Approximate Global Illumination by combining Screen Space Directional Occlusion and Recursive Non-local Mean Denoising

Gururaj U Malipatil
School of Computer Science and Engineering
REVA University & Qualcomm pvt. ltd
Bangalore, India
guru.patil9@gmail.com

Sunilkumar S Manvi
School of Computer Science and Engineering
REVA University
Bangalore, India
ssmanvi@reva.edu.in

***Abstract*—Complex techniques with substantial processing expenses are required to produce photorealistic computer-based photographs. Implementing these methods requires the use of techniques like Ray tracing and Path tracing, which are still expensive in terms of computation even with the most recent hardware-optimized GPUs. As a result, faster and more precise techniques for rendering 3D environments are required to give visual programme users a near-perfect photo-realistic interactive experience.**

For real-time interactive rendering, our proposed hybrid ray-marching/rasterization strategy uses fast non-local mean denoising methods and computes Screen Space Directional Occlusion(SSDO), Screen Space Ambient Occlusion(SSAO) based on the type of surface. Recursive non-local mean Spatio-temporal filter and SSDO with realistic graphics can be achieved by using a rendering model based on raymarching/raytracing, which uses sphere trace or bounding volume hierarchy to calculate intersections with scene objects. For global illumination, we take two indirect bounces of lights in the scene and direct light at rasterized implicit shape surfaces, and we estimate this with one ray traced sample per pixel. We separate the surfaces into matte and glossy to apply SSAO and SSDO and with confidence measurement accumulated over frames by fast approximated non-local mean denoising filter. A rapid approximated recursive non-local filter is used to avoid noise and smooth edge areas in areas where reprojection fails. We demonstrate generating photo-realistic images with fewer artifacts against the offline denoising algorithms

Keywords— global illumination, Denoising, BRDF, SSAO, SSDO, ray tracing, lighting, non-local mean

I. INTRODUCTION

Engines such as Embree and Optix, which are parallel ray casting by efficiently use of CPU and GPU, may conduct path trace global illumination in real-time with considerable noisy results. There is a compromise between ray bounce/sample per pixel and computation cost, ray budgets are anticipated to expand ten-thousand-fold in the future. Offline rendering procedures that require hundreds of samples per pixel represent most of the earlier high-quality denoising approaches for path tracing.

Ray bounce count(cost budgets) required for processing increase with the advancement of CPU and GPU hardware capabilities. Path tracing will eventually increase to a thousand-fold bounce and the convergence threshold will be limited however, we need to do post-processing on a frame and apply denoising filters. Most of the photo-realistic algorithms are based on denoising filter techniques are offline rendering. In this paper, We use Screen Space Ambient Occlusion(SSAO) and Screen Space Directional Occlusion(SSDO) based on the factored Bidirectional reflectance distribution function(BRDF) for the surfaces like glossy and matte. In addition, we add non-local mean temporal filtering across the sequence of the frames to get refined and attractive results. This produces slightly faster and smoother rendering results without blurred surfaces than offline rendering.

The Contribution to the paper is about designing the Spatio-temporal framework and using SSAO and SSDO based on the classified surfaces by ray bounce. The key idea is about approximating the monte-carlo integrator by applying SSAO and SSDO for the matte and glossy surfaces and is also based on the confidence interval of the non-local mean filter over the frames. This produces accountable results in terms of lights reflective and refractive on the surfaces in the scene.

Based on several observations for photo-realistic algorithms, the details covered by reflected light are because of material property and direct light illumination than the light in the scene. We consider SSAO for matte surfaces because we need to consider light bounce in the entire semi-hemisphere and therefore its computationally expensive and exceedingly noisy-but the same dispersion means that the light can be intrusively denoised. In specular surfaces, the specular highlight region is where a lot of light gets clustered. So, we use SSDO where specular lobes direction is considered and can be denoised with filters. In temporal non-local mean, reflection is based on object reflection and not on surface points but the failure is negotiable and can be improved by filters.

Our Method has a different path for the type of lights in the scene (i.e indirect and direct) and material properties. As the frame gets rendered the results are observed to suppress the artifacts and blurring with flicker-free frames.

The following is how the paper is structured. Section II is about related works and the limitation of previous techniques. Section III presents the proposed rendering model algorithm and Section IV presents the system requirement used for Experimental purposes. The results were discussed in Section V . Finally, in Section VI, we provide conclusions.

II. RELATED WORK

Using the instant radiosity method developed by Keller[1], the indirect light component of global illumination can be approximated using a set of Virtual Point Lights (VPLs) in a scene. By shooting photons into the scene and having them intersect with the surface there, this technique formed VPLs, which were then used to light up the scene to render it. For a large number of point lights, there is a computational cost in the calculation of shadows. Lightcuts Walter et al.,[5] is a technique that reduces the number of shadow queries by clustering the VPLs in sets, however, it does not increase performance significantly. This is done by interpolating indirect illumination from an onboard camera's cache Ward et al.,[14]. It's possible to get accurate estimates of the irradiance of some surfaces using ray tracing, and for the rest, interpolation is utilized.

By using spherical harmonics instead of irradiance to store and interpolate indirect illumination, Radiance caching Krivanek et al., [4] adds to the irradiance cache. Specular surfaces can be handled by this method, which requires fewer cache samples. Although some methods for global lighting via denoising have been viewed as an offline rendering problem, the path-tracing method was a prevalent offline process until recently.

The survey by Zwicker et al.[10] Methods based on non-local means (NLM) by Rousselle et al.[15] uses a component of a discontinuously segmented image's zeroth- or first-order gradient. Buffers are utilized to establish the weighting of each sample for each term based on material geometry and other metadata such as light visibility. The motion vector buffer is known as the G-buffer. The G-buffer is used to determine the weights of several machine learning algorithms Kalantari et al.[16] and other edge-avoiding filters e.g., Dammertz et al.[17]). Eisemann and Durand,[18]; Petschnigg et al. and colleagues used Cross bilateral filters in real-time stochastic transparency and computational photography can also be used in offline denoising. Gaussian kernels with each tap weighted by a G-buffer function are used in these. Antialiasing and real-time stochastic transparency Salvi.[19] both make heavy use of projection and temporal filtering. The first time they were used for interactive denoising in Monte Carlo rendering was by Bauszat et al.[21]. Previous Ambient Occlusion(AO) approximation approaches (Lehtinen and Kautz. 2003) required a great deal of pre-computation for Directional Occlusion(DO) and interreflections parameters to save data in a compressed format with restricting spatial or direction design. For example, check related works for matte Durand et al.[18]; Kontkanen et al.[20]; Soler et al.[22] and glossy reconstruction filters aimed to minimize/attenuate bias instead of the visual artifacts that we eliminated by combining SSDO and temporal non-local mean filter kernels.

Real-time denoising was developed by Schied and colleagues.[23] using route-traced images with one path per pixel. Their study was independent and conducted at the same time as ours, so we plan to compare our findings with theirs in the future.

III. PROPOSED WORK

The proposed work comprises five stages (shown in Figure.2): Pre-Pass, PathTrace, Accumulation(Spatio-temporal), Approximation, and Reconstruction. Before proceeding with stages, Our stages are based on the rendering equation and the factorization of the Bidirectional Reflectance Distribution Function(BRDF) function.

To achieve global illumination, we have to know the characteristics of light behavior, characteristics of objects in the scene, interacting with objects in the scene, and finally reaching the eye/camera. The illumination in the scene is dependent on direct and indirect lighting given by the rendering equation:

$$L_o(\omega_o) = \int_\pi L_{direct}(\omega_i)\cos\theta_i f_r(\omega_i, \omega_o) d\omega_i + \int_\pi L_{indirect}(\omega_i)\cos\theta_i f_r(\omega_i, \omega_o) d\omega_i \quad (1)$$

where, ω_o is outgoing light direction, ω_i incoming light direction, $\cos\theta_i$ is dot product of normal and incoming light, $f_r(\omega_i, \omega_o)$ is Bidirectional Reflectance Distribution Function.

Bidirectional reflectance distribution function(BRDF): It is a function that describes the distribution of how much light is reflected from a material or surface when light arrives or falls on it. In general, its ratio of radiance(direction of the camera) by irradiance(direction of light). Refer to Figure 1.

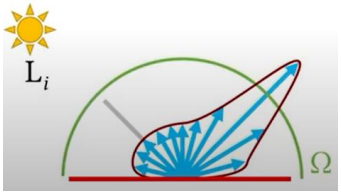

Figure 1: Representation of BRDF.

Our pipline rendering system is processed in stages and in each stage we calculate depth, normals and non-local mean search window data and store in GPU buffers. Figure 2 describes various stages used in our method.

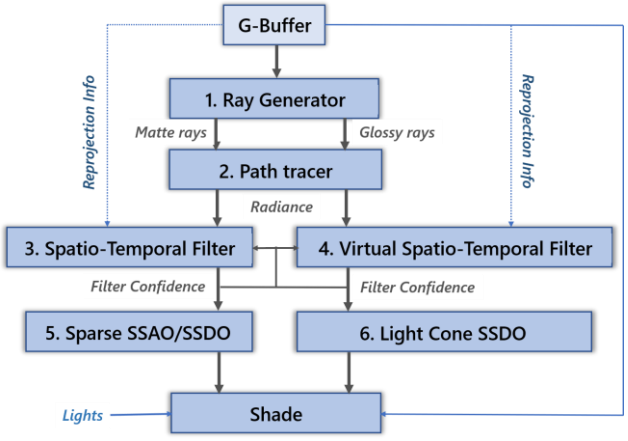

Figure 2: Rendering system (Blue arrow represent G-Buffer and gray arrow represents radiance buffer)

Figure 2. consists of stages and is numbered. In the following sections, we will explain the stages used.

A. *PrePass*

Prepass - Write albedo, normals, and other common G-Buffer attachments to the scene's G-Buffer file. A ray-trace pre-pass (stage 1) is required instead of a raster prepass for the first bounce versions of those buffers.

B. PathTrace

Split denoising with Specular Reflections and Global Illumination in different attachments would be great because reflection denoising performs better on first bounce data, whereas Global Illumination/Ambient Occlusion/Shadows are calculated with spherical harmonic coefficients (stage 2). However, we have found the optimal bounce count by comparing vs rendering.

C. PathTrace

Recursive non-local denoising algorithm:

However, a non-local mean of the frame is more compatible with the approximate RNLM [2] method's purpose of using spatiotemporal information effectively(Fig 2). Pixel estimates from the current frame and the preceding frame are combined to create the RNLM estimates. RNLM additionally uses the difference in convolutions in the search window for consecutive frames to execute on the GPU in parallel. In other words, the current output is generated by fusing the current input frame with the previous output frame. Because temporal recursive (stages 3 and 4) processing reduces overall computing complexity without considerably extending the search window, it helps to gain temporal signal correlations.

The estimate given for the proposed RNLM is:

$$\hat{e}_k(i) = \frac{1}{W_{k,i}}\left[\sum_{j\in\epsilon(i)} w_{\hat{e},k}(i)\hat{e}_{k-1}(p_k(i)) + \sum_{j\in\epsilon(i)} w_{y,k}y_k(j)\right] \quad (2)$$

where $\hat{e}_k(i)$ represents the estimated image at pixel i in frame k. and $\hat{e}_{k-1}s_k(i)$ is an estimate of the previous frame $(i.e, frame\ k-1)$ at pixel $p_k(i) \in \{1,2,â€¦,N\}$. Pixel $p_k(i)$ is selected from $\{\hat{e}_{k-1}(\cdot)\}$ based on the standard block-matching algorithm (BMA) for blocks of input frame k centered around pixel i. The choice of $p_k(i)$ takes into account block and search sizes that may differ from those used for in-frame processing. In particular, the block that corresponds to the block size is $N_bÃ—N_b$, with a search window of $N_sÃ—N_s$. The recursive weight of the expression (2) is $w_{\hat{e},k}(i)$, and the non-recursive weights is $w_{y,k}(i,j)$.

Screen Space directional occlusion:

For specular surfaces, we use SSDO (stage 6). SSDO is to improve screen space ambient occlusion (SSAO) and better approximate global lighting, the direction in which ambient light (both the light that hits the object directly and the light that is reflected by the object immediately behind it) is sampled is taken into consideration. Contributes to the direction of the incident light and,

- Incorporate one or two bounces of ray for calculating indirect illumination
- Complements standard object-based global illumination and
- The extra calculation time is small.

Factored BRDF:

Start by usually splitting the BRDF function into the type of surface like diffuse/matte layers [3]. Glossy/specular term in combination with Fresnel coefficient:

$$f(\hat{\omega}_i,\hat{\omega}_o) = m(\hat{\omega}_i,\hat{\omega}_o)(1-F(\hat{\omega}_i,\hat{\omega}_o))^2 + g(\hat{\omega}_i,\hat{\omega}_o)F(\hat{\omega}_i,\hat{\omega}_o) \quad (3)$$

The shiny term g() may contain a specular impulse. The diffuse term m() needs to change slowly with respect to $\hat{\omega}_i$ and $\hat{\omega}_o$ Like the Lambert model and the Oren-Nayar model.

Split Monte Carlo Integration:

By sampling a Monte Carlo integrator for standard materials(for light or more) using indirect light L_i we calculate outward radiance L_o at point X on the surface with key visibility. The Eq is given below:

$$L_o(X,\hat{\omega}_o) = \frac{1}{2N}\sum_{i}^{2N} L_i(X,\hat{\omega}_i)\frac{f(\hat{\omega}_i,\hat{\omega}_o)|\hat{n}\cdot\hat{\omega}_i|}{\rho(\hat{\omega}_i)} \quad (4)$$

Here, each $\hat{\omega}_i$ in the incident direction of 2N is independent. sampled from distributions $\rho\hat{\omega}$ and L_i is calculated by path tracing.

Non-zero distribution can be selected for ρ, where f is non-zero. The integrator makes the sample optimally important regarding the material, $\rho(\hat{\omega}) \propto f(\hat{\omega},\hat{\omega}_o)|\hat{n}\cdot\hat{\omega}|$. Different sampling distributions can be chossen for the various BRDF terms and its optional. In case of diffuse estimator (stage 5), we choose

$$\rho_m(\hat{\omega}) = \frac{max(\hat{\omega}\cdot\hat{n},0)}{\pi} \quad (5)$$

to cancel the numerator. For optimal important sampling when $m()$ is almost constant throughtout the hemisphere. To estimate specular term (stage 6), Select the incident direction $\hat{\omega}_j$ for the specular term from some distribution ρ_g, which is close to $g()$ but can be sampled efficiently like a power-cosine. (We differentiate between the use of index j for specular and index i for diffuse that these directions are sampled independently.) The complete estimator is:

$$L_o(X,\hat{\omega}_o) = \frac{\pi}{N}\sum_{i}^{N} L_i(X,\hat{\omega}_i)(1-F(\hat{\omega}_i,\hat{\omega}_o))^2 m(\hat{\omega}_i,\hat{\omega}_o)$$
$$+\frac{1}{N}\sum_{j}^{N} L_i(X,\hat{\omega}_j)\frac{F(\hat{\omega}_j,\hat{\omega}_o)g(\hat{\omega}_j,\hat{\omega}_o)|\hat{n}\cdot\hat{\omega}_j|}{\rho_g(\hat{\omega}_j)} \quad (6)$$

D. Approximation

Both the F and m parameters used in Eq 6. have a slow change in incidence and outgoing vector that can factor them from the integrator to separate material and light for approximation. Therefore, we need to select a single representative incident vector $\hat{\omega}_i$ outside the operator summation by selecting $\hat{\omega}_i = \hat{\omega}_s$,the specular mirror reflection of $\hat{\omega}_o$

As a result, the diffuse/matte part of the estimator (Eqn. 6) is reduced as follows

$$\frac{(1-F(\hat{\omega}_s,\hat{\omega}_o))^2 m(\hat{\omega}_s,\hat{\omega}_o)\pi}{N}\sum_{i}^{N} L_i(X,\hat{\omega}_i) \quad (7)$$

A similar approximation is made for specular estimators. However, because we know that g can be very sensitive to the incident direction $\hat{\omega}_i$ (that is, potentially have narrow lobes), g cannot be out of the summation and should be

evaluated for each sample. The glossy part of the estimator (Eqn. 6) is as follows.

$$\frac{F(\hat{\omega}_s, \hat{\omega}_o)}{N} \sum_{j}^{N} L_i(X, \hat{\omega}_j) \frac{g(\hat{\omega}_j, \hat{\omega}_o)|\hat{n} \cdot \hat{\omega}_j|}{\rho_g(\hat{\omega}_j)} \quad (8)$$

E. Reconstruction

The chain of filters, SSAO/SSDO-Filter using the G-buffer by camera-space plane distance in both normals and depth from the center sample with corresponding weights. Every G-buffer frame contains illumination obtained from neighboring samples that are spatially and temporally (recursively and nonlocally) nearby, allowing the convolution of the search window, X-position variation, and the implicit time parameter to be calculated (stages 3 and 4). The temporal (stage 4) using non-local mean filter does reverse re-projection from data of the previous frame. It computes the confidence interval of how well the reprojection positions match. The high-frequency noise is removed significantly by stages 3 and 4. Because of the aforementioned three factors, over-blurring of the image is not apparent when using this method

- Previous section approximation excludes material features (matt reflectance m and specular magnitude and albedo F0) from the total of the material properties.
- According to the roughness of the material, the diameter of the specular spatial core is determined.
- Compensate for motion with reverse reprojection.
- Caustic reflections are blurred and are hard to reconstruct from sparse samples.

(stage 5) does 3 x 3 SSAO filter for matte surfaces, which eliminates the low-frequency noise. SSAO is usually faster as it's operating in image space. (stage 6) does SSDO filter where the light direction is considered for glossy surfaces. The filters SSAO and SSDO (stages 3 and 4) are optimized for matte surfaces but because of reprojection, the temporal filter will lead to some blur for glossy surfaces when in motion.

IV. System Requirement

In Table I, we give a detailed account of the specification of the software and hardware of the machine we used for all test cases documented in this work, as well as for the render images presented. Finally, we want to note that all the results for rendering time and graphics fidelity presented in the following sections are rendered in 1280x720p resolution unless specified otherwise

Software	Qt Qml and OpenGL 3.3
Operating System	Windows 10.1 Version 1809 (build 17763.1098)
CPU	Intel i3-6600k
GPU	MSI Radeon R9 390 8GB GDDR5
GPU driver version	Radeon Software Adrenalin 2020 Edition 20.1.4

TABLE I System Specification

V. RESULTS

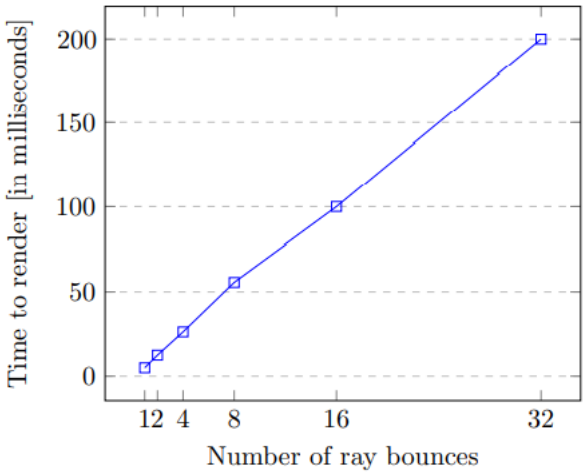

Figure 3: Data visualization of how changing the number of ray bounces affects the total rendering time of our model for the renders

The number of bounces that we take into account each time we march a new ray is critical to the image fidelity we want to provide. This parameter introduces a compromise between image fidelity and the time to render the final image: the higher the number of bounces, the more realistic the image will look but will be more expensive to compute. In order to understand the differences each ray bounce count imposes in the time to render the final image, we decided to plot the different data we measured from the multiple renders. The visualization of how each bounce count affects the time to render is displayed in Figure.3.

From Figure.3., we observe that in order to provide each frame in less than 41.6 milliseconds, we must pick a number of ray bounces that is smaller than 7. From this case study, we conclude that our range of optimal ray bounces is between the 4 and 6 range for a simple scene. Since we don't observe major differences between the 4 and 8 bounce count renders, we decided to set our default number of bounces to 4 for the rest of the case studies of our model.

A visualization of this temporal method is presented in Figure 4, where we can observe the progressive denoising applied over time to render fractal geometry. Our denoising method proved successful in delivering the same fidelity of graphics provided by renders of over 32 samples but only using 1 sample per pixel/ray without any visible noise after accumulating frames for a second. Applying the denoising process seen in Figure 4, we can provide the user with a time to render 76.9ms between frames. While this is only 13fps and is surely beneath our limit for real-time graphics rendering, rendering the same scene using a multi-sample approach would require 10 samples for it to converge to the last render seen in Figure 4 (10 frames accumulation). The suggested 10 sample render would require 10x the rendering time of the denoised render, thus providing each frame at 770ms, rendering the application unusable from a user interaction standpoint.

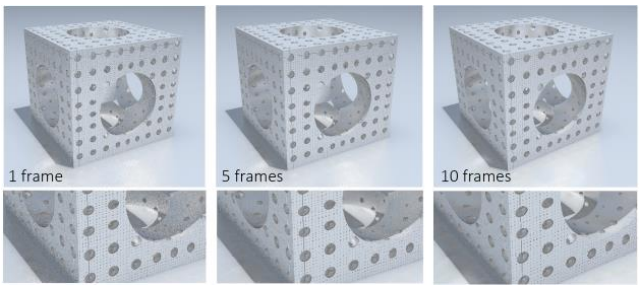

Figure 4: A comparison between noise accumulation in a different count of total frames elapsed

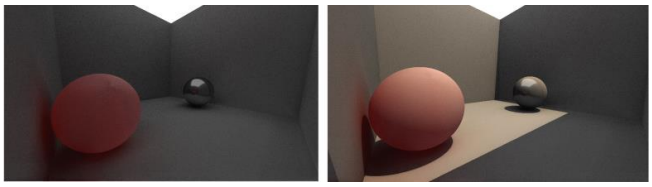

Figure 5: Global illumination and emmisive objects effects with denoised

Our denoising study case is a render of a simple scene composed of two spheres inside a room with no roof. Let's use it to analyze and compare the effects of global illumination produced by our rendering model against Blender's Cycles path tracer. The output produced by our method is shown in Figure 5 and the rendering time in ms of each stage(Figure 2) in our rendering pipeline in Table II

Stages	Time
Spatio-Temporal Filter (Matte&Glossy)	~45
SSAO (Matte Surface)	~5
SSDO (Glossy Surface)	~15

TABLE II: Per Frame rendering time in milliseconds at 1280 X 720 for filter GPU pipeline stages.

We observe that the render produced by our proposed model has some noise in a few cases. Nevertheless, we believe that for this scene our model provides a higher dynamic range of light when compared to Cycles's render: shadows are darker and the ambient occlusion factor is higher. Although this makes the version rendered by our proposed model look more realistic, we can't say that this is a clear win since these variables are easily configured in both models to match the wanted visuals without using extra computational power.

VI. CONCLUSION

Despite our conclusion that the model we proposed in this work is capable of delivering high-fidelity graphics in real-time, we are aware that it has a few limitations like blurring at low or high-frequency noise, in SSAO noise occurs for depth discontinuity at object edges, the closer glossy surfaces are blurr compare to distant ones.

Consequently, we wish to specify a couple of improvements in FPS by reducing the frame buffer pixel format and datatype precision. To make results stable and spatially smoother, we may improve denoise filter algorithms or replace them with faster denoising filters.

REFERENCES

[1] Keller, A. (1997). Instant radiosity. In Computer Graph ics (ACM SIGGRAPH 97 Proceedings), volume 31, pages 49-56.

[2] Redha A. Ali & Russell C. Hardie, "Recursive non-local means filter for video denoising," EURASIP Journal on Image and Video Processing, pp. 3–5, 2017.

[3] Michael Mara & Morgan McGuire & Benedikt Bitterli & Wojciech Jarosz, "An Efficient Denoising Algorithm for Global Illumination," in ACM SIGGRAPH / EuroGraphics High Performance Graphics 2017, July 28, 2017, pp. 1–7.

[4] Krivanek, J., Gautron, P., Pattanaik, S., and Boua- Â´ touch, K. (2005). Radiance caching for efficient global illumination computation. IEEE Transactions on Visualization and Computer Graphics, 11(5). Also available as Technical Report 1623, IRISA, http://graphics.cs.ucf.edu/RCache/index.php.

[5] Walter, B., Fernandez, S., Abree, A., Bala, K., Onikian, M., and Greenberg, D. P. (2005). Lightcuts: A scalable approach to illumination. In ACM SIGGRAPH 2005 Full Conference DVD-ROM, pages 1098-1107.

[6] Nvidia. Nvidia turing gpu architecture whitepaper, 2018.

[7] Simple DirectMedia Layer. https://www.libsdl.org/, 2020.

[8] B. Bitterli, F. Rousselle, B. Moon, J. A. Iglesias-Guitian, D. Adler, K. Mitchell, W. Jarosz, and J. NovÃ¡k. 2016. Nonlinearly Weighted First-order Regression for Denoising Monte Carlo Renderings.

[9] Y. Tokuyoshi. 2015. Specular Lobe-Aware Filtering and Upsampling for Interactive Indirect Illumination. Comp. Graph. Forum 34, 6 (2015), 135-147.

[10] M. Zwicker, W. Jarosz, J. Lehtinen, B. Moon, R. Ramamoorthi, F. Rousselle, P. Sen, C. Soler, and S.E. Yoon. 2015. Recent Advances in Adaptive Sampling and Reconstruction for Monte Carlo Rendering. Comput. Graph. Forum 34, 2 (May 2015), 667-681.

[11] Henning Zimmer, Fabrice Rousselle, Wenzel Jakob, Oliver Wang, David Adler, Wojciech Jarosz, Olga Sorkine-Hornung, and Alexander Sorkine-Hornung. 2015. Path-space Motion Estimation and Decomposition for Robust Animation Filtering. In Proc. of EGSR. 131-142.

[12] Sergei Karmalsky. Global illumination in metro exodus: An artist's point of view. Game developers conference, 2019.

[13] Kehl et al. Coloured signed distance fields for full 3d object reconstruction. British Machine Vision Conference, 2014.

[14] Ward, G. J., Rubinstein, F. M., and Clear, R. D. (1988). A Ray Tracing Solution for Diffuse Interreflection. In Computer Graphics (ACM SIGGRAPH 88 Proceedings), volume 22, pages 85-92.

[15] Rousselle, C. Knaus, and M. Zwicker. 2012. Adaptive Rendering with Non-local Means Filtering. ACM Trans. Graph. 31, 6 (Nov. 2012), 195:1-195:11.

[16] Nima Khademi Kalantari, Steve Bako, and Pradeep Sen. 2015. A Machine Learning Approach for Filtering Monte Carlo Noise. ACM Trans. Graph. 34, 4 (July 2015).

[17] H. Dammertz, D. Sewtz, J. Hanika, and H. P. A. Lensch. 2010. Edge-avoiding Ã-

[18] Trous Wavelet Transform for Fast Global Illumination Filtering. In Proc. of HPG. 67–75. Elmar Eisemann and Fredo Durand. 2004. Flash Photography Enhancement via Intrinsic Relighting. ACM Trans. Graph. 23, 3 (Aug. 2004), 673–678.

[19] Marco Salvi. 2016. An Excursion in Temporal Supersampling. (2 March 2016). https: //developer.nvidia.com/gdc-2016 Talk at GDC.

[20] Janne Kontkanen, Jussi Rasanen, and Alexander Keller. 2004. Irradiance Filtering for Monte Carlo Ray Tracing. In Proc. of MC2QMC. Springer, 259–272.

[21] Pablo Bauszat, Martin Eisemann, Marcus Magnor, and Naveed Ahmed. 2011. Guided Image Filtering for Interactive High-quality Global Illumination. Proc. of EGSR 30, 4 (Jun 2011), 1361–1368.

[22] Cyril Soler, Kartic Subr, Fredo Durand, Nicolas Holzschuch, and Francois Sillion. 2009. Fourier Depth of Field. ACM Trans. Graph. 28, 2 (May 2009), 18:1–18:12

[23] Christoph Schied, Anton Kaplanyan, Anjul Patney, Chris Wyman, Chakravarty Reddy Alla Chaitanya, John Burgess, Shiqiu Liu, Carsten Dachsbacher, Aaron Lefohn, and Marco Salvi. 2017. Spatiotemporal Variance-Guided Filtering: Real-Time Reconstruction for Path Traced Global Illumination. In Proc. of HPG. 12.

[24] . McGuire, M. Mara, D. Nowrouzezahrai, and D. Luebke. 2017. Real-Time Global Illumination using Precomputed Light Field Probes. In Proc. of I3D. 11.