Real-time subsurface scattering with single pass variance-guided adaptive importance sampling

Tiantian Xie*
University of Maryland, Baltimore County

Marc Olano
University of Maryland, Baltimore County

Brian Karis
Epic Games, Inc.

Krzysztof Narkowicz
Epic Games, Inc.

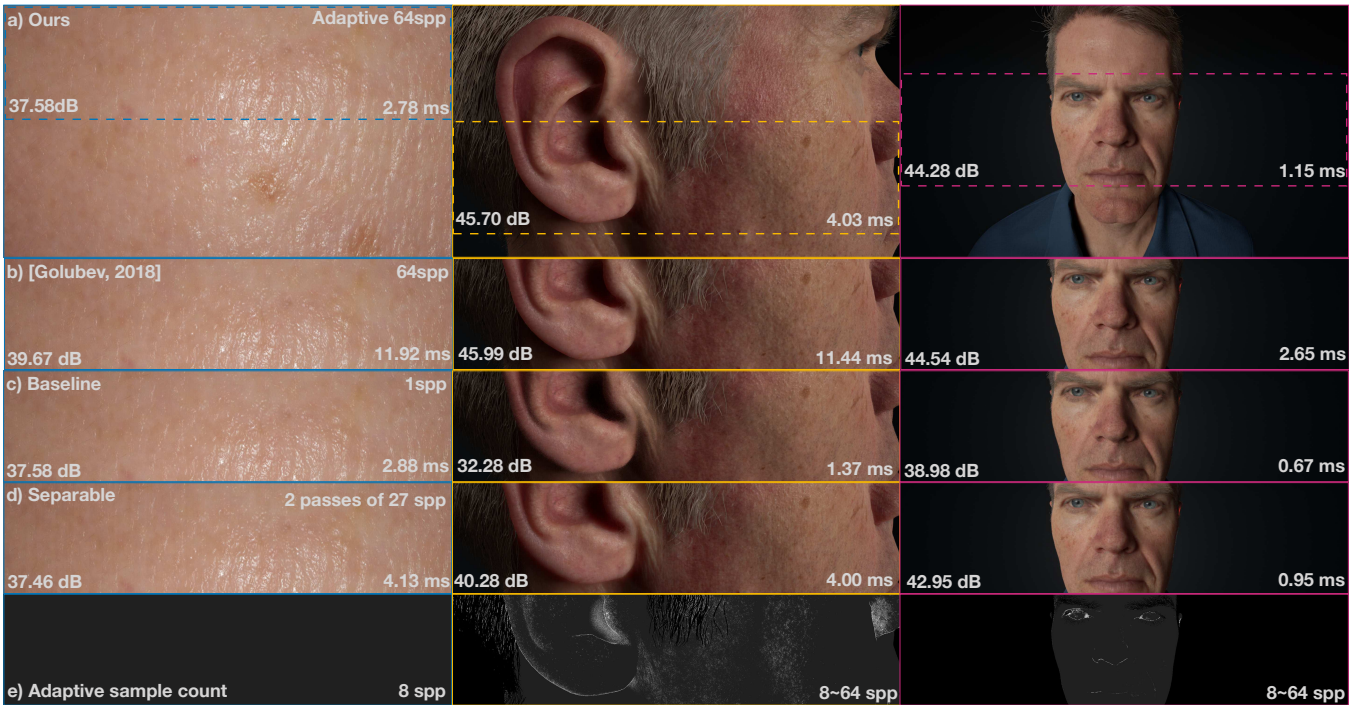


Figure 1: Subsurface rendering comparison from close to far at 1920x1080 on NVIDIA Quadro P4000 (implemented in UE4). (a) our adaptive sampling algorithm ($\sigma_0^2 = 0.001$, $\kappa = 0.2$, $b_{min} = 8$ spp, $b_{max} = 64$ spp), (b) Golubev [2018]'s sampling model in our framework with 64spp, (c) a *Baseline* fixed 64-sample implementation without our proposed acceleration techniques, (d) *Separable* screen-space diffusion. (e) Visualization of our adaptive sample count for each view. Our quality is higher than Baseline in all three scenarios (close skin patch, ear, and front). Moreover, our algorithm runs faster on the close skin patch with an acceleration of up to 91.07× (2.78 ms vs 253.18 ms). In addition, our algorithm enables better quality with run time comparable or even better than Separable. Error measurements are PSNR for the subsurface, as compared to a reference image at 2k samples per pixel. *Digital Mike* ©Epic Games, Inc.

ABSTRACT

In real-time applications, it is difficult to simulate realistic subsurface scattering with differing degrees translucency. Burley's reflectance approximation by empirically fitting the diffusion profile as a whole makes it possible to achieve realistic looking subsurface scattering for different translucent materials in screen space. However, achieving a physically correct result requires real-time Monte Carlo sampling of the analytic importance function per pixel per frame, which seems prohibitive to achieve. In this paper, we propose an approximation of the importance function that can be evaluated in real-time. Since subsurface scattering is more pronounced in certain regions (e.g., with light gradient change), we propose an adaptive sampling method based on temporal variance to lower the required number of samples. We propose a one phase adaptive sampling pass that is unbiased, and able to adapt to scene changes due to motion and lighting. To further improve the quality, we explore temporal reuse with a guiding pass prior to the final temporal

*: The fundamental work is done while as an intern at Epic Games.

Authors' addresses: Tiantian Xie*, University of Maryland, Baltimore County; Marc Olano, University of Maryland, Baltimore County; Brian Karis, Epic Games, Inc. Krzysztof Narkowicz, Epic Games, Inc.

anti-aliasing (TAA) phase that further improves the quality. Our local guiding pass does not constrain the TAA implementation, and only requires one additional texture to be passed between frames. Our proposed variance-guided algorithm has the potential to make stochastic sampling algorithm effective for real-time rendering.

CCS CONCEPTS

• **Theory of computation → Stochastic approximation**; • **Computing methodologies → Reflectance modeling**; *Image processing*; • **Computer systems organization** → *Real-time systems.*

KEYWORDS

Subsurface scattering, Diffuse profile, Real-time rendering, TAA, Variance-guided sampling, adaptive sampling, UE4

ACM Reference Format:
Tiantian Xie*, Marc Olano, Brian Karis, and Krzysztof Narkowicz. 2020. Real-time subsurface scattering with single pass variance-guided adaptive importance sampling. *Proc. ACM Comput. Graph. Interact. Tech.* 3, 1 (May 2020), 12 pages. https://doi.org/10.1145/3384536

1 INTRODUCTION

Subsurface scattering is a demanding feature in real-time applications. Especially in games, it is used to create the appearance of translucent materials (e.g., skin, jade, wax, fruit, and marble) to either mimic real-world materials or create imaginary ones. A full evaluation of subsurface scattering relies on expensive Monte-Carlo (MC) simulation of subsurface scattering events [Pharr and Hanrahan 2000]. The dipole diffusion approximation [Jensen et al. 2001] was the major enabler to make efficient subsurface rendering possible. Methods like texture space [d'Eon and Luebke 2007], pre-integration [Penner and Borshukov 2011] and screenspace subsurface scattering [Jimenez et al. 2009, 2015] attempt to provide faster approximations to the dipole model for real-time applications. More recently, Burley fit a sum of two simple analytic equations directly to Monte-Carlo data [Burley 2015; Christensen and Burley 2015]. His formulation provides a simple importance sampling function, but as a density function, goes to infinity near zero scattering radius, so is problematic for the direct convolution used by other real-time models. A recent excellent work [Golubev 2018] brought the method in screen space with a compromise of pre-calculated sampling and pattern rotation.

In this work, we introduce a real-time *stochastic MC sampling per pixel per frame* in screen space using Burley's approximation, without pre-calculated sampling. The major enabler is our proposed approximated importance function and a novel single pass variance-guided adaptive sampling algorithm (SPVG) which is capable of distributing samples to each frame, with just one temporal reuse texture maintained between frames. The major reason that adaptive sampling is helpful in subsurface scattering is that there is sparsity in the sample count requirement, which has already led to the great pre-integration subsurface scattering method. Nevertheless, there are three challenging problems to resolve in this paper:

1) Generate samples online efficiently. The cumulative density function for sampling with Burley's approximation requires either an iterative solver (e.g., Newtons or Haley's method), a multiple importance sampling of two importance functions, a LUT for each subsurface profile, or an analytic inverse that is more expensive than we desire for real-time rendering. We present an approximation to the inverse function that is simple to evaluate online.

2) Design a one pass adaptive sampling algorithm. A classic adaptive sampling algorithm consists of two phases, which is inefficient for real-time application due to sample disposal and might lead to bias due to low sample counts in the first phase. Usually, in the first phase, samples are gathered to determine the variance and then discarded to avoid bias. In the second phase, an adaptive number of samples is used to reduce the variance. Recent research tries to use all samples by posing the two sample sets as a multiple importance sampling problem [Grittmann et al. 2019]. In this paper, we make it a one adaptive sampling guiding pass for subsurface scattering that makes use of each individual sample. Specifically, sample variances at each frame are continuously updated into a temporal buffer based on exponential moving variance. With temporal reuse, this variance at each frame reflects the pixel condition in the previous adjacent frames and is capable to control the sample counts for the next frame without visible bias for stochastic sampling.

3) Make the algorithm generic. Since we cannot afford too many samples per frame in real-time rendering, we make use of temporal reuse, which has recently been explored for improving the quality of 1 sample per pixel path tracing [Schied et al. 2017]. However, this work makes changes to the existing temporal reuse algorithm, which does not fit well into existing rendering engines. We explore in a different way by adding a pass for guiding the subsurface scattering that assumes there is an existing temporal reuse pass like Temporal Anti-aliasing (TAA), but without assumptions about the implementation of that temporal reuse pass, as long as it tries to make use of temporal information and to resolve artifacts like ghosting, blurring, lag and flickering.

Our contributions include:

- We approximate the Burley importance function to sample subsurface scattering in real-time.
- We present a one pass adaptive sampling method, which is shown to be suitable for accelerating subsurface scattering in real-time. This method adaptively concentrates samples on demanding regions and reduces samples where not required. It also leverages the existing TAA process to improve the quality, yet without constraint on the implementation of that TAA. We get a total acceleration up to 91.07×. For all test scenes, adaptive sampling provides 2× to 6× acceleration when compared to fixed sampling.
- Our variance-guiding pass for a stochastic sampling process (subsurface scattering) only needs one additional texture to enable per-pixel per-frame quality control. Our algorithm is general and can be easily implemented for other algorithms. We believe it is may also have potential for offline adaptive importance sampling.

2 RELATED WORK

Our work focuses on real-time subsurface scattering, adaptive rendering, and temporal reuse. Subsurface scattering for offline rendering is outside the scope of this paper.

Real-time subsurface scattering. Early rendering algorithms for subsurface scattering used Monte Carlo path tracing to solve the radiative transfer equation. Even with the introduction of Bidirectional scattering-surface reflection distribution function (BSSRDF) [Nicodemus et al. 1977] to simplify subsurface integration onto

surface domain, these algorithms took hours to generate an image on hardware of the day. Jensen introduced a BSSRDF light diffusion model with the dipole [Jensen et al. 2001] and multi-dipole [Donner and Jensen 2005] approximations. The simpler dipole method only considers isotropic subsurface scattering in highly scattering media, but introduced the *diffuse reflectance profile*, which allowed compact representation and efficient evaluation.

Two different directions that reduce the complexity for real-time rendering when using diffuse profiles are online sampling and pre-integration. Online sampling is designed with an assumption that subsurface scattering can be approximated by a weighted sum of artist-friendly kernels [Borshukov and Lewis 2005] or dipole based Gaussian kernels [d'Eon et al. 2007]. The sample count is further reduced by shifting from texture space [d'Eon and Luebke 2007] per subsurface object to screen space [Jimenez et al. 2009, 2015] after deferred rendering. In contrast, pre-integration methods offload online sampling based on the dipole approximation into texture look-ups based on the assumptions that subsurface scattering is visible when there is light gradient change [Penner and Borshukov 2011] and the distance to shadow can be correctly reconstructed.

Recent advances in physically-correct subsurface rendering fit an approximation to Monte-Carlo results instead of fitting the dipole model, which is already an approximation [Christensen and Burley 2015]. This technique has been recently incorporated into the Unity game engine with pre-calculated sampling [Golubev 2018]. In this work, we propose an approximation of the importance sampling function that can be used for real-time subsurface scattering.

Adaptive rendering. Adaptive rendering adjusts sampling density and filtering to improve efficiency. Sparse linear models and optimal sampling windows are recursively updated to best reconstruct neighboring pixels [Moon et al. 2015]. An error estimator guides sampling to high variance regions. Adaptive sampling algorithms often require significant memory and computing power to solve an optimization function, e.g., in gradient domain [Manzi et al. 2016] for path tracing. Typically, two passes are required [Moon et al. 2014]. In the first phase, a small number of uniform samples are allocated to gather the error. In the second pass, additional samples are allocated based on error metrics in the first pass. We propose a single pass adaptive sampling model that only needs to store one additional texture across frames and with little calculation overhead. The proposed one pass model is able to pick up dynamic light and motion changes and adjust samples effectively.

Temporal reuse. Real-time rendering demonstrates high spatio-temporal coherence [Scherzer et al. 2012]. This coherence has been used for temporal anti-aliasing [Karis 2014](TAA) and accumulation with shading or geometry information to amortize cost and improve the rendering quality. The major problems dealt with in the literature are blurring, ghosting, lag and flickering [Iglesias-Guitian et al. 2016; Patney et al. 2016; Xiao et al. 2018] due to the use of exponential moving average that blends history with the current frame, and heuristics to tell if they are from the same object.

Temporal information has also been integrated with other systems to amortize sample costs. Recent work replaces the TAA result with ray tracing when ghosting and blurring are detected based on variance and depth [Marrs et al. 2018]. The *spatio-temporal variance guided filter* (SVGF) method includes geometry information and an image-space wavelet filter to achieve one sample per pixel real-time path tracing[Schied et al. 2017]. Schied et al. [2018] adaptively change the exponential weights based on temporal gradients to improve the scene temporal stability (reducing temporal overblurring) caused by fast light changes when using SVGF.

In this work, we guide the stochastic sampling process through temporally-accumulated variance and effective sample count. This allows us to amortize the sample count to meet quality requirements and time budget per pixel. We do this without modifying the existing TAA process or constraining its implementation, providing more stable and lower variance input to the TAA process, while still relying on that TAA to combine samples across frames. Unlike existing work [Iglesias-Guitian et al. 2016] where linear models and inverse covariance matrix are fit through recursive least squares to provide variance information, we use a light-weight algorithm that needs only one additional texture to be passed between frames.

3 BACKGROUND

We briefly review subsurface scattering and temporal anti-aliasing as used in real-time rendering.

3.1 Subsurface scattering

At a surface point p, the outgoing illuminance for direction ω_o is

$$L_o(p,\omega_o) = \int_{\partial\Omega}\int_{S^2} L_i(q,\omega_i)S(q,w_i,p,w_o)d\omega_i dq, (q \in \partial\Omega), \quad (1)$$

where S is the *bidirectional scattering-surface reflectance distribution function* (BSSRDF). Eq. 1 is often simplified by assuming a radially symmetric S in a homogeneous semi-infinite planar medium as

$$S(q,w_i,p,w_o) = CF_t(q,\omega_i)R(r_q)F_t(p,\omega_o), \quad (2)$$

where $r_q = ||p-q||$, $F_t(\cdot)$ is the directional fresnel transmission term and C is a constant term. Then, for a given flux reflection direction ω at p, the simplified subsurface scattering function with symmetric diffusing profile at the surface point p becomes

$$L_o(p,\omega) = \int_{\partial\Omega} 2\pi r_q R(r_q)\cdot \int_{S^2} CF_t(p,\omega)F_t(q,\omega_i)L_i(q,\omega_i)\langle\omega_i,n_q\rangle d\omega_i dr_q. \quad (3)$$

To use this formula, e.g., in deferred rendering, the internal integration can be pre-calculated into irradiance textures for all lights. The outer one can be implemented as a post-processing pass. In this paper, we focus on the efficient sampling of the outer integration.

Burley's profile approximation. In the literature, most work decomposes S into single scattering, multi-scattering, and/or radiance reduction terms as $S = S^{(0)} + S^{(1)} + S_d$ [Habel et al. 2013] and handles them separately.

Instead, Christensen and Burley [2015] directly approximate the diffuse reflectance profile $R(\cdot)$ based on empirical MC simulation data including all scattering terms. $R(\cdot)$ can be well approximated by a sum of two exponential functions in terms of distance r as

$$R(r) = A\frac{e^{-r/d} + e^{-r/(3d)}}{8\pi dr}, \quad (4)$$

where A is the surface albedo with $A = \int_0^\infty R(r)2\pi r dr$ [Jensen and Buhler 2005]. The d term is fit to a free path length parameter ℓ

based on the configuration Θ,

$$d = \begin{cases} \ell \,/\, (1.85 - A + 7|A - 0.8|^3) & \Theta = \Theta_1 \\ \ell \,/\, (1.9 - A + 3.5(A - 0.8)^2) & \Theta = \Theta_2 \\ \ell \,/\, (3.5 + 100(A - 0.33)^4) & \Theta = \Theta_3 \end{cases} \quad (5)$$

where we have:

(1) *Search light configuration* ($\Theta = \Theta_1$). Light enters the volume perpendicular to the surface. ℓ is the volume mean free path.
(2) *Diffuse surface transmission* ($\Theta = \Theta_2$). ℓ is the volume mean free path for a rough material after ideal diffuse transmission.
(3) *Diffuse mean free path as parameter* ($\Theta = \Theta_3$). ℓ is the diffuse mean free path on the surface.

We choose Θ_3 as being more artist friendly for typical game uses.

3.2 Temporal Anti-aliasing

Temporal anti-aliasing (TAA) [Karis 2014] is the de facto standard for anti-aliasing in real-time rendering engines. It amortizes sampling over time with an exponential moving average to accumulate consecutive frames in color space per jittered pixel. When an object moves, TAA reprojects samples from the accumulated history buffer along a per-pixel velocity vector. We summarize it as

$$\mu_i = (1-\alpha)\mathcal{C}(x_i, \Lambda) + \alpha\mathcal{S}(p_i);\ \alpha = \mathcal{M}(\alpha_0, \Lambda); p_i = \mathcal{N}(x_i, f(i)), \quad (6)$$

where μ_i is the estimated value at pixel $x_i \in \mathbb{R}^2$ in frame i. α is the exponential weight between the clamped history context term $\mathcal{C}$ and current-frame shading term $\mathcal{S}$. α is the per-pixel exponential moving average weight as computed by the weight update function $\mathcal{M}$ based on the user defined max weight α_0 and a context Λ that includes velocity, geometry type (e..g, transparent or not), neighbors etc. $\mathcal{C}$ is an operator to resample and reject the history projected from x_i. Both $\mathcal{C}$ and $\mathcal{M}$ are designed to minimize artifacts (i.e., ghosting, blurring, lag, and flickering) while preserving anti-aliasing results [Karis 2014; Yang et al. 2009].

$\mathcal{S}$ is the shading function at p_i, where p_i is a jittered pixel position computed by the neighbor sampling function $\mathcal{N}$ from pixel position x_i and filter-kernel importance sampling offset f.

When used to accumulate MC results across multiple frames, the clamping and rejection performed by $\mathcal{C}$ and $\mathcal{M}$ need to either be modified to explicitly account for the MC process, or the incoming variance in $\mathcal{S}$ must be reduced to fit their rejection model. In modern game engines, TAA is used to accumulate samples from many MC processes, including glossy reflection, ambient occlusion, shadowing, and subsurface scattering. Methods that modify the clamping and rejection models for one MC method used in isolation are not effective given the multiple types of accumulation being performed by TAA. In this paper, we do not modify the existing TAA process. Instead, we create one local variance-guided phase starting from Eq. 6 that outputs an adaptive sampling count per pixel per frame. The sampling result goes to the standard post-process pipeline and uses the existing TAA to further improve quality.

4 VARIANCE-GUIDED MC SAMPLING

There are two problems to solve to efficiently use MC sampling in real-time: **I)** The *minimal number of samples* to reach a target quality level, and **II)** Efficient online *sample generation.*

4.1 Minimal sample count estimation

To estimate the minimal sample count, we need a metric to adaptively increase the number of samples when there is high perceivable noise and decrease fewer samples are sufficient.

4.1.1 Basic Metrics. We estimate the minimal number of samples,

$$n_{(i)} = max\left(\sigma^2_{M_{(i-1)}} \cdot n_{(i-1)} \Big/ \sigma_0^2,\ \beta_{(i-1)}\right), \quad (7)$$

where $n_{(i-1)}$ is the minimal sample count estimated in the previous frame, $\sigma^2_{M_{(i-1)}}$ is the pixel variance of the distribution mean estimated in the previous frame. The purpose of this formula is to reduce the variance of the distribution mean to a target level σ_0^2. Due to lighting and the chance that we might miss some details if we have too few samples, we always use at least $\beta_{(i-1)}$ samples.

We do not know the exact distribution per pixel, due to motion or lighting conditions, but by the central limit theorem, given a population of a finite mean and variance, the sampling distribution of the mean μ_M becomes a *normal distribution* of $(\mu, \sigma^2/N)$ as sample size increases, regardless of the shape of the original distribution. Therefore, we can estimate the variance in means as the distribution is sampled repeatedly across frames.

Using this algorithm to determine the minimal number of sample count in two phases would still be inefficient (i.e., in the first pass, collect samples to gather the variance, and estimate the sample count for rendering in the second pass). We adopt the idea of temporal accumulation from TAA to reduce the number of samples per frame and make it a one pass adaptive sampling technique.

4.1.2 Metrics within Temporal Accumulation. To reduce sample count in each frame, we perform variance and sample count estimation in a local phase. With Eq. 6, we calculate an exponential moving average (EMA) over jittered pixel values. When α is small enough, it will eventually converge to $\mu_M = \mu$, the population mean [Karis 2014]. Instead of recording all three channels, we record the moving average of gamma-corrected luminance to consider human perception. Knowing the sample count $n_{(i)}$ at frame i, we estimate the mean sample count, $\bar{n}_{(i)}$ as

$$\bar{n}_{(i)} = (1-\alpha)\bar{n}_{(i-1)} + \alpha n_{(i)}, \quad (8)$$

and the population variance σ^2 with exponential moving variance (EMV) [Finch 2009] as

$$\sigma_i^2 = (1-\alpha)\sigma_{i-1}^2 + \alpha(1-\alpha)(\mathcal{S}(p_i) - \mathcal{C}(x_i, \Lambda))^2. \quad (9)$$

In this way, we can accumulate sample counts to solve Eq. 7. We now ignore $\beta_{(i-1)}$, it can be set after we get $n_{(i)}$. If a consecutive k frames are accumulated, the equation becomes

$$\sum_{i-k+1}^{i} n_{(j)} = \frac{\sigma^2_{M_{(i-1)}} \sum_{i-k+1}^{i} n_{(j-1)}}{\sigma_0^2}. \quad (10)$$

Then the estimated sample count for the current frame is

$$\hat{n}_{(i)} = \sum_{i-k+1}^{i} n_{(j)} - \sum_{i-k+1}^{i} n_{(j-1)} + n_{(i-k)} \tag{11}$$

$$\approx \frac{(\sigma^2_{M_{(i-1)}} - \sigma_0^2)}{\sigma_0^2} \cdot \bar{n}_{(i-1)} \cdot k + \bar{n}_{(i-1)} \tag{12}$$

$$= \frac{(\sigma^2_{M_{(i-1)}} - \sigma_0^2)}{\sigma_0^2} \cdot \bar{n}_{(i-1)} \cdot (k-1) + E(\bar{n}_{(i)}). \tag{13}$$

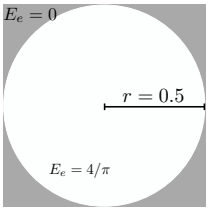
Since we are using EMA and EMV, we cannot maintain $n_{(i-k)}$, we approximate it by $\bar{n}_{(i-1)}$. Then, we estimate k as $k = 2/\alpha - 1$ based a common conversion between N-day EMA and simple moving average in analysis of financial data [Bauer and Dahlquist 1998]. Eq. 13 is the expected sample mean, $E(\bar{n}_{(i)}) = \sigma^2_{M_{(i-1)}}/\sigma_0^2 \cdot \bar{n}_{(i-1)}$ at frame i derived from Eq. 10, plus a correction term. When $|\sigma^2_{M_{(i-1)}} - \sigma_0^2|$ is large, this correction term will be very large to aggressively reduce the variance to the target level in a single frame. We add a control factor κ to limit the per-frame correction. Then the final formula based on Eq. 13 is

$$\hat{n}_{(i)} = \kappa \cdot \Delta(i) + E(\bar{n}_{(i)}), \kappa \in [0, 1] \tag{14}$$

$$\Delta(i) = \frac{(\sigma^2_{M_{(i-1)}} - \sigma_0^2)}{\sigma_0^2} \cdot \bar{n}_{(i-1)} \cdot (k-1), \tag{15}$$

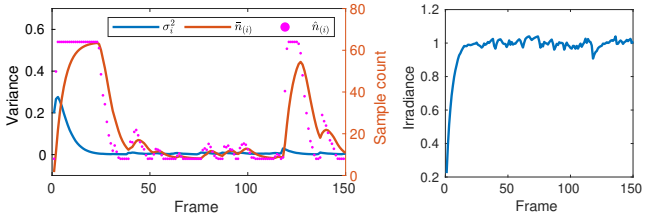
where $\Delta(i)$ is the correction term. At $\kappa = 1$, this favors faster convergence at the cost of firing up to the maximum sample budget per frame, while at $\kappa = 0$ it uses time to accumulate enough samples.

4.1.3 Circle scenario. We create a simplified subsurface scattering scenario to help understand this estimation. We simplify the subsurface scattering scenario model to sample uniform irradiance in a circle of radius 0.5 centered at (.5, .5). We sample with a uniform 2D random number $(\xi_1, \xi_2) \in [0, 1)^2$ (instead of using Burley's subsurface scattering model). The accumulated irradiance is $4/\pi$ within the circle and 0 outside. We give the sample budget of $b_{min} = 8$ spp and $b_{max} = 64$ spp, use $\alpha = 0.2$ and a target quality level $\sigma_0^2 = 0.08^2$. The history tuple $\mathcal{H}_i = (\bar{n}_{(i)}, \mu_i, \sigma_i^2)$ is initialized to **0**. Fig. 2 shows the sample count, variance and the scattering result over 150 frames for $\kappa = 0$ and $\kappa = 1$. After a *cold start* session, both methods try to converge to 1 within the given quality level. $\hat{n}_{(i)}$ in Fig. 2a and 2c shows how κ affects the actual samples used in each frame. The $\kappa = 1$ case has higher sample count peaks, usually lasting for a single frame, while $\kappa = 0$ uses fewer samples, but with sample count increases smoothed over several frames.

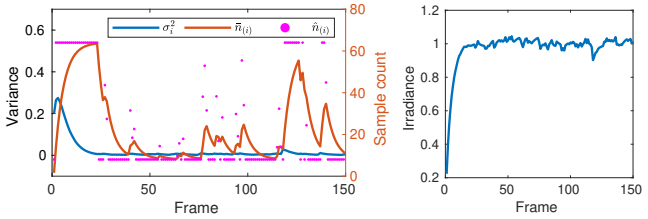


4.1.4 Disocclusion. Fig. 2 shows the sample count per frame for a *cold start*, which is rare in typical rendering scenarios. The more common case for missing history data is disocclusion, when a previously hidden portion of an object becomes visible. We estimate the initial history when disocclusion happens.

Denote $C_s(\cdot)$ as a point sampling operator without rejection on subsurface mask history. $C_s(x_i, \Lambda) = 0$ when the subsurface mask

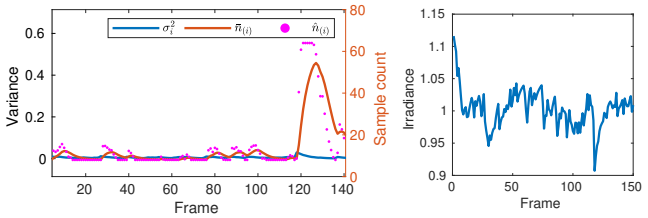


(a) Sample count and variance ($\kappa = 0$). (b) Result ($\kappa = 0$).

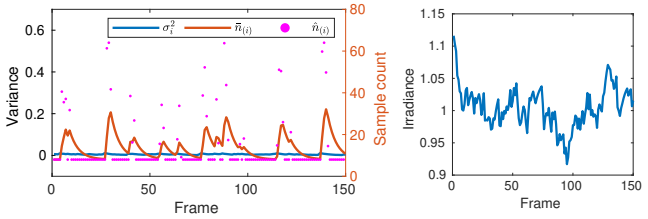


(c) Sample count and variance ($\kappa = 1$). (d) Result ($\kappa = 1$).

Figure 2: Sample count estimation. Effect of using different correction factors ($\kappa = 0$ vs $\kappa = 1$) in Eq. 14 in *Circle Scenario*.



(a) Sample count and variance ($\kappa = 0$). (b) Result ($\kappa = 0$).



(c) Sample count and variance ($\kappa = 1$). (d) Result ($\kappa = 1$).

Figure 3: Sample count estimation considering disocclusion with the same configuration in Fig. 2. We initialize the history $\mathcal{H}_i$ with estimation from the initial sampling to solve the overestimation due to disocclusion (*cold start*).

history is not available in the previous frame. Then our new $\mathcal{M}$ has

$$\mathcal{M}'(\alpha_0, \Lambda) = \begin{cases} 1 & C_s(x_i, \Lambda) = 0 \\ \alpha_0 & \text{otherwise} \end{cases}. \tag{16}$$

This operator enables the estimation of the initial sample value as $\mathcal{S}(p_i)$ with an initial sample count of $n_{(i)} = max(\hat{n}_{(i)}, \beta_{(i-1)})$. We have no variance history, so estimate the variance as

$$\hat{\sigma}_i^2(\alpha_0, \Lambda) = \begin{cases} \sigma_0^2 & C_s(x_i, \Lambda) = 0 \\ \sigma_i^2 & \text{otherwise} \end{cases}. \tag{17}$$

Fig. 3 shows how the updated weight function $\mathcal{M}$ and variance estimation work under the same configuration as in the *Circle Scenario*. The initial high sample counts are eliminated.

4.1.5 Integration with Global TAA. In the previous section, we estimated the sample counts at each frame with the assumption that

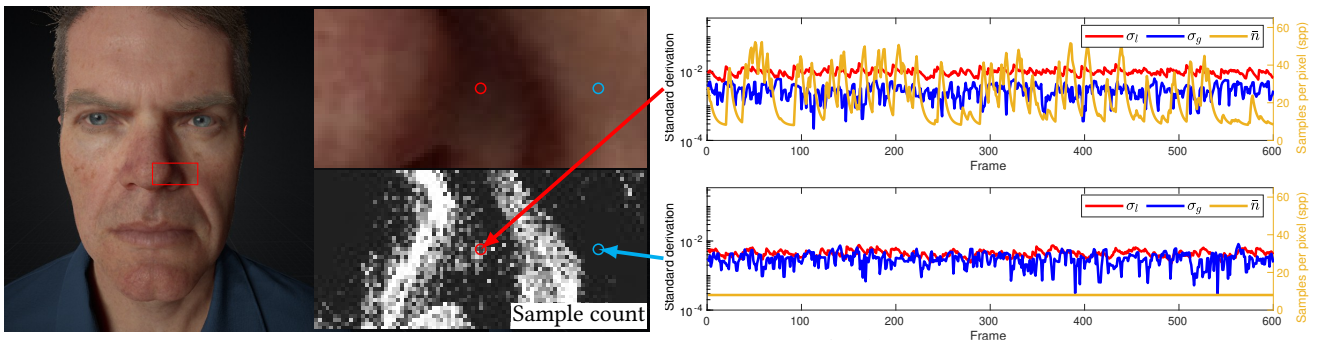


Figure 4: Per-frame local variance guided Screen-space Subsurface Scattering (SSS) sampling to reach a target final rendering quality. The image shows a test scene and two representative points, $P1$ (red) in shadow, and $P2$ (blue) fully lit. The graphs show per-frame local (σ_l) and global (σ_g) pixel standard derivation (*SD*) when $\bar{n}$ is estimated based on Eq. 8. Local target quality level is $\sigma_0^2 = 0.01^2$, with EMA weight $\alpha_l = 0.2$, and control factor $\kappa = 1$. The sample budget is $[b_{min}, b_{max}] = [8, 64]$ spp. The mean of local and global *SD* is $\bar{\sigma}_l = 0.0097$ and $\bar{\sigma}_g = 0.0029$ for $P1$, $\bar{\sigma}_l = 0.0044$ and $\bar{\sigma}_g = 0.0031$ for $P2$.

local and global TAA use the same configuration (i.e., $C_l = C_g, \mathcal{S}_l = \mathcal{S}_g, \Lambda_l = \Lambda_g, \mathcal{M}_l = \mathcal{M}_g$ and $\mathcal{M}_l(\alpha_0, \Lambda_l) = \alpha_0$) so that the variance guided sampling guides the global TAA quality. However, this is not guaranteed in a real-time rendering engine. The TAA weight and functions are primarily tuned for artifacts outside of subsurface scattering. In addition, the same subsurface scattering pixel could have other contributions like overlaid transparent objects.

To make it a general technique for subsurface scattering without assumptions on the global TAA, we choose to *target a lower bound on subsurface quality* to improve the overall TAA quality. Denote $\gamma = (C, \mathcal{M}, \mathcal{N})$ as a solution in the space Γ, $\gamma \in \Gamma$. The variance for a given context Λ_i at frame i of a pixel has $\sigma_{i,\gamma}^2 = \varsigma^2(\gamma, \Lambda_i)$. We chose $\gamma = \gamma_0$ such that for any given sequence of length N_b:

$$\gamma_0 = \underset{\gamma_0 \neq \gamma_j, \gamma_0, \gamma_j \in \Gamma}{\operatorname{argmax}} \sum_{i=1}^{N_b} \mathbb{1}_{\sigma_{i,\gamma_0}^2} \sigma_{i,\gamma_j}^2 \tag{18}$$

where the indicator function has $\mathbb{1}_A(B) = 1$ if $A <= B$, otherwise $\mathbb{1}_A(B) = 0$. Eq. 18 selects the solution that consistently produces lowest variance every frame. To achieve this, we choose C to be a nearest neighbor sampler without history rejection, $\mathcal{M}$ to be a max operator with $\mathcal{M}(\alpha_0, \Lambda) = \alpha_0$ and $\mathcal{N}$ as the default sample position when TAA is used (e.g., positions following a Gaussian distribution). In practice, it is sufficient to be the lower bound.

To illustrate the effect of this selection, we show a sequence of per frame local variance and global variance in Fig. 4 at a shadow pixel and a direct lighting pixel. We fix α_l to 0.2 (this is the maximal weight in UE4 where we implemented our algorithm) and $\sigma_0^2 = 0.01^2$. In this example, the mean global pixel standard derivation is smaller. Moreover, in high variance pixel $P1$, the average global target quality is 11.19 times better than the actual local quality. $P2$ has lower variance than the target quality, at the minimal allowed sample count of $\beta_{i-1} = 8$ spp. It results in a variance 2× better.

When the minimal sample count is determined, we need an efficient sampling method to meet the rendering requirement.

4.2 Efficient sample generation

We simplify Eq. 3 for Monte Carlo sampling at p to

$$L_o = \frac{1}{n} \sum_{j=1}^{n} \frac{2\pi r_{q_j} R(r_{q_j}) \cdot L_{q_j}}{pdf_{q_j}}. \tag{19}$$

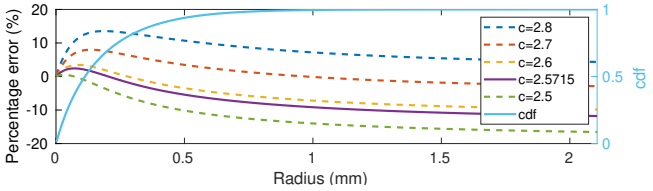
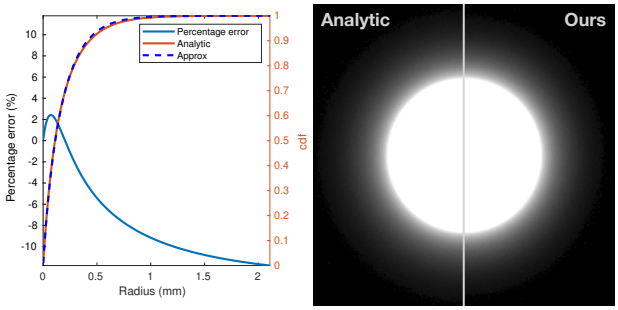


Figure 5: Approximation error for $cdf^{-1}(\xi)$. *Left axis)* Percentage error of $g(\xi)$ compared to $cdf^{-1}(\xi)$ for different c. $c = 2.5715$ has the minimal cost. *Right axis / light blue*) The CDF in Eq. 20. d derives with $A = 0.9$ and $\ell = 1.0$.

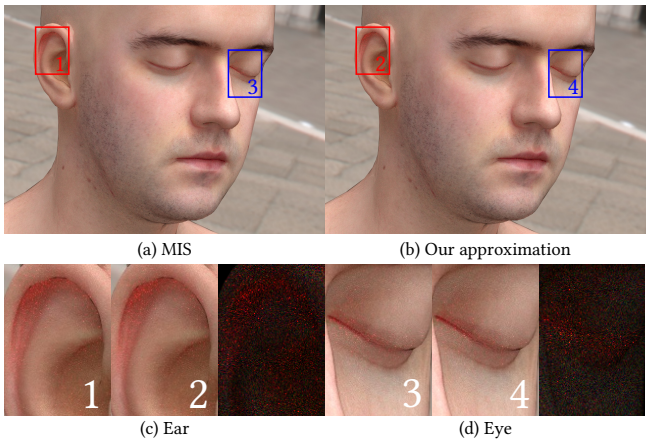


(a) The CDF **(b) 2D profile**

Figure 6: Analytic inverse vs. our approximation. Graphs of the analytic (orange) and approximate (dashed blue) are shown, along with the percentage error. The image shows that the errors do not produce significant visual differences between the analytic and our approximation for the configuration $c = 2.5715$ in Fig. 5 for the CDF, and 2D profile rendered in UE4.

where L_o is the scattering result, r_{q_j} and pdf_{q_j} are the radius to center p and PDF of the jth sample. L_{q_j} is the accumulated diffuse irradiance at q_j. To solve this equation, apart from an efficient 2D sampling sequence (ξ_1, ξ_2) (which is out of the scope of this paper), we need to importance sample the density function based on the CDF for radius sampling:

$$cdf(r) = 1 - \frac{1}{4}e^{-r/d} - \frac{3}{4}e^{-r/(3d)} \tag{20}$$



(a) MIS (b) Our approximation

(c) Ear (d) Eye

Figure 7: Infinite head (©Lee Perry-Smith) rendered with (a) MIS and (b) our approximation from Eq. 21 in PBRT. RMS error is 1.87%. We show a close comparison of high scattering region (c) ear and (d) eye with their pixel differences.

Christensen [2015] suggests to use multiple importance sampling (MIS) of the two exponents, Newton iterations, or a look up table. Golubev [2019] derived an analytic inverse solution. Instead, we propose to use a simpler function $g(\xi)$ that approximates $cdf^{-1}(\xi)$ as:

$$g(\xi) = d((2-c)\xi - 2)log(1-\xi); \tag{21}$$

where c is a parameter to control the damping of the function. We find that when $c = 2.5715$, we have the minimal mean squared error of $cdf(g(\xi|c))$ vs. ξ as optimized using least squares nonlinear curve-fitting [Coleman and Li 1996] at the tolerance of 10^{-6}. Fig. 5 shows the CDF function and the approximation error.

Since this is an approximation to the inverse CDF, samples generated with it are only a (close) approximation to the Burley PDF. To evaluate its quality, Fig. 6 illustrates how close the analytic inverse and our approximation are. In the beam light scenario (Fig.6b), the circle light (10 flux) radius is 1 cm. Zooming into the figure, a tiny shrink in scattering distance can be observed, but is negligible at normal viewing distances. Further more, we implemented our approximation into PBRT [Pharr et al. 2016] and compared it with the built-in MIS method. Fig. 7 shows the rendering comparison within PBRT using the *head* scene. The *dmfp* parameter is derived from the *skin1* configuration in Jensen et al. [2001]. Our approximation has a small RMS error of 1.87% compared to multi-importance sampling for $2k$ spp.

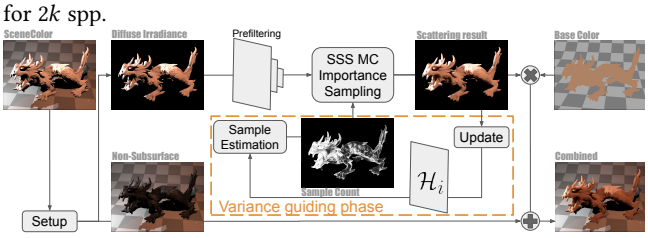


Figure 8: Subsurface pass overview. The variance guiding phase enables one pass adaptive sampling. In each frame, this pass 1) estimates the number of samples per pixel per frame and 2) updates the history $\mathcal{H}_i$.

5 IMPLEMENTATION

We implemented our subsurface scattering as a single screen-space adaptive sampling post-processing pass in UE4, without modifying the final TAA pass. Fig. 8 summarizes the subsurface scattering pass. The scene color is broken into diffuse irradiance and non-subsurface irradiance. Then we prefilter the diffuse irradiance map to accelerate cache hits when incoherently sampling the irradiance. During screen-space subsurface scattering, we use our importance approximation Eq. 21 to sample in the subsurface plane that is perpendicular to our view direction. The number of samples is estimated in the variance guiding phase. Since we could have different adjacent subsurfaces, we cache an 8-bit profile ID texture to resolve how bleeding color between profiles is mixed. After sampling, the result is used to update the history texture that stores $\mathcal{H}_i$ per pixel with $\alpha_0 = 0.2$ (the max weight for the non-transparent object in UE4). Finally, the scattering result is combined with surface albedo and the non-subsurface part to form the final output. In order to make it real-time and have a realistic rendering result, two existing techniques are tailored to our needs.

Pre-filtering. Prefiltering has been successfully used in environmental map importance sampling [Křivánek and Colbert 2008]. We use it for screen-space scattering irradiance sampling. Specifically, we generate the mip level with

$$m = \frac{1}{2} \cdot max\left(-log_2\left(\frac{a \cdot pdf \cdot \hat{n}_{(i)}}{\ell^2_{max} \cdot t}\right), 0\right) \tag{22}$$

$$t = \frac{w}{D}\frac{w \cdot AspectRatio}{D} = \left(\frac{w}{D}\right)^2 \cdot AspectRatio \tag{23}$$

where ℓ_{max} is the max diffuse mean free path of the three-channel subsurface profile. a is a constant factor to scale the mip level. t is the texel size in screen space considering world unit scale w, D is the scene depth at the center sample. We find $a = \frac{1}{16}$ gives a good balance between quality and performance.

Bilateral filtering. We adopt depth-based bilateral filtering [Golubev 2018] to solve the bleeding problem between distinct scattering surfaces. We extended Eq. 19 as:

$$L_o = \frac{\sum_{j=1}^{n_{(i)}} \mathbb{1}_s(q_j) \cdot r'_{q_j} R(r'_{q_j})/pdf_{q_j} \cdot L_{q_j}}{\sum_{j=1}^{n_{(i)}} \mathbb{1}_s(q_j) \cdot r'_{q_j} R(r'_{q_j})/pdf_{q_j}} \tag{24}$$

where $r'_{q_j} = \sqrt{r^2_{q_j} + \Delta D^2_{q_j}}$, ΔD_{q_j} is the depth difference between q_j and the center sampling point, and the indicator function $\mathbb{1}_s(q_j)$ is 1 if there is subsurface and 0 otherwise at q_j.

6 RESULTS

We include comparisons to evaluate the quality and speed of our adaptive sampling algorithm. For quality, we compare root mean square error (RMSE) and gray-scale peak signal-to-noise ratio (PSNR). For real-time performance, we compare speed and quality to Burley's method without the adaptive sampling, and to the separable screen-space method that is the standard implementation in UE4. We modified it to approximate Burley's model instead of Dipole. Please refer to Appendix A.1 for more information about our approach and validation of this approximation. Unless otherwise specified, all performance numbers are measured on NVIDIA Quadro P4000 with a resolution of 1366x1024. We measure time in *ms* for just the subsurface work.

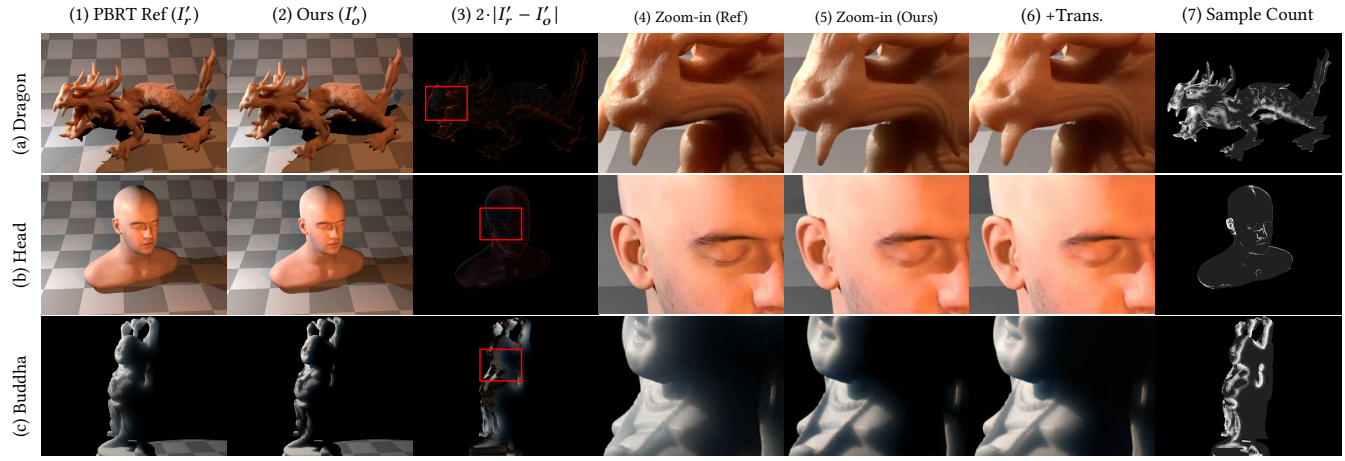


Figure 9: Subsurface ground truth comparison (without transmission). (a) Stanford Asian Dragon, (b) Infinite-Realities head, and (c) Stanford Happy Buddha at 1366×1024. In each row, we show (1) PBRT reference, (2) our scattering, and (3) difference from PBRT. We also zoom into a high difference region for (4) PBRT and (5) ours (6) with transmission. The (7) sample count from our one pass adaptive sampling (white = 64 spp, black = 0 spp). For our algorithm, $\kappa = 0.2$, $\sigma_0^2 = 0.001$, $[b_{min}, b_{max}] = [8, 64]$ spp.

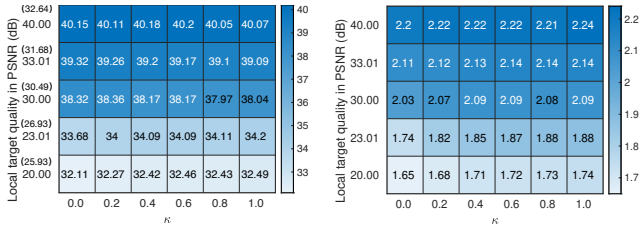
6.1 Quality comparisons

We compare against PBRT ground truth in Fig. 9 for three scenes: Dragon, Infinite-Realities head, and Happy Buddha. We compare our screen-space subsurface scattering (with no transmission) to the PBRT *path* integrator with *Disney* material with *maxdepth* = 1. To focus on subsurface and minimize the difference caused by different light and tone mapping implementations, only point lights are used, with maximum shadow resolution in the UE4 rendering. Tone mapping is not applied in either renderer for ground truth comparison. The results show that, while some differences are visible, they remain qualitatively low. We hypothesize the most significant differences are due to the PBRT renderings including transmission paths, while diffusion models cannot. UE4 does include a separate translucent object transmission model. To confirm this as the major source of observed errors, we replaced the UE4 transmission profile with Burley's model as shown in Fig. 9(6). This produces a closer (though still not exact) match. We believe a better real-time transmission model could further reduce the difference.

6.2 Adaptive sampling quality

For real-time timing evaluation, we compare our adaptive algorithm to a fixed sample-count interactive implementation of the Burley model, and the UE4's separable screen-space diffusion model, tuning each for approximately equal quality as compared to a 2k sample per pixel ground truth within UE4. Performance for the sampling models is worst at a close viewing distance since the texture accesses are least coherent then, with cache misses causing significant performance degradation. Therefore, we evaluate the quality and performance of adaptive sampling at both a normal viewing distance, and at a view close to the surface.

Regular distance. We compare a fixed 2k-sample rendering of the Buddha model to our adaptive algorithm, with a max sample count of 64 and varying σ_0^2 and κ. For timing, both the Buddha and checkerboard base plane use the subsurface material, though the PSNR is only calculated for the Buddha pixels. Fig. 10 shows the



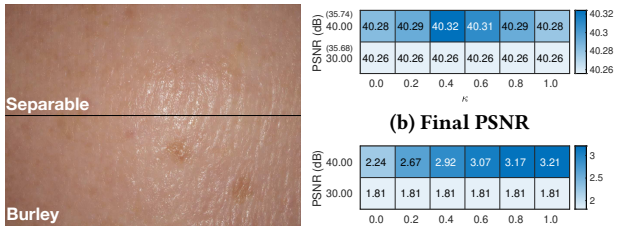
(a) PSNR for Buddha region **(b) Pass time (ms)**

Figure 10: Varying PSNR and κ for the Buddha scene with $b_{max} = 64$ spp. Color shows pass time in ms. PSNR numbers in parentheses are single-frame without TAA. Fixed 64 spp runs in 4.38ms with final PSNR= 40.51(32.91) dB.

PSNR of the Buddha and the subsurface time. Our algorithm runs faster (1.97×) with negligible quality difference (40.51-40.2=.31 dB) when local target quality is $\sigma_0^2 = 0.0001$ (PSNR=40.00 dB). Moreover, our algorithm can make use of TAA to boost the performance with small quality degradation. For examples, when the local target quality is $\sigma_0^2 = 0.001$ (PSNR = $30.00dB$), the final quality reaches 38.36 dB with a speedup by 2.12× in 2.07 *ms*. The last column of Fig. 9(c) shows the sample count of the Buddha in this configuration.

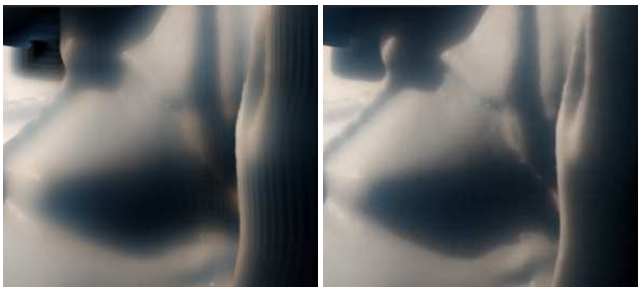
Close distance. We use a forehead skin patch from Digital Mike model that has high scattering distance in UV space to investigate this case. Fig. 11 shows the final PSNR and subsurface time. Especially, when local target quality is $\sigma_0^2 = 0.001$ (PSNR = 30 dB), our algorithm adaptively reduced sample count to $b_{min} = 8$ spp for all κ. We almost achieved equal quality when compared to fixed 64spp (40.26 dB vs. 40.39 dB) with a speedup of 4.15× (from 7.51 ms to 1.81 ms). We expect better performance for adaptive sampling in these cases vs. fixed sampling, since adaptive sampling reduces the number of incoherent non-cached texture accesses.

With the sampling bandwidth bottleneck of close views, the 8-bit cached profile ID texture is critical for performance. Though computed each frame, profile ID texture generation is coherent,



(a) Close skin patch (b) Final PSNR (c) Pass time (ms)

Figure 11: Varying PSNR and κ for the close skin patch with $b_{max} = 64$ spp. Color shows pass time in ms. PSNR numbers in parentheses are single-frame without TAA. Fixed 64 spp runs in 7.51 ms with final PSNR= 40.39(36.02) dB.



(a) Separable (b) Ours

Figure 12: Quality comparison of high scattering marble material. Observable vertical banding artifacts in (a) Separable.

and it reduces the incoherent accesses from 16 bytes per sample to one byte per sample. Without the texture, quality level 0.001 took 4.23ms (6.13×) while fixed 64 spp took 25.94 ms. When compared to Separable, the rendering time of this case is faster than our model, at 3.06 ms, but the separable approximation shows significant banding artifacts in close views, resulting in a worse PSNR = 39.95 dB. We show a better comparison of the artifacts to ours in Fig. 12.

6.3 Equal quality comparison

Fig. 13 compares a single frame Buddha scene quality between our adaptive sampling algorithm and fixed sampling, given approximately the same execution time. In the Buddha scene, we fixed $b_{min} = 8, b_{max} = 64$ with two different σ_0^2, κ settings that lead to the best quality ($\sigma_0^2 = 0.001, \kappa = 0.2$, and $\sigma_0^2 = 0.0001, \kappa = 0.6$). The quality compared to 2k-spp ground-truth is better using our adaptive sampling algorithm.

Fig. 1(a) and (c)/left compares quality on the Digital Mike model for equal time comparison between our adaptive sampling algorithm and the fixed sampling algorithm without pre-filtering and profile caching as a baseline. We also show two other views with the same settings (middle and right). Fig. 1(d) shows the separable screen-space filtering algorithm for the same views. Fig. 1(e) visualizes the sample counts for our adaptive algorithm in each view. The result shows that our algorithm targets best quality with a single setting that runs comparable or even better than Separable.

6.4 Real-time counterpart comparison

We also implemented the state-of-art screenspace subsurface scattering Golubev [2018]'s sampling into our framework as a fixed 64-sample ground truth with the Burley diffuse profile instead of

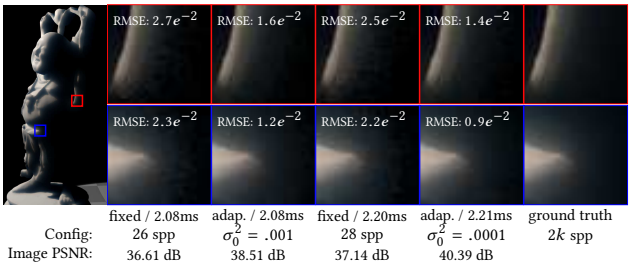


Figure 13: Equal time comparison: fixed vs. adaptive.

our approximation. The result is shown in Fig. 1(b). Based on the approximation comparison in Fig. 6, the bias should be small. Differences between Burley's model and our approximation are small in the ear (0.29 dB) and front (0.26 dB) scenarios. However, those differences are relatively large in the skin patch scenario (2.09 dB). Further inspection of the adaptive sampling count in Fig. 1(e), shows that only 8 samples are used per pixel to meet the target quality. Although there is a small quality degradation, our sampling algorithm is a good mechanism to prevent oversampling when the quality is already met for real-time rendering. Despite the minor quality drop, we have a significant performance gain ranging from 2.3× to 4.3×.

Table 1: Adaptive vs fixed phase breakdown for Digital Mike in Fig. 1 (*ms*).

Scenario	Setup	Pre-filtering	Sampling	Update	Combine	Total
(L)+fixed	0.38	0.16	10.73	N/A	0.20	11.47
(L)+adt.	0.38	0.16	1.50	0.54	0.20	2.78
(C)+fixed	0.41	0.17	9.72	N/A	0.27	10.22
(C)+adt.	0.41	0.17	2.72	0.46	0.27	4.03
(R)+fixed	0.35	0.17	1.45	N/A	0.11	2.08
(R)+adt.	0.35	0.17	0.38	0.14	0.11	1.15

6.5 Performance breakdown

To help understand the cost of the variance guiding phase, the update pass in Fig.8 is separated from the sample estimation and sampling process for time measurement. We also compared fixed and adaptive 64 spp cost to illustrate how techniques utilizing our method might perform on different hardware. The result is presented in Table 1. The performance is measured as the median time (in ms) for the left (L) close patch, center (C) ear, and right (R) front image in Fig.1.

6.6 Discussion and Limitations

Our variance-guided algorithm produces the greatest speedups when sampling is most incoherent, since the bandwidth reduction matters most in those cases. Thus the speedup is best for distant sampling (close views or large mean free path), or sparse sampling.

We also might overestimate the pixel variance, leading to more samples than needed. Because we haven't distinguished the variance due to TAA jittering from variance due to insufficient sampling. For subsurface, this might be good as everything will be diffused.

Our final quality is bounded by the local target quality. Once met, no more samples will be added. Nevertheless, our algorithm is an approximation. It might have created biased sample counts per frame even though the bias is negligible in terms of the rendering result. When $\kappa = 1$, the sampling contribution cannot always be

correctly reflected in the final rendering. Because the weight of the final TAA limits the contribution of each frame.

In the paper, we choose a minimum number of 8 spp. This is because, if b_{min} is set to 1 spp, the estimated sample count series might oscillate or flicker due to noise from the low sample count and TAA history rejection. $b_{min} = 4$ spp was observed to also reduce these flickering effects, and could be a good choice for particularly time-critical use. However, we chose $b_{min} = 8$ spp to as sufficient to remove all TAA flickering artifacts, while still benefiting from adaptive sampling.

7 CONCLUSION

We present a single pass adaptive algorithm SPVG that guides the MC sampling of subsurface scattering to enable real-time performance even with incoherent resource access in uv space. We also present an importance sampling approximation for subsurface scattering that is suitable for online sample generation. We believe our algorithm could also have the potential to be used for offline rendering to guide sample distribution.

ACKNOWLEDGMENTS

This project is supported by a gift from Epic Games. I would like to express my very great appreciation to my advisor, and coworkers while working as an intern at Epic Games. I am also particularly thankful to all reviewers for providing insightful feedback and suggestions to improve the quality of this paper. Special thanks to Mike Seymour and Epic Games for the Digital Mike model, and Stanford University Computer Graphics Laboratory for Asian Dragon and Happy Buddha model.

A APPENDIX

A.1 Separable approximation to the Burley model

To make sure that the separable filter implementation is valid and can be compared to our method, we provide our fitting from Gaussian kernel parameterization [Jimenez et al. 2015] to Burley based on non-linear least squares fitting, and also a comparison between these different techniques. Having a separable fit to the Burley model eases rendering upgrades where the separable model is already deployed, but also provides insight into how existing Gaussian kernel parameters interact with Burley's.

Note that this fitting is to make the separable and Burley models appear as similar as possible visually. This process itself is not physically meaningful. The original separable kernel is an approximation of the dipole model, which is already an approximation of the actual profile. We are trying to fit it to Burley's profile, which is a direct approximation of the actual profile.

A.1.1 Fitting. The fitting problem is to find the best parameter fitting between the Burley $R_B(r, \theta_B)$ and separable $R_S(r, \theta_S)$ diffuse profile, where $\theta_B = \{A, \ell\}$, and $\theta_S = \{falloff\}$. To achieve this, we explore three error functions:

(1) **Difference**. The direct difference between the Burley and separable curves. The minimization error function is

$$S_\Delta = \sum (R_B(r, \theta_B) - R_S(r, \theta_S))^2 \tag{25}$$

(2) **Ratio**. The ratio between the Burley and separable curves should be close to 1. The minimization error function is

$$S_R = \sum (R_B(r, \theta_B)/R_S(r, \theta_S) - 1)^2 \tag{26}$$

(3) **Derivative**. The first derivative of both curves in terms of r should have highest similarity. The minimization error function is

$$S_D = \sum (R'_B(r, \theta_B) - R'_S(r, \theta_S))^2 \tag{27}$$

We observed that S_Δ is the best error function for non-linear least squares fitting. Fig. 14 shows an example of the result with $falloff = 0.8$ for different error functions. Fig. 15 shows the corresponding mapping between θ_B and θ_S. This figure clearly shows the range of values that can be expressed by the default separable parameterization and Burley's. To have the fitting on the fly in an engine, a LUT or a linear fitting function can be deployed.

A.1.2 Validation. We created a beam light scene where a 1 cm radius circle surface receives 10 flux within. The subsurface profile configuration is based on the fitting shown in Fig. 16. Subsurface color for the separable model [Jimenez et al. 2015] is set to 1. Non-subsurface processes like tone mapping, bloom, eye adaptation, specular, and auto exposure are turned off. The result is shown in Fig. 17. We compared the separable and Burley model in different configurations and with different diffuse mean free paths (ℓ). Due to the low sampling count, the separable and Burley models have different artifacts when ℓ increases. The separable model has banding artifacts while Burley has energy loss due to TAA clamping. We believe better clamping in TAA could further reduce the energy loss.

REFERENCES

Richard J Bauer and Julie R Dahlquist. 1998. *Technical Markets Indicators: Analysis & Performance*. Vol. 64. John Wiley & Sons.

George Borshukov and John P Lewis. 2005. Realistic human face rendering for the matrix reloaded. In *ACM SIGGRAPH 2005 Courses*. ACM, New York, NY, 13.

Brent Burley. 2015. Extending the Disney BRDF to a BSDF with integrated subsurface scattering. In *SIGGRAPH Course: Physically Based Shading in Theory and Practice*. ACM, New York, NY, 19.

Per H Christensen. 2015. An approximate reflectance profile for efficient subsurface scattering. In *ACM SIGGRAPH 2015 Talks*. ACM, New York, NY, 25.

Per H. Christensen and Brent Burley. 2015. *Approximate Reflectance Profiles for Efficient Subsurface Scattering*. Technical Report. Pixar.

Thomas F Coleman and Yuying Li. 1996. An interior trust region approach for nonlinear minimization subject to bounds. *SIAM Journal on optimization* 6, 2 (1996), 418–445.

Eugene d'Eon, David Luebke, and Eric Enderton. 2007. Efficient rendering of human skin. In *Proceedings of the 18th Eurographics conference on Rendering Techniques*. Eurographics Association, Aire-la-Ville, Switzerland, 147–157.

Craig Donner and Henrik Wann Jensen. 2005. Light diffusion in multi-layered translucent materials. *ACM Transactions on Graphics (ToG)* 24, 3 (2005), 1032–1039.

Eugene d'Eon and David Luebke. 2007. Advanced techniques for realistic real-time skin rendering. *GPU Gems* 3, 3 (2007), 293–347.

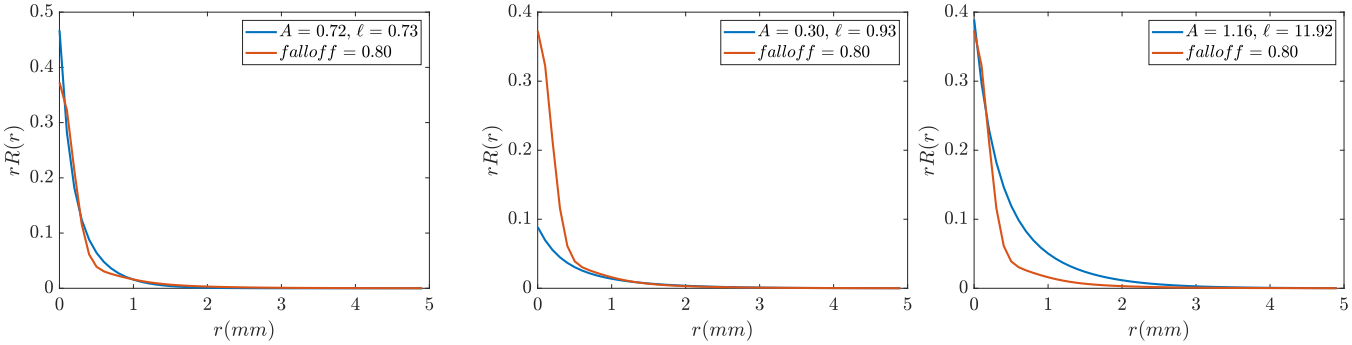
Tony Finch. 2009. Incremental calculation of weighted mean and variance. *University of Cambridge* 4, 11-5 (2009), 41–42.

Evgenii Golubev. 2018. Efficient screen-space subsurface scattering using Burley's normalized diffusion in real-time. Retrieved Aug 29, 2019 from http://advances.realtimerendering.com/s2018/Efficient%20screen%20space%20subsurface%20scattering%20Siggraph%202018.pdf

Evgenii Golubev. 2019. Sampling Burley's Normalized Diffusion Profiles. Retrieved Nov 17, 2019 from https://zero-radiance.github.io/post/sampling-diffusion/

Pascal Grittmann, Iliyan Georgiev, Philipp Slusallek, and Jaroslav Křivánek. 2019. Variance-Aware Multiple Importance Sampling. *ACM Trans. Graph. (SIGGRAPH Asia 2019)* 38, 6 (2019), 9. https://doi.org/10.1145/3355089.3356515

Ralf Habel, Per H Christensen, and Wojciech Jarosz. 2013. Photon beam diffusion: a hybrid Monte Carlo method for subsurface scattering. In *Proceedings of the Eurographics Symposium on Rendering*. Eurographics Association, Aire-la-Ville, Switzerland, 27–37.



(a) By Difference (S_Δ). **(b) By Ratio (S_R).** **(c) By Derivative (S_D).**

Figure 14: Diffuse profile fitting result with three different methods.

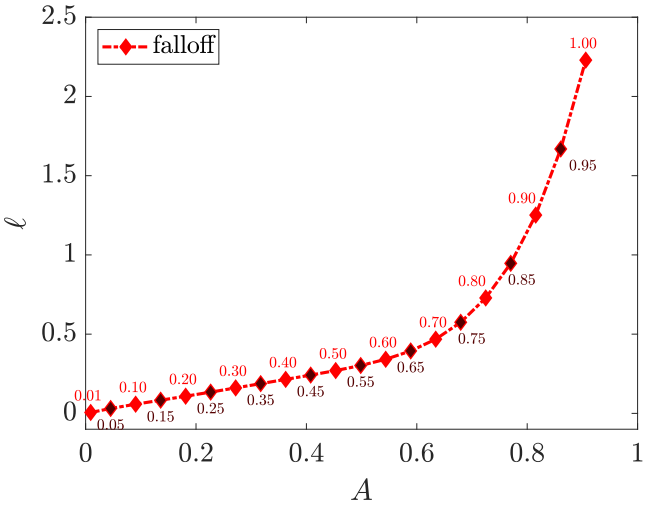


Figure 15: The fitting from *falloff* color to Albedo (A) and Dmfp (ℓ) with S_Δ.

Jose A Iglesias-Guitian, Bochang Moon, Charalampos Koniaris, Eric Smolikowski, and Kenny Mitchell. 2016. Pixel History Linear Models for Real-Time Temporal Filtering. *Computer Graphics Forum* 35, 7 (2016), 363–372.

Henrik Wann Jensen and Juan Buhler. 2005. A rapid hierarchical rendering technique for translucent materials. In *ACM SIGGRAPH 2005 Courses*. ACM, New York, NY, 12.

Henrik Wann Jensen, Stephen R Marschner, Marc Levoy, and Pat Hanrahan. 2001. A practical model for subsurface light transport. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. ACM, New York, NY, 511–518.

Jorge Jimenez, Veronica Sundstedt, and Diego Gutierrez. 2009. Screen-space perceptual rendering of human skin. *ACM Transactions on Applied Perception (TAP)* 6, 4 (2009), 23.

Jorge Jimenez, Károly Zsolnai, Adrian Jarabo, Christian Freude, Thomas Auzinger, Xian-Chun Wu, Javier von der Pahlen, Michael Wimmer, and Diego Gutierrez. 2015. Separable Subsurface Scattering. *Computer Graphics Forum* 34, 6 (Sept. 2015), 188–197.

Brian Karis. 2014. High Quality Temporal Supersampling. Retrieved Aug 29, 2019 from http://advances.realtimerendering.com/s2014/epic/TemporalAA.pptx

Jaroslav Křivánek and Mark Colbert. 2008. Real-time shading with filtered importance sampling. *Computer Graphics Forum* 27, 4 (2008), 1147–1154.

Marco Manzi, Markus Kettunen, Frédo Durand, Matthias Zwicker, and Jaakko Lehtinen. 2016. Temporal gradient-domain path tracing. *ACM Transactions on Graphics (TOG)* 35, 6 (2016), 246.

Adam Marrs, Josef Spjut, Holger Gruen, Rahul Sathe, and Morgan McGuire. 2018. Adaptive temporal antialiasing. In *Proceedings of the Conference on High-Performance Graphics*. ACM, New York, NY, 1.

Bochang Moon, Nathan Carr, and Sung-Eui Yoon. 2014. Adaptive rendering based on weighted local regression. *ACM Transactions on Graphics (TOG)* 33, 5 (2014), 170.

Bochang Moon, Jose A Iglesias-Guitian, Sung-Eui Yoon, and Kenny Mitchell. 2015. Adaptive rendering with linear predictions. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 121.

Fred Edwin Nicodemus, Joseph C Richmond, Jack J Hsia, Irving W Ginsberg, and Thomas Limperis. 1977. *Geometrical considerations and nomenclature for reflectance*. US Department of Commerce, National Bureau of Standards, Gaithersburg, MD.

Anjul Patney, Marco Salvi, Joohwan Kim, Anton Kaplanyan, Chris Wyman, Nir Benty, David Luebke, and Aaron Lefohn. 2016. Towards foveated rendering for gaze-tracked virtual reality. *ACM Transactions on Graphics (TOG)* 35, 6 (2016), 179.

Eric Penner and George Borshukov. 2011. Pre-integrated skin shading. *Gpu Pro* 2 (2011), 41–55.

Matt Pharr and Pat Hanrahan. 2000. Monte Carlo evaluation of non-linear scattering equations for subsurface reflection. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., New York,NY, 75–84.

Matt Pharr, Wenzel Jakob, and Greg Humphreys. 2016. *Physically based rendering: From theory to implementation*. Morgan Kaufmann, Cambridge, MA.

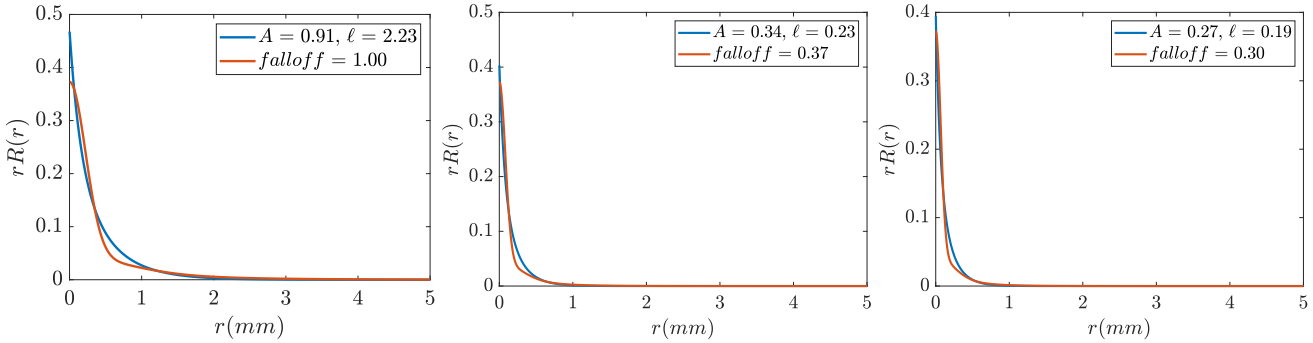
Daniel Scherzer, Lei Yang, Oliver Mattausch, Diego Nehab, Pedro V Sander, Michael Wimmer, and Elmar Eisemann. 2012. Temporal coherence methods in real-time rendering. 31, 8 (2012), 2378–2408.

Christoph Schied, Anton Kaplanyan, Chris Wyman, Anjul Patney, Chakravarty R 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 *Proceedings of High Performance Graphics*. ACM, New York, NY, 2.

Christoph Schied, Christoph Peters, and Carsten Dachsbacher. 2018. Gradient estimation for real-time adaptive temporal filtering. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 1, 2 (2018), 24.

Kai Xiao, Gabor Liktor, and Karthik Vaidyanathan. 2018. Coarse pixel shading with temporal supersampling. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*. ACM, New York, NY, 1.

Lei Yang, Diego Nehab, Pedro V Sander, Pitchaya Sitthi-amorn, Jason Lawrence, and Hugues Hoppe. 2009. Amortized supersampling. *ACM Transactions on Graphics (TOG)* 28, 5 (2009), 135.



(a) Red channel. **(b) Green channel.** **(c) Blue channel.**

Figure 16: Diffuse profile fitting for the default Separable configuration in UE4.

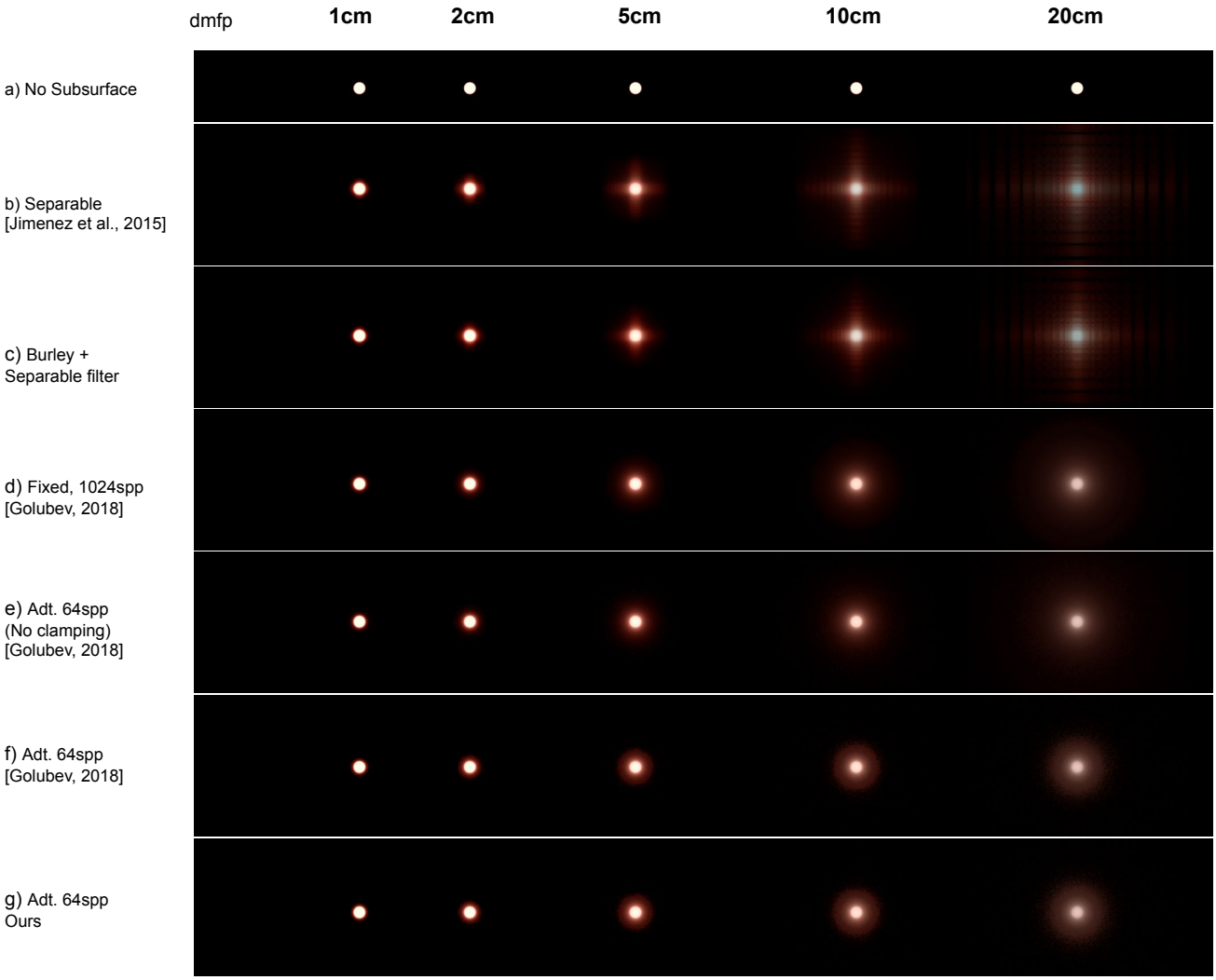


Figure 17: Subsurface test for the beam light scene with a 1 cm circle receiving 10 flux. a) No subsurface. b) Separable [Jimenez et al. 2015]. c) Burley profile fit with a separable filter. d) Burley with fixed 1024 spp. e) Burley with adaptive 64spp + no history clamping. f) Burley with adaptive 64spp. g) Burley with adaptive 64spp using our approximation. Halley's method introduced in [Golubev 2018] is used in d)-f) for sampling. TAA is on for all tests.