ISSN 2464-4617 (print) WSCG 2016 - 24th Conference on Computer Graphics, Visualization and Computer Vision 2016
ISSN 2464-4625 (CD-ROM)

Optimized Skin Rendering for Scanned Models

Roger Hernando
ViRVIG Group - UPC,
roger.hernando@gmail.com

Antoni Chica
ViRVIG Group - UPC,
achica@cs.upc.edu

Pere-Pau Vazquez
ViRVIG Group - UPC,
pere.pau@cs.upc.edu

ABSTRACT

Skin is one of the most difficult materials to reproduce in computer graphics, mainly due to two major factors: First, the complexity of the light interactions happening at the subsurface layers of skin, and second, the high sensitivity of our perceptual system to the artificial imperfections commonly appearing in synthetic skin models. Many current approaches mix physically-based algorithms with image-based improvements to achieve realistic skin rendering in realtime. Unfortunately, those algorithms still suffer from artifacts such as halos or incorrect diffusion. Some of these artifacts (e.g. incorrect diffusion) are especially noticeable if the models have not been previously segmented. In this paper we present some extensions to the Separable Subsurface Scattering (SSSS) framework that reduce those artifacts while still maintaining a high framerate. The result is an improved algorithm that achieves high quality rendering for models directly obtained from scanners, not requiring further processing.

Keywords: Skin Rendering, Subsurface Scattering, Physically-based Rendering.

1 INTRODUCTION

There are several factors that distinguish skin from other materials and put it in a very special category. The first one is the complexity of the skin itself, because the skin is made up of multiple layers (epidermis, dermis and subcutis), which are composed of different types of cellular level elements. Hence, they scatter light according to their own composition [12]. The second factor is a perceptual one. Human perception of skin is very accurate. In any rendered scene where a human-like character with visible skin appears, slight errors in its simulation are spotted easier. Imperfections in color or shading easily make the model to look awkward for our perceptual system. Thus, the accurate believable simulation of the subsurface scattering is very important to make the scene convincing. There have been huge advances the last years in the simulation of skin for synthetic imaging. Nowadays, quite realistic effects can be achieved in realtime using screen-space techniques such as SSSS [15]. Unfortunately, the speed comes at some cost, and such fast techniques still suffer from artifacts that become visible if the user zooms in, or if a real image is compared side by side with a synthetic one. Although the perceptual quality of such renderings has been demonstrated previously, there is still room for improvement. Screen-space subsurface algorithms are prone to spreading the filter outside the skin. Although this may be alleviated by some correction factors, still may present itself in the form of halos outside the silhouette. A second artifact appears when the model is not previously segmented (e.g. when it is applied to scanned models), in the form of incorrect diffusion: the algorithm spreads away the skin region thus blurring other elements such as the eyes or the hair. And third, irregular scattering distribution: screen-space techniques do not properly account for the distance, in object space, of the distribution, making the scattering more noticeable in high curvature regions. In this paper we deal with these limitations and propose approximations that solve them, while still maintaining high framerates. Thus, our contributions improve screen-based subsurface skin algorithms in three ways: i) halo removal, ii) limiting diffusion, and iii) curvature-aware scattering. The rest of the paper is organized as follows: Section 2 will review some previous work as well as outline the framework we work upon, Section 3 will detail the improvements of our system, and finally Section 4 will discuss the results and point some lines for future research.

2 RELATED WORK

Subsurface skin rendering/simulation techniques, according to their temporal cost, can be initially classified into off-line or on-line rendering techniques. Off-line techniques are used, for instance, in movies, or in applications which need to compute accurately and in a photorealistic way skin appearance and do not require interactive manipulation. Such techniques involve the accurate simulation of light rays going through the skin simulating their scattering effects, which is a very demanding process in terms of computational time, especially if solved for a high number of ray bounces. In contrast, on-line techniques are useful for real-time environments such as video games, which need real-time interaction and manipulation. The main challenge

of such techniques is to compute an approximation of the complex subsurface scattering effects, which should be good enough to be perceptually plausible, but at the same time fast enough to allow for real-time rendering. Furthermore, they should be easy to implement so that they integrate well with existing pipelines (e.g. rendering engines).

2.1 Off-line techniques

The simulation of scattering inside translucent materials dates back to the radiative transfer equation [2]. Off-line techniques compute the *BSSRDF* accurately, although the full multiple scattering simulation within a *BSSRDF* might be computationally prohibitive. A *BSSRDF* is an 8D function (Equation 1) that describes the light transport from one point to another for a given illumination and viewing direction. Monte Carlo simulation (ray tracing) is often the tool of choice to solve the light transport problem.

Jensen et al. [14, 13] use the complete *BSSRDF* along with a diffusion approximation to model subsurface scattering. The main idea behind this paper is to decouple the incident illumination from the evaluation of the *BSSRDF* by using a two-pass approach. In the first pass they compute the irradiance at selected points on the surface, and in the second pass the diffusion approximation is calculated using the dipole diffusion approximation from pre-computed irradiance samples. The dipole diffusion approximation assumes that the material is homogeneous and semi-infinite, which is not the case of the human skin. This approach is substantially faster than directly sampling the *BSSRDF* since it only evaluates the incident illumination once at a given surface location. Later, Donner and Jensen [8] extended the dipole model into a multipole one, which allows the modeling of multi-layered translucent materials, such as skin. They present a multipole diffusion approximation for light scattering in thin slabs, which generalizes to an arbitrary number of layers. This way, it enables the composition of arbitrary multi-layered materials with different optical parameters for each layer (i.e. roughness and refraction indices). This method is both accurate and efficient, and can be easily integrated into ray-tracing simulation methods using the dipole diffusion approximation to compute the scattering effects.

In a further work, Donner and Jensen introduce a photon diffusion technique to combine photon tracing and the diffusion approximation [9]. This combination enables an efficient render of highly scattering translucent materials while accounting for internal blockers, complex geometry, translucent inter-scattering, and transmission and refraction of light at the boundary causing internal caustics. Instead of sampling lighting at the surface as the previous techniques, this technique performs a photon tracing step to distribute photons in the material and store them volumetrically at the first scattering interaction with the material. Then, the radiant emittance at points on the material surface is computed by hierarchically integrating the diffusion of the light from photons.

More recently, D'Eon and Irving [6] presented a new *BSSRDF* for rendering images of translucent materials. Previous diffusion *BSSRDF*s are limited by the accuracy of classical diffusion theory. However, they introduce a modified diffusion theory which is more accurate for highly absorbing materials near the point of illumination. This new diffusion solution separates single and multiple scattering terms. Moreover, the authors derive an extended-source solution to the multi-layer searchlight problem by quantizing the diffusion Green's function obtaining a quantized-diffusion (QD) model. This can be done because the contribution from many depth sources at once arises from the separability of Gaussian functions. This allows the application of the QD multipole model to material layers several orders of magnitude thinner than previously possible and creates accurate results under high-frequency illumination.

Finally, Habel, Christensen and Jarosz [10] introduce the photon beam diffusion method. Their approach interprets incident light as a continuous beam of photons inside the material. They leverage the improved diffusion model [6], but propose an efficient and numerically stable Monte Carlo integration scheme that gives equivalent results using only 3-5 samples instead of 20-60 Gaussians. This method can account for finite and multi-layer materials, and additionally supports directional incident effects at surfaces. Besides, their numerical approach allows to extend the accuracy and capabilities of the diffusion model and even combine it efficiently with more general Monte Carlo rendering algorithms.

Unfortunately, those methods are not suited for real-time because they require more than a few milliseconds to be computed, limiting the framerate. Moreover, such methods are intended to be used with Monte Carlo rendering algorithms (e.g. path tracing, photon mapping) [20], which definitely are not able to produce high quality noiseless results in real time.

2.2 On-line techniques

On-line techniques are mainly based on, or try to improve, the subsurface scattering by Borshukov and Lewis [1], which approximates subsurface scattering by blurring a 2D diffuse irradiance texture using a gaussian filter. While it is efficient and maps well to the GPU, it neglects the more subtle details of subsurface scattering.

The previous idea is extended by D'Eon and Luebke [7] to develop a high-quality real-time skin shader. The key element is to approximate the multipole diffusion profiles of thin homogeneous slabs [8] of a multi-layered translucent material such as human skin, as a

linear combination of carefully chosen gaussian basis functions, in order to use them to blur the irradiance signal in texture space. Since the gaussian convolution is separable, this allows transforming the expensive 2D convolutions into a cheaper set of 1D convolutions. This representation greatly accelerates the computation of multi-layer profiles and enables improved algorithms for texture-space diffusion and global scattering. In order to compute the light transmitted through thin parts of the object, the technique by Dachsbacher and Stamminger is used [5].

Although the previously mentioned techniques are based on blurring the irradiance signal in texture space providing real-time performance, they scale poorly with the number of translucent objects in the scene, since subsurface-scattering simulation needs to be performed on a per-object basis. To overcome this issue, Jimenez and Gutierrez [15] proposed to translate the simulation from texture to screen space. Diffuse irradiance of all objects is blurred once as a preprocessing step employing sum-of-Gaussians, thereby limiting subsurface scattering computations to visible parts of the objects. Although the algorithm is faster due to the fact that it works on screen space, the algorithm has less information to work with, as opposed to algorithms that work in 3D or texture space. Therefore, the screen-space algorithm loses irradiance in all points of the surface not seen from the camera, since only the visible pixels are rendered. Thus, the method cannot calculate the transmittance of light through thin parts of an object. Moreover, due to this screen space lack of information, the method produces artifacts such as thin halos near the silhouette of the surface. Mikkelsen [18] showed that the surface convolution by a Gaussian function can be weighted with a cross bilateral filter (*CBF*) over an image containing the edges from the observer point of view, thus solving these silhouette errors.

The aforementioned method also fails to simulate light transmitted through high-curvature features because of the lack of lighting information behind objects. For this reason, its authors extended the method to simulate the transmittance of light through skin [16]. They basically propose an approximation to reconstruct the irradiance on the back of an object. This, in turn, is used to approximate the transmittance based on the multipole theory [8]. Such technique requires standard shadow maps as input, which eases its integration with rendering pipelines, also reducing the memory usage compared to previous work techniques which take transmittance into account.

Shah *et al.* [21] propose a method to compute *BSSRDF* using the dipole diffusion model. They employ the dipole diffusion model with a splatting approach to evaluate the integral over the surface area in an image-space framework, in order to compute the illumination due to multiple scattering. The main contribution is to take sample points on the surface visible from the light source, and splat the scattering contribution to all points visible to the viewer within the effective scattering range from each point. Finally, each point on the rendered surface receives the scattering contribution from all points that have an influence on it.

A recent approach by Jimenez *et al.* [17] proposes two real-time models to generate separable approximations of diffuse reflectance profiles to simulate subsurface scattering. It uses just two 1D convolutions, reducing both execution time and memory consumption, while delivering results comparable to techniques with higher cost. To approximate a 2D diffuse reflectance profile by a single separable kernel, the authors relax the requirement of radial symmetry of diffusion models. They also show how by combining importance sampling and jittering strategies (e.g. [11]), a small number of samples per pixel are enough in many cases of practical interest. They use the approach by Jimenez *et al.* to compute the light transmitted through thin parts of the object [16].

Unlike the previous described methods, which are based on gathering the neighboring light in order to simulate the subsurface scattering effects, other authors pre-integrate the effects of scattered light into a texture [19]. They define three regions of the mesh where the subsurface scattering is important to achieve realism: zones with high surface curvature, zones with small surface bumps, and the zones which next to shadow edges. To obtain the scattering that occurs due to the curvature of the surface and the shadow edges, a precomputed subsurface texture is used, and accessed with the surface local curvature and the shadowness level of the region. To take into account the subsurface scattering due the small surface bumps, they propose a strategy of diffuse normals in which they filter the mesh normal map with R/G/B skin profiles. The authors claim that this strategy allows them to achieve non-local effects of subsurface scattering using only locally stored information.

Finally, Chen et al. [3] presented Pre-integrated Deferred Subsurface Scattering (*PDSS*), a technique that adapts pre-integrated skin scattering to screen space, making it suitable for use in a deferred lighting pipeline and increasing its visual quality. Surface curvature is calculated in real time by evaluating the curvature from the gradient of world space normals in the G-Buffer, avoiding curvature calculation artifacts. PDSS has the advantages of being independent of the scene geometry and scaling well in the number of lights and the number of objects. They use the method by Penner and Borshukov [19] to calculate the subsurface scattering, which uses the curvature and a shadowing factor to look up into a pre-baked scattering texture and also the diffused normals. Light transmitted through thin parts of

the object is calculated using the approach by Jimenez *et al.* [16].

3 BACKGROUND

Subsurface scattering is a complex phenomenon which describes how light enters an object, interacts with its different layers, and may exit at various points around the incident point or be transmitted through the object. This effect is described in terms of the *BSSRDF S* which relates the outgoing radiance $L_0(x_0, \overrightarrow{\omega_0})$ at a point x_0 to the radiant flux $\Phi_i(x_i, \overrightarrow{\omega_i})$ at the point x_i from the direction ω_i:

$$dL_0(x_0, \overrightarrow{\omega_0}) = S(x_i, \overrightarrow{\omega_i}; x_0, \overrightarrow{\omega_0}) d\Phi_i(x_i, \overrightarrow{\omega_i}) \quad (1)$$

The subsurface scattering effect can also be described using radially symmetric diffusion profiles. A diffusion profile is a function $R_d(x,y)$ that describes the light reflected around a normally incident pencil beam on the origin of a surface of an infinite half-space. For an homogeneous material, R_d is radially symmetric and can be characterized by a 1D diffusion profile $R_d(r)$, which describes how the light attenuates at each point as a function of the distance $r = ||(x,y)||$ from the incident point. To obtain such diffusion profiles, diffusion theory can be used to reach to a diffusion equation [14]:

$$D\nabla^2\phi(x) = \rho_\alpha(x) - Q_0(x) + 3D\overrightarrow{\nabla} \cdot \overrightarrow{Q}_1(x) \quad (2)$$

where Q_0 is the 0^{th} order source distribution, Q_1 is the 1^{st} order source distribution, D is the diffusion constant, and ρ_α is the absorption coefficient. For an infinite medium this equation has a simple solution, however for a finite media this equation has no analytical solution. Some authors propose techniques to obtain such diffusion profiles numerically [14, 4]. Applying a diffusion profile is simple. Consider a point $P(x,y)$ on the surface. We want to obtain the contribution of all points around P. Part of the light arriving at such adjacent points will penetrate into the object and exit at P, with the specific attenuation given by the diffusion profile $R(r)$, expressed by:

$$M(x,y) = \int\int E(x',y') R_d(r') dx' dy' \quad (3)$$

$M(x,y)$ being the radiant exitance at point P, $E(x,y)$ the irradiance around P, and R_d the diffuse BSSRDF. Equation 3 sums the contribution of each point around P, each of them weighted by the diffusion profile $R(r)$ according to its distance r to P. Therefore, it can be rewritten as a two-dimensional convolution:

$$M(x,y) = E(x,y) * R_d(r) \quad (4)$$

Carrying out the 2D convolution of Equation 4 is costly for real-time applications. However, if $R_d(r)$ can be approximated by a sequence of $2N$ $1D$ separable convolutions, A, represented as:

$$A(r) = \sum_{i=1}^{N} a_i(r) \quad (5)$$

where the approximation A is defined by $1D$ functions a_i. Due to the radial symmetry of R_d the same functions a_i can be employed in both coordinate directions.

3.1 Screen-Space gaussians sum

From Equation 4, D'Eon and Luebke [7] observed that the skin diffusion profile resembles the aspect of a Gaussian, so a sum of Gaussian functions (Table 1) is suitable for approximation, being $R_d(r)$:

$$R_d(r) = \sum_{i=1}^{k} w_i G(v_i, r) \quad (6)$$

Following the previous idea, Jimenez et al. [15] proposed to perform this sum of gaussians approach in screen space instead of texture space. The method requires the diffuse render, the linear depth of the scene, and the stencil buffer to distinguish which zones are skin and which are not. With them, it generates different levels of Gaussian blurring, and adds up all these levels using the weights of Table 1 in order to obtain the subsurface scattering contribution. Finally, it adds up the specular term to obtain the final render. It is worth noting that pixels located far from the camera should have narrower kernel sizes than pixels near the camera, so the width of the kernel should be modified according to the distance to the camera. Besides, a correction component is introduced to prevent scattering through neighboring pixels in screen space but farther away in the geometry.

Variance	Color Weights		
	Red	Green	Blue
0.0064	0.233	0.455	0.69
0.0484	0.1	0.336	0.344
0.187	0.118	0.198	0
0.567	0.113	0.007	0.007
1.99	0.358	0.004	0
7.41	0.078	0	0

Table 1: Sum-of-gaussians parameters for a skin model depicted by D'Eon and Luebke [7].

This way, the technique mimics the results of the method proposed by D'Eon and Luebke [7], at a fraction of its cost both in time and memory usage. What this method can neither reproduce nor match from the previous method is the simulation of transmitted light through the thin slabs of skin. Therefore, this method must be used along with those that simulate forward scattering.

Our technique is based on this approach, but adapting the shaders to handle an arbitrary number of samples.

4 OPTIMIZED SKIN RENDERING

Raw scanned acquired models are 3D photographs of an object, just including color and geometry information, and sometimes a normal map depicting the fine details of the skin (e.g. pores, wrinkles).

In computer games, models are further processed, to identify skin, eyes, and so on. However, in a more general case, such work is not possible, and thus, using the models as is, causes some artifacts or worsen other problems that still appear with the mentioned algorithms. We illustrate some of these problems in Figure 1, namely halos and incorrect diffusion.

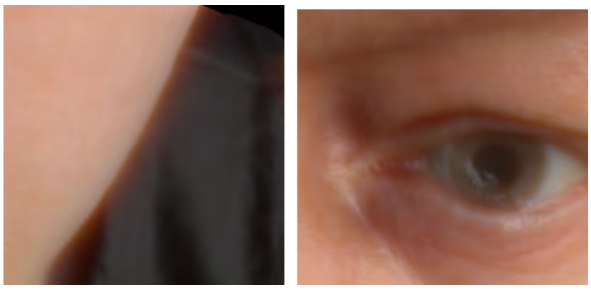

Figure 1: Screen-space subsurface scattering algorithms may produce halos (left) and incorrect diffusion (right) on scanned models.

In this Section we propose some optimizations to address these problems with raw models.

4.1 Halo Removal

The first obvious problem that appears is halos. The screen space approaches produce halos between neighboring zones in image space but at different depth levels. The authors of the aforementioned subsurface scattering methods noticed the halos problems as well, and tried to tackle them with the correction factor (central image of Figure 6). The approach modulates the color of the samples which, although being near the central point of the diffusion profile in image space, are far away in the geometry, using the difference in depth between the central and the sampled points. Unfortunately, the correction factors are not enough and Mikkelsen [18] showed that using a cross bilateral filter (*CBF*) to weight the diffusion profile fixes the halos problem.

A *CBF*, works like a bilateral filter (Equation 9), but uses an auxiliary image to compute the weights instead of the image that is being filtered. *CBF* is characterized by the following equation:

$$CBF[I,E]_p = \frac{\sum_{q\in S} G_{\sigma_s} e^{-||p-q||} G_{\sigma_r} e^{-(E_p-E_q)} I_q}{\sum_{q\in S} G_{\sigma_s} e^{-||p-q||} G_{\sigma_r} e^{-(E_p-E_q)}} \quad (7)$$

where I is the original input image, E is the auxiliary image used to compute the difference of intensities, p are the coordinates of the current pixel to be filtered, S is the window centered in p, and G_{σ_r} and G_{σ_s} are the distance and color weighting factors, respectively.

The auxiliary image E is defined as an image that distinguishes between zones whose normal is perpendicular to the view direction and zones which are not. This creates an image of contours from the point of view of the observer, highlighting the edges between continuous areas in screen space but not in the geometry. This image is defined as follows:

$$E(p) = I(x(p)) * cos^3(\phi_i) \frac{||x(p)||^2}{cos(\phi_j)} \quad (8)$$

where $x(p)$ is the object point which is drawn in pixel p, ϕ_i is the angle between z-axis and the direction from the observer to the point $x(p)$, ϕ_j is the angle between the surface normal and the vector from $x(p)$ to the observer, and $I(x(p))$ the intensity of the pixel p.

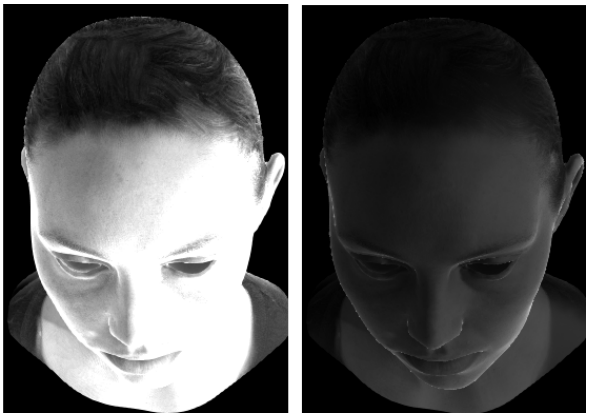

Figure 2: The image shows the unnormalized strategy proposed by Mikkelsen [18] (left) vs. our normalized strategy (right).

Figure 3: The image shows the artifacts (black dots near the edges) introduced by the *CBF* method when used directly with our shaders.

However, the efficacy of this approach as proposed depends on how far the model is from the projection plane, losing the power of detecting edges and therefore not removing the halos, as can be seen in Figure 2. We

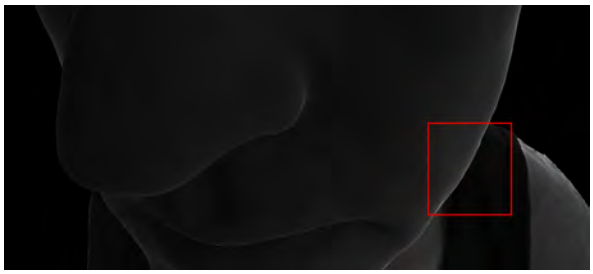

Figure 4: The auxiliary image used in the *CBF* weighting, in order to reduce the halos. The highlighted section corresponds to the region used in Figure 5.

refine this method by modifying the way the auxiliary image is computed. In our case, we make it invariant to the distance by taking the point $x(p)$ as if it was always placed on the projection plane, we call this one the *normalized* image. Moreover, using this cross bilateral weighting directly within our shaders produces some ugly artifacts as can be seen in Figure 3. We fix these issues by modifying the shaders so that if the final color is black, the original diffuse color is used. In contrast to the proposal by Mikkelsen, we use this technique along with the correction factors.

Figure 4 shows the auxiliary image used by the *CBF* with a highlighted area which is the same as used in the Figure 5, which shows the halos effect and its reduction using this method.

Figure 5: Halos comparison: not using any method to correct them (left), using the correction factors (center), and using **our modified *CBF* approach** (right).

4.2 Limiting scattering diffusion

When scanned models are not artist-processed, and thus skin and non-skin areas are not properly segmented, the screen-space subsurface scattering algorithms generate a second artifact: incorrect diffusion. Since the boundaries of the elements are not identified, the method produces blurring over surfaces such as the eyes, as illustrated in Figure 1-right.

We alleviate this problem with a bilateral filter weighted by the color distance of neighboring pixels, which modulates the contribution of each sample. To take into account human perception, the color distance is performed in CIE Lab color space. The bilateral filter we used is defined by the following equation:

$$BF[I]_p = \frac{\sum_{q \in S} G_{\sigma_s} e^{-||p-q||} G_{\sigma_s} e^{-(I_p - I_q)} I_p}{\sum_{q \in S} G_{\sigma_s} e^{-||p-q||} G_{\sigma_s} e^{-(c_p - c_q)}} \quad (9)$$

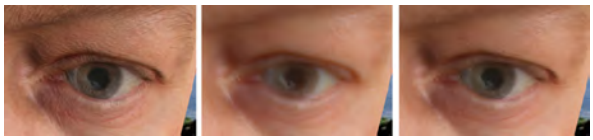

Figure 6: Reducing blur between skin and non-skin zones: without subsurface scattering (left), simple subsurface scattering (centre), and using a bilateral filter to avoid blurring between skin and non-skin zones (right).

where I is the original input image, p are the coordinates of the current pixel to be filtered, S is the window centered in p, I_p and I_q are the colors of the image I at pixel p and q respectively, and G_{σ_r} and G_{σ_s} are the distance and color weighting factors, respectively. As for the *CBF* method, we used G_{σ_r} as the diffusion kernel weight and G_{σ_s} is set to one.

Figure 6 shows the blurring of skin and non-skin zones, and how the bilateral filtering deals with the problem. It is not a perfect solution because it still blurs some high frequency details (i.e. thin hair), but it substantially improves the render quality.

4.3 Scattering modulation

As already stated, scattering is caused by the light entering the surface, bouncing several times, and getting out of it at a different point. If the surface is curved, there will be more light entering and exiting the object. This should be reflected as an increase in the subsurface scattering in higher curvature regions [19]. Unfortunately, the screen-space algorithms presented so far, do not use this information to modulate the amount of scattering. To solve this we modulate the scattering in screen-space so that the effect is stronger at zones with higher curvature and weaker at lower curvature zones.

Like the rest of our method, we are going to compute this in screen-space. To obtain the oriented gradient of a pixel, we use the normals of the neighbors, and obtain the curvature by analyzing the magnitude of the variation of these axes. It is worth noting that, in order not to introduce high frequency discontinuities, the normals used to compute the curvature are the geometry normals and not the normal map normals (Figure 7-left). Besides, the curvature should be smoothed (i.e. mean blurring) to avoid such artifacts.

We have modulated the subsurface scattering effect with the curvature in three different ways (Figure 8):

- Increasing the subsurface scattering strength of a pixel according to its local curvature. However, since screen space curvature is higher at the contours of the geometry, this causes the subsurface scattering effect to be stronger on the edges. Therefore, increasing the filter size at the contours and making the halo artifacts more noticeable.
- Reducing the subsurface scattering effect at zones with lower curvature and keeping it at zones with

Figure 7: Using the normal map to compute the screen space curvature results in a noisier curvature (left). Geometry normals reduce the noise (center). We smooth this map to feed the algorithm with a smoother curvature map (right) free of high frequency discontinuities.

higher curvature. This strategy caused the zones with nearly zero curvature not to simulate the subsurface scattering at all, and breaking the high quality skin rendering.

- Reducing the subsurface scattering effect at zones with lower curvature up to a minimum and leaving it unchanged at zones with higher curvature. This proved to be a good strategy because the subsurface scattering is simulated and strengthens the effect at the high curvature zones.

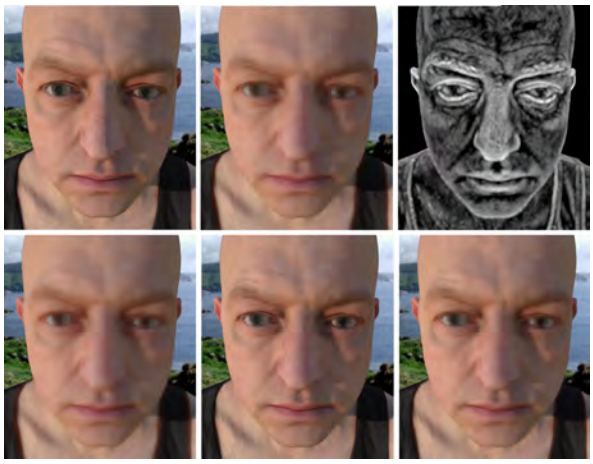

Figure 8: Top row shows a face without and with subsurface scattering. Top right shows the blurred screen space curvature used to modulate the scattering strength. The bottom row shows the three attempted strategies: increasing the subsurface scattering strength according to its local curvature (bottom left), reducing the subsurface scattering effect at zones with lower curvature (middle botom), and reducing the subsurface scattering effect at zones with lower curvature up to a minimum (bottom right).

We have also tried to modulate the forward scattering strength according to the mesh local curvature, which proved to be a bad idea since it produces ugly artifacts (i.e. extremely bright translucency areas) at high curvature zones as shown in Figure 9, where the high curvature areas suffer from artifacts.

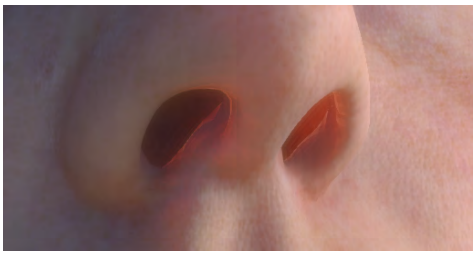

Figure 9: When the forward scattering is modulated with the screen space curvature, it produces bright artifacts at high curvature areas (e.g. nostrils).

5 RESULTS AND CONCLUSIONS

Our implementation also implements other features such as forward scattering to simulate light sources illuminating from back of the object, or gamma correction. The pipeline of our application is shown in Figure 10. In the first step, we get a shadowmap from the light source position. Then, a rendering stage generates the information required for the subsurface scattering: a diffuse map, the stencil buffer, a depth map with linear depth, a specular map, and a curvature map. Then, the rendering stage is the one that generates the subsurface scattering visualization. Finally, a simple step combines the previous result with the specular lighting, and a final tone mapping step generates the final result. Although we work upon the screen-space subsurface scattering work by Jimenez *et al.* [17], our optimizations can also be used on other screen-space methods. The algorithm runs in realtime. The most costly part is the Gaussian sum, which amounts to less than 10 ms for a close view of the face, as shown in Table 2. The remaining steps (shadow map, main render, specular, and tone mapping) add a total of less than 3 ms to the subsurface algorithm.

View	Gausian sum	Artistic	Pre-int Kernel
Close	9.428 ms	1.731 ms	1.799 ms
Mid	2.01 ms	0.492 ms	0.487 ms
Far	0.676 ms	0.312ms	0.36 ms

Table 2: Elapsed time of each subsurface scattering simulation algorithm, at different distances.

To sum up, we have presented a number of optimizations that improve the quality of the screen-space subsurface scattering algorithm: *i)* a technique to avoid halos spreading on different depth regions, *ii)* a method to reduce the scattering diffusion, and *iii)* an improvement tailored to increase the scattering in high curvature regions. The first and third improvements can be applied to any kind of models, while the second is especially suitable for general scanned models that have not been segmented to identify skin and other elements. All of these improvements have a low impact on rendering and thus we have realtime framerates. In future we

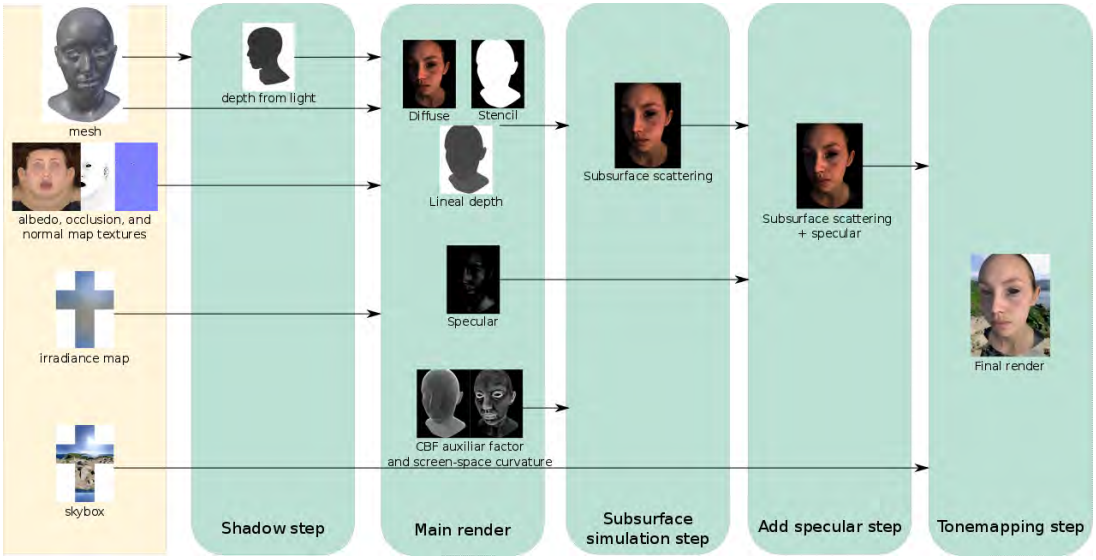

Figure 10: Pipeline of our application.

want to improve the screen-space subsurface scattering approach to better represent the different layers of skin.

ACKNOWLEDGEMENTS

Supported by project TIN2014-52211-C2-1-R by the Spanish Ministerio de EconomÃa y Competitividad with EU FEDER funds.

REFERENCES

[1] G. Borshukov and J. P. Lewis. Realistic human face rendering for "the matrix reloaded". In *ACM SIGGRAPH 2003 Sketches &Amp; Applications*, SIGGRAPH '03, pages 1–1, New York, NY, USA, 2003. ACM.

[2] S. Chandrasekhar. *Radiative Transfer*. Dover Books on Intermediate and Advanced Mathematics. Dover Publications, 1960.

[3] X.M. Chen, T. Lambert, and E. Penner. Pre-integrated deferred subsurface scattering. In *ACM SIGGRAPH 2014 Posters*, SIGGRAPH '14, pages 98:1–98:1, New York, NY, USA, 2014. ACM.

[4] Craig D. and H.W. Jensen. A spectral bssrdf for shading human skin. In *Rendering Techniques*, EGSR '06, pages 409–417, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[5] C. Dachsbacher and M. Stamminger. Translucent shadow maps. In *Proceedings of the 14th Eurographics Workshop on Rendering*, EGRW '03, pages 197–201, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[6] E. D'Eon and G. Irving. A quantized-diffusion model for rendering translucent materials. *ACM Trans. Graph.*, 30(4):56:1–56:14, July 2011.

[7] E. d'Eon and D. Luebke. Advanced techniques for realistic real-time skin rendering. In Hubert Nguyen, editor, *GPU Gems 3*, pages 293–347. Addison-Wesley, 2008.

[8] C. Donner and H.W. Jensen. Light diffusion in multi-layered translucent materials. *ACM Trans. Graph.*, 24(3):1032–1039, July 2005.

[9] C. Donner and H.W. Jensen. Rendering translucent materials using photon diffusion. In *ACM SIGGRAPH 2008 Classes*, SIGGRAPH '08, pages 4:1–4:9, New York, NY, USA, 2008. ACM.

[10] R. Habel, P.H. Christensen, and W. Jarosz. Photon beam diffusion: A hybrid monte carlo method for subsurface scattering. *Computer Graphics Forum (Proceedings of EGSR)*, 32(4), June 2013.

[11] J. Huang, T. Boubekeur, T. Ritschel, M. Holländer, and E. Eisemann. Separable approximation of ambient occlusion. In *Eurographics 2011 - Short papers*, 2011.

[12] T. Igarashi, K. Nishino, and S. K. Nayar. The appearance of human skin: A survey. *Foundations and Trends® in Computer Graphics and Vision*, 3(1):1–95, 2007.

[13] H.W. Jensen and J. Buhler. A rapid hierarchical rendering technique for translucent materials. *ACM Trans. Graph.*, 21(3):576–581, July 2002.

[14] H.W. Jensen, S.R. Marschner, M. Levoy, and P. Hanrahan. A practical model for subsurface light transport. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pages 511–518, New York, NY, USA, 2001. ACM.

[15] J. Jimenez and D. Gutierrez. *GPU Pro: Advanced Rendering Techniques*, chapter Screen-Space Subsurface Scattering, pages 335–351. AK Peters Ltd., 2010.

[16] J. Jimenez, D. Whelan, V. Sundstedt, and D. Gutierrez. Real-time realistic skin translucency. *IEEE Computer Graphics and Applications*, 30(4):32–41, 2010.

[17] J. Jimenez, K. Zsolnai, A. Jarabo, C. Freude, T. Auzinger, X.C. Wu, J. v.d. Pahlen, M. Wimmer, and D. Gutierrez. Separable subsurface scattering. *Computer Graphics Forum*, pages n/a–n/a, 2015.

[18] M.S. Mikkelsen. Skin rendering by pseudo–separable cross bilateral filtering. *Naughty Dog Inc*, page 1, 2010.

[19] E. Penner and G. Borshukov. *GPU Pro 2: Advanced Rendering Techniques.*, chapter Pre-Integrated Skin Shading, pages 41–55. AK Peters Ltd., 2010.

[20] M. Pharr and G. Humphreys. *Physically Based Rendering, Second Edition: From Theory To Implementation*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2nd edition, 2010.

[21] M. A. Shah, J. Konttinen, and S. Pattanaik. Image-space subsurface scattering for interactive rendering of deformable translucent objects. *IEEE Comput. Graph. Appl.*, 29(1):66–78, January 2009.