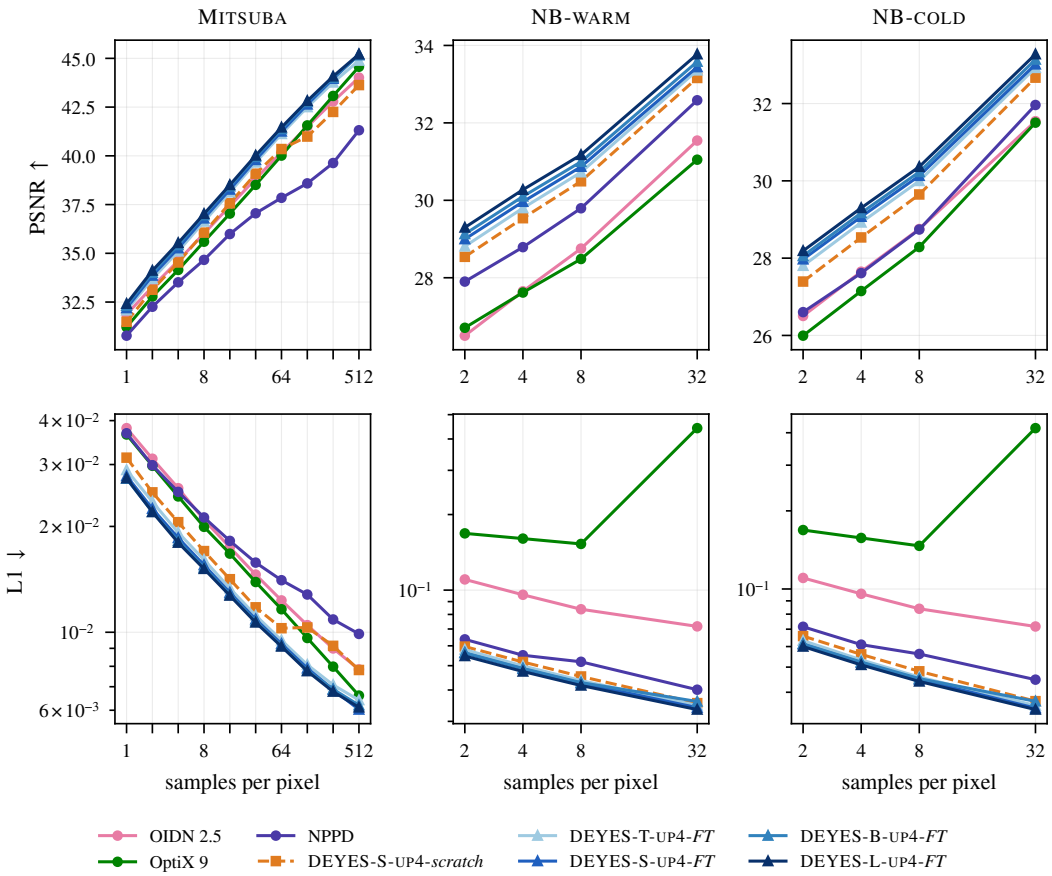

Figure 14: Quality against sample count on all three protocols (rows: PSNR ↑, $\mathcal{L}_1$ ↓; columns: MITSUBA 1–512 spp, NB-WARM and NB-COLD 2–32 spp). Lines: DEYES-{T,S,B,L}-UP4-*FT* (seeds as in Table 7) and DEYES-S-UP4-*scratch*. NPPD is the best of its four checkpoints per rung on MITSUBA and the matched checkpoint on Noisebase. Every DEYES size is ahead of every baseline on both metrics at every sample count of all three protocols.

Synthetic high-spp data. Figure 15 isolates the synthetic high-spp rows of Section 3 with a matched pair: DEYES-S-UP4-*LoRA* (10,000 steps) with and without them, otherwise identical. Without them the model is unchanged up to 16 spp and loses 0.20, 0.37, 0.70 and 1.16 dB at 64, 128, 256 and 512 spp, falling below the OIDN/OptiX envelope from 256 spp; with them its gain over the noisy input shrinks from 18.1 dB at 1 spp to 9.3 at 512 (OIDN/OptiX: 18.0 and 8.8). Genuinely rendered inputs above 32 spp are never seen in training; the model sees the synthetic ones, whose rescaled residual keeps the heavy-tailed shape of 32-spp noise rather than the more Gaussian noise of a true k-spp estimate.

Clean input. A denoiser that filters radiance can only redistribute what its input contains; one that synthesizes output can add structure that is not there. An input that is already clean tests this directly: there is no noise left to remove, so any change the denoiser makes, whether smoothing, sharpening or

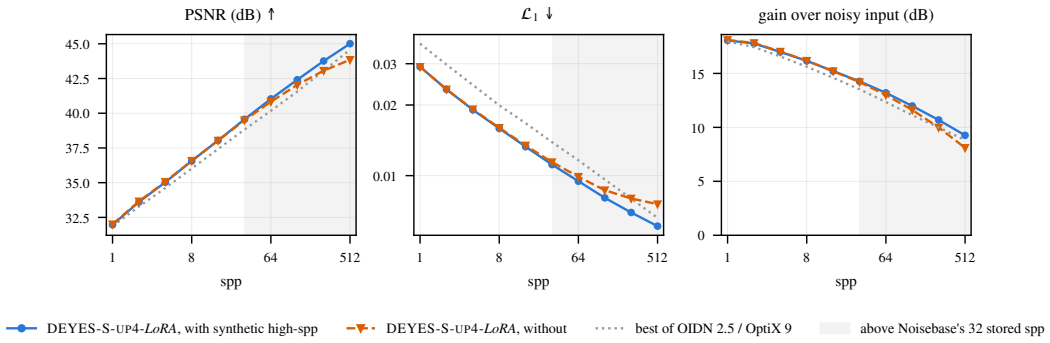

Figure 15: **Convergence in sample count and the synthetic high-spp ablation**, MITSUBA, 39 scenes, 1–512 spp. Without the synthetic high-spp training rows, both PSNR and $\mathcal{L}_1$ regress at every sample count from 64 spp, by 1.16 dB and 24% at 512 spp. (a, b) PSNR and $\mathcal{L}_1$ of DEYES-S-UP4-*LoRA* (10,000 steps) with and without synthetic high-spp training rows, a matched pair differing only in that setting; dotted gray, the per-rung best of OIDN 2.5 and OptiX 9. (c) PSNR gain over the noisy input (gray: the envelope's). Shaded: sample counts above the 32 samples per pixel that Noisebase stores.

invented texture, is its own contribution, and a well-behaved denoiser should leave the image nearly untouched. We feed each MITSUBA scene's reference (Appendix B) as the radiance input, with the auxiliary buffers of its 512-spp render, and score the output against that same image, so PSNR, $\mathcal{L}_1$ and SSIM measure how much each method alters an image it should leave alone (Table 13, 39 scenes). Where the reference still carries residual noise, removing it also counts as alteration: on *kitchen* and *veach-ajar* every method scores within 1 dB of the others (46.8–47.8 and 39.7–40.0 dB), close to the reference's own noise level, so these scenes do not separate the methods. DEYES-S-UP4-*FT* alters the clean input about as little as the production denoisers, and less than DEYES-S-UP4-*scratch*. On no scene does any method alter the clean input more than its own output on the real 512-spp render differs from the reference. Telling the model the true sample count rather than the learned "unknown" value is worth 2.2 dB, so the conditioning makes it gentler, as intended. Every method alters hair and fur the most, and DEYES-S-UP4-*FT* alters them least (*curly-hair*: 43.7 dB, against 34.6 for OIDN and 38.2 for OptiX); its weakest scene is *glass-of-water* (38.2 dB, OptiX 40.3). DEYES rounds its radiance input to bf16 before the fp32 pyramid (at most 0.2% relative), which caps a perfect pass-through, so its numbers are conservative.

Table 13: **Clean-input test**, MITSUBA, 39 scenes: each scene's reference is the input, and the output is scored against it, so higher PSNR and lower $\mathcal{L}_1$ and $1-\text{SSIM}$ mean the method changed the clean image less. "512-spp render": the method's PSNR on the real 512-spp render (as in Table 8), for scale. Last column: scenes where the clean input is altered more than that.

	clean input, output vs. input			512-spp render	clean input
method	PSNR ↑	$\mathcal{L}_1 \times 10^3$ ↓	$(1-\text{SSIM}) \times 10^4$ ↓	PSNR	altered more
DEYES-S-UP4-*FT*, spp plane at 512	52.6	3.00	7.3	45.2	0/39
DEYES-S-UP4-*FT*, spp "unknown"	50.4	3.45	7.6	45.2	0/39
DEYES-S-UP4-*scratch*, spp plane at 512	51.4	3.20	8.1	43.6	0/39
DEYES-S-UP4-*scratch*, spp "unknown"	49.6	3.49	8.6	43.6	0/39
OIDN 2.5	51.1	4.00	8.4	44.0	0/39
OptiX 9	53.2	2.64	7.1	44.6	0/39

Does the sample saving pay for the denoiser? Tables 14 and 15 are the tabular form of Figures 3 and 16, generated by the same script from the same data; At each baseline rung the quality target is what the baseline group delivers at that sample count; each side's time is the cheapest per-frame wall clock (render + denoise) at which it reaches that target, with quality interpolated linearly in $\log_2$ spp between measured rungs. Render times are measured per scene and per rung by the renderer that produced the frames (Mitsuba 3's `cuda_ad_rgb` variant on one A40), excluding writing the frame and its buffers to disk (our dataset writer's tonemapped PNG and EXR buffers, 1.57 s per frame on average, which is not part of a denoising pipeline); scene load (0.87 s mean, 0.23–5.8 s) is counted in the isolated still, and Dr.Jit's one-off kernel compilation (0.04–0.17 s with a warm cache, up to 7 s cold) is not; denoise time is compute time measured at each scene's own resolution (ten resolutions

from 0.26 to 2.07 Mpx). Every table has two blocks, one per figure, both on all 39 MITSUBA scenes: the *isolated still* (scene load and the renderer's per-frame launch overhead included) and the *marginal sampling cost*.

Against the fastest production baseline (Table 14), the default model alone is 0–14% slower to the same quality at 1–4 spp on PSNR and saves 27–38% of the render + denoise time from 32 to 512 spp; on $\mathcal{L}_1$ and LPIPS it is ahead from 4 spp and saves 37–55% and 63–83% from 32 spp. With DEYES-T-UP2-*FT* allowed it is at most 5% slower anywhere. With fixed costs amortized, the baselines' faster denoisers win below 4 spp on $\mathcal{L}_1$ and LPIPS and at or below 4 spp on PSNR (the interactive model narrows the PSNR gap to -30% at 1 spp and -12% at 2, and wins from 2 spp on $\mathcal{L}_1$ and LPIPS), and the high-spp savings are 27–40% on PSNR. Every saving follows the break-even condition: a method with denoise time d_D that needs s_D samples beats one with d_B and s_B when $c\,(s_B - s_D) > d_D - d_B$, where d is the scene-mean denoise time and c the scene-mean marginal render cost per sample (0.143 s). The condition is exact for the marginal block; in the isolated still, scene load and launch overhead are paid by both sides and cancel, and c becomes the mean cost per sample between s_D and s_B. One scene dominates c: *banner_06* costs 4.27 s per sample against a median of 0.017 s, 77% of the mean. Without it (38 scenes, quality and time both recomputed), the isolated-still savings from 32 to 512 spp are 13–32% on PSNR, 27–46% on $\mathcal{L}_1$ and 37–80% on LPIPS, and 22–38% on PSNR with fixed costs amortized; DEYES first leads on PSNR at 16 spp rather than 8.

Table 14: **Time to the fastest production baseline's quality**, MITSUBA, 39 scenes. At each rung the target is the best quality OIDN 2.5, OptiX 9 or NPPD (best of its four checkpoints) reaches at that sample count. "base s" is the least per-frame render + denoise time, in seconds, at which any of them reaches it (superscript: which one, O OIDN, X OptiX, N NPPD); "ours s" is the least time at which DEYES-S-UP4-*FT* or DEYES-T-UP2-*FT* reaches it (^{t}: DEYES-T-UP2-*FT* is the faster); "saved" $= 1-$ ours/base. Render time is the renderer's own time, without writing the frame or its buffers to disk. Upper block: an isolated still, paying scene load and the per-frame launch overhead (Figure 3); lower block: marginal sampling cost only, as an animation that amortizes fixed costs would (Figure 16). Each saving is specific to its metric.

	PSNR (dB) ↑			$\mathcal{L}_1$ ↓			LPIPS ↓		
rung	base s	ours s	saved	base s	ours s	saved	base s	ours s	saved
isolated still: scene load and per-frame launch overhead included, 39 scenes (Figure 3)									
1	1.11^O	1.16^t	-5%	1.09^X	1.14^t	-4%	1.11^O	1.14^t	-3%
2	1.25^O	1.29^t	-3%	1.23^X	1.16^t	+6%	1.25^O	1.14^t	+9%
4	1.54^O	1.54	-0%	1.52^X	1.30^t	+15%	1.52^X	1.17^t	+23%
8	2.11^O	1.93	+9%	2.10^X	1.56	+26%	2.10^X	1.32	+37%
16	3.26^O	2.65	+19%	3.25^X	1.97	+39%	3.25^X	1.52	+53%
32	5.57^O	4.06	+27%	5.56^X	2.86	+48%	5.56^X	2.05	+63%
64	10.20^O	6.65	+35%	10.19^X	4.61	+55%	10.19^X	3.19	+69%
128	19.20^X	11.93	+38%	19.20^X	8.83	+54%	19.20^X	5.09	+73%
256	37.53^X	24.47	+35%	37.53^X	18.44	+51%	37.53^X	8.12	+78%
512	73.89^X	54.15	+27%	73.89^X	46.83	+37%	73.89^X	12.62	+83%
marginal sampling cost only, 39 scenes (Figure 16)									
1	0.16^O	0.21^t	-30%	0.15^X	0.20^t	-32%	0.16^O	0.20^t	-19%
2	0.31^O	0.35^t	-12%	0.29^X	0.22^t	+26%	0.31^O	0.20^t	+36%
4	0.59^O	0.60	-1%	0.58^X	0.36^t	+38%	0.58^X	0.23^t	+60%
8	1.16^O	0.98	+16%	1.15^X	0.61	+47%	1.15^X	0.38	+67%
16	2.31^O	1.70	+26%	2.29^X	1.03	+55%	2.29^X	0.58	+75%
32	4.59^O	3.10	+33%	4.58^X	1.91	+58%	4.58^X	1.10	+76%
64	9.16^O	5.66	+38%	9.15^X	3.64	+60%	9.15^X	2.23	+76%
128	18.29^X	10.91	+40%	18.29^X	7.81	+57%	18.29^X	4.12	+77%
256	36.57^X	23.54	+36%	36.57^X	17.51	+52%	36.57^X	7.11	+81%
512	73.14^X	53.29	+27%	73.14^X	45.92	+37%	73.14^X	11.62	+84%

Table 15: **Time to the from-scratch network's quality**, MITSUBA, 39 scenes, as Table 14 with DEYES-S-UP4-*scratch* (the same network trained without pretraining, 40,000 steps) as the baseline and DEYES-S-UP4-*FT* alone as "ours": inference cost is identical, so every difference is samples. Of the six metrics, shown are PSNR and the three others with the largest mean saving over all 20 cells (PSNR 28%, $\mathcal{L}_1$ 29%, LPIPS 36% and SMAPE 27%); SSIM and MS-SSIM average 17% each. The from-scratch network's quality is not monotone in sample count on $\mathcal{L}_1$, LPIPS and SMAPE, so its cheapest route to its own quality at 128 and 256 spp is a lower sample count, which "base s" reflects.

	PSNR (dB) ↑			$\mathcal{L}_1$ ↓			LPIPS ↓			SMAPE ↓		
rung	base s	ours s	saved	base s	ours s	saved	base s	ours s	saved	base s	ours s	saved
isolated still: scene load and per-frame launch overhead included, 39 scenes (Figure 3)												
1	1.26	1.26	+0%	1.26	1.26	+0%	1.26	1.26	+0%	1.26	1.26	+0%
2	1.41	1.34	+5%	1.41	1.33	+5%	1.41	1.32	+6%	1.41	1.34	+5%
4	1.70	1.54	+9%	1.70	1.53	+10%	1.70	1.49	+12%	1.70	1.54	+9%
8	2.27	1.96	+13%	2.27	1.95	+14%	2.27	1.80	+21%	2.27	1.95	+14%
16	3.42	2.81	+18%	3.42	2.84	+17%	3.42	2.38	+30%	3.42	2.79	+18%
32	5.73	4.57	+20%	5.73	4.65	+19%	5.73	3.52	+39%	5.73	4.50	+21%
64	10.36	7.54	+27%	10.36	7.68	+26%	10.36	5.78	+44%	10.36	7.02	+32%
128	19.38	9.70	+50%	9.97	7.34	+26%	7.14	4.22	+41%	7.02	5.17	+26%
256	37.70	16.55	+56%	37.70	11.06	+71%	10.17	5.68	+44%	10.07	6.89	+32%
512	74.06	31.63	+57%	74.06	20.38	+72%	74.06	10.96	+85%	74.06	12.39	+83%
marginal sampling cost only, 39 scenes (Figure 16)												
1	0.32	0.32	+0%	0.32	0.32	+0%	0.32	0.32	+0%	0.32	0.32	+0%
2	0.46	0.40	+14%	0.46	0.39	+16%	0.46	0.38	+18%	0.46	0.40	+15%
4	0.75	0.59	+21%	0.75	0.59	+21%	0.75	0.54	+27%	0.75	0.60	+20%
8	1.32	1.02	+23%	1.32	1.01	+24%	1.32	0.86	+35%	1.32	1.00	+24%
16	2.46	1.86	+25%	2.46	1.89	+23%	2.46	1.44	+42%	2.46	1.84	+25%
32	4.75	3.61	+24%	4.75	3.68	+23%	4.75	2.56	+46%	4.75	3.53	+26%
64	9.32	6.54	+30%	9.32	6.68	+28%	9.32	4.80	+49%	9.32	6.02	+35%
128	18.46	8.66	+53%	8.94	6.34	+29%	6.15	3.26	+47%	6.02	4.20	+30%
256	36.75	15.59	+58%	36.75	10.03	+73%	9.13	4.70	+48%	9.03	5.90	+35%
512	73.31	30.68	+58%	73.31	19.46	+73%	73.31	9.92	+86%	73.31	11.37	+84%

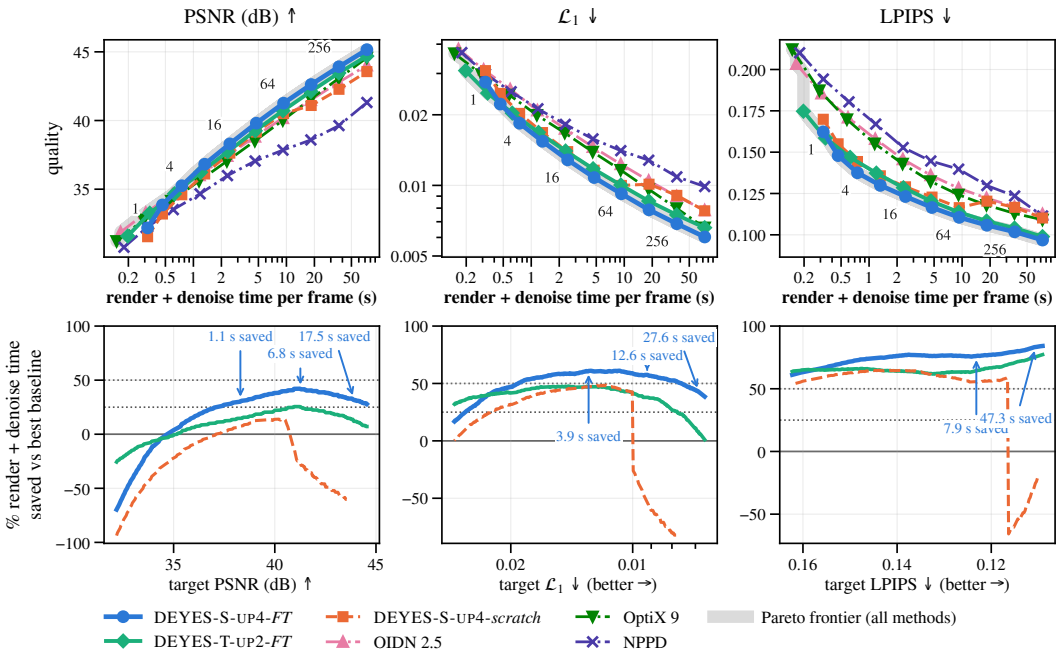

Figure 16: Figure 3 with render time counted as marginal sampling cost only (scene load and per-frame launch overhead removed, as an animation would amortize them), on the same 39 scenes. Denoise time then weighs more in a low-spp frame: the baselines are faster than DEYES-S-UP4-*FT* below 4 spp on $\mathcal{L}_1$ and LPIPS (below 2 spp for DEYES-T-UP2-*FT*) and at or below 4 spp on PSNR, while the savings at 32–512 spp are 27–40% PSNR, 37–60% $\mathcal{L}_1$ and 76–84% LPIPS.

M QUALITATIVE GALLERY

Crop selection. The crops in Figures 1 and 5 and in this appendix were chosen automatically from every 192^2 window of every scored frame (MITSUBA 1–64 spp, NB-WARM 2–32 spp). A window qualifies when its reference is neither mostly black nor clipped and is at least as detailed as the frame's 40th-percentile window; qualifying windows are ranked by how much absolute error DEYES removes relative to the *best* baseline on that window. These are illustrative best cases; Appendix N shows the windows where we lose.

Capacity, qualitatively. Figure 17 shows the four fully fine-tuned encoder sizes beside OIDN 2.5 and NPPD on the windows where the large model improves most over the tiny one while the tiny model already beats every baseline, at 1–4 spp. Large gains 1.5–2.5 dB over tiny on these crops; the ordering tiny $<$ small $<$ base $<$ large holds on six of the eight, and on the other two base falls below small by 0.2 and 0.05 dB. The differences show as texture and edge placement in the darkest, noisiest regions.

Figures 18–20 show 24 more rows, ordered by sample count; the DEYES column is our chosen model, DEYES-S-UP4-*FT*.

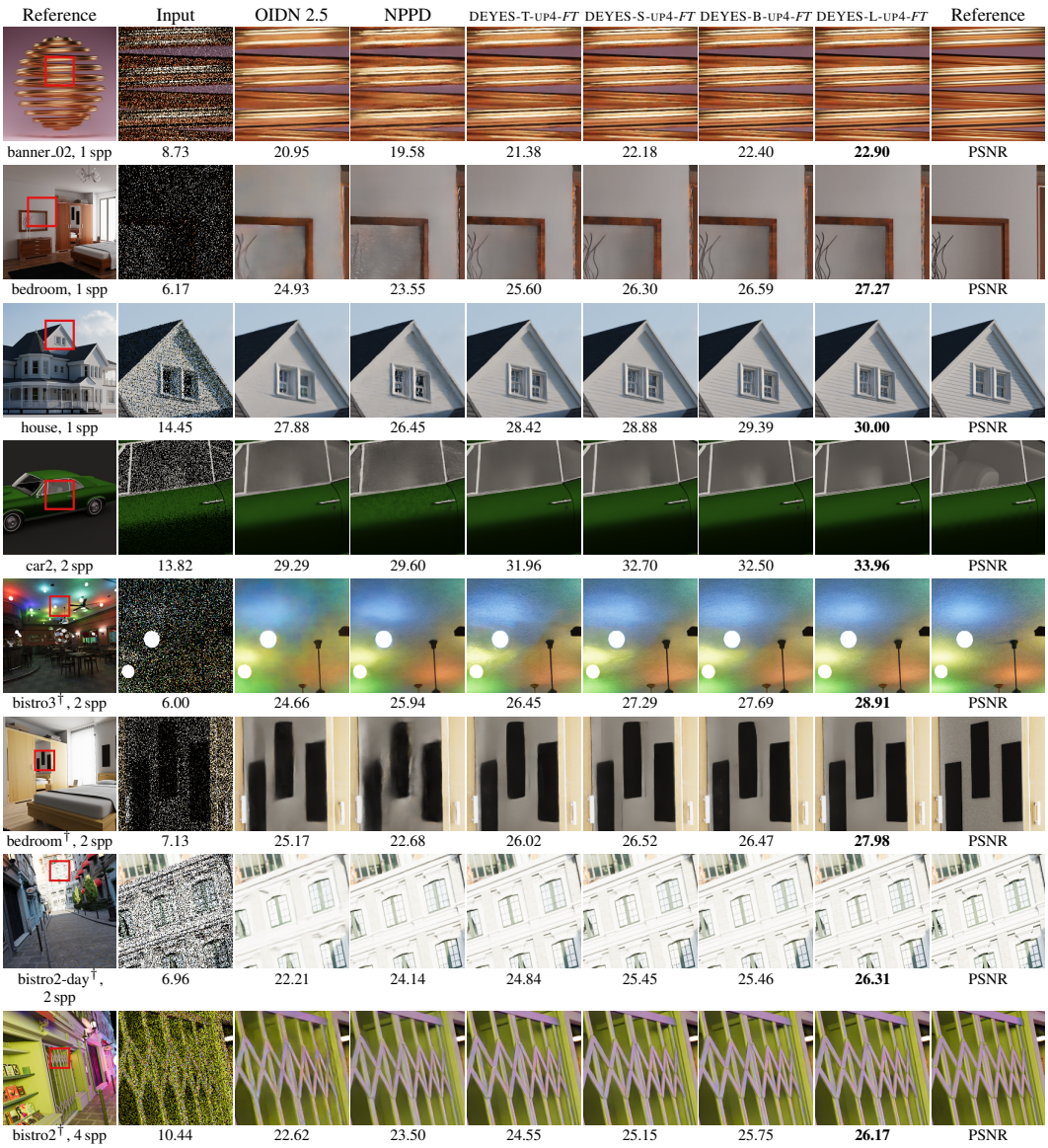

Figure 17: **Encoder sizes:** reference frame (crop in red), noisy input, OIDN 2.5, NPPD (on MITSUBA, the best of its four checkpoints for each crop), DEYES-{T,S,B,L}-UP4-*FT* and the reference; crop PSNR under each crop, best method in bold; † Noisebase with temporal history.

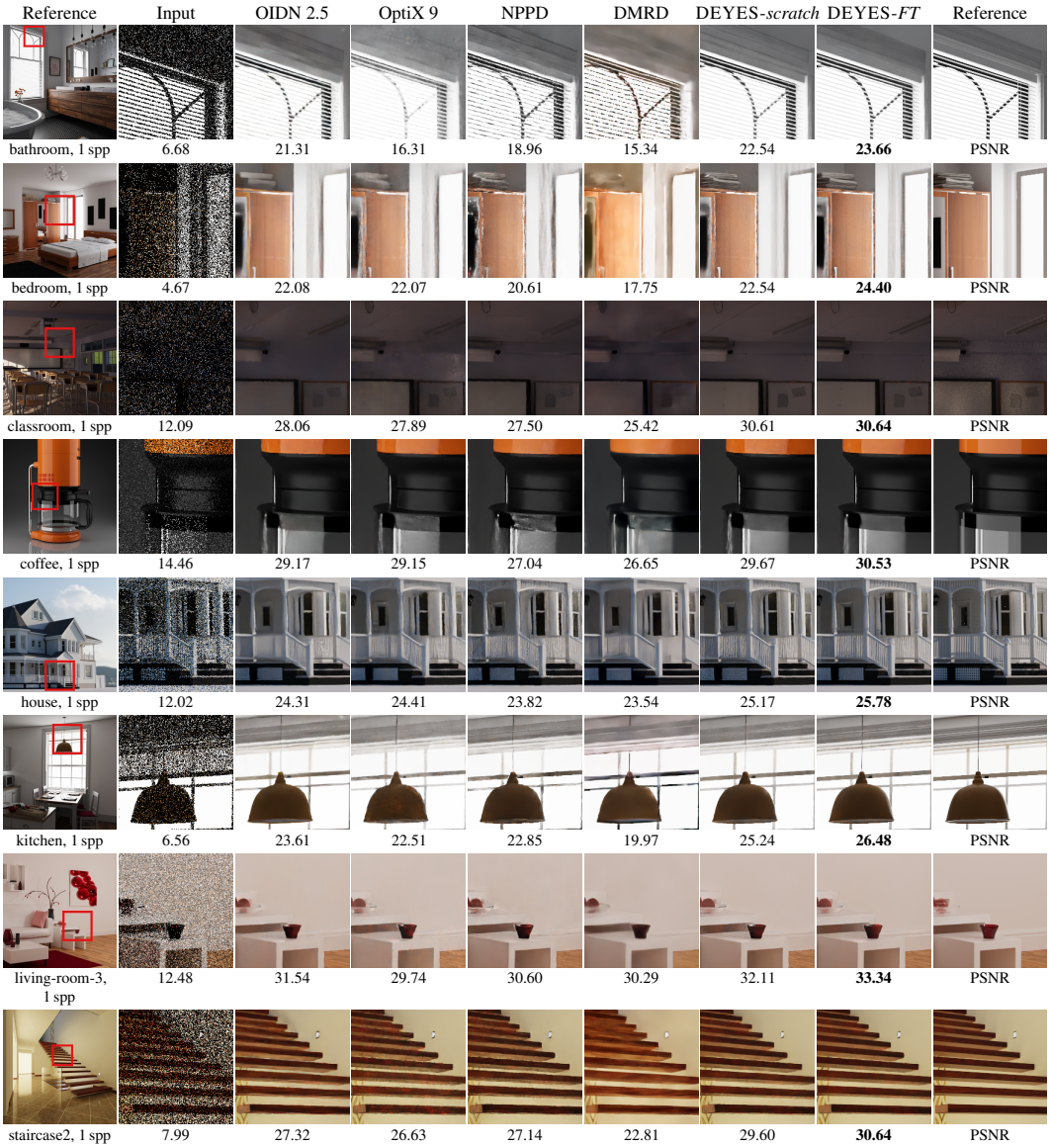

Figure 18: **Qualitative gallery (1/3):** MITSUBA at 1 spp. Columns as in Figure 5; crop PSNR under each crop, best method in bold.

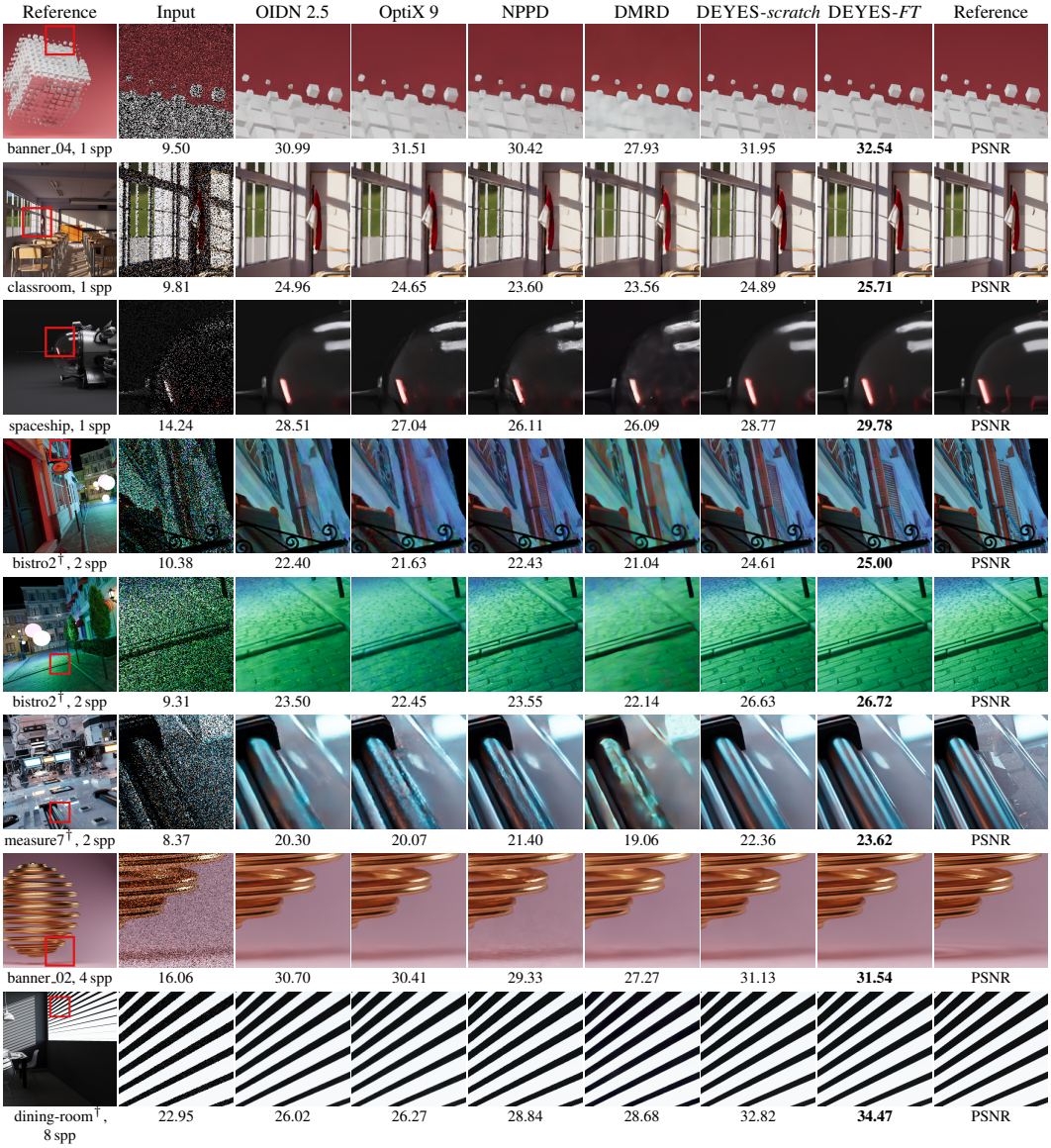

Figure 19: **Qualitative gallery (2/3):** 1–8 spp, MITSUBA and Noisebase (†, with temporal history). Columns as in Figure 5; crop PSNR under each crop, best method in bold.

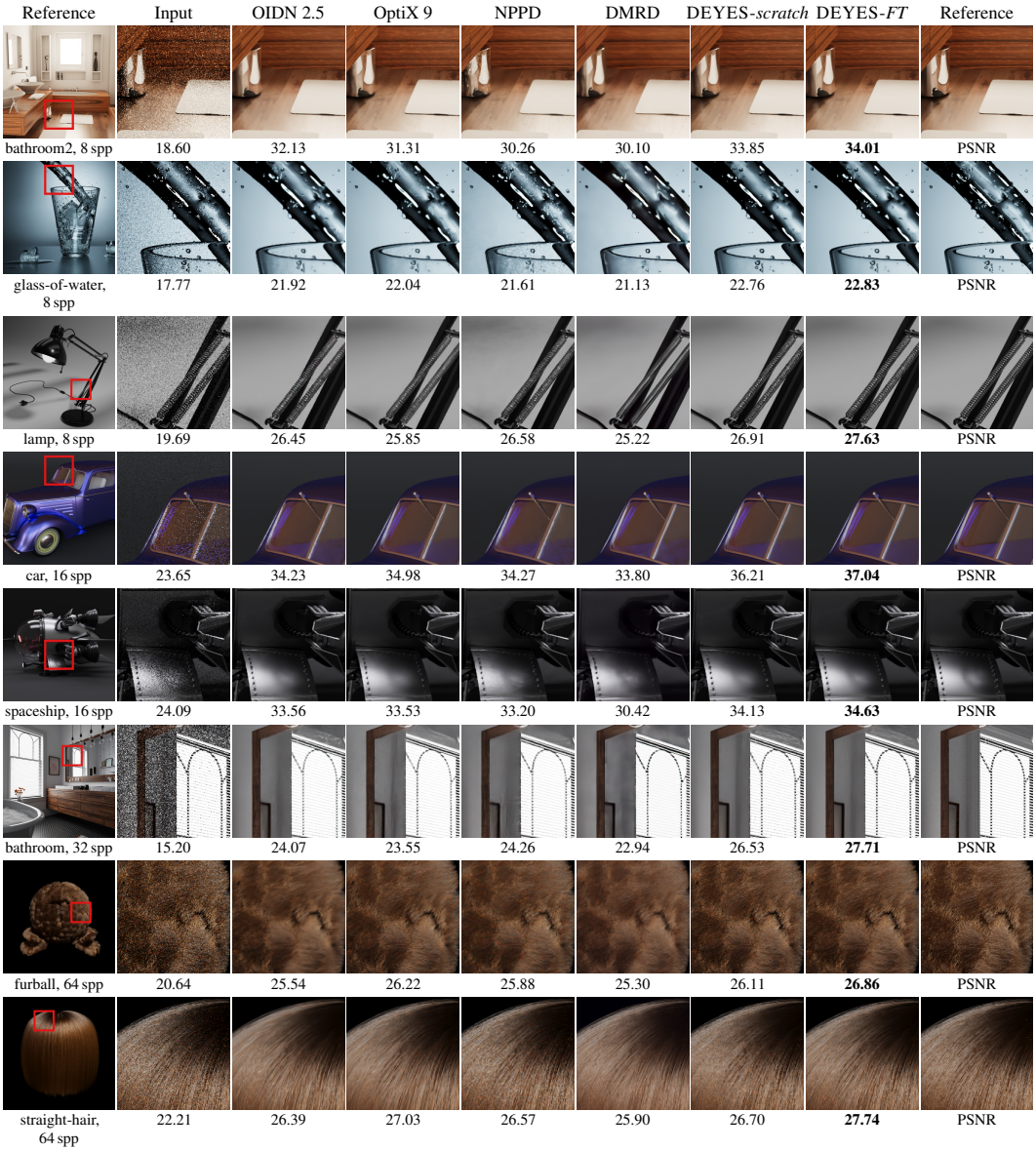

Figure 20: **Qualitative gallery (3/3):** MITSUBA at 8–64 spp. Columns as in Figure 5; crop PSNR under each crop, best method in bold.

N WHERE WE LOSE: FAILURE CASES

Cell count. A cell is (protocol, metric, rung) and counts when at least one external method (OIDN, OptiX, NPPD) has a number there: 108 cells over MITSUBA (six metrics, ten rungs) and NB-WARM and NB-COLD (six metrics, four rungs). The default model (three-seed mean) loses three: MITSUBA SSIM (0.9967 vs. 0.9968) and MS-SSIM (0.9958 vs. 0.9961) at 512 spp, and NB-WARM LPIPS at 32 spp (0.2544 vs. NPPD's 0.2526). Across the full-fine-tune size ladder (Figure 10), cold LPIPS at 32 spp is won by every size, warm LPIPS at 32 spp is won by large (0.2515) and missed by base by 0.02% (0.25266 vs. 0.25260), and the two 512-spp MITSUBA cells are lost by every size; base and large also lose MITSUBA SSIM and MS-SSIM at 256 spp, so large loses four cells and base five. The interactive DEYES-T-UP2-*FT* loses 11 cells, all at 1–8 spp or at the 512-spp and 32-spp cells above. DEYES-S-UP4-*LoRA* loses seven. Figure 21 shows eight windows where the best baseline beats DEYES by the largest visible margin (the ranking of Appendix M, reversed), chosen from the candidates to cover different failure modes.

Where the from-scratch network fails. On three dark, mostly indirectly lit interiors (*veach-ajar*, *bedroom*, *veach-bidir*), DEYES-S-UP4-*scratch* gets worse as the sample count rises past 64 spp, on all three seeds: on *veach-ajar* it falls from 31.9 dB at 64 spp to 23.5 dB at 128 spp, below its own 1-spp result, while DEYES-S-UP4-*FT* holds 35.0–35.6 dB. This causes the dip in DEYES-S-UP4-*scratch*'s SSIM and MS-SSIM at 128 spp (Table 8b). Without these scenes the pretraining gain on the MITSUBA mean is 0.87 dB instead of 1.01 dB (1.07–1.21 instead of 1.56–1.70 dB at 128–512 spp): part of the held-out gain is robustness to a failure mode the pretrained network did not exhibit.

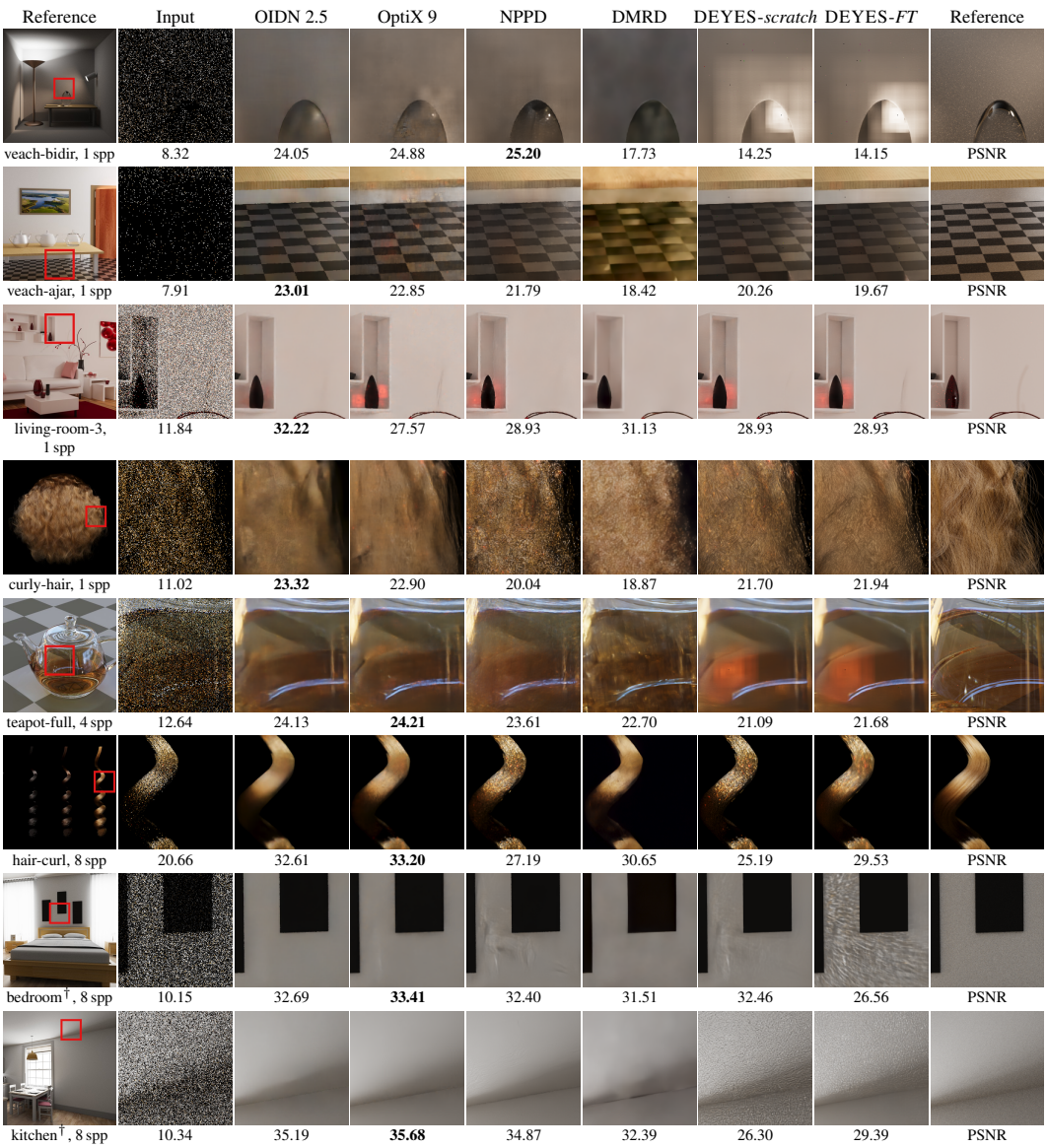

Figure 21: **Failure cases:** the windows where the best baseline beats DEYES by the largest visible margin, eight failure modes ordered by sample count. Columns as in Figure 5; crop PSNR under each crop, best method in bold; † Noisebase with temporal history.

O FULL SCALING GRIDS

Provenance and seeds. Data axis: EMA weights; compute axis: raw weights, because EMA's averaging window is not comparable across schedules of different length. Seeds per point: the pretrained frozen-trunk (LoRA) data and compute ladders, three seeds on all three protocols; the scratch data ladder, two; the scratch compute ladder, one (two at 10,000 steps, three at 40,000); the full fine-tune compute line, one, except three at 10,000 steps (MITSUBA) and three at 20,000 steps (MITSUBA; one on Noisebase). Bootstrap 95% intervals on the pretrained α resample seeds and are $[0.88, 1.12]$ (data) and $[1.01, 1.18]$ (compute). The scratch data points are drawn but not fitted, because the scratch compute line still climbs at 10,000 steps ($+0.31$ dB to 20,000).

Axes and provenance. Both axes of every scaling panel are in images: the data axis is the number of *unique* training images (training sequences $\times$ 64 frames; the sequence count is the upper tick row), the compute axis the number of images seen (optimizer steps $\times$ batch 64, A40-hours on top). Consecutive frames of a sequence are not independent examples, so the sequence count is a meaningful unit. The data panels fix the step budget: every run is a complete 10,000-step cosine schedule, so more data means fewer passes over each sequence, and the panels measure how well each initialization uses additional data at that budget, not a converged data-scaling curve. Only the pretrained data law is fitted; the from-scratch data points are drawn unfitted because that initialization is still improving at this horizon (its compute line climbs +0.31 dB from 10,000 to 20,000 steps while the frozen-trunk one moves -0.03). On the compute axis both initializations are trained to completion on 750–80,000-step ladders, with the full fine-tune as a third series. The from-scratch line peaks at 40,000 steps and changes by -0.03 dB at 80,000 (one run, below the three-seed 40,000-step mean on 6 of the 10 rungs, under raw and EMA weights) while its training loss keeps improving; a monotone $E + A\,x^{-\alpha}$ fit through it returns an asymptote (38.0 dB) above both of its highest measured points. We therefore report the measured separation of the best points: 1.04 dB between the full fine-tune (38.9 dB at 20,000 steps, three seeds) and scratch (37.9 dB at 40,000, three seeds), and 0.82 dB with the trunk frozen. The 80,000-step scratch rung is a single run; the separation is several times the spread of repeated runs (Appendix B), while the 40,000-to-80,000 change is not, so the scratch line is flat, not turning, past 40,000 steps. All panels, including the Noisebase columns and the per-rung grids, come from the same scored ladders.

Per-metric laws. Figures 23 and 24 repeat Figure 22 (PSNR) for $\mathcal{L}_1$ and SSIM, aggregated over rungs as in Appendix B; Figures 25–27 show $\mathcal{L}_1$ at every rung separately, with the per-rung baseline lines. SSIM is plotted as $1 - x$ so every non-PSNR panel is an error (lower is better, logarithmic y axis) and the $E + A\,x^{-\alpha}$ fit is on a decreasing quantity; above 64 spp the per-rung points are non-monotone in places and the fitted line is a level.

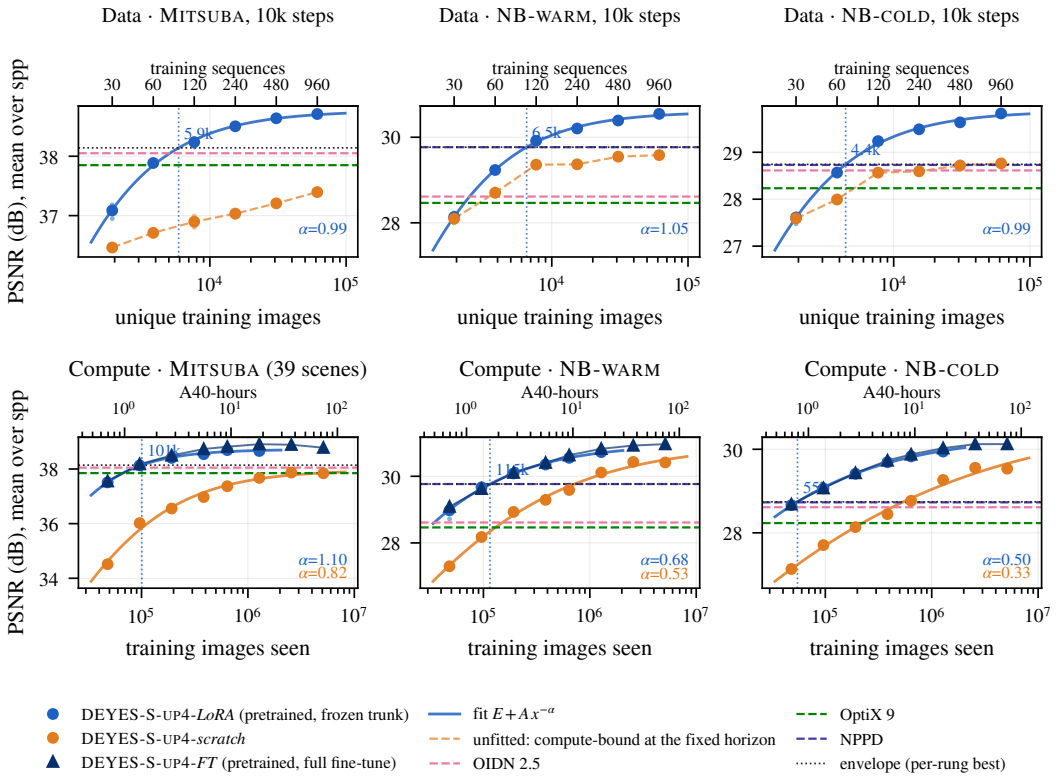

Figure 22: **Scaling laws on PSNR** (mean over rungs, as in Appendix B). Rows: data (unique training images; sequences of 64 frames on the upper ticks; both initializations at the same 10,000-step schedule) and compute (training images seen over complete 750–80,000-step schedules, A40-hours on top); columns: MITSUBA, NB-WARM, NB-COLD. Points are complete training runs of the small encoder from the pretrained trunk with the trunk frozen (blue, LoRA) and from scratch (orange); curves are $E + Ax^{-\alpha}$ fits on MSE, drawn over the measured range (the scratch data points are compute-bound at this horizon and are connected, not fitted). Navy triangles: the full fine-tune (every trunk weight trainable). Seeds: pretrained frozen-trunk points, three; scratch data, two; scratch compute, one (two at 10,000 steps); full fine-tune, one, except three at 10,000 and 20,000 steps on MITSUBA. Dashed horizontals are OIDN 2.5, OptiX 9 and NPPD (Noisebase only), the dotted black line their per-rung envelope; the dotted vertical marks where the pretrained fit crosses it.

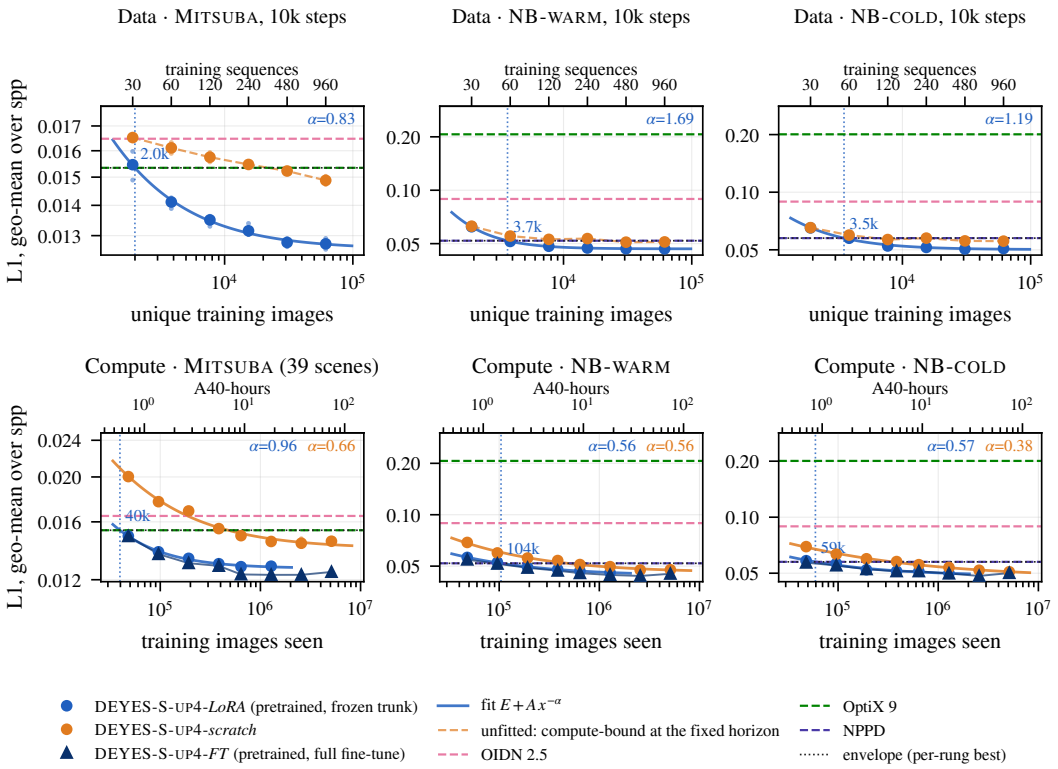

Figure 23: Scaling laws on $\mathcal{L}_1$ (geometric mean over rungs), data and compute axes, for MITSUBA, NB-WARM and NB-COLD; same runs, axes and fits as Figure 22.

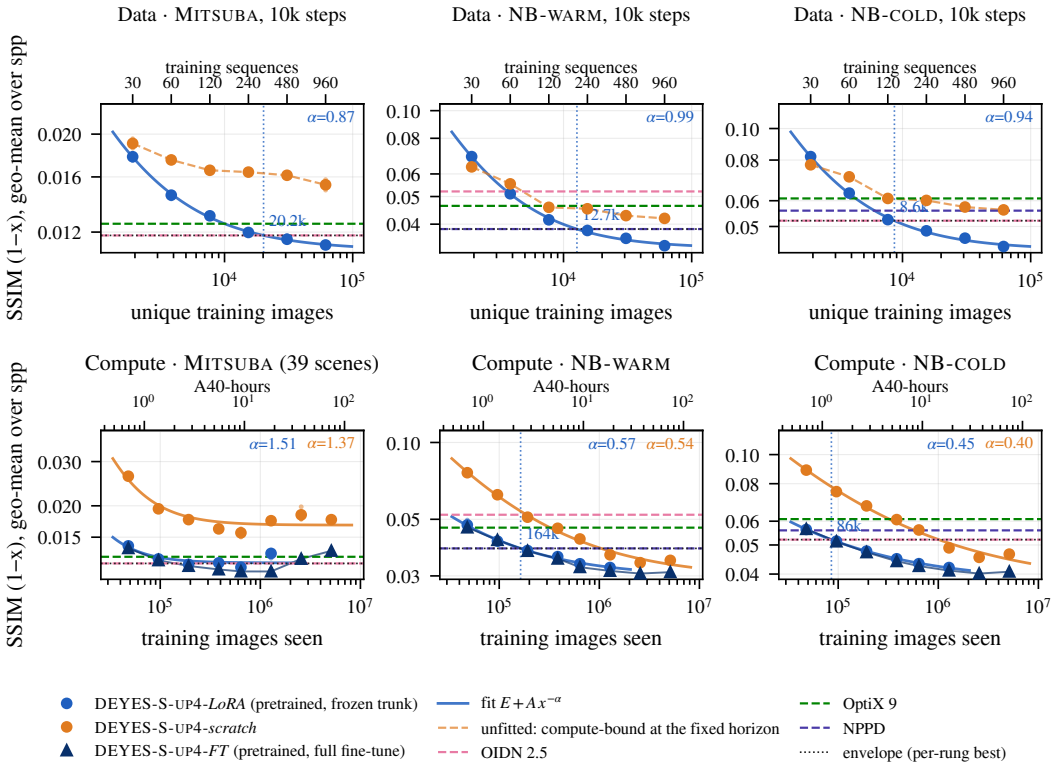

Figure 24: Scaling laws on SSIM, plotted as $1 - \mathrm{SSIM}$ (geometric mean over rungs), data and compute axes, for MITSUBA, NB-WARM and NB-COLD; same runs, axes and fits as Figure 22.

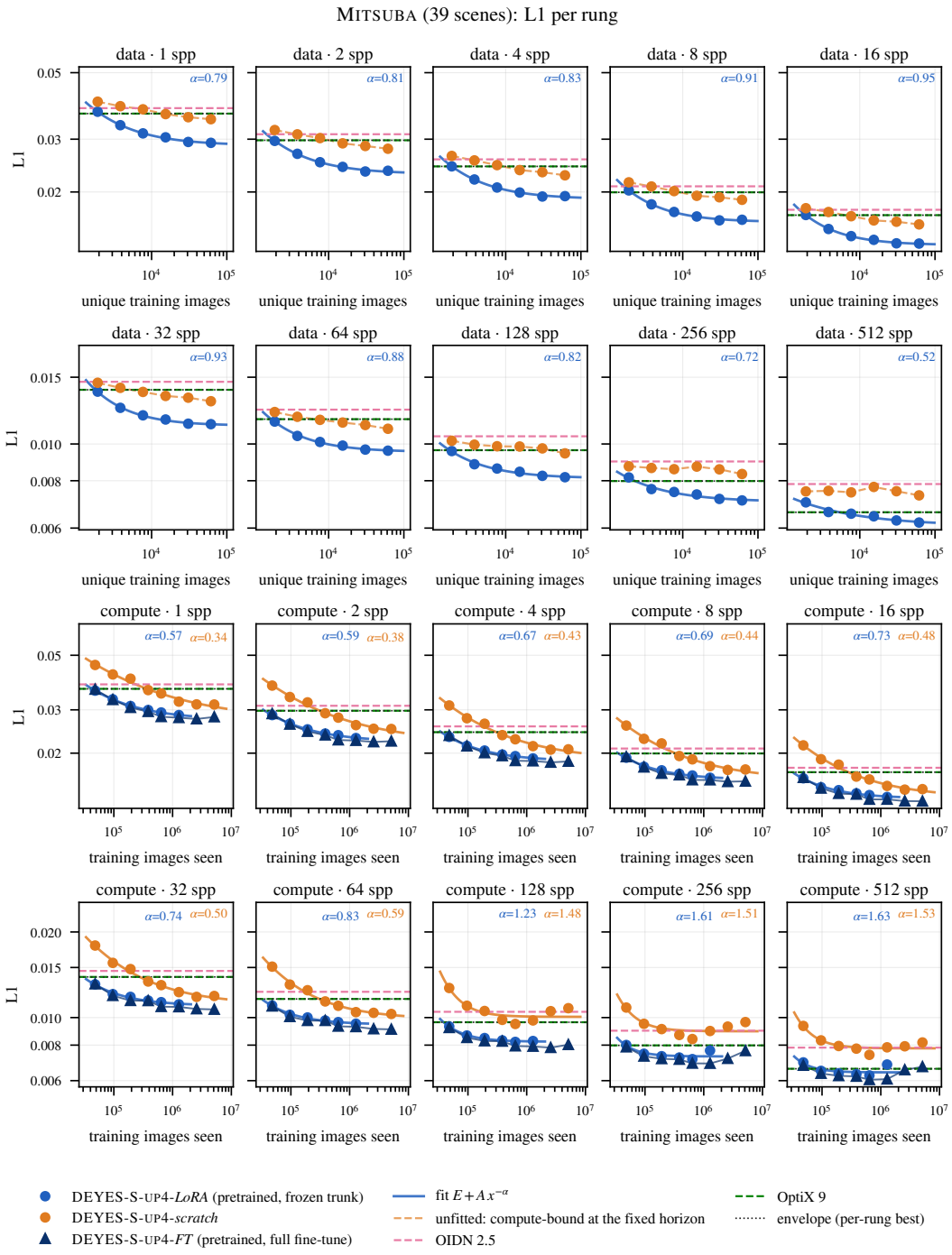

Figure 25: Per-rung scaling laws, $\mathcal{L}_1$, MITSUBA (39 scenes): data axis (top two rows, unique training images) and compute axis (bottom two, images seen), one panel per sample-count rung, with the per-rung OIDN, OptiX and envelope lines; scratch data points unfitted. Panels in a row share the y axis; navy triangles: the full fine-tune (compute rows).

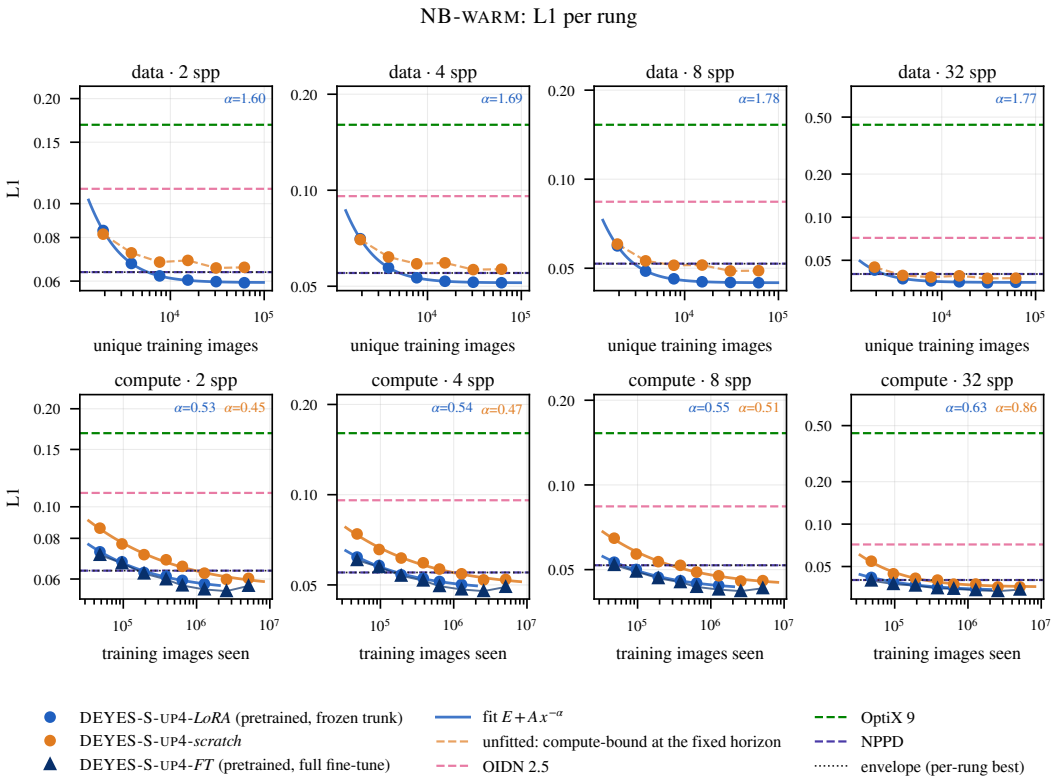

Figure 26: Per-rung scaling laws, $\mathcal{L}_1$, NB-WARM: data axis (top) and compute axis (bottom), one panel per rung, with the per-rung OIDN, OptiX, NPPD and envelope lines; navy triangles: the full fine-tune (compute rows).

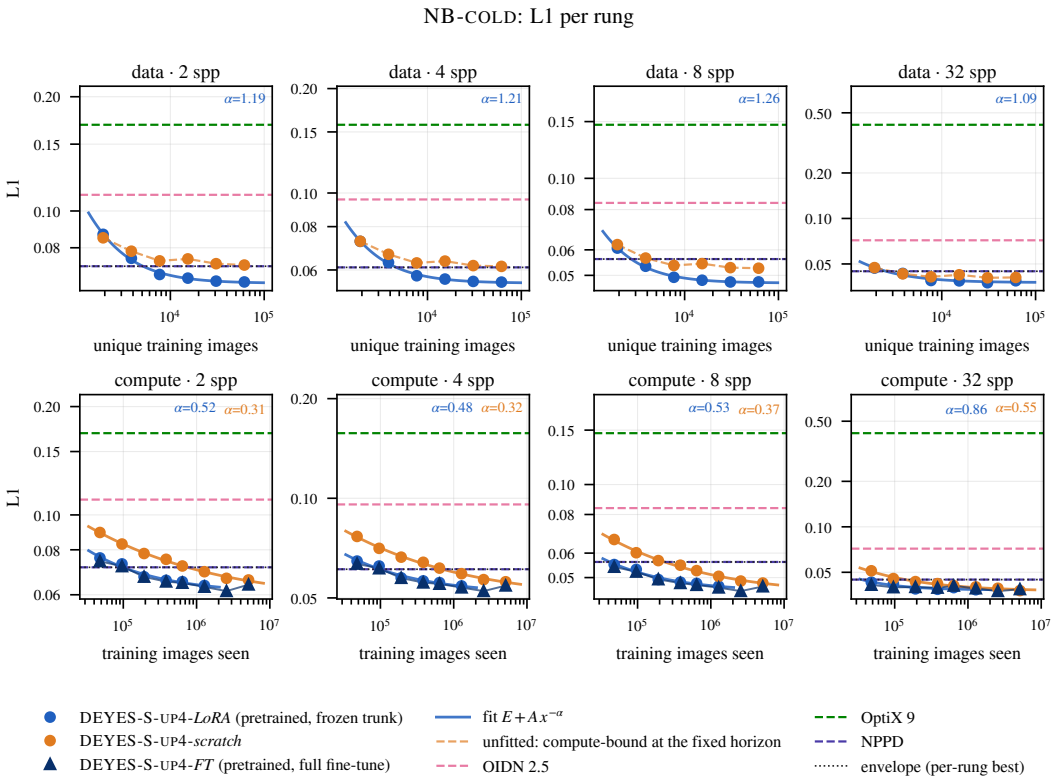

Figure 27: Per-rung scaling laws, $\mathcal{L}_1$, NB-COLD: data axis (top) and compute axis (bottom), one panel per rung, with the per-rung OIDN, OptiX, NPPD and envelope lines; navy triangles: the full fine-tune (compute rows).