Ragged Neighborhood Attention for Spatiotemporal Neural Denoising of Deep Monte Carlo Renderings

XIANYAO ZHANG*, Disney Research Studios, Switzerland
GERHARD RÖTHLIN*, Disney Research Studios, Switzerland
TUNC OZAN AYDIN, Disney Research Studios, Switzerland
FARNOOD SALEHI, Disney Research Studios, Switzerland
MARIOS PAPAS†, Disney Research Studios, Switzerland

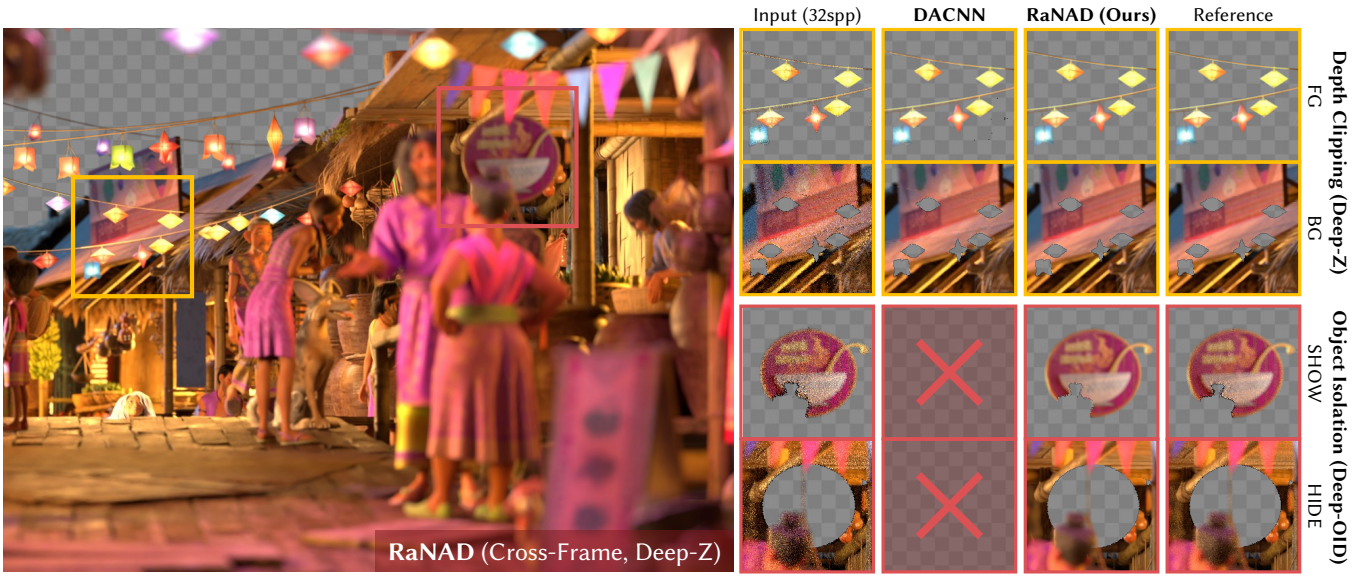

Fig. 1. RaNAD is capable of processing both deep-Z and deep-OID variants of deep images, preserving the depth separation offered by deep-Z images and the clean object isolation offered by deep-OID images. On the left, we show a denoised deep-Z frame with cross-frame RaNAD. On the right, we show edited versions of the two highlighted regions in this frame, listing the noisy input, a previous CNN-based deep-Z denoiser DACNN [Zhang et al. 2024], our proposed RaNAD, and the reference. The two top rows show that our RaNAD method maintains a clean separation between the foreground (FG) and the background (BG) whereas DACNN leaves traces of BG in FG (vice versa). The bottom two rows show that, when applied to a deep-OID image, RaNAD preserves the separation between objects; DACNN does not support denoising deep-OID images. © Disney

Deep images store a variable number of "bins" within each pixel, enabling deep compositing workflows with clean separation of overlapping objects, but Monte Carlo noise limits their practical use. Existing deep image denoisers are limited in quality, generality, and temporal processing because ragged bin neighborhoods are incompatible with fixed convolution kernels. We introduce the ragged neighborhood attention (RaNA) operator, which extends neighborhood attention to semi-structured deep images by dynamically resolving bin-to-bin relationships across spatial and temporal neighborhoods. Using RaNA, we build the first spatiotemporal neural denoiser for deep Monte Carlo renderings, termed RaNAD, with a multi-U-Net backbone, multi-scale reconstruction, and an optimized CUDA implementation. RaNAD handles both deep-Z and deep-OID variants and supports temporal windows of up to 7 frames. Compared with the previous state of the art for deep image denoising, RaNAD improves denoising quality substantially while preserving the layered structure needed for deep compositing; when its output is flattened for evaluation, it also attains quality competitive with strong flat image denoisers. As a kernel-based method, RaNAD efficiently denoises multi-AOV images, and temporal processing further improves quality and stability, making the method practical for offline production rendering and compositing workflows.

*Both authors contributed equally to this work.
†Corresponding author.

Authors' Contact Information: Xianyao Zhang, Disney Research Studios, Zürich, Switzerland, xianyao.zhang@disneyresearch.com; Gerhard Röthlin, Disney Research Studios, Zürich, Switzerland, gerhard@disneyresearch.com; Tunc Ozan Aydin, Disney Research Studios, Zürich, Switzerland, tunc@disneyresearch.com; Farnood Salehi, Disney Research Studios, Zürich, Switzerland, farnood.salehi@disneyresearch.com; Marios Papas, Disney Research Studios, Zürich, Switzerland, marios.papas@disneyresearch.com.

SIGGRAPH Conference Papers '26, Los Angeles, CA, USA

ACM ISBN 979-8-4007-2554-8/26/07
https://doi.org/10.1145/3799902.3811128

CCS Concepts: • **Computing methodologies** → **Rendering**; **Image processing**; **Neural networks**.

Additional Key Words and Phrases: Deep EXR images, Monte Carlo rendering, denoising, attention, spatiotemporal

ACM Reference Format:
Xianyao Zhang, Gerhard Röthlin, Tunc Ozan Aydin, Farnood Salehi, and Marios Papas. 2026. Ragged Neighborhood Attention for Spatiotemporal Neural Denoising of Deep Monte Carlo Renderings. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers (SIGGRAPH Conference Papers '26), July 19–23, 2026, Los Angeles, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3799902.3811128

1 Introduction

Compositing is essential in animated film and VFX production, where different elements within a shot are adjusted independently [Brinkmann 2008]. A key step is isolating objects of interest, typically via alpha matting [Yao et al. 2024] or multiple render passes. Both approaches have limitations: matting creates edge artifacts when objects overlap within pixels, while render passes require knowing in advance which objects will be modified.

Deep images, or more specifically deep EXRs [Kainz 2013b], facilitate compositing by separately storing different entities within the same pixel. These sub-pixel units are termed "bins"; each bin records color, opacity, and arbitrary additional attributes, and each pixel can contain an arbitrary number of bins. Therefore, deep images can provide more accurate opacity and avoid edge artifacts in compositing because the different bins offer ideal separation of geometric boundaries.

One hurdle preventing rendered deep images from widespread adoption is noise, which increases the difficulty of achieving the desired effect. High-quality denoisers for deep images, especially ones that compete with the quality of denoisers on flat images [Vogels et al. 2018; Zhang et al. 2023], will greatly improve the practicality of deep images in offline production workflows. Figure 2 shows why flat denoising is not a replacement for this workflow: compositing noisy deep data before flat denoising can cause color shifts under nonlinear edits, while flattening before compositing loses layer separation and leaves boundary artifacts.

Recent machine learning methods adapt convolutional neural networks (CNNs) [Zhang et al. 2024] for denoising deep-Z images, filtering each bin based on information from neighboring bins in image space and in depth. Though advancing over the traditional cross-filter [Vicini et al. 2019], the quality of these methods is not sufficient, and they cannot be easily extended to handle deep-OID images or ingest temporal information.

The key challenge of neural denoising for deep images lies in their ragged structure. Image-space operations that assume a regular 2-D image plane, including convolutions, pooling, and unpooling, cannot be easily adapted to process deep images, as the definition of neighborhood is unclear. We propose the ragged neighborhood attention (RaNA) operator that enables efficient information gathering on semi-structured data. Our RaNA-based denoiser, RaNAD, greatly improves the denoising quality on deep Monte Carlo renderings and supports both deep-Z and deep-OID formats (Figure 1).

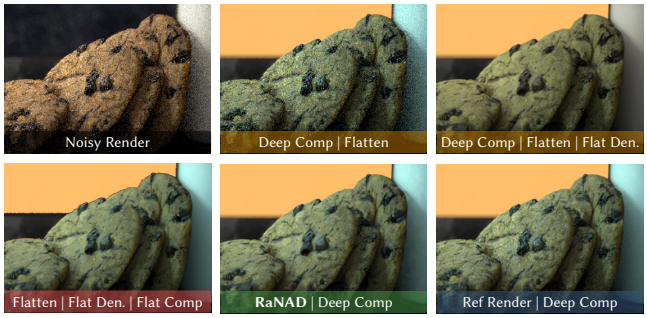

Fig. 2. Deep image denoising preserves the deep compositing workflow. The example inserts a layer between foreground and background and recolors the foreground. Compositing noisy deep data before flat denoising causes color shifts (*e.g.*, to the right of the image), while flat denoising before compositing loses separation and creates boundary artifacts. RaNAD denoises before compositing, matching the reference workflow.

Another critical aspect that offline production-ready denoisers need to consider is cross-frame denoising, *i.e.*, making use of neighboring frames. Not only is cross-frame denoising essential for maintaining temporal coherence, but it can also improve denoising quality. To the best of our knowledge, previous deep image denoising methods work with only spatial neighbors, and an effective cross-frame denoising technique for deep images is yet to be developed.

Our RaNAD model is kernel-based, performing reconstruction via attention-induced implicit kernels. Kernel-based denoisers are preferred in production for their stability, quality, and robustness [Bako et al. 2017; Vogels et al. 2018]. A key benefit is efficient denoising of arbitrary output variables (AOVs) through kernel reuse: by caching embeddings and rerunning only the reconstruction module, RaNAD can consistently denoise multiple AOVs without repeated inference.

Our contribution can be summarized as follows.

- We introduce the ragged neighborhood attention (RaNA) operator which, through our custom CUDA implementation, enables efficient processing of semi-structured data with neural networks.
- Based on RaNA, we propose the ragged neighborhood attention denoiser (RaNAD) for deep images, which greatly advances denoising quality over previous state-of-the-art deep image denoisers.
- We further adapt RaNAD to perform cross-frame denoising, introducing the first spatiotemporal neural denoising method for deep images and improving temporal stability and denoising quality.

2 Background and Related Work

Our study focuses on denoising deep images produced by Monte Carlo rendering using attention-based neural networks. We hereby review the deep image format, the relevance of attention mechanisms to ragged data, and the landscape of Monte Carlo denoising.

2.1 Deep Images

Unlike traditional images where every spatial location (x, y) contains a fixed-size data vector, deep images consist of sub-pixel "bins"

whose count varies dynamically from pixel to pixel [Kainz 2013a]. This transforms the image from a dense 3-D tensor into a ragged 4-D array of shape $H \times W \times B \times C$, where the bin dimension B has variable length. Deep images are typically generated by Monte Carlo rendering, aggregating light path samples based on either first-hit depth (deep-Z) or Object ID (deep-OID) [Vicini et al. 2019]. While this structure facilitates downstream compositing [Foundry 2023], the inherent rendering noise is a major obstacle. Existing denoisers [Vicini et al. 2019; Zhang et al. 2024] struggle to meet production-quality requirements and lack support for cross-frame temporal consistency. Our proposed RaNAD addresses these limitations, offering a production-ready solution for offline rendering and compositing workflows.

2.2 Attention and Neighborhood Attention

The lack of a regular grid in the bin dimension makes defining consistent convolution kernels on deep images non-trivial. The attention mechanism [Vaswani et al. 2017] offers a solution by handling variable-length sequences naturally. It computes a weighted sum of values (V) based on the compatibility of a query (Q) with keys (K):

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_{\text{KQ}}}}\right)V \tag{1}$$

While standard attention is computationally expensive ($O(N^2)$ for self-attention of a length-N sequence), local variants like Neighborhood Attention (NA) [Hassani et al. 2023] reintroduce local inductive biases similar to convolutions. However, existing NA implementations require dense tensors. Our RaNA operator bridges this gap, leveraging a custom CUDA implementation to apply NA within the ragged data structure. Related work has also explored hybrid convolution–attention architectures for flat image processing [Bello et al. 2019] and transformers for point clouds [Zhao et al. 2021], but these methods are designed for either dense tensors or unstructured point neighborhoods, and thus do not directly support semi-structured deep images.

2.3 Monte Carlo Denoising

Kernel-based denoising. Kernel-predicting convolutional networks (KPCN) [Bako et al. 2017; Vogels et al. 2018] have become the standard for denoising rendered content. By predicting per-pixel filter kernels based on auxiliary features (*e.g.*, albedo, normal, depth), they preserve high-frequency details better than direct color prediction. Sample-based Monte Carlo denoisers [Gharbi et al. 2019; Munkberg and Hasselgren 2020] operate on individual Monte Carlo samples with a kernel-based approach, but they still target flat image quality. RaNAD adopts the kernel-based denoising philosophy but generates filter kernels implicitly via attention, and it optimizes the denoising quality while preserving deep image structure, instead of targeting flat image quality.

Attention-based architectures. While Vision Transformers are increasingly popular in low-level vision tasks [Liang et al. 2021; Zamir et al. 2022], their adoption in Monte Carlo denoising is recent [Chen et al. 2024; Yu et al. 2021]. Unlike prior works that use attention primarily for feature extraction on dense images, we utilize it specifically to resolve the structural irregularity of deep images.

Deep image denoising. As shown in Figure 2, high-quality flat denoisers cannot be easily applied in deep compositing workflows. On the other hand, solutions tailored to deep images are scarce. Vicini et al. [2019] adapted the cross-NL-Means filter for deep-Z bins, and Zhang et al. [2024] introduced a 3-D KPCN that adapts kernels based on depth; neither approach supports deep-OID images or temporal denoising. RaNAD significantly improves denoising quality and introduces cross-frame capabilities, establishing a robust pipeline for offline deep image denoising and compositing.

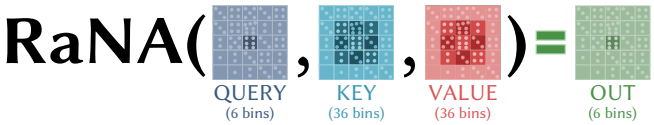

Fig. 3. One step of ragged neighborhood attention (RaNA). Here Query, Key, Value, and Output are all 5×5 deep images, where each dot denotes one *bin*. For each query bin in the center pixel, RaNA gathers the bins from neighboring pixels in Key and Value and applies attention to compute the Output for the query bin. The final Output has the same bin layout as Query and the same channel count as Value.

3 Ragged Neighborhood Attention

The core operator enabling our deep image denoiser is the Ragged Neighborhood Attention (RaNA) mechanism. This approach is similar to the concept of Neighborhood Attention (NA) [Hassani et al. 2023], which mitigates the quadratic cost of global self-attention [Vaswani et al. 2017] by restricting the attention scope to a local sliding window. However, while the original NA relies on a fixed grid topology, deep images possess a ragged structure where the number of bins varies per pixel. In this context, determining which neighbors are relevant is non-trivial due to depth discontinuities and overlapping strata. RaNA extends the principles of neighborhood attention to this irregular domain, using attention to dynamically resolve bin-to-bin relationships within the ragged neighborhood, as conceptually illustrated in Figure 3.

3.1 Formal Definition of the RaNA Operator

Let $\mathcal{B}$ denote the set of all populated bin indices in the deep image. A bin index $\mathbf{p} \in \mathcal{B}$ is a 3D vector that can be decomposed into its spatial 2D pixel coordinates $\mathbf{x}$ and its depth-wise bin index b, such that $\mathbf{p} = (\mathbf{x}, b)$.

For a given center bin $\mathbf{p}$, we define a local spatial neighborhood $\mathcal{N}(\mathbf{p})$ comprising all bins belonging to pixels within a Euclidean distance (radius) R of the center pixel $\mathbf{x}$ in the image plane:

$$\mathcal{N}(\mathbf{p}) = \left\{\mathbf{p}' \in \mathcal{B} \mid \mathbf{p}' = (\mathbf{x}', b') \text{ and } \left\|\mathbf{x} - \mathbf{x}'\right\|_2 \leq R\right\}. \tag{2}$$

We deliberately employ the L_2 distance (defining a pixelated circle) rather than the Chebyshev (L_∞) distance commonly used in convolutional or sliding window operators because RaNA is not bound by the rectangular memory constraints of fixed grids and the L_2 norm allows us to select neighbors based on consistent spatial proximity. An illustration of the neighborhood can be found in the

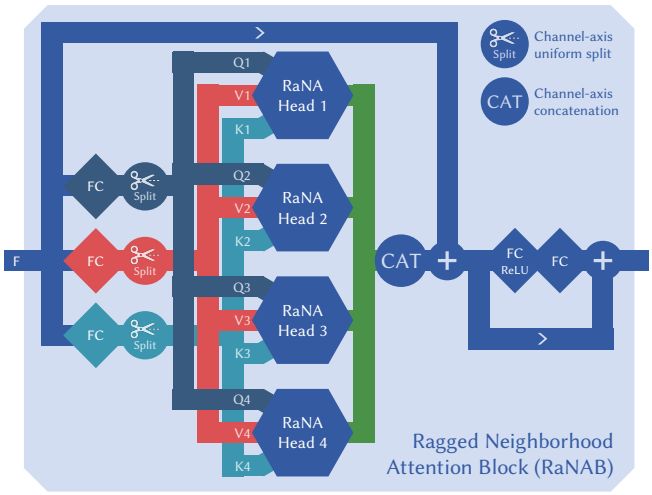

Fig. 4. RaNA block, consisting of multi-head RaNA (4-head example), fully connected (FC) layers, and residual connections.

supplemental material. Due to the ragged nature of the data, the cardinality $|\mathcal{N}(\mathbf{p})|$ varies for every $\mathbf{p}$.

The core RaNA operator takes a center query and its ragged neighborhood; key and value vectors are looked up for each neighboring bin. For a specific center bin $\mathbf{p}$, the attention weight $A_{\mathbf{p}\leftarrow\mathbf{p}'}$ represents how much a neighbor $\mathbf{p}'$ contributes to the update of $\mathbf{p}$. The weight is computed via a scaled dot-product between the query of the center and the keys of the neighbors, normalized by a softmax over the ragged neighborhood:

$$A_{\mathbf{p}\leftarrow\mathbf{p}'} = \left[\operatorname{softmax}_{\mathbf{m}\in\mathcal{N}(\mathbf{p})}\left(\frac{\mathbf{q}_\mathbf{p}^\top \mathbf{k}_\mathbf{m}}{\sqrt{d_k}}\right)\right]_{\mathbf{p}'}. \tag{3}$$

The output of the RaNA operator, $\mathbf{y}_\mathbf{p}$, is the aggregation of information flowing from the neighborhood:

$$\operatorname{RaNA}(\mathbf{q}_\mathbf{p}, \mathcal{N}(\mathbf{p})) = \sum_{\mathbf{p}'\in\mathcal{N}(\mathbf{p})} A_{\mathbf{p}\leftarrow\mathbf{p}'}\mathbf{v}_{\mathbf{p}'}. \tag{4}$$

3.2 RaNA Block (RaNAB)

We incorporate the core RaNA operator into a complete processing block, the RaNA Block (RaNAB), employing multi-head attention and residual connections as illustrated in Figure 4.

Let F denote the input feature tensor to the block, where $F_\mathbf{p} \in \mathbb{R}^C$ is the feature vector for bin $\mathbf{p}$. First, F is projected by three separate fully connected (FC) layers to generate queries Q, keys K, and values V. These are subsequently split along the channel dimension into N_{head} independent heads. For each head $h \in \{1, \ldots, N_{\text{head}}\}$, we obtain query, key, and value vectors $\mathbf{q}_\mathbf{p}^{(h)}, \mathbf{k}_\mathbf{p}^{(h)}, \mathbf{v}_\mathbf{p}^{(h)}$. We apply the core RaNA operator (Equation (4)) to each head independently in parallel. The outputs of the heads are concatenated to form a single combined feature vector:

$$\mathbf{y}_\mathbf{p}^{\text{cat}} = \operatorname{Concat}\left\{\operatorname{RaNA}\left(\mathbf{q}_\mathbf{p}^{(1)}, \mathcal{N}(\mathbf{p})\right), \ldots, \operatorname{RaNA}\left(\mathbf{q}_\mathbf{p}^{(N_{\text{head}})}, \mathcal{N}(\mathbf{p})\right)\right\}. \tag{5}$$

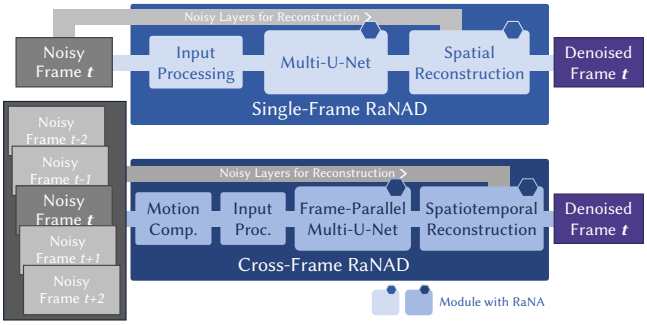

Fig. 5. High-level organization of single-frame and cross-frame RaNAD. The single-frame model processes one noisy deep frame through input processing, a RaNA-based multi-U-Net, and spatial per-layer reconstruction. The cross-frame model extends this pipeline to a frame window using motion compensation, frame-parallel multi-U-Net backbone, and spatiotemporal per-layer reconstruction for the center frame.

Following standard transformer design principles, we employ residual connections. The concatenated output is added to the original input features $F_\mathbf{p}$:

$$\mathbf{z}_\mathbf{p} = F_\mathbf{p} + \mathbf{y}_\mathbf{p}^{\text{cat}}. \tag{6}$$

Finally, this intermediate result passes through a feed-forward network (FFN), consisting of two FC layers separated by a ReLU activation, with its own residual connection:

$$\text{Output}_\mathbf{p} = \mathbf{z}_\mathbf{p} + \text{FC}_2\big(\text{ReLU}\big(\text{FC}_1(\mathbf{z}_\mathbf{p})\big)\big). \tag{7}$$

This complete definition allows the RaNA block to learn complex local relationships in ragged deep image data efficiently.

4 RaNA Denoiser for Deep Images

In this section, we describe RaNAD for spatiotemporal deep image denoising, starting with the single-frame model and then extending it to cross-frame feature mixing and reconstruction. Figure 5 provides a high-level overview of both models.

4.1 Single-Frame RaNAD

Figure 6 shows the single-frame RaNAD architecture, formed by three main modules: input processing, multi-U-Net backbone, and per-layer reconstruction. The model input consists of diffuse, specular, alpha, depth, albedo, and normal layers from a noisy deep image. In addition to image channels, we also provide the indices along the width, height, and bin dimensions for positional encoding.

The model denoises diffuse, specular, alpha, and depth; albedo and normal are auxiliary feature buffers. We adopt the diffuse–specular split because of its quality gains over color-only denoising [Vogels et al. 2018; Xu et al. 2019], and we additionally denoise the alpha and depth layers because noise in these layers can introduce artifacts in compositing operations [Zhang et al. 2024].

4.1.1 Input processing. The input processing module prepares a per-bin feature vector as the input to the multi-U-Net backbone. First, each input layer, as listed above, is transformed with a specialized function based on its respective meaning and range, following previous works on both flat and deep-Z image denoising [Zhang

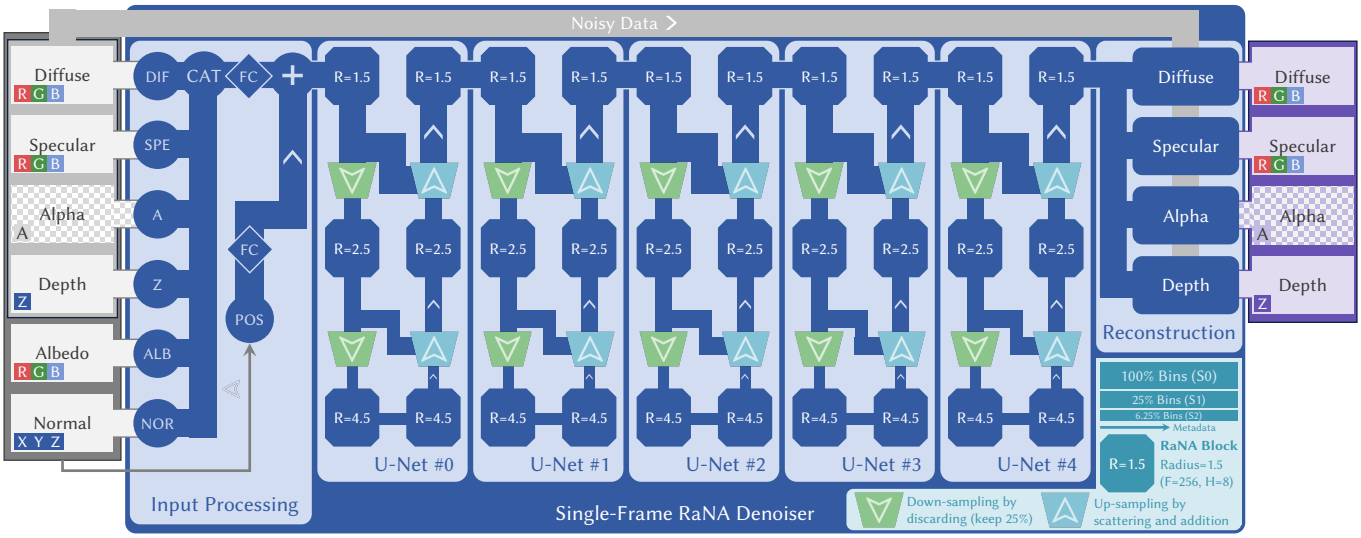

Fig. 6. Single-frame ragged neighborhood attention denoiser (RaNAD) architecture. The pipeline consists of three stages: input processing (Section 4.1.1), multi-U-Net backbone (Section 4.1.2), and multi-scale reconstruction (Section 4.1.3). The width of the data pipes indicates the scale in terms of bin density.

et al. 2021, 2024]. Next, positional encoding is appended to each bin's feature vector, using not only the pixel location (x, y) but also the bin index b, so that the downstream RaNA operators are aware of the neighborhood structure of a bin. This step is necessary because RaNA, derived from the attention operation, requires position information to distinguish close-by neighbors from distant ones, in contrast with fully convolutional denoisers [Hasselgren et al. 2020; Vogels et al. 2018] whose convolution kernels encode the neighbors' relative position inherently. The exact inputs used by our RaNAD models are detailed in supplemental material.

4.1.2 Multi-U-Net Backbone. As depicted in the middle part of Figure 6, the majority of RaNAD's trainable parameters reside in a backbone network termed *multi-U-Net*, which consists of a sequence of multiple RaNA-based U-Net submodules. The multi-U-Net backbone receives per-bin feature vectors from the input processing module and produces per-bin learned embeddings, which are passed to the reconstruction module for the eventual denoising step.

U-Nets have been widely adopted in Monte Carlo denoising to efficiently gather context from larger neighborhoods [Hofmann et al. 2023; Zhang et al. 2021]. In RaNAD, each U-Net submodule contains encoder and decoder blocks at multiple scales, with skip connections at each scale between the encoder blocks' output and the decoder blocks' input. The distinct difference from previous work is that RaNA, instead of convolution layers, is used to implement RaNAD's U-Nets. More precisely, RaNA drives the feature aggregation inside the encoder and decoder blocks, while the actual changes in bin layout are performed by dedicated down/up operators. Down-sampling selects a sparser query layout through deterministic bin dropout, and up-sampling restores a finer bin layout through up-scattering and shortcut merging (see supplemental material for details). For coarser scales with fewer bins, we expand the RaNA operators' radius to gather information from more distant neighbors.

The output of a U-Net submodule will serve as input to the next U-Net submodule, and this chain of U-Nets forms the multi-U-Net backbone of RaNAD. The reason behind this choice of chaining U-Nets instead of increasing the depth of the encoder and decoder within a single U-Net is twofold. First, down-sampling deep images in RaNAD is achieved by discarding a selection of bins (see supplemental material). The multi-U-Net architecture allows alternating between different dropout patterns, alleviating the possibility that the model focuses on one selection of bins. Second, as we will discuss next, cross-frame denoising of deep images requires efficient information propagation not only spatially but also temporally, which is facilitated by the multi-U-Net. Our ablation study shows that a capacity-matched single-U-Net backbone can achieve slightly lower error than the multi-U-Net, but extending the single-U-Net for a practical spatiotemporal denoiser is less straightforward.

4.1.3 Reconstruction Module. To the far right of Figure 6 are reconstruction modules for each denoised layer. Inspired by multi-scale kernel-based denoisers [Bako et al. 2017; Vogels et al. 2018; Zhang et al. 2024], our reconstruction module processes deep images at multiple scales, but its denoising kernels are implicitly predicted by RaNA operations. The reconstruction modules turn the shared embeddings from the backbone into layer-specific denoising kernels with different effective radii and numbers of scales.

Figure 7 shows the architecture of the reconstruction module. The first stage performs RaNA-based down-sampling: we apply bin dropout to create sparser queries, then use RaNA with these as queries and full-resolution features as keys, paired with the noisy layer as values. Recursive down-sampling with increasing radii produces a pyramid of progressively blurred deep images.

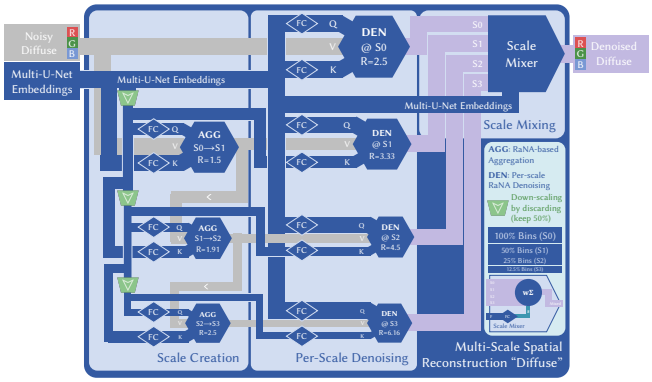

Fig. 7. Four-scale reconstruction module. It receives per-bin embeddings from the multi-U-Net backbone together with the noisy image layer, forms progressively sparser multi-scale views, and combines the resulting per-scale denoised outputs into the final layer.

The second stage performs per-scale denoising. We apply RaNA with full-resolution features as queries and each scale's features and blurred images as key–value pairs, producing a sequence of full-resolution images with different effective kernel sizes (see supplemental material for per-scale visualization).

Finally, a scale mixer (FC layer with softmax) predicts per-bin mixing weights to combine the per-scale results into the final denoised layer. Each output layer uses a dedicated reconstruction module with layer-specific configuration (number of scales, dropout rate, neighborhood size).

4.2 Cross-Frame RaNAD

Denoising while considering neighbor frames in an image sequence gives access to more information, which both helps with gathering information on the underlying image content structure (*e.g.*, borders, corner points) in an image, and gathering samples to average over, at no additional rendering cost in the offline rendering scenario. Being able to access the same data as used to denoise neighbor frames reduces temporal artifacts such as flickering and improves the overall temporal stability of the result. Our cross-frame RaNAD architecture is shown in Figure 8; detailed submodule diagrams are provided in the supplemental material. Note that the input processing module of the cross-frame RaNAD includes an additional one-hot encoding of the frame ID, which is useful for identifying the center frame from neighboring frames.

4.2.1 Motion compensation. Aligning the neighboring frame(s) with the to-be-denoised *center* frame using motion vectors, *a.k.a.* motion compensation, is a typical step for cross-frame denoising to maximize temporal reuse [Işık et al. 2021; Vogels et al. 2018; Zhang et al. 2023]. Cross-frame RaNAD also benefits from motion compensation, as shown to the left of Figure 8. Either per-pixel or *per-bin* motion vectors can be used for this warping step, but per-bin motion vectors, when supplied by the renderer, provide superior warping accuracy over their per-pixel counterparts, thanks to accurate motion information at object boundaries and depth discontinuities. Our final cross-frame RaNAD can be trained on a mixture of datasets where per-bin motion vectors are only available for a subset of training images.

4.2.2 Frame-Parallel Multi-U-Net. At the core of the cross-frame RaNAD is a *frame-parallel* multi-U-Net. The term "frame-parallel" refers to the design where each frame is processed by a U-Net independently, or practically, in parallel. The embeddings of the frames then go through a temporal mix (T-Mix) module, which consists of a RaNA block that attends only to each pixel's *temporal* neighborhood. Each U-Net in the multi-U-Net chain is followed by a T-Mix module, enabling an alternation between spatial and temporal information propagation. Compared to a canonical architecture that concatenates all frames at the input stage [Vogels et al. 2018], this alternating architecture greatly reduces the computation cost for deep images with a high number of bins per pixel. Facilitated by the multi-U-Net architecture, this optimization is crucial to alleviate the performance challenges posed by the huge sizes of deep images and the ragged data structure.

4.2.3 Spatiotemporal Reconstruction. The per-layer reconstruction modules in cross-frame RaNAD, positioned to the right of Figure 8, are also upgraded to incorporate neighboring frames. More specifically, we introduce an additional temporal reconstruction step preceding the spatial reconstruction submodule. The noisy images from all frames are passed to the reconstruction module, along with the learned embeddings. The reconstruction module first performs a T-Mix-like operation to denoise each bin using only its temporal neighborhood (bins at the same pixel location across frames). The intermediate result, a "temporally denoised" version of the center frame, is then further denoised by the spatial reconstruction submodule. The two-step approach also avoids the high computational cost of directly processing the spatiotemporal neighborhood.

5 Implementation

We implement RaNAD in TensorFlow 2 [Abadi et al. 2015], with a FlashAttention-2-inspired [Dao 2023] custom CUDA kernel for the RaNA operator. Deep images are represented using TensorFlow's `RaggedTensor` [TensorFlow Authors 2025], flattened to dense value tensors with cumulative bin count splits for efficient GPU access. Full implementation details, including the CUDA forward/backward passes, input transforms, and training losses, are provided in supplemental material.

Table 1. Datasets for training and evaluating RaNAD. The column "Split" denotes how many *scenes* are used for training and evaluation, respectively. For single-frame RaNAD, the center frames from the three datasets are used; for cross-frame RaNAD, all available frames are used as input (including Dataset A2, which only contains center frames), and the metrics are computed on the reconstructed center frames.

Dataset	Type	Split	Resolution	Temporal	Motion
A1	Deep-Z	90+10	1920×804	7 frames	Per-bin
A2	Deep-OID	90+10	1920×804	—	—
B	Deep-Z	90+9	1920×1080	3 frames	Per-pixel

We train RaNAD on three datasets (Table 1) containing noisy and reference deep images rendered with production-grade path

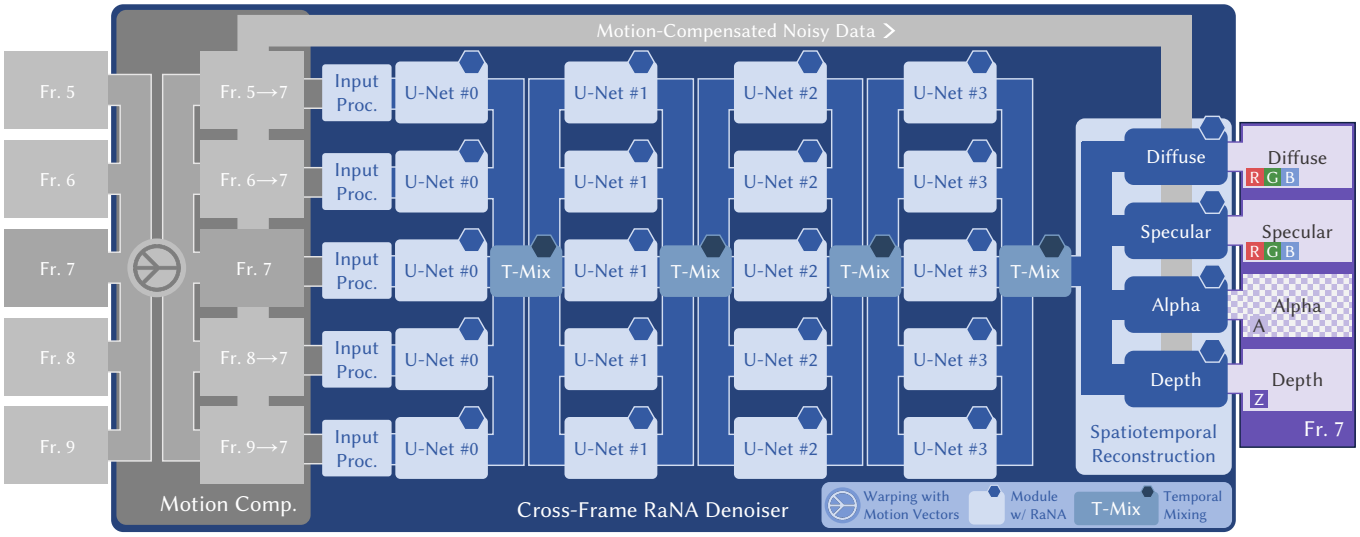

Fig. 8. High-level view of cross-frame RaNAD with the frame-parallel Multi-U-Net. We illustrate denoising frame 7 using a 5-frame context window, *i.e.*, frames 5–9. Each frame first undergoes motion compensation and the same input processing as in the single-frame model (Figure 6), then passes through an alternation of per-frame U-Nets and temporal mix modules. The final spatiotemporal reconstruction module converts the learned embeddings into the denoised center-frame output.

tracers, with spp levels starting from 16. Datasets A1 and A2 are rendered with Disney's Hyperion renderer [Burley et al. 2018]; they share the same scenes but use deep-Z and deep-OID formats, respectively; Dataset A1 provides 7-frame temporal windows with per-bin motion vectors. Dataset B, rendered with Pixar's RenderMan [Christensen et al. 2018], offers complementary deep-Z data with 3-frame windows and per-pixel optical flow motion vectors. All datasets were used to train both the single- and cross-frame RaNAD, and the mixed-dataset training is enabled by our attention-based architecture, which can naturally handle varying sizes of temporal windows. Around 10% of scenes are held out for evaluation.

We train on a single NVIDIA RTX A6000 GPU using AdamW [Loshchilov and Hutter 2019] with batch size 1 for approximately 1M iterations. A piecewise-constant learning rate schedule with warm-up and staged decay is employed, combined with gradient clipping and L_2 regularization for training stability. Training losses include SMAPE on individual bins for radiance layers, combined with ꟻLIP*, a training-friendly modification of ꟻLIP [Andersson et al. 2020], on composited results. Hyperparameters and architecture configurations are detailed in supplemental material.

6 Results

We evaluate RaNAD in terms of denoising quality and computation cost, comparing against DACNN [Zhang et al. 2024] for deep-Z denoising and production-quality flat denoisers [Áfra 2023; Vogels et al. 2018]. We demonstrate that RaNAD significantly outperforms previous deep image denoisers while achieving competitive quality with state-of-the-art flat denoisers when evaluated after flattening, and that RaNAD's computation costs are practical for offline rendering.

6.1 Baseline and Evaluation Setup

As listed in Table 1, we evaluate RaNAD on the test set comprising 19 production-quality scenes held out from training. We use the 3-D depth-aware CNN (DACNN) [Zhang et al. 2024] as our baseline for deep-Z denoising. Following the neural network setup and training procedure described by Zhang et al. [2024], we trained a single-frame DACNN on our deep-Z Dataset A1 and obtained very similar results to those reported by Zhang et al. [2024]. As DACNN requires depth information to operate, we can only perform comparisons on deep-Z images from Dataset A1. To evaluate the flat image denoising quality of RaNAD (computed by flattening RaNAD's denoised deep output), we compare it with a state-of-the-art proprietary denoiser, termed KPAL, which is a variant of the kernel-predicting CNN approach [Vogels et al. 2018; Zhang et al. 2023]. KPAL features single- and cross-frame versions, both of which are used in our comparisons. KPAL denoisers were trained on flat noisy–clean image pairs, and their training sets are more than five times larger than RaNAD's in terms of the number of scenes, since flat renderings are generally more available and lighter in storage. Note that KPAL is only a contextual flat denoising baseline because it cannot preserve the deep image structure; OIDN [Áfra 2023] provides supplemental context for KPAL's flat quality.

We employ SMAPE [Flores 1986], DSSIM (1−SSIM [Wang et al. 2004]), and ꟻLIP [Andersson et al. 2020] as metrics, applying gamma-2.2 tone mapping before computing the latter two.

6.2 Improvements on Still-Image Deep-Z Denoising

Previous deep image denoisers focus on processing static deep-Z imagery, where the neural DACNN method [Zhang et al. 2024] achieves much higher denoising quality than the traditional method based on

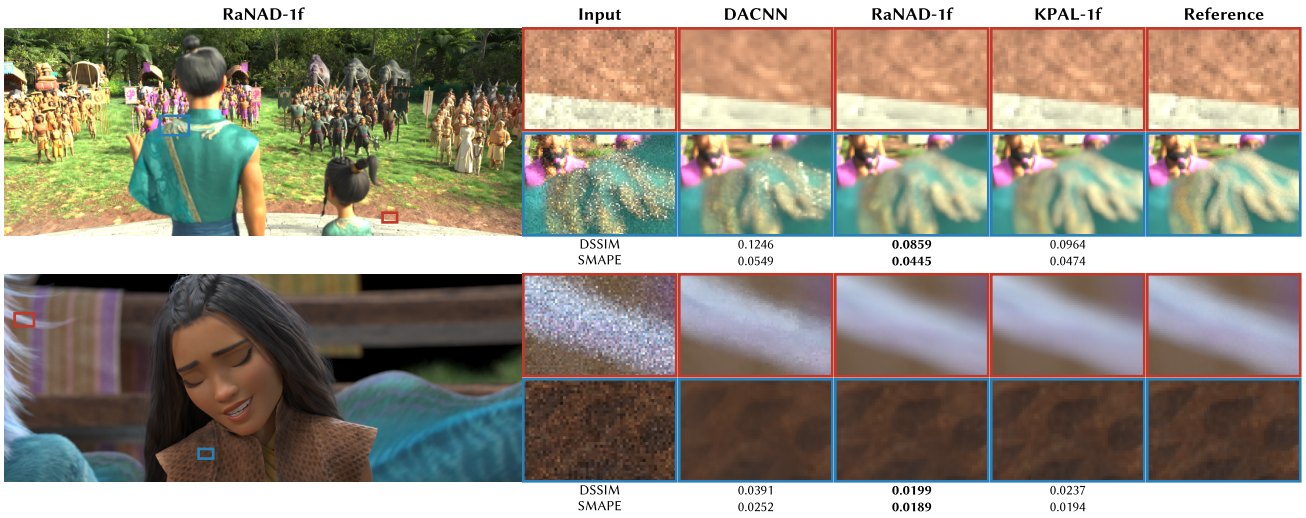

Fig. 9. Still-image deep-Z denoising comparison between DACNN and RaNAD-1f. RaNAD-1f produces sharper details and fewer artifacts across various scene types. Full results for all test scenes are available in the supplemental image viewer. © Disney

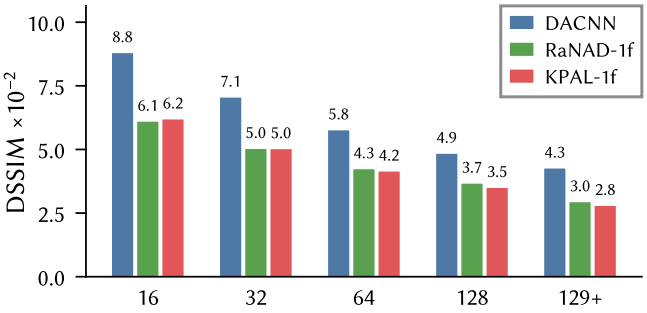

Fig. 10. Single-frame flattened denoising comparison on Dataset A1. RaNAD-1f significantly outperforms DACNN for deep-Z denoising and achieves competitive flat denoising quality compared to KPAL-1f. Lower is better. Additional metrics are in the supplemental.

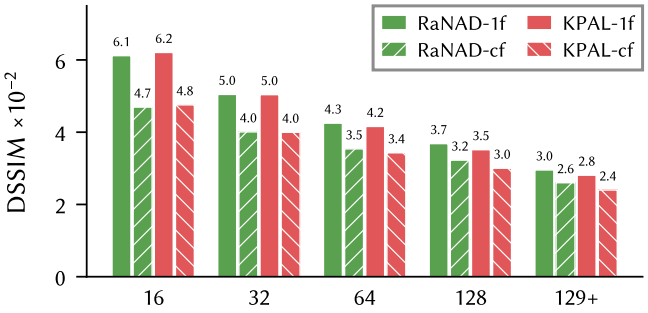

Fig. 11. Cross-frame flattened denoising comparison on Dataset A1. Hatched bars indicate cross-frame versions. RaNAD processes deep images; flat metrics are computed on the flattened output. Lower is better. Additional metrics are in the supplemental.

NL-Means [Vicini et al. 2019]. In this category, our method significantly outperforms the current state-of-the-art DACNN denoiser. Based on our quantitative evaluation shown in Figure 10, our single-frame RaNAD model achieves lower error than DACNN using deep images rendered with half the path samples. When evaluated after flattening, RaNAD attains quality competitive with high-quality flat denoisers such as KPAL, while preserving the deep representation needed for downstream compositing rather than denoising only the composited flat result. Visually, RaNAD produces sharper details and fewer artifacts than DACNN (Figure 9).

We also evaluate depth separation by progressively clipping foreground or background bins. RaNAD achieves lower error than DACNN at all bin-preservation percentages (see supplemental material), further supporting its single-frame deep-Z denoising advantage.

We also find that a capacity-matched single-U-Net is slightly stronger for flattened single-frame evaluation: 3.6% lower SMAPE and 6.4% lower DSSIM on Dataset A1, and 1.3% lower SMAPE and 2.1% lower DSSIM on Dataset B (see supplemental material). However, extending it directly to spatiotemporal deep denoising would increase computation and memory, as discussed in Section 4.1.2.

6.3 Cross-Frame Deep EXR Denoising

One of the main benefits of RaNAD compared to DACNN is its ability to perform spatiotemporal denoising. Below, we provide evidence on the importance of utilizing information from neighboring frames by showing the denoising quality improvements from single-frame to cross-frame RaNAD. Besides, as there are no previous spatiotemporal denoisers for deep images to the best of our knowledge, we compare cross-frame RaNAD (after flattening) and cross-frame KPAL as contextual evidence.

Figure 11 summarizes the error metrics for the cross-frame methods on Dataset A1. Notably, RaNAD-cf improves significantly over

RaNAD-1f in most scenes for all metrics we tested. The comparison between RaNAD-cf and KPAL-cf is similar to that between the single-frame versions: RaNAD-cf shows comparable performance to KPAL-cf, and on multiple scenes and SPPs it even outperforms KPAL-cf. Notably, RaNAD-cf achieves error levels comparable to RaNAD-1f at 2–4× higher sample counts. Visual comparisons for all test scenes are available in the supplemental image viewer, and the temporal stability comparison between RaNAD-1f and RaNAD-cf is shown in the supplemental video with KPAL-cf provided as context.

6.4 Deep-ObjectID Denoising

RaNAD models are also capable of denoising deep-OID images, as they were trained on both deep-Z and deep-OID data. Results confirm that denoising either format produces nearly identical flat metrics (see supplemental). Notably, deep-OID can separate objects that are close in depth, and Figure 1 shows examples of this separation being preserved. We do not include cross-frame deep-OID denoising results due to data unavailability, though the RaNAD architecture can ingest deep-OID frame sequences.

6.5 Computation Costs

Deep images pose significant challenges for denoisers' computational efficiency, not only due to their ragged data structure but also due to the significant increase in size now that each pixel can hold multiple bins. On a machine with an AMD EPYC 7313P 16-Core CPU, 48GiB RAM, and an NVIDIA GeForce RTX 3090 Ti GPU (20 GiB VRAM), we evaluate RaNAD's performance and show its practicality within the context of high-quality offline rendering.

6.5.1 Denoising Timings. RaNAD's execution time correlates with the number of bins rather than pixels. On GPU, single-frame denoising takes 15–150 s depending on bin count, while cross-frame with a 7-frame window takes 190–515 s (3–4× slower); execution on the CPU is 10–20 times slower on our test hardware (see supplemental material). KPAL runs in seconds on GPU because dense images are more GPU-friendly, despite comparable parameter counts: 15M/8M for RaNAD-1f/RaNAD-cf vs. 16M/9M for KPAL-1f/KPAL-cf (cross-frame models have fewer parameters due to the memory constraint to process more frames). Compared to typical render times on the order of hours, RaNAD's speed remains practical for offline denoising and compositing workflows.

6.5.2 Multi-Scale Reconstruction. Compared to single-scale reconstruction with a large radius ($R = 8.5$), our multi-scale reconstruction module reduces computation cost by up to 20% while providing slight quality benefits (see supplemental for ablation).

6.5.3 Denoising Arbitrary Output Variables. Kernel-based methods enable efficient and stable denoising of arbitrary output variables (AOVs) through kernel reuse. A typical use case for this is denoising *per-light outputs*, where each AOV layer contains only contribution from a certain light group; kernel reuse ensures that the denoised AOVs would mathematically sum up to the denoised color layer. For RaNAD models, denoising kernels are attention weights and are not directly predicted, and kernel reuse means saving the embeddings from the multi-U-Net backbone and rerunning the reconstruction blocks for the AOVs. On a test example with 33 light groups, kernel reuse allows denoising all the AOVs in 46.3 seconds with single-frame RaNAD, less than twice the cost of the primary denoising pass (29.7 seconds) which produced the core network embeddings. Denoising each AOV with a separate denoising pass would have taken 980.1 seconds instead—also without recombination consistency.

Fig. 12. Representative still-image hard case with residual noise. The crop shows a challenging transparent foreground where many pixels contain either no reliable foreground bin or only extremely noisy foreground bins. RaNAD-1f improves over DACNN but can still leave residual noise because few synergistic neighboring bins are available; KPAL-1f can reduce the flattened appearance by blurring, but does not preserve the deep separation needed for compositing. © Disney

7 Limitations and Future Work

We acknowledge several limitations of our approach. First, despite our regularization techniques, training can occasionally be unstable due to the large range of deep image data values. Second, attention-based down-sampling can produce subtle low-sample grid artifacts; alternative deep-image resampling remains open (further discussion in supplemental material). Third, while our trained models can process volumetric effects (smoke, fire, fog), they show more residual noise than flat denoisers on such content, as our training data focuses on hard surfaces; extending RaNAD to volumetric deep images with appropriate auxiliary features [Zhang et al. 2022] is a natural direction for future work. Finally, highly noisy transparent or particle-like foregrounds can leave residual artifacts in complex depth structures (Figure 12). Adaptive sampling that balances samples across bins within a pixel could alleviate this issue.

Beyond addressing these limitations, we believe RaNA presents opportunities for adapting other neural image processing methods (*e.g.*, frame interpolation, stylization) to deep images, where object separation can help resolve boundaries.

8 Conclusion

We introduced the ragged neighborhood attention denoiser (RaNAD) to tackle the problem of deep image denoising. RaNAD uses the ragged neighborhood attention (RaNA) operator as the building block, which is designed to gather information on semi-structured deep images. Thanks to RaNA's flexibility, RaNAD can perform multi-scale feature embedding and reconstruction, and can be easily extended to process information from neighboring frames. Experiments on production-quality data showed that, compared to previous state of the art in deep image denoising, RaNAD improves denoising quality substantially while preserving the deep image structure. It also supports both deep-Z and deep-OID images, extends naturally to cross-frame denoising, and remains competitive

with strong flat-image denoisers after flattening. We believe that RaNAD greatly enhances the practicality of deep compositing in offline production, and the RaNA operator opens doors for adapting other neural image processing techniques to work with deep images.

Acknowledgments

We thank the reviewers for their constructive feedback, David Adler and Per Christensen for proofreading, and Sam Cordingley, André Mazzone, Mark Meyer, Akshay Shah, and Shilin Zhu for useful discussions.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, et al. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. Software available from https://tensorflow.org (accessed: 2026-05-04).

Attila T. Áfra. 2023. Intel® Open Image Denoise. https://www.openimagedenoise.org (accessed: 2026-05-04).

Pontus Andersson, Jim Nilsson, Tomas Akenine-Möller, Magnus Oskarsson, Kalle Åström, and Mark D. Fairchild. 2020. FLIP: A Difference Evaluator for Alternating Images. *Proc. ACM on Computer Graphics and Interactive Techniques* 3, 2, Article 15 (Aug. 2020), 23 pages. doi:10.1145/3406183

Steve Bako, Thijs Vogels, Brian McWilliams, Mark Meyer, Jan Novák, Alex Harvill, Pradeep Sen, Tony DeRose, and Fabrice Rousselle. 2017. Kernel-Predicting Convolutional Networks for Denoising Monte Carlo Renderings. *ACM Trans. Graph.* 36, 4, Article 97 (July 2017), 14 pages. doi:10.1145/3072959.3073708

Irwan Bello, Barret Zoph, Ashish Vaswani, Jonathon Shlens, and Quoc V. Le. 2019. Attention Augmented Convolutional Networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 3286–3295. doi:10.1109/ICCV.2019.00338

Ron Brinkmann. 2008. *The Art and Science of Digital Compositing, Second Edition: Techniques for Visual Effects, Animation and Motion Graphics* (2 ed.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Brent Burley, David Adler, Matt Jen-Yuan Chiang, Hank Driskill, Ralf Habel, Patrick Kelly, Peter Kutz, Yining Karl Li, and Daniel Teece. 2018. The Design and Evolution of Disney's Hyperion Renderer. *ACM Trans. Graph.* 37, 3, Article 33 (July 2018), 22 pages. doi:10.1145/3182159

Chuhao Chen, Yuze He, and Tzu-Mao Li. 2024. Temporally Stable Metropolis Light Transport Denoising using Recurrent Transformer Blocks. *ACM Trans. Graph.* 43, 4, Article 123 (July 2024), 14 pages. doi:10.1145/3658218

Per Christensen, Julian Fong, Jonathan Shade, Wayne Wooten, Brenden Schubert, Andrew Kensler, Stephen Friedman, Charlie Kilpatrick, Cliff Ramshaw, Marc Bannister, Brenton Rayner, Jonathan Brouillat, and Max Liani. 2018. RenderMan: An Advanced Path-Tracing Architecture for Movie Rendering. *ACM Trans. Graph.* 37, 3, Article 30 (Aug. 2018), 21 pages. doi:10.1145/3182162

Tri Dao. 2023. FlashAttention-2: Faster Attention with Better Parallelism and Work Partitioning. arXiv:2307.08691. https://arxiv.org/abs/2307.08691 arXiv preprint.

Benito E Flores. 1986. A Pragmatic View of Accuracy Measurement in Forecasting. *Omega* 14, 2 (1986), 93–98. doi:10.1016/0305-0483(86)90013-7

Foundry. 2023. Deep Compositing in Nuke. https://learn.foundry.com/nuke/content/comp_environment/deep/deep_compositing.html (accessed: 2026-05-04).

Michaël Gharbi, Tzu-Mao Li, Miika Aittala, Jaakko Lehtinen, and Frédo Durand. 2019. Sample-based Monte Carlo Denoising Using a Kernel-splatting Network. *ACM Trans. Graph.* 38, 4, Article 125 (July 2019), 12 pages. doi:10.1145/3306346.3322954

Ali Hassani, Steven Walton, Jiachen Li, Shen Li, and Humphrey Shi. 2023. Neighborhood Attention Transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 6185–6194. doi:10.1109/CVPR52729.2023.00599

J. Hasselgren, J. Munkberg, M. Salvi, A. Patney, and A. Lefohn. 2020. Neural Temporal Adaptive Sampling and Denoising. *Computer Graphics Forum* 39, 2 (May 2020), 147–155. doi:10.1111/cgf.13919

Nikolai Hofmann, Jon Hasselgren, and Jacob Munkberg. 2023. Joint Neural Denoising of Surfaces and Volumes. *Proc. ACM on Computer Graphics and Interactive Techniques* 6, 1 (2023), 1–16. doi:10.1145/3585497

Mustafa Işık, Krishna Mullia, Matthew Fisher, Jonathan Eisenmann, and Michaël Gharbi. 2021. Interactive Monte Carlo Denoising Using Affinity of Neural Features. *ACM Trans. Graph.* 40, 4, Article 37 (July 2021), 13 pages. doi:10.1145/3450626.3459793

Florian Kainz. 2013a. Interpreting OpenEXR Deep Pixels. https://openexr.com/en/latest/InterpretingDeepPixels.html (accessed: 2026-05-04).

Florian Kainz. 2013b. Technical Introduction to OpenEXR. https://openexr.com/en/latest/TechnicalIntroduction.html (accessed: 2026-05-04).

Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. 2021. SwinIR: Image Restoration Using Swin Transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*. 1833–1844. doi:10.1109/ICCVW54120.2021.00210

Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations (ICLR)*. https://openreview.net/forum?id=Bkg6RiCqY7

Jacob Munkberg and Jon Hasselgren. 2020. Neural denoising with layer embeddings. *Computer Graphics Forum* 39, 4 (2020), 1–12. doi:10.1111/cgf.14049

TensorFlow Authors. 2025. Ragged Tensors. https://www.tensorflow.org/guide/ragged_tensor. Accessed: 2026-05-04.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 6000–6010.

Delio Vicini, David Adler, Jan Novák, Fabrice Rousselle, and Brent Burley. 2019. Denoising Deep Monte Carlo Renderings. *Computer Graphics Forum* 38, 1 (2019), 316–327. doi:10.1111/cgf.13533

Thijs Vogels, Fabrice Rousselle, Brian McWilliams, Gerhard Röthlin, Alex Harvill, David Adler, Mark Meyer, and Jan Novák. 2018. Denoising with Kernel Prediction and Asymmetric Loss Functions. *ACM Trans. Graph.* 37, 4, Article 124 (July 2018), 15 pages. doi:10.1145/3197517.3201388

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

Bing Xu, Junfei Zhang, Rui Wang, Kun Xu, Yong-Liang Yang, Chuan Li, and Rui Tang. 2019. Adversarial Monte Carlo Denoising with Conditioned Auxiliary Feature Modulation. *ACM Trans. Graph.* 38, 6, Article 224 (Nov. 2019), 12 pages. doi:10.1145/3355089.3356547

Jingfeng Yao, Xinggang Wang, Shusheng Yang, and Baoyuan Wang. 2024. ViTMatte: Boosting image matting with pre-trained plain vision transformers. *Information Fusion* 103 (2024), 102091. doi:10.1016/j.inffus.2023.102091

Jiaqi Yu, Yongwei Nie, Chengjiang Long, Wenju Xu, Qing Zhang, and Guiqing Li. 2021. Monte Carlo denoising via auxiliary feature guided self-attention. *ACM Trans. Graph.* 40, 6, Article 273 (Dec. 2021), 13 pages. doi:10.1145/3478513.3480565

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 (CVPR)*. 5728–5739. doi:10.1109/CVPR52688.2022.00564

Xianyao Zhang, Marco Manzi, Thijs Vogels, Henrik Dahlberg, Markus Gross, and Marios Papas. 2021. Deep Compositional Denoising for High-quality Monte Carlo Rendering. *Computer Graphics Forum* 40, 4 (July 2021), 1–13. doi:10.1111/cgf.14337

Xianyao Zhang, Melvin Ott, Marco Manzi, Markus Gross, and Marios Papas. 2022. Automatic Feature Selection for Denoising Volumetric Renderings. *Computer Graphics Forum* 41, 4 (2022), 63–77. doi:10.1111/cgf.14587

Xianyao Zhang, Gerhard Röthlin, Marco Manzi, Markus Gross, and Marios Papas. 2023. Deep Compositional Denoising on Frame Sequences. In *Eurographics Symposium on Rendering*, Tobias Ritschel and Andrea Weidlich (Eds.). The Eurographics Association. doi:10.2312/sr.20231142

Xianyao Zhang, Gerhard Röthlin, Shilin Zhu, Tunç Ozan Aydin, Farnood Salehi, Markus Gross, and Marios Papas. 2024. Neural Denoising for Deep-Z Monte Carlo Renderings. *Computer Graphics Forum* 43, 2 (2024), 18 pages. doi:10.1111/cgf.15050

Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. 2021. Point Transformer. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. 16239–16248. doi:10.1109/ICCV48922.2021.01595