Eurographics Symposium on Rendering (2025)
B. Wang and A. Wilkie (Guest Editors)

Supplementary Material: Neural Path Guiding with Distribution Factorization

Pedro Figueiredo, Qihao He and Nima Khademi Kalantari

Texas A&M University

1. Sampling Derivation

Here, we discuss the process of sampling from the learned PDF for both cases of nearest and linear interpolations. Given a randomly generated value between 0 and 1 with uniform distribution u the goal is to obtain ϵ' such that:

$$u = \int_0^{\epsilon'} \hat{p}_{\mathbf{w}}(\epsilon|\mathcal{C})\, \mathrm{d}\epsilon \tag{1}$$

Nearest Neighbor We start by breaking down the integral into two terms:

$$u = \int_0^{\frac{i}{M}} \hat{p}_{\mathbf{w}}(\epsilon|\mathcal{C})\, \mathrm{d}\epsilon + \int_{\frac{i}{M}}^{\epsilon'} \hat{p}_{\mathbf{w}}(\epsilon|\mathcal{C})\, \mathrm{d}\epsilon, \tag{2}$$

where i is the index of the first bin for which u is greater than the first integral term. Computing the second integral as the area of the bar (see Fig. 2 of the paper; top-left) corresponding to $\mathbf{v}[i]$ from i/M to ϵ' we have:

$$u = \hat{P}_{\mathbf{w}}(\frac{i}{M}|\mathcal{C}) + \mathbf{v}[i](\epsilon' - \frac{i}{M}), \tag{3}$$

where we have replaced the first integral in Eq. 8 with the CDF at i/M. Therefore, we can obtain ϵ' as follows:

$$\epsilon' = \frac{i}{M} + \frac{u - \hat{P}_{\mathbf{w}}(\frac{i}{M}|\mathcal{C})}{\mathbf{v}[i]}. \tag{4}$$

Linear Similar to the nearest neighbor case, we break down the integral in Eq. 1 into two terms:

$$u = \int_0^{\frac{i-0.5}{M}} \hat{p}_{\mathbf{w}}(\epsilon|\mathcal{C})\, \mathrm{d}\epsilon + \int_{\frac{i-0.5}{M}}^{\epsilon'} \hat{p}_{\mathbf{w}}(\epsilon|\mathcal{C})\, \mathrm{d}\epsilon, \tag{5}$$

where again i is the index of the first bin for which u is greater than the first integral term. Since $\hat{p}_{\mathbf{w}}(\epsilon|\mathcal{C})$ between the bounds of the second integral is linear, we can write it as:

$$u = \hat{P}_{\mathbf{w}}(\frac{i-0.5}{M}|\mathcal{C}) + \int_{\frac{i-0.5}{M}}^{\epsilon'} m\epsilon + b\, \mathrm{d}\epsilon, \tag{6}$$

where m and b are the slope and y-intercept of the line passing through $(\mathbf{v}[i-1], \frac{i-0.5}{M})$ and $(\mathbf{v}[i], \frac{i+0.5}{M})$. Calculating the integral we have:

$$u = \hat{P}_{\mathbf{w}}(\frac{i-0.5}{M}|\mathcal{C}) + \frac{m}{2}\epsilon'^2 - \frac{m}{2}(\frac{i-0.5}{M})^2 + b\epsilon' - b\frac{i-0.5}{M}. \tag{7}$$

We rewrite this equation by grouping the quadratic A, linear B, and constant C terms as follows:

$$u = A\epsilon'^2 + B\epsilon' + C. \tag{8}$$

This equation has two solutions, one of which is invalid. We choose the maximum of the two numbers if the slope m is positive, and the minimum of the two for negative m. Note that the boundary conditions where $i = 0$ or M is handled in a special manner depending on whether ϵ corresponds to θ or ϕ.

2. Equal-Sample Comparisons

Quantitative and Qualitative. We compare our technique against Müller et al. [MGN17] (PPG), Rath et al. [RGH*20] (Variance), Müller et al. [MMR*19] (NIS), and Dong et al. [DWL23] (NPM) in an equal sample count setting to show the effectiveness of our technique in learning the target distributions. We render each scene using 750 samples per pixel (spp) and report the trimmed relative mean squared error [RKZ11] (relMSE) with a threshold of 0.1% averaged over 10 independent runs in Table 7. As seen, our linear interpolation variant ranks first in all scenes, with nearest neighbor in second or third, depending on the scene. Additionally, our method with linear interpolation has a small overhead compared to nearest neighbor, but produces better results on all the scenes. We show visual comparisons on six selected scenes in Fig. 3. Our results consistently exhibit less noise than faster methods, and better results than NIS, demonstrating the effectiveness of our method in modeling the target distribution for a fixed number of samples.

Convergence Analysis. We further analyze the convergence behavior of our approach in an equal-sample setting in Fig. 1. Similar to our quantitative results, we observe best results for our approach in all scenes.

Table 1: *Equal-time comparison (120s) of our method with linear interpolation with (DF-L) and without (DF-L NC) radiance caching. Radiance caching impact varies by scene, with higher gains for scenes with difficult light paths.*

	DF-L NC		DF-L	
BATHROOM	0.2281	1231spp	0.2106	1131spp
BEDROOM	0.0153	1390spp	0.0138	1328spp
BREAKFAST	0.0183	2191spp	0.0171	2046spp
KITCHEN	0.0148	1445spp	0.0146	1382spp
SALLE DE BAIN	0.0173	1427spp	0.0155	1355spp
STAIRCASE	1.2844	1891spp	0.2861	1729spp
VEACH DOOR	1.2976	1875spp	0.4129	1454spp

Table 2: *Average runtime (in milliseconds) per operation for the training and guiding-only phases on the equal-sample (750spp)* SALLE DE BAIN *scene.*

	Training Phase		Guiding-only Phase	
	NIS	DF-L	NIS	DF-L
Method Forward	103	59.2	58.5	33.8
Method Backward	32.2	29.5	–	–
Cache Forward	–	20.5	–	–
Cache Backward	–	8.31	–	–

3. Radiance Caching Comparison

We analyze the impact of radiance caching via an equal-time comparison in Table 1. As seen, radiance caching improvements vary by scene, with higher gains in scenes with indirect lights such as VEACH DOOR.

4. Memory and Runtime Analysis

We collect detailed memory usage and runtime costs of our method during training and guiding-only phases for the BREAKFAST scene with 750spp.

Runtime. As seen in Table 3, the majority of the time budget (54%) in the training-phase is spent optimizing the guiding distribution p_Θ. Additionally, the training phase accounts for 70% of the total time spent to render 750spp, even though we only train for the first 250 samples. This training-time bottleneck highlights the importance of our input conditioning that allows for parallel execution when evaluating the PDF and computing gradients.

Memory. Table 4 shows that the additional buffers we statically allocate to enable fast radiance caching constitute a significant portion of the total memory usage (VRAM) in the training phase. As we only use radiance caching for optimization, the approach deallocates these buffers, greatly reducing the memory requirement for the guiding-only phase.

5. Additional Comparisons with NIS

We extensively compare our approach with NIS [MMR*19]. First, we describe the key differences of PDF evaluation and sampling in both methods, then we compare runtime metrics to show the impact of these differences in practice. Lastly, we match the input encoding of NIS and disable radiance caching to show that our method achieves equal expressiveness in less time than NIS.

Table 3: *Runtime (in seconds) for training and guiding-only phases on the 750spp render of the* BREAKFAST *scene.*

	Training Phase	Guiding-only Phase
Raytrace	4.8 (11%)	5.0 (27%)
Method Inference	8.3 (19%)	13.5 (73%)
Method Train	23.8 (54%)	–
Cache Inference	1.9 (4%)	–
Cache Train	5.4 (12%)	–
Total	44.2	18.5

Table 4: *Memory usage (in GB) for training and guiding-only phases on the 750spp render of the* BREAKFAST *scene.*

	Training Phase	Guiding-only Phase
Raytrace	3.3 (52%)	3.3 (80%)
Method	1.1 (17%)	0.8 (20%)
Cache	1.9 (30%)	–
Total	6.3	4.1

PDF Evaluation and Sampling. We detail the 2-dimensional PDF evaluation and sampling processes of our approach and NIS in Fig. 2. As seen, our approach models the PDF as a product of conditional and marginal PDFs in a parallel-friendly formulation. Specifically, MLP_2 is conditioned on ϵ_1 (provided during evaluation), requiring only a single synchronization step to compute the final product $\hat{p}_\Theta(\epsilon_1, \epsilon_2 | \mathbf{x}, \omega_o)$. During sampling, we first find ϵ_1 by inverse CDF sampling our marginal PDF (MLP_1), then ϵ_2 by performing an analogous operation on MLP_2. In contrast, NIS is formulated primarily as a sampling process that warps coordinates α_1, α_2 using coupling functions (modeled by MLPs) to obtain ϵ_1, ϵ_2. The PDF evaluation is a byproduct of this sampling process obtained by executing the same process in reverse order and inverse sampling operations. To compute the PDF, NIS requires sequential evaluation of both MLPs given the dependence on both α_1 and α_2. In practice, our parallel evaluation yields faster training inference (forward), and gradient computation (backward). During the guiding-only phase, our formulation also results in faster inference given the need to evaluate the PDF of BSDF-sampled paths during multiple importance sampling.

Runtime. We compare the runtime of NIS [MMR*19] and our method (DF-L) for the SALLE DE BAIN scene in Table 2. Specifically, we highlight the runtime cost of optimizing the method and cache during the training phase, and the cost of inference in the guiding-only phase. In both stages, our method performs inference and training faster than NIS on average, which contributes to the lower overall runtime shown in Table 7.

Expressiveness We compare the linear interpolation variant of our method *without* radiance caching to NIS with two piecewise

Table 5: *Equal-time comparison (120s) of increasing VMF counts for NPM [DWL23] using the relMSE metric. As seen, NPM does not benefit from additional VMFs above the author-recommended value of 8.*

	NPM (8 VMFs)		NPM (16 VMFs)		NPM (32 VMFs)	
BATHROOM	0.3761	1507spp	0.4602	1214spp	0.5638	975spp
BEDROOM	0.0182	1830spp	0.0205	1512spp	0.0265	1154spp
BREAKFAST	0.0286	3267spp	0.0404	2224spp	0.0584	1671spp
KITCHEN	0.0196	2091spp	0.0231	1467spp	0.0359	1157spp
SALLE DE BAIN	0.0233	2249spp	0.0303	1543spp	0.0392	1196spp
STAIRCASE	1.1155	1981spp	1.5215	1782spp	2.0558	1446spp
VEACH DOOR	1.8092	2828spp	3.8952	1824spp	6.5413	1422spp

quadratic coupling layers. For a fair comparison, both methods use the same one-blob [MMR*19] input encoding with 32 bins and same network configuration. We render scenes with 750 samples per pixel (spp) and report the relMSE metric in Fig. 4. As shown, our method without radiance caching produces results comparable to NIS, demonstrating equal expressiveness while being faster.

6. Additional Comparisons with NPM

VMF Count. In our comparisons, we model NPM [DWL23] using 8 VMFs, as specified by the authors. Here, we evaluate how NPM performs using an increasing number of VMFs that trade expressiveness for increased runtime. Table 5 shows that NPM with 8 VMFs performs best in all scenes in an equal-time comparison.

7. Product Sampling Assessment

We evaluate the impact of product sampling on the equal-time performance across all methods. To isolate this factor, we render a modified VEACH DOOR scene that uses only diffuse BSDFs, thus removing the influence of the product term with the BSDF. As shown in Table 6, the ranking of the methods is consistent with original VEACH DOOR discussed in the main paper. Notably, both versions of our method maintain the top two positions.

Table 6: *Equal-time comparison (120s) on a modified* VEACH DOOR *scene with only diffuse BSDFs (*DIFFUSE VD*), using the relMSE metric. As seen, the ranking of all methods remain similar to the equal-time results on the unmodified* VEACH DOOR *scene, indicating that learning the product with BSDF is not the defining reason for overall lower scores for neural (product-based) approaches.*

	PT	PPG	Variance	NIS	NPM	DF-N	DF-L
DIFFUSE VD	13.6338	3.2713	7.6464	1.9056	1.7107	0.4262	0.2306
VEACH DOOR	14.182	3.0940	9.3482	1.9415	1.8092	0.7132	0.4129

References

[DWL23] DONG H., WANG G., LI S.: Neural parametric mixtures for path guiding. In *ACM SIGGRAPH 2023 Conference Proceedings* (New York, NY, USA, 2023), SIGGRAPH '23, Association for Computing Machinery. 1, 3, 6

[MGN17] MÜLLER T., GROSS M., NOVÁK J.: Practical path guiding for efficient light-transport simulation. *Computer Graphics Forum 36*, 4 (2017), 91–100. 1, 6

[MMR*19] MÜLLER T., MCWILLIAMS B., ROUSSELLE F., GROSS M., NOVÁK J.: Neural importance sampling. *ACM Trans. Graph. 38*, 5 (oct 2019). 1, 2, 3, 6

[RGH*20] RATH A., GRITTMANN P., HERHOLZ S., VÉVODA P., SLUSALLEK P., KŘIVÁNEK J.: Variance-aware path guiding. *ACM Trans. Graph. 39*, 4 (aug 2020). 1, 6

[RKZ11] ROUSSELLE F., KNAUS C., ZWICKER M.: Adaptive sampling and reconstruction using greedy error minimization. *ACM Transactions on Graphics (TOG) 30*, 6 (2011), 1–12. 1

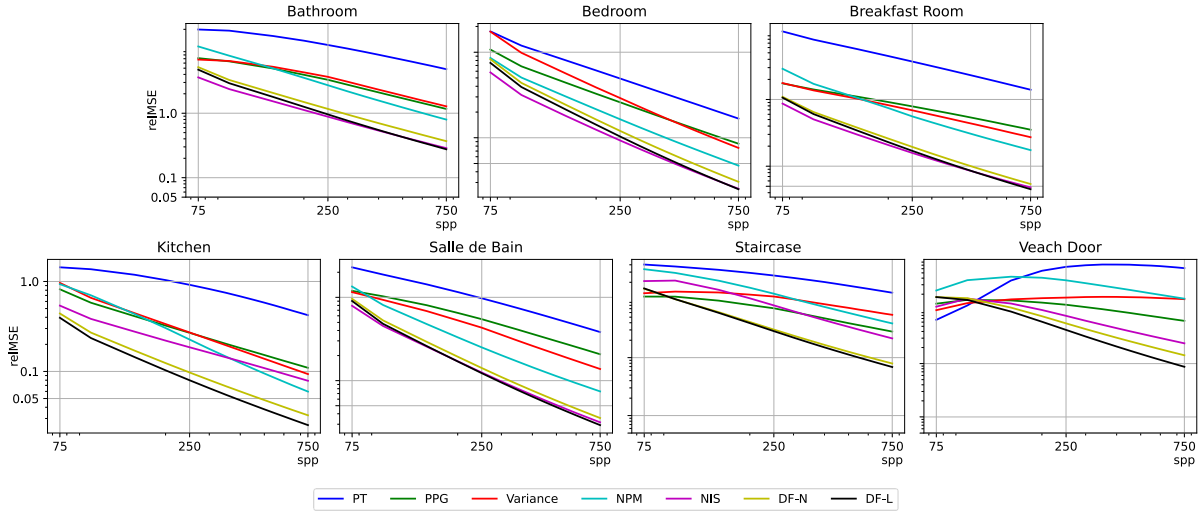

Figure 1: *We show convergence plot of all the approaches on the seven scenes from 75 to 750 spp. The 250 spp line is where all the approaches (except the unidirectional path tracer) stop learning and use the learned distributions to sample the remaining paths.*

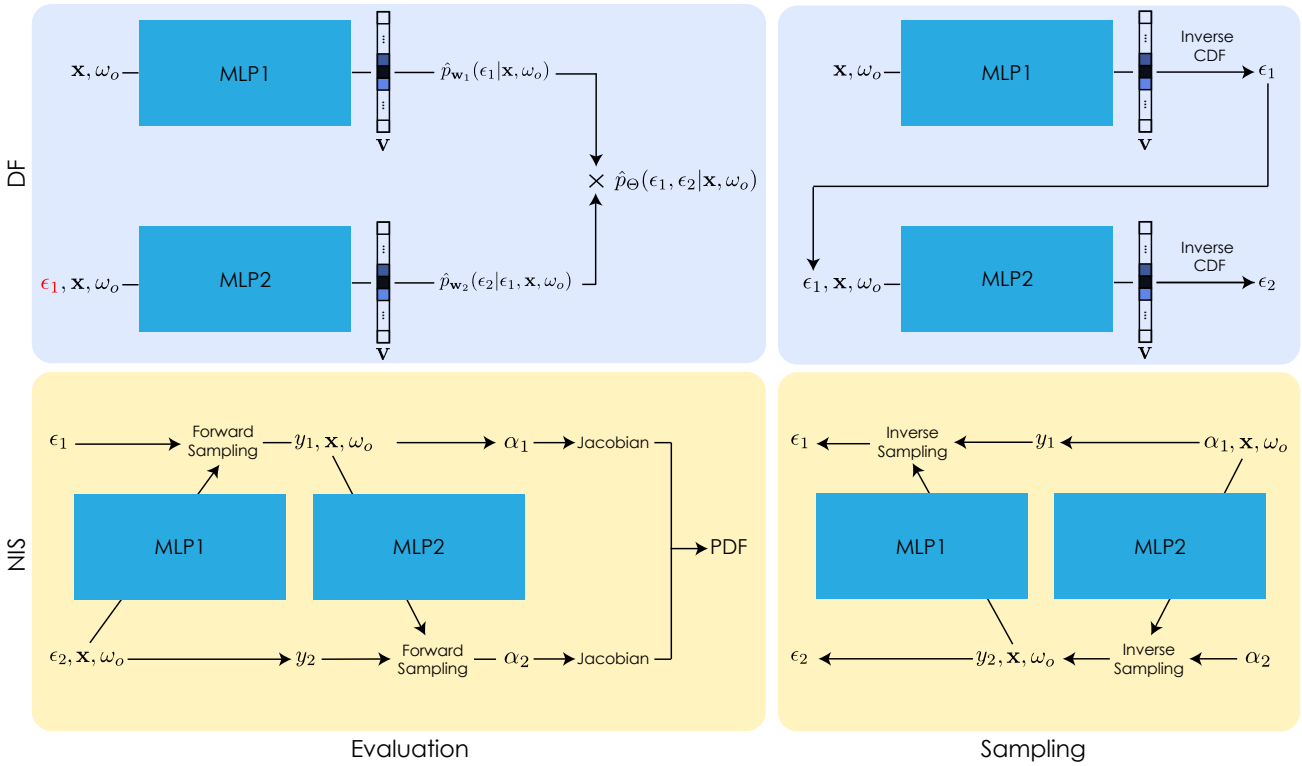

Figure 2: *PDF evaluation and sampling diagrams for our method (DF) and NIS in the 2-dimensional case. Our approach models the PDF as a product of conditional and marginal PDFs that can be evaluated in parallel since ϵ_1 (highlighted in red) is given. Sampling is performed by first finding ϵ_1 in the marginal PDF, then providing it as input to the conditional PDF to find ϵ_2. In contrast, NIS requires sequential evaluation of both MLPs to compute α_1 and α_2 and evaluate the PDF. NIS samples target ϵ_1, ϵ_2 from α_1, α_2 by performing consecutive inverse sampling operations in the reverse order of evaluation.*

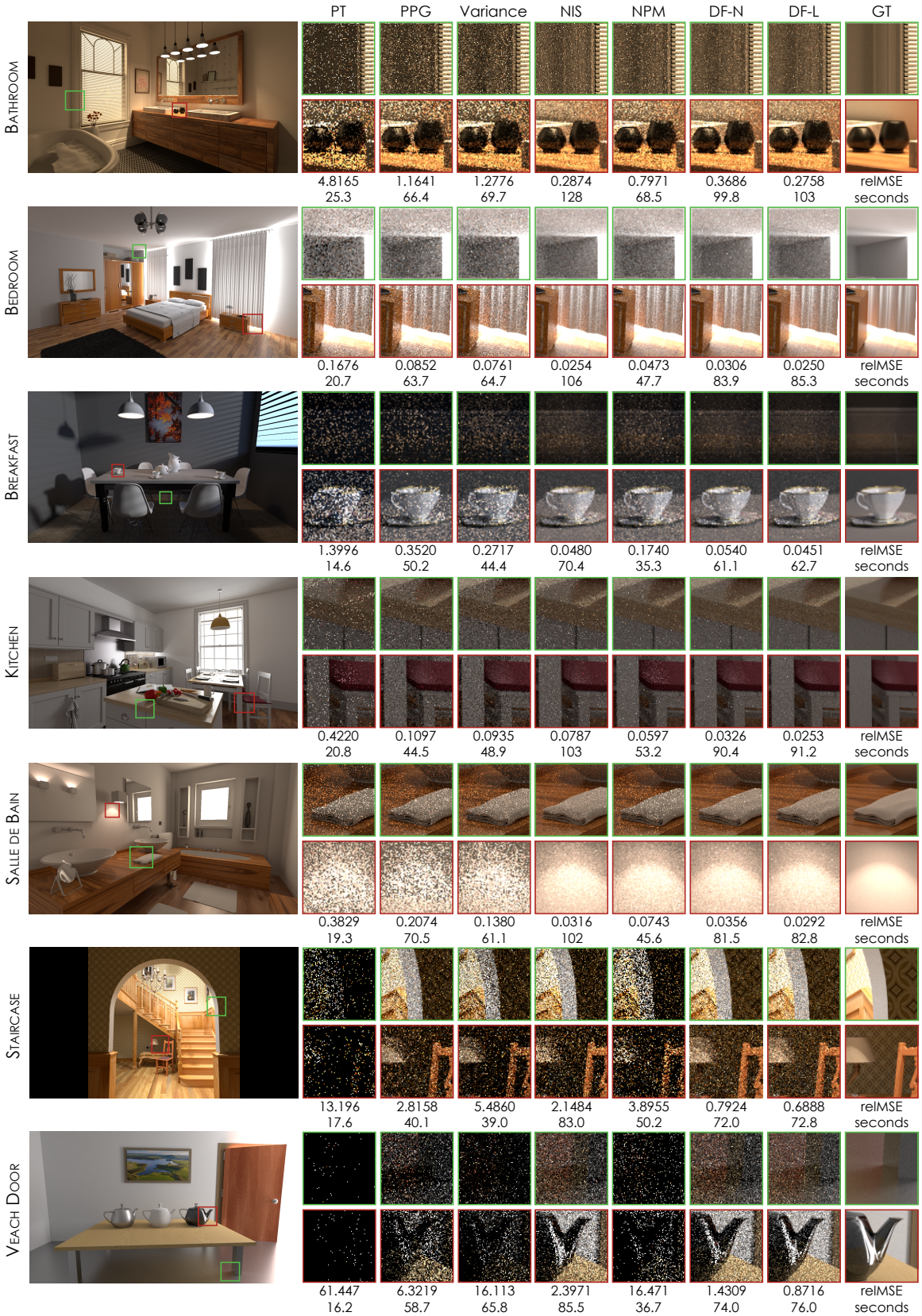

Figure 3: *Equal-sample comparison against several state-of-the-art methods. We use 750 spp for all the scenes, with a 30% training budget.*

Table 7: *Equal-spp comparisons measured in relMSE. We compare our approach with nearest neighbor (DF-N) and linear (DF-L) interpolation against unidirectional path tracing (PT), PPG [MGN17], Variance [RGH*20], NIS [MMR*19], and NPM [DWL23]. All the scenes are rendered using each approach for 750 samples per pixel. We color code the first, second, and third lowest numbers.*

	PT		PPG		Variance		NIS		NPM		DF-N		DF-L	
BATHROOM	4.8165	25.3s	1.1641	66.4s	1.2776	69.7s	0.2874	128s	0.7971	68.5s	0.3686	99.8s	0.2758	103s
BEDROOM	0.1676	20.7s	0.0852	63.7s	0.0761	64.7s	0.0254	106s	0.0473	47.7s	0.0306	83.9s	0.0250	85.3s
BREAKFAST	1.3996	14.6s	0.3520	50.2s	0.2717	44.4s	0.0480	70.4s	0.1740	35.3s	0.0540	61.1s	0.0451	62.7s
KITCHEN	0.4220	20.8s	0.1097	44.5s	0.0935	48.9s	0.0787	103s	0.0597	53.2s	0.0326	90.4s	0.0253	91.2s
SALLE DE BAIN	0.3829	19.3s	0.2074	70.5s	0.1380	61.1s	0.0316	102s	0.0743	45.6s	0.0356	81.5s	0.0292	82.8s
STAIRCASE	13.196	17.6s	2.8158	40.1s	5.4860	39.0s	2.1484	83.0s	3.8955	50.2s	0.7924	72.0s	0.6888	72.8s
VEACH DOOR	61.447	16.2s	6.3219	58.7s	16.113	65.8s	2.3971	85.5s	16.471	36.7s	1.4309	74.0s	0.8716	76.0s

Figure 4: *Equal-sample comparison of our method without radiance caching (DF-L NC) and NIS [MMR*19]. All scenes are rendered with 750 spp under a training budget of 30%. As shown, our approach achieves results comparable to NIS, demonstrating equal expressiveness while offering faster performance.*