Neural Partitioning Pyramids for Denoising Monte Carlo Renderings: Supplementary Material

MARTIN BALINT, Max Planck Institute for Informatics, Germany
KRZYSZTOF WOLSKI, Max Planck Institute for Informatics, Germany
KAROL MYSZKOWSKI, Max Planck Institute for Informatics, Germany
HANS-PETER SEIDEL, Max Planck Institute for Informatics, Germany
RAFAŁ MANTIUK, University of Cambridge, United Kingdom

CCS Concepts: • **Computing methodologies** → *Image processing*; **Ray tracing**.

ACM Reference Format:
Martin Balint, Krzysztof Wolski, Karol Myszkowski, Hans-Peter Seidel, and Rafał Mantiuk. 2023. Neural Partitioning Pyramids for Denoising Monte Carlo Renderings: Supplementary Material. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Proceedings (SIGGRAPH '23 Conference Proceedings), August 6–10, 2023, Los Angeles, CA, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3588432.3591562

A BACKBONE ARCHITECTURE

A.1 Input feature mappings

When feeding $\boldsymbol{L}_{xyst}$ and $\boldsymbol{r}_{xyst}$ to our encoder, we compress radiance $\boldsymbol{L}$ and depth disparity $1/d$ using the log +1 curve to lessen the numerical effect of outliers in our training data:

$$\tau : \boldsymbol{L} \mapsto \log\left(\frac{\boldsymbol{L}}{\mathbb{E}_{xys}[\boldsymbol{L}]} + 1\right), \tag{1}$$

$$d \mapsto \log\left(\frac{1}{d} + 1\right). \tag{2}$$

As our training and evaluation scenes have varying exposure, we scale radiance per frame, mapping the average over all (x, y) coordinates, colour channels and samples to one when providing radiance data as input to our neural networks. In Equation 1, dividing by $\mathbb{E}_{xys}[\boldsymbol{L}]$ denotes this operation.

A.2 Sample encoder

For our sample encoder we use a 32-channel, three-layer, fully-connected network with leaky ReLU activations.

A.3 OursSmall

For our smaller network we consider a typical U-Net structrue commonly used in previous work [Hasselgren et al. 2020; Işık et al. 2021; Munkberg and Hasselgren 2020]. Here, we use max-pooling, LeakyReLU activations and concatenating skip connections.

Our channel counts are the following:

$$\begin{aligned}\boldsymbol{c}96\boldsymbol{c}96\boldsymbol{d} \rightarrow \boldsymbol{c}96\boldsymbol{c}128\boldsymbol{d} \rightarrow \boldsymbol{c}128\boldsymbol{c}192\boldsymbol{d} \rightarrow \boldsymbol{c}192\boldsymbol{c}256\boldsymbol{d} \\ \rightarrow \boldsymbol{c}256\boldsymbol{c}384\boldsymbol{d} \rightarrow \boldsymbol{c}512\boldsymbol{c}512\boldsymbol{c}384\boldsymbol{u} \rightarrow \boldsymbol{c}384\boldsymbol{c}256\boldsymbol{u} \rightarrow \boldsymbol{c}256\boldsymbol{c}192\boldsymbol{u} \\ \rightarrow \boldsymbol{c}192\boldsymbol{c}128\boldsymbol{u} \rightarrow \boldsymbol{c}128\boldsymbol{c}96\boldsymbol{u} \rightarrow \boldsymbol{c}96\boldsymbol{c}96\end{aligned} \tag{3}$$

A.4 OursLarge

Our 30 million parameter network (OursLarge) uses ConvNext [Liu et al. 2022] blocks in the Restormer-like [Zamir et al. 2022] configuration as described in Figure 1.

B DATASET

Inspired by Hypersim [Roberts et al. 2021], we leverage Evermotion's Archinteriors and Archexteriors collections to build our training dataset. These scenes contain production quality assets with detailed physically based materials, far exceeding the quality and diversity of datasets used in previous work. The light transport of our training scenes also better matches photorealistic production scenarios, in which we expect our denoiser to have the largest impact.

We optimise 7 exterior and 8 interior scenes for the Falcor [Kallweit et al. 2022] renderer and manually add camera trajectories. We generate 1024 64-frame-long training sequences. We randomly pick a scene, a one-second camera trajectory segment, and an environment map for each sequence. We further perturb the camera trajectory and add randomly moving sphere lights and objects nearby the trajectory. We pick objects from the Amazon Berkeley Objects Dataset [Collins et al. 2022] containing 7941 high-quality 3D models with physically based materials and environment maps from the Poly Haven HDRI Dataset containing 388 4K environment maps. We randomise our generated sphere lights' colour, size, and intensity. We extract 256×256 motion compensated patches cropped from a 1080×1920 virtual camera frames. The motion is compensated by adjusting the crop offset according to the average optical flow in the cropped region. This allows our training patches to capture more temporal information. See Figure 2 for example frames of our training set. We will share our dataset with rendered content upon acceptance.

We choose to render our scenes with Falcor [Kallweit et al. 2022], a fast GPU-based renderer suitable for real-time and interactive previews, better matching potential applications. Most previous works were performed using PBRT [Pharr et al. 2016], a CPU-based offline renderer using slower but more advanced sampling algorithms; Falcor's path tracer requires double the sample count to meet the same

SIGGRAPH '23 Conference Proceedings, August 6–10, 2023, Los Angeles, CA, USA

ACM ISBN 979-8-4007-0159-7/23/08.
https://doi.org/10.1145/3588432.3591562

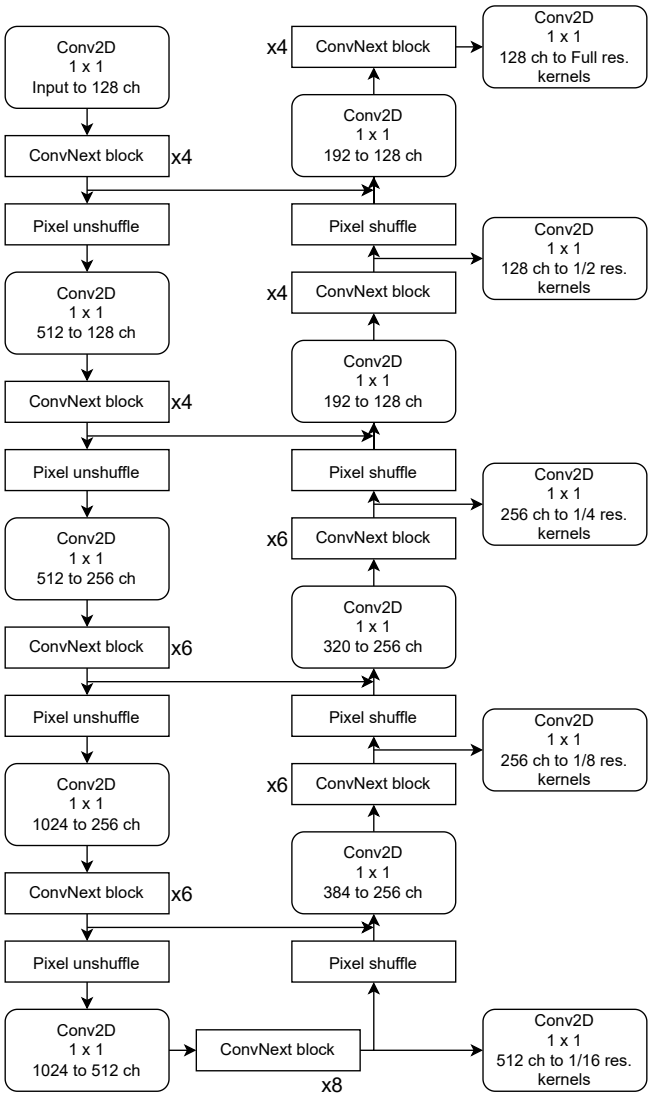

Fig. 1. Our ConvNext based architecture.

noise level. Please keep this in mind when comparing to metrics reported in previous works.

We render the supervision reference images for our training dataset at 6144 samples per pixel. Due to our scenes' complexity, some noise and fireflies are present in these images. Mitigating the noise by increasing our reference sample count by several orders of magnitude would make the generation of our dataset impractical. Therefore, we use OIDN [Intel 2022] to denoise three uncorrelated 2048 spp estimates and take their median as our training reference. We do not apply denoising to our test set.

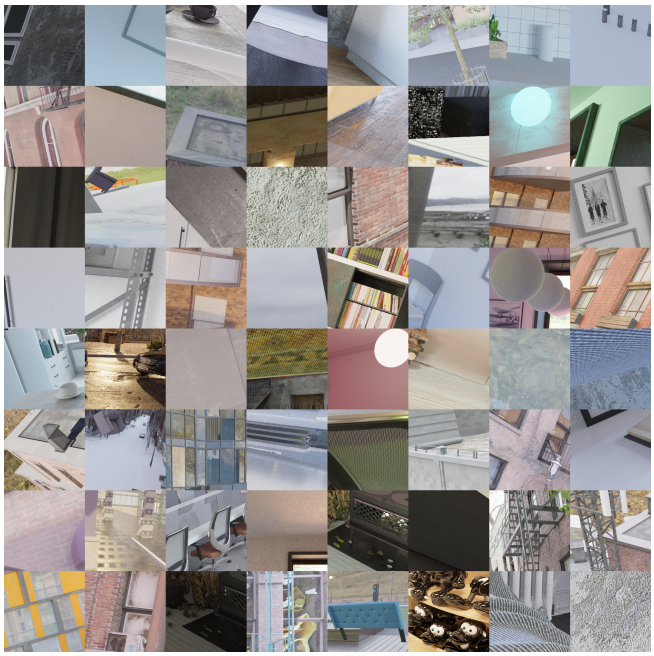

Fig. 2. Randomly chosen frames from our training dataset.

C LOSS FUNCTION

We aim to train our model to reconstruct temporally stable videos clean of noise. Previous work used Symmetric Mean Absolute Percentage Error as their metric of reconstruction quality:

$$\mathrm{SMAPE}(\hat{L}_t, L_t^\star) = \mathbb{E}_{x,y}\left[\frac{|\hat{L}_{x,y,t} - L^\star_{x,y,t}|}{|\hat{L}_{x,y,t}| + |L^\star_{x,y,t}| + \epsilon}\right], \tag{4}$$

where $\hat{L}_t$ and $L_t^\star$ denote the denoised and reference frames at the time t, and $\epsilon = 10^{-3}$.

Unfortunately, we found that SMAPE, being a per-pixel loss function, does not strongly condition in the lower frequencies. SMAPE also ignores the structural content of the images, often resulting in a blurry look. Furthermore, while it is similar to the median seeking L1 loss, it instead converges to brighter images than the median due to the denominator in the formula. For these reasons, we revise our loss function with a genuinely median seeking, perceptual, and multiscale component in feature space as proposed in [Thomas et al. 2022]. We get the best results by blending the perceptual loss and SMAPE:

$$\mathscr{L}_t^{\text{spatial}} = 0.2\|f(PU21(\hat{L}_t)) - f(PU21(L_t^\star))\|_1 + 0.8 \cdot \mathrm{SMAPE}(\hat{L}_t, L_t^\star), \tag{5}$$

where f extracts feature maps for frames $\hat{L}_t$ and $L_t^\star$. For f, we implement the perceptual loss proposed by [Thomas et al. 2022], with PU21 [Mantiuk and Azimi 2021] mapping from HDR inputs.

We also revise the temporal loss; previous methods take per-pixel differences between frames. This approach corresponds to the viewer not tracking any objects in the scene. While this scenario is improbable, we cannot tell which object the viewer might be

tracking without additional eye-tracker hardware. Therefore, we assume the viewer tracks each object in the scene, and thus we take the difference between warped images:

$$\mathscr{L}_t^{\text{temporal}} = 0.2 \left\| \left[f(PU21(\hat{L}_t)) - f(PU21(\mathscr{W}_t \hat{L}_{t-1})) \right] - \left[f(PU21(L_t^\star)) - f(PU21(\mathscr{W}_t L_{t-1}^\star)) \right] \right\|_1 + 0.8 \cdot \text{SMAPE}(\hat{L}_t - \mathscr{W}_t \hat{L}_{t-1}, L_t^\star - \mathscr{W}_t L_{t-1}^\star), \quad (6)$$

where $\mathscr{W}_t$ stands for the warping operator from the previous frame $t-1$ to the current frame t. The resulting videos generated by our denoiser appear significantly more stable, as demonstrated in our supplementary video.

Our complete loss is the sum of our temporal and spatial losses:

$$\mathscr{L}_t = \mathscr{L}_t^{\text{spatial}} + \mathscr{L}_t^{\text{temporal}}. \quad (7)$$

Table 1. Radiance notations used in our work

$\boldsymbol{L}_{xyst}$	Per-sample radiance
$\boldsymbol{L}_{xyt}$	Per-pixel radiance
$\bar{\boldsymbol{L}}_{xyt}$	Temporally accumulated radiance (until frame t)
$\bar{\boldsymbol{L}}^l_{xyt}$	Downsampled (and partitioned) per-layer radiance
$\hat{\boldsymbol{L}}^l_{xyt}$	Denoised per-layer radiance
$\check{\boldsymbol{L}}^l_{xyt}$	Composed per-layer radiance (until layer l from coarser layers)
$\check{\boldsymbol{L}}^0_{xyt}$	Unstable denoised output
$\boldsymbol{O}_{xyt}$	Temporally stabilised output

REFERENCES

Jasmine Collins, Shubham Goel, Kenan Deng, Achleshwar Luthra, Leon Xu, Erhan Gundogdu, Xi Zhang, Tomas F Yago Vicente, Thomas Dideriksen, Himanshu Arora, Matthieu Guillaumin, and Jitendra Malik. 2022. ABO: Dataset and Benchmarks for Real-World 3D Object Understanding. *CVPR* (2022).

J Hasselgren, J Munkberg, M Salvi, A Patney, and A Lefohn. 2020. Neural Temporal Adaptive Sampling and Denoising. In *Computer Graphics Forum*, Vol. 39. Wiley Online Library, 147–155.

Intel. 2022. Intel Open Image Denoise. https://www.openimagedenoise.org/.

Mustafa Işık, Krishna Mullia, Matthew Fisher, Jonathan Eisenmann, and Michaël Gharbi. 2021. Interactive Monte Carlo denoising using affinity of neural features. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–13.

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

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

Rafał K. Mantiuk and Maryam Azimi. 2021. PU21: A novel perceptually uniform encoding for adapting existing quality metrics for HDR. In *2021 Picture Coding Symposium (PCS)*. 1–5. https://doi.org/10.1109/PCS50896.2021.9477471

Jacob Munkberg and Jon Hasselgren. 2020. Neural denoising with layer embeddings. In *Computer Graphics Forum*, Vol. 39. Wiley Online Library, 1–12.

Matt Pharr, Wenzel Jakob, and Greg Humphreys. 2016. *Physically Based Rendering: From Theory to Implementation (3rd ed.)* (3rd ed.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Mike Roberts, Jason Ramapuram, Anurag Ranjan, Atulit Kumar, Miguel Angel Bautista, Nathan Paczan, Russ Webb, and Joshua M. Susskind. 2021. Hypersim: A Photorealistic Synthetic Dataset for Holistic Indoor Scene Understanding. In *International Conference on Computer Vision (ICCV) 2021*.

Manu Mathew Thomas, Gabor Liktor, Christoph Peters, Sungye Kim, Karthik Vaidyanathan, and Angus G Forbes. 2022. Temporally Stable Real-Time Joint Neural Denoising and Supersampling. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 5, 3 (2022), 1–22.

Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. 2022. Restormer: Efficient transformer for high-resolution image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 5728–5739.

Fig. 3. Further examples of our predicted partitioning. The images correspond to the fourth column of Figure 6, showing the layers after denoising and upsampling. In the *Zero-Day* scene on the left, our denoiser seems to partition diffuse lighting affecting larger areas into coarser layers and keep local highlights in fine-resolution layers. The middle column shows a synthetic example of the contrast sensitivity function, illustrating the frequency sensitivity of our partitioning. Note how our upsampler preserves the transitions between different regions, despite the varying frequency. In the *Bistro3* scene on the right, our denoiser again uses the coarse layers to reconstruct the diffuse lighting of the scene accurately, handling the challenging scenario with many light sources.