Interactive Monte Carlo Denoising using Affinity of Neural Features – Supplemental Material

MUSTAFA IŞIK, Technical University of Munich, Germany
KRISHNA MULLIA, Adobe, California, USA
MATTHEW FISHER, Adobe, California, USA
JONATHAN EISENMANN, Adobe, California, USA
MICHAËL GHARBI, Adobe, California, USA

CCS Concepts: • **Computing methodologies** → **Ray tracing**; **Neural networks**.

Additional Key Words and Phrases: Monte Carlo denoising, deep learning

ACM Reference Format:
Mustafa Işık, Krishna Mullia, Matthew Fisher, Jonathan Eisenmann, and Michaël Gharbi. 2021. Interactive Monte Carlo Denoising using Affinity of Neural Features – Supplemental Material. *ACM Trans. Graph.* 40, 4, Article 37 (August 2021), 7 pages. https://doi.org/10.1145/3450626.3459793

1 TRAINING AND ARCHITECTURE DETAILS OF THE BASELINE METHODS

NBG, NBG-large and KPCN demodulate albedo before denoising and re-modulate afterwards. Our albedo buffer might be noisy as we store the values at the first non-specular hit. So, we replace the albedo of these noisy regions with white. We use this modified albedo only for demodulation and re-modulation. We feed the original albedo as input to the networks. For ONND, we tried both noisy and modified albedo as input. We present the results of the latter as it achieves better results in our experiments.

All the baselines are trained using ADAM optimizer with an initial learning rate of 10^{-4}. The loss is selected to be SMAPE. Trainings take approximately 300 to 400 epochs for NBG and NBG-large, and 600 to 750 epochs for the other baselines. We used the exact same dataset for all the baselines. Only NBG and NBG-large account for temporal stability. For other methods, we treated each frame of an animation sequence as a single image input.

We used the architectures reported by Munkberg and Hasselgren [2020] for NDLE and NDLE-ms. Our method and its ablations, NDLE, NDLE-ms and SBMC use 3 layers with 32 units for their fully-connected sample embedding networks. Modified U-net architecture of the SBMC is as follows:

$$\mathbf{c}112\mathbf{c}112\mathbf{d} \rightarrow \mathbf{c}224\mathbf{c}224\mathbf{d} \rightarrow$$
$$\mathbf{c}472\mathbf{c}472\mathbf{u} \rightarrow \mathbf{c}224\mathbf{c}224\mathbf{u} \rightarrow \mathbf{c}112\mathbf{c}32.$$

Note that SBMC employs three U-Nets [Gharbi et al. 2019] while other methods use only one.

2 RESULTS

We present the results for single-image denoising on Table 1 and for video denoising on Table 2. We provide plenty of metrics to be analyzed. Metrics that start with the prefix *Temporal* measures the quality of temporal stability. They are always applied between the temporal gradients of the prediction and the temporal gradients of the ground truth. Moreover, metrics that require tonemapped images (e.g. SSIM, PSNR), calculate temporal gradients on the tonemapped predictions and tonemapped ground truth images. In Table 3, we analyze the effect of the kernel size quantitatively. Table 4 shows the runtime and storage cost of affinity-based kernels (ours) and gather kernels. In Figure 1, we provide a few examples to demonstrate how affinity-based kernels are superior to gather kernels at preserving high-frequency details. Table 5 provides a numerical comparison between our method and two ablations that introduce alternatives to our sample embedding mechanism. In Figure 2, we observe the impact of using series of dilated kernels instead of using a single large kernel.

REFERENCES

Michaël Gharbi, Tzu-Mao Li, Miika Aittala, Jaakko Lehtinen, and Frédo Durand. 2019. Sample-based Monte Carlo denoising using a kernel-splatting network. *ACM Transactions on Graphics (TOG)* 38, 4 (2019), 1–12.

Jacob Munkberg and Jon Hasselgren. 2020. Neural Denoising with Layer Embeddings. In *Computer Graphics Forum*, Vol. 39. Wiley Online Library, 1–12.

Authors' addresses: Mustafa Işık, Technical University of Munich, Munich, Germany, m.isik@tum.de; Krishna Mullia, Adobe, San Jose, California, USA, mulliala@adobe.com; Matthew Fisher, Adobe, San Francisco, California, USA, matfishe@adobe.com; Jonathan Eisenmann, Adobe, San Francisco, California, USA, j@adobe.com; Michaël Gharbi, Adobe, San Francisco, California, USA, mgharbi@adobe.com.

© 2021 Association for Computing Machinery.
This is the author's version of the work. It is posted here for your personal use. Not for redistribution. The definitive Version of Record was published in *ACM Transactions on Graphics*, https://doi.org/10.1145/3450626.3459793.

Table 1. **Results of static-image denoising.** All the metrics are averaged over the scenes for each method. The first and the second best results are highlighted for each evaluation metric. Higher PSNR and SSIM, and lower L_2 and SMAPE are better.

		ONND	NBG	NBG-large	KPCN	SBMC	NDLE	ours-single	ours
2 spp	Relative L_2	0.1176	369.08	30.396	0.0692	0.0629	0.0821	0.0508	0.0467
	SMAPE	0.0776	0.1115	0.0758	0.0871	0.0745	0.0738	0.0605	0.0583
	PSNR	29.404	26.042	28.580	27.950	28.975	30.089	30.935	30.932
	SSIM	0.8502	0.7798	0.8425	0.8277	0.8470	0.8615	0.8761	0.8805
4 spp	Relative L_2	0.0944	257.24	18.458	0.0494	0.0473	0.0579	0.0460	0.0451
	SMAPE	0.0690	0.0865	0.0643	0.0624	0.0556	0.0580	0.0513	0.0513
	PSNR	30.620	27.584	30.091	30.863	32.339	32.207	32.677	32.526
	SSIM	0.8671	0.8204	0.8669	0.8731	0.8906	0.8898	0.8971	0.8962
8 spp	Relative L_2	0.0764	197.58	13.110	0.0614	0.0408	0.0438	0.0456	0.0456
	SMAPE	0.0620	0.0730	0.0578	0.0518	0.0484	0.0491	0.0483	0.0482
	PSNR	31.793	28.527	31.204	32.796	33.973	33.630	33.669	33.537
	SSIM	0.8820	0.8438	0.8825	0.8978	0.9079	0.9036	0.9057	0.9053
16 spp	Relative L_2	0.0664	198.41	31.107	0.0486	0.0350	0.0378	0.0458	0.0460
	SMAPE	0.0563	0.0667	0.0535	0.0484	0.0462	0.0450	0.0465	0.0463
	PSNR	32.889	29.116	31.947	33.903	34.577	34.455	34.343	34.252
	SSIM	0.8946	0.8564	0.8931	0.9082	0.9127	0.9107	0.9110	0.9118
32 spp	Relative L_2	0.0595	215.47	24.431	0.0518	0.0324	0.0379	0.0463	0.0457
	SMAPE	0.0515	0.0638	0.0500	0.0470	0.0466	0.0435	0.0451	0.0447
	PSNR	33.913	29.429	32.589	34.543	34.441	34.733	34.817	34.785
	SSIM	0.9054	0.8628	0.9013	0.9120	0.9144	0.9142	0.9149	0.9170

Table 2. **Results of video denoising.** All the metrics are averaged over the scenes for each method. The first and the second best results are highlighted for each evaluation metric. Higher PSNR and SSIM, and lower L_1, L_2, TRMAE and SMAPE are better.

		ONND	NBG	NBG-L	NDLE	NDLE-ms	ours-$K1$	ours-single	ours
2 spp	Relative L_2	0.0835	534.83	247.19	0.0315	0.0341	0.0272	0.0212	0.0208
	SMAPE	0.0667	0.1062	0.0747	0.0650	0.0614	0.0653	0.0532	0.0498
	PSNR	30.669	26.676	29.318	30.985	31.024	30.487	32.179	32.341
	SSIM	0.8174	0.7626	0.8145	0.8298	0.8212	0.8030	0.8394	0.8454
	TRMAE	1.9881	2.0583	1.0691	0.7644	0.8066	0.8035	0.6404	0.5170
	Temporal Relative L_2	4.6321	1339.1	631.91	0.6288	0.7569	0.6398	0.4267	0.3574
	Temporal SMAPE	0.3637	0.3662	0.3376	0.3361	0.3421	0.3457	0.3213	0.3105
	Temporal PSNR	34.252	31.394	33.560	34.711	34.316	33.951	35.499	35.977
	Temporal SSIM	0.8763	0.8270	0.8722	0.8859	0.8784	0.8654	0.8942	0.8990
4 spp	Relative L_2	0.0549	453.32	202.67	0.0189	0.0204	0.0182	0.0164	0.0141
	SMAPE	0.0593	0.0813	0.0633	0.0517	0.0512	0.0526	0.0464	0.0444
	PSNR	31.749	28.487	30.679	33.056	32.910	32.642	33.672	33.936
	SSIM	0.8297	0.7933	0.8292	0.8469	0.8407	0.8316	0.8511	0.8550
	TRMAE	1.6939	1.6956	1.0011	0.6180	0.6625	0.6504	0.5566	0.4671
	Temporal Relative L_2	3.2075	1117.0	487.43	0.3088	0.3922	0.3436	0.2732	0.1777
	Temporal SMAPE	0.3474	0.3486	0.3245	0.3139	0.3203	0.3209	0.3070	0.3009
	Temporal PSNR	35.253	32.660	34.653	36.478	36.096	35.858	36.832	37.279
	Temporal SSIM	0.8869	0.8538	0.8849	0.9017	0.8965	0.8900	0.9049	0.9083
8 spp	Relative L_2	0.0366	397.72	139.30	0.0146	0.0145	0.0141	0.0140	0.0120
	SMAPE	0.0534	0.0692	0.0571	0.0444	0.0461	0.0462	0.0431	0.0416
	PSNR	32.748	29.569	31.608	34.375	33.961	33.979	34.542	34.814
	SSIM	0.8397	0.8087	0.8376	0.8549	0.8498	0.8463	0.8569	0.8602
	TRMAE	1.4681	1.4536	0.9066	0.5718	0.5951	0.5656	0.5116	0.4463
	Temporal Relative L_2	2.2628	970.56	345.03	0.6803	0.2807	0.2293	0.2041	0.1361
	Temporal SMAPE	0.3340	0.3382	0.3155	0.3017	0.3077	0.3053	0.2982	0.2944
	Temporal PSNR	36.166	33.396	35.426	37.526	37.115	37.110	37.675	38.047
	Temporal SSIM	0.8955	0.8667	0.8923	0.9087	0.9045	0.9023	0.9102	0.9131

Table 3. **Ablation on the kernel size.** When working with fewer samples (2spp), filtering over larger neighborhoods is beneficial. As the number of samples increases, the advantage of large kernels diminishes. The kernel size reported are the number of taps in *each* of the spatial kernels. This evaluation uses ours-single model on static images.

		7×7	9×9	11×11	13×13	15×15	17×17	19×19
2 spp	Relative L_2	0.0582	0.0540	0.0520	0.0508	0.0501	0.0502	0.0501
	SMAPE	0.0676	0.0638	0.0617	0.0605	0.0597	0.0593	0.0591
	PSNR	30.194	30.575	30.796	30.935	31.024	31.079	31.113
	SSIM	0.8603	0.8695	0.8734	0.8761	0.8775	0.8785	0.8790
4 spp	Relative L_2	0.0486	0.0468	0.0461	0.0460	0.0461	0.0466	0.0471
	SMAPE	0.0542	0.0523	0.0516	0.0513	0.0513	0.0515	0.0518
	PSNR	32.260	32.509	32.624	32.677	32.694	32.686	32.664
	SSIM	0.8895	0.8945	0.8962	0.8971	0.8974	0.8975	0.8973
8 spp	Relative L_2	0.0461	0.0454	0.0453	0.0456	0.0459	0.0465	0.0470
	SMAPE	0.0491	0.0483	0.0481	0.0483	0.0485	0.0488	0.0492
	PSNR	33.492	33.632	33.670	33.669	33.644	33.602	33.550
	SSIM	0.9025	0.9051	0.9056	0.9057	0.9054	0.9051	0.9047
16 spp	Relative L_2	0.0457	0.0453	0.0455	0.0458	0.0462	0.0467	0.0471
	SMAPE	0.0464	0.0461	0.0462	0.0465	0.0468	0.0472	0.0476
	PSNR	34.339	34.395	34.380	34.343	34.293	34.232	34.165
	SSIM	0.9100	0.9112	0.9112	0.9110	0.9105	0.9101	0.9096
32 spp	Relative L_2	0.0457	0.0457	0.0460	0.0463	0.0467	0.0472	0.0476
	SMAPE	0.0444	0.0445	0.0448	0.0451	0.0455	0.0459	0.0464
	PSNR	34.951	34.939	34.882	34.817	34.745	34.669	34.591
	SSIM	0.9152	0.9157	0.9153	0.9149	0.9143	0.9138	0.9133

Table 4. **Impact of the kernel size for predicted and affinity-based kernels on memory and runtime requirements.** Our method uses affinity-based kernels, and the *predicted* denotes the ablation that replaces our affinity-based kernels with gather kernels. All versions are timed on cuDNN implementation for 4spp inputs with 1920×1024 resolution on a GeForce RTX 2080 Ti graphics card. The timings are averaged over 120 frames of the *Bedroom* scene. We use 3 spatial and 1 temporal kernels for each version.

		5×5	7×7	9×9	11×11	13×13
Runtime (ms)	affinity-based	34.17	36.36	40.08	**45.91**	**52.74**
	predicted	**33.65**	**35.85**	**39.90**	46.45	53.42
Memory (GB)	affinity-based			**0.25**		
	predicted	0.77	1.51	2.50	3.74	5.22

Table 5. **Quantitative evaluation of different pixel embedding generation strategies.** We test the ablations and our method on single-image denoising test scenes and average the results. The first and the second best results are highlighted for each evaluation metric. Higher PSNR and SSIM, and lower Relative L_2 and SMAPE are better.

		ours-me	ours-pixel	ours-single
2 spp	Relative L_2	0.0489	0.0520	0.0508
	SMAPE	0.0582	0.0663	0.0605
	PSNR	30.998	29.974	30.935
	SSIM	0.8814	0.8669	0.8761
4 spp	Relative L_2	0.0468	0.0437	0.0460
	SMAPE	0.0519	0.0539	0.0513
	PSNR	32.503	32.209	32.677
	SSIM	0.8959	0.8907	0.8971
8 spp	Relative L_2	0.0475	0.0412	0.0456
	SMAPE	0.0487	0.0476	0.0483
	PSNR	33.545	33.639	33.669
	SSIM	0.9059	0.9053	0.9057
16 spp	Relative L_2	0.0473	0.0410	0.0458
	SMAPE	0.0470	0.0445	0.0465
	PSNR	33.983	34.581	34.343
	SSIM	0.9128	0.9148	0.9110
32 spp	Relative L_2	0.0482	0.0418	0.0463
	SMAPE	0.0462	0.0429	0.0451
	PSNR	33.931	35.181	34.817
	SSIM	0.9171	0.9205	0.9149

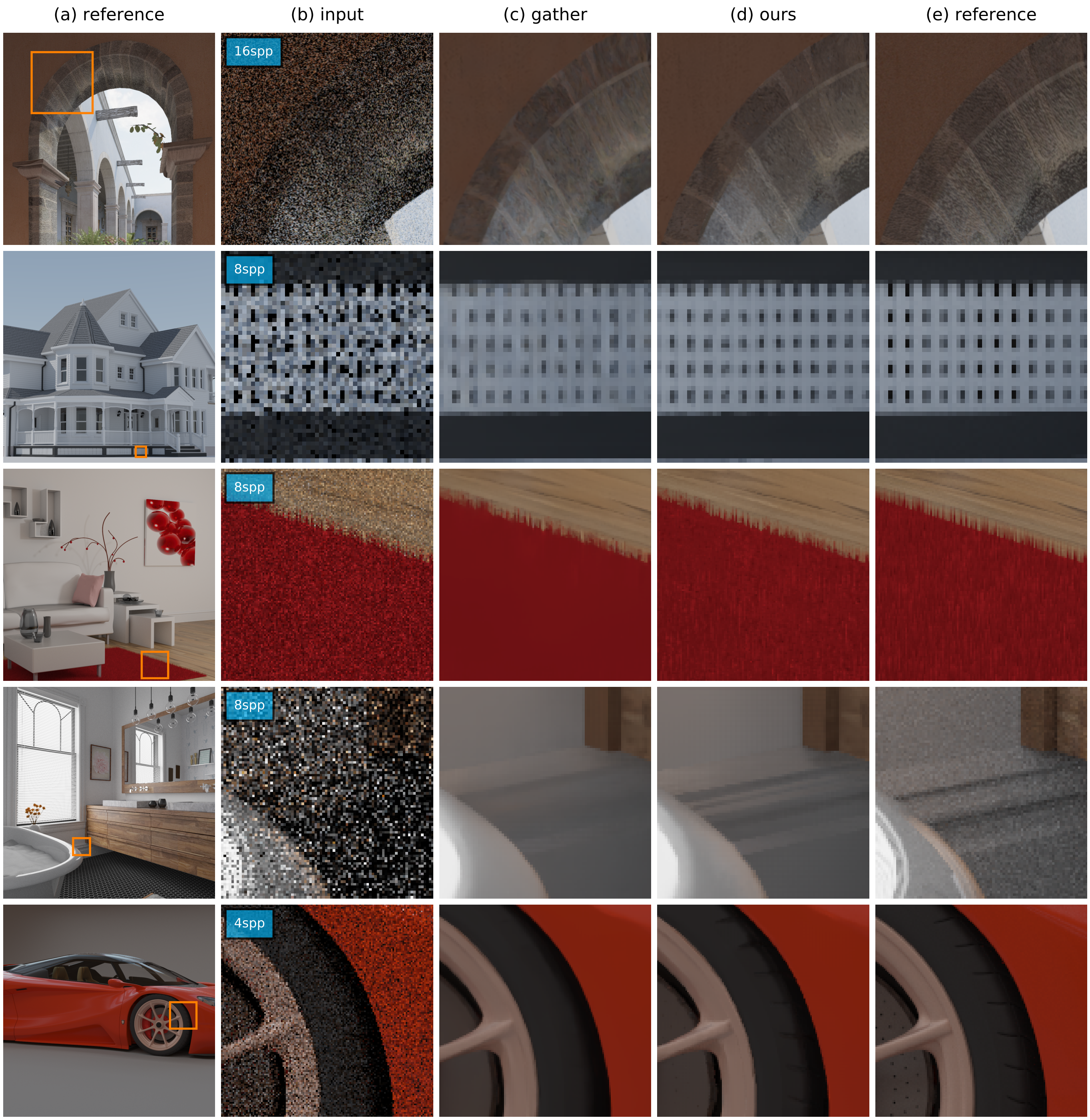

Fig. 1. **Affinity-based kernels (ours) are better at reconstructing fine details.**

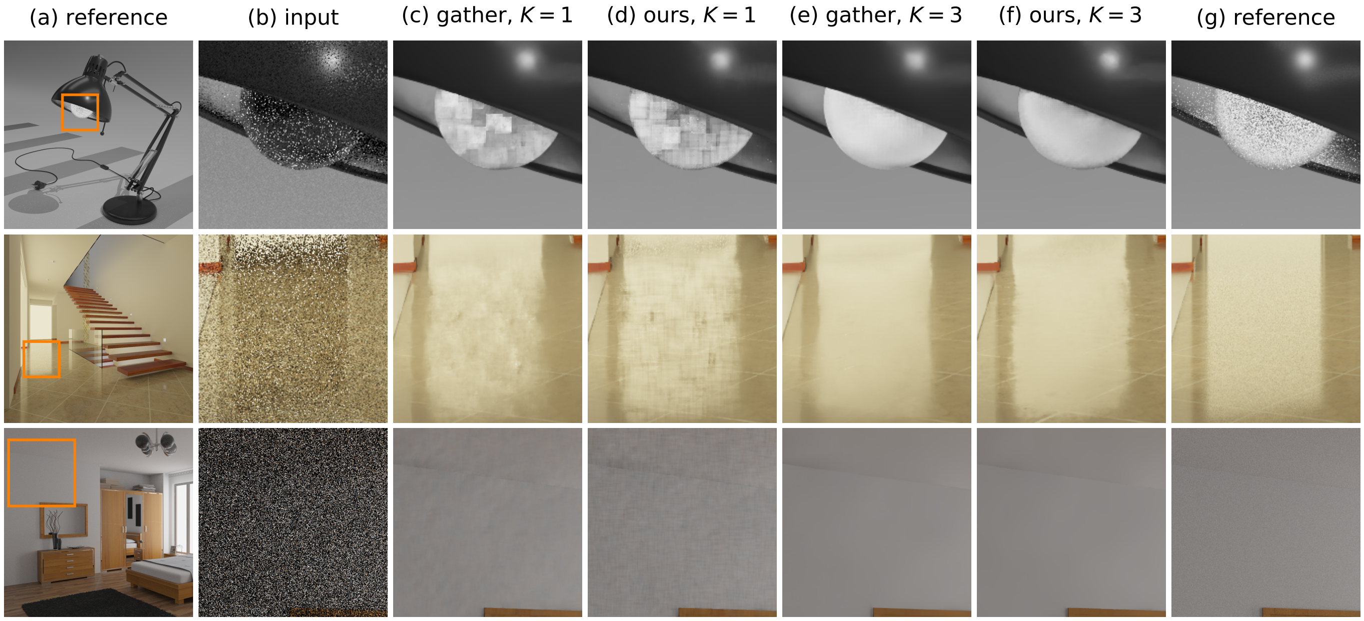

Fig. 2. **Advantage of using iterative dilated kernels.** Both gather (c) and affinity-based (d) kernels fail to spread the energy of bright pixels in case of severe noise when they use a single large kernel (17×17 here). Applying three dilated kernels (size 13×13) iteratively enlarges the kernel footprint significantly, and solves the problem (e, f).