Foveated Monte-Carlo Denoising

Nicholas Milef
nicholas.milef@tamu.edu
Texas A&M University
College Station, Texas, USA

Nima Khademi Kalantari
nimak@tamu.edu
Texas A&M University
College Station, Texas, USA

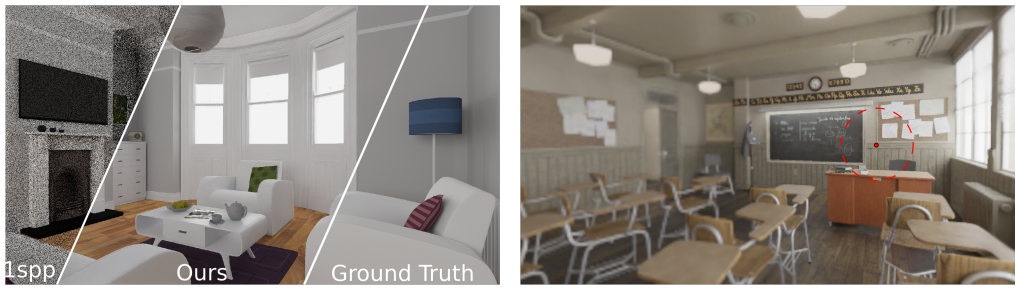

Figure 1: Our approach accurately denoises 1spp Monte-Carlo renderings at full resolution (left) and is capable of denoising in an accelerated foveated manner (right).

ABSTRACT

In this work, we propose a temporally-stable denoising system that is capable of reconstructing MC renderings in a foveated manner. We develop a multi-scale convolutional neural network that starts at a base (downsampled) resolution and denoises progressively higher resolutions. Our network learns to use the lower resolutions and the previous frames to denoise each foveal layer. We demonstrate how this architecture produces accurate denoised results at a much lower computational cost.

CCS CONCEPTS

• **Computing methodologies** → **Ray tracing**; *Neural networks*; Image processing.

KEYWORDS

foveated rendering, denoising, Monte-Carlo rendering, neural networks

ACM Reference Format:
Nicholas Milef and Nima Khademi Kalantari. 2021. Foveated Monte-Carlo Denoising. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Posters (SIGGRAPH '21 Posters), August 09-13, 2021.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3450618.3469140

1 INTRODUCTION

Path-tracing has largely become the de facto technique for rendering animations in the film industry. While path-tracing produces more realistic visuals, convergence towards a visually noise-free image often requires thousands of samples and long computation times. This is particularly problematic for video game and virtual reality applications that must deliver high frame rates at increasingly high resolutions, severely limiting the computational budget for path-tracing. To reduce the number of samples needed, denoising filters can be used to reconstruct missing details and remove noise in rendered images. Denoising images with an extremely low number of samples still poses a major challenge and can itself add processing overhead. Is it possible to use foveation to speed up Monte-Carlo denoising?

In this work, we propose a multi-scale deep-learning approach that supports foveated denoising. Inspired by the network designs of Hasselgren et al.[Hasselgren et al. 2020] and Vogels et al. [Vogels et al. 2018], we perform the denoising process at multiple scales using a sequence of small CNNs. Our system downsamples the input image and auxiliary features, such as shading normal and depth, to different resolutions and denoises these progressively, feeding the lower resolution as input to the next layer. We reproject past denoised frames and use these warped frames as input into our network to improve temporal stability. Our denoiser is inherently able to perform foveated denoising by producing denoised images for different retinal eccentricity levels, which can be composited into a foveated image [Guenter et al. 2012]. This allows us to significantly cut down on inference time by generating only a small portion of the image around the gaze point at finer scales.

2 APPROACH

In our system, we progressively downsample the input by a factor of 2 and use it at each scale. In our implementation, we use a set of $N = 4$ scales. We perform the denoising process at each scale using a small UNet. Specifically, starting from the coarsest scale, our network's input includes 6 channels (3 for illumination, 2 for normal, and 1 for depth) and outputs the denoised illumination. To ensure temporal coherence, we warp the previous frame's denoised using screen-space motion vectors. Similar to the other auxiliary

SIGGRAPH '21 Posters, August 09-13, 2021, Virtual Event, USA

ACM ISBN 978-1-4503-8371-4/21/08.
https://doi.org/10.1145/3450618.3469140

features, we downsample the reprojected denoised illumination by a factor of 2 and use it as input to the neural network for each layer.

We compute the denoised illumination at the current scale, through a weighted combination of the intermediate output at the current scale, a low frequency extraction of the intermediate output, and the upsampled denoised image from the coarser scale. We also compute a soft mask to blend the last frame's reprojected denoised illumination with the current frame's denoised illumination. Training is done without foveation (i.e., all layers have the same resolution).

2.1 Foveation

Full-resolution rendering is only necessary for a small area centered around the gaze point [Guenter et al. 2012]. As eccentricity increases, spatial resolution decreases substantially. In our system, we take advantage of this to significantly reduce the computational cost for generating foveated images. We follow the findings, presented by Guenter et al. [Guenter et al. 2012], that demonstrate a user-validated resolution model for foveation. Specifically, we discretize the eccentricity angles into 4 different scales and obtain the radial eccentricity e_i (measured in degrees) at each scale as follows:

$$e_i = \frac{s_{i+1} w^* - w_0}{m} \tag{1}$$

where we use the default parameters by Guenter et al. for a 51cm-wide display, distance of 59cm to the screen, and resolution of 1920×1080: $w_0 = 1/48$, $w^* = 0.0516$, and $m = 0.0220$. m is a modifiable slope parameter that can be changed depending on the scene, but we use the suggested conservative threshold that was validated through user studies [Guenter et al. 2012]. Moreover, s_i is the downsampling factor of the scale i, i.e., $s_3 = 4$. The angular eccentricity in degrees e_i can then be easily converted eccentricity in pixels, r_i, based on the distance from the screen and the resolution of the display. Note that, we compute the eccentricities only for scale $i = \{1, 2, 3\}$, since the coarsest scale is essentially the base layer and needs to be generated at full size.

We obtain the final foveated image by alpha blending different scales from the coarsest to the finest. We compute the blending map for each scale according to the eccentricity at that scale, using the smoothstep equation (Hermite cubic). Only a small area around the gaze point is denoised at full resolution. Other areas are denoised at lower resolutions, depending on the eccentricity, and then bilinearly upsampled to save computation time.

3 RESULTS

Our network's frame time is approximately 68.4ms on a 1920×1080 resolution image without foveation. With foveation, the best case frame time is 9.6ms (7.1x speedup) and the worst case is 14.1ms (4.8x speedup). Times were measured on an NVIDIA RTX 2080 Super GPU. The best case occurs when the gaze point is focused on the corner of the screen while the worst case occurs while the gaze point is focused on the center of the screen. It is important to note that the speedup amount largely depends on the central gaze region, which can also change with more aggressive parameters.

As shown in Fig. 2, our denoiser performs denoising at a higher resolution only around the gaze point. The reference images uses

Figure 2: Our foveated results for 1spp input. The red dot represents the gaze point.

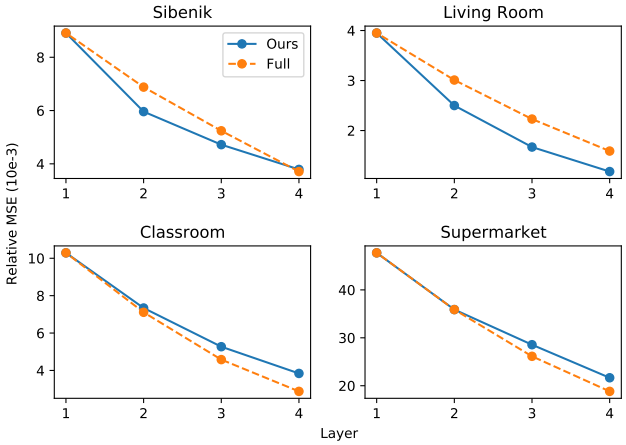

Figure 3: Image quality for each foveal layer. Layer 1 is denoised at full resolution for various scenes.

the same foveation logic [Guenter et al. 2012] as our denoiser. Despite only using a 1spp input, our denoiser is able to produce output close to the reference images.

Foveation quality depends on the quality of each layer output. To this end, we run two tests to evaluate our network's per-layer output quality. In our first test, we compare our method's per-layer output to the ground truth image bilinearly downsampled to the size of the respective layer. In our second test, we bilinearly downsample our network's full resolution output (i.e., layer 1) and compare it to a bilinearly downsampled full-resolution ground truth image. We compute the relative MSE for the comparisons and depict our results in Fig. 3. Our network produces a similar relative MSE for lower levels despite requiring significantly less computation.

REFERENCES

Brian Guenter, Mark Finch, Steven Drucker, Desney Tan, and John Snyder. 2012. Foveated 3D graphics. *ACM Transactions on Graphics (TOG)* 31, 6 (2012), 1–10.

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.

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 Transactions on Graphics (Proceedings of SIGGRAPH 2018)* 37, 4, Article 124 (2018), 124:1–124:15 pages. https://doi.org/10.1145/3197517.3201388