Efficient Implementation of the Dual Scattering Model in RenderMan

Iman Sadeghi*
Walt Disney Animation Studios
University of California, San Diego

Rasmus Tamstorf†
Walt Disney Animation Studios

Figure 1: *Some early rendering results based on the dual scattering implementation presented in this report.*

Abstract

Simulating the multiple scattered light is critical for rendering light colored hair. Most of the existing models for simulating the exact behavior of multiple scattered light are computationally expensive and require additional data structures. The Dual Scattering model [Zinke et al. 2008] is a fast and relatively accurate model that approximates the behavior of multiple scattering component. In this report we provide the necessary details needed for implementing the Dual Scattering model efficiently without any ray tracing steps and any extra data structures. We have implemented this model in RenderMan and it has formed the foundation for our work in [Sadeghi et al. 2010].

Keywords: rendering, multiple scattering, dual scattering model

1 Motivation

It is critical to consider multiple scattering of light to ensure the correct perception of hair color, especially for light colored hair [Moon and Marschner 2006; Zinke and Weber 2006]. To capture the exact behavior of the multiple scattered light one needs elaborate methods like brute force path tracing [Zinke and Weber 2007], photon mapping [Moon and Marschner 2006; Zinke and Weber 2006], or other grid based approaches like [Moon et al. 2008]. Path tracing approaches are computationally expensive and their results converge very slowly. Photon mapping and grid based approaches are faster than path tracing methods but are still relatively expensive. Besides, the latter two are two pass methods which need ray tracing capabilities in their first pass and extra data structures to store extra information in the scene for the second pass (e.g. photons or spherical harmonics coefficients). Therefore, integrating these methods into RenderMan is more complicated than a typical surface shader.

Another class of methods try to approximate the multiple scattering component by considering the physical properties of human hair fibers. The most recent method in this category is the Dual Scattering method [Zinke et al. 2008]. It is a fast and good approximation that produces results very similar to path tracing. We decided to use this method as the basis of our multiple scattering component because it does not require extra data structures and with careful considerations one can implement this method without any ray tracing steps.

In Section 2 we explain the details of our implementation. We show how one can implement the global multiple scattering component without any ray tracing steps in Section 2.1. We also explain our implementation of the local multiple scattering component and provide extra clarifications and formula needed for implementation of this component in Section 2.2. In Section 2.3 we present the pseudo code of our implementation and highlight its differences with the original dual scattering method [Zinke et al. 2008]. We show some rendering results of our model in Section 3 and present a summarized version of our RenderMan Shading Language code in Appendix A.

2 Implementation

To implement the Dual Scattering method we follow the instructions of the original paper [Zinke et al. 2008]. However, there are some notes and clarifications which might be useful for anyone who wants to implement this method. In particular, in the original paper, the longitudinal inclination angles θ, θ_h, θ_d, θ_i and θ_o have been used interchangeably. We try to state the subscripts of all θ angles explicitly. Throughout this report we have highlighted all modifications and formula which are not present in the original paper in red. Table 1 summarizes all the terms and their corresponding descriptions that will be used in this report.

The Dual Scattering method approximates the multiple scattering component Ψ as a combination of two components: global multiple scattering Ψ^G and local multiple scattering Ψ^L. The global multiple scattering component approximates the amount of light that reaches the neighborhood of the shading point after it travels through the hair volume. The local multiple scattering approximates the scattering events within the neighborhood of the shading point (See Figure 2). The multiple scattering component is derived according to the following equation :

$$\Psi(x, \omega_d, \omega_i) = \Psi^G(x, \omega_d, \omega_i)(1 + \Psi^L(x, \omega_d, \omega_i)) \quad (1)$$

where ω_d is the direction of the incoming light entering the hair volume and ω_i is the direction of the multiple scattered light reaching the shading point x. This equation states that the multiple scattered

*iman@graphics.ucsd.edu
†rasmus.tamstorf@disney.com

light reaching the eye is either the result of global multiple scattering or its combination with the scattering events around the shading point (i.e. local multiple scattering).

Symbol	Description
Ψ	Multiple scattering function
Ψ^G / Ψ^L	Global/local multiple scattering function
ω_i	Incoming light direction
ω_o	Outgoing view direction
ω_d	Direct lighting direction
θ_i / θ_o	Incoming/outgoing longitudinal angle
θ_h / θ_d	Half/difference angle between θ_i and θ_o
T_f	Forward scattering transmittance
S_f	Forward scattering spread
$\bar{S}_b$	Average backward scattering spread
$\bar{A}_b$	Average backscattering attenuation
$\bar{\Delta}_b$	Average backscattering longitudinal shift
d_f / d_b	Forward/backward scattering density factor
$\bar{a}_f$ / $\bar{a}_b$	Average forward/backward attenuation
$\bar{\alpha}_f$ / $\bar{\alpha}_b^2$	Average forward/backward scattering shift
$\bar{\beta}_f^2$ / $\bar{\beta}_b^2$	Average forward/backward scattering variance
$\bar{\sigma}_f^2$ / $\bar{\sigma}_b^2$	Total variance of forward/backward scattering
f_s^{direct} / $f_s^{scatter}$	Single scattering for direct/indirect lighting
f_{back}^{direct} / $f_{back}^{scatter}$	Backscattering for direct/indirect lighting
M_X	Longitudinal scattering function for component X
N_X	Azimuthal scattering function for component X
α_X	Longitudinal shift for component X
β_X	Longitudinal width for component X

Table 1: *Summary of all terms used in this report.*

2.1 Global Multiple Scattering

Global multiple scattering Ψ^G is especially important for rendering light colored hair since it accounts for the light that penetrates the hair volume. Since the dominant subcomponent of the hair scattering function is the transmission component (TT), Ψ^G is approximated by only the front scattered light. In other words, Ψ^G is the contribution of one or more TT scattering events along the line connecting the shading point to the light source. Therefore, Ψ^G at any point is dependent on the orientations of all the hairs between the light source and that point. It is approximated by the multiplication of the forward scattering transmittance T_f and spread S_f of the light that reaches the shading point from all light sources.

$$\Psi^G(x, \omega_d, \omega_i) \approx T_f(x, \omega_d) S_f(x, \omega_d, \omega_i) \tag{2}$$

Global multiple scattering will be computed separately for different points. One of the easiest way of computing this component is by using a ray shooting method. In this method, for a given shading point, we shoot a ray toward the light and intersect it with any hair that occludes the light source. Then from the orientations of those intersecting hairs we compute the forward scattering transmittance according to :

$$T_f(x, \omega_d) = d_f \prod_{k=1}^{n} \bar{a}_f(\theta_d^k) \tag{3}$$

Here d_f is the forward scattering density factor which accounts for the hair density around the shading point. In the original paper, this

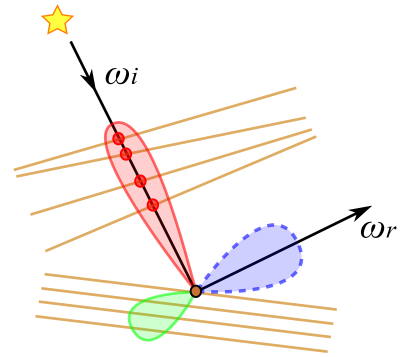

Figure 2: *The dual scattering method separates the multiple scattering component (blue) into global multiple scattering (red) and local multiple scattering (green). The global multiple scattering depends on the orientation of all hair strands between the shading point and the light sources while the local multiple scattering depends only on the orientation of the hair strand at the shading point.*

density factor is set to 0.7. The term θ_d^k represents the longitudinal inclination at the k'th scattering event and $\bar{a}_f(\theta_d)$ is the average attenuation which is computed from the fiber scattering function f_s. The computation of the single scattering components is identical to [Marschner et al. 2003] and is the product of longitudinal scattering functions M, which are modeled as Gaussian functions, with precomputed azimuthal scattering functions N.

We also compute the forward scattering spread as follows

$$S_f(x, \omega_d, \omega_i) = g(\theta_h, \bar{\sigma}_f^2(x, \omega_d))/(\pi \cos\theta_d) \tag{4}$$

Here $\bar{\sigma}_f^2$ is the total variance of forward scattering in the longitudinal directions and is the sum of variances of all scattering events along the shadow path:

$$\bar{\sigma}_f^2(x, \omega_d) = \sum_{k=1}^{n} \bar{\beta}_f^2(\theta_d^k) \tag{5}$$

Ray shooting is very straightforward in the ray tracing context but it is very expensive when using a REYES based renderer like RenderMan. We therefore approximate the global multiple scattering component by assuming that the hairs in front of any point have the same orientation as the shading point. This is a good approximation for long flat hair style which was the main focus for our production. Moreover, in a production environment, the inaccuracies in this approximation can be compensated by the extra artistic controls. At each shading point we estimate the number of hair strands n in front of a shading point by looking up the shadow opacity value stored in the deep shadow maps. We then approximate T_f and $\bar{\sigma}_f^2$ according to the following formula:

$$T_f(x, \omega_d) \approx d_f\, \bar{a}_f(\theta_d)^n \tag{6}$$

$$\bar{\sigma}_f^2(x, \omega_d) \approx \bar{\beta}_f^2(\theta_d) \times n \tag{7}$$

Here θ_d is the longitudinal inclination of the hair strands at the shading point.

These approximations have been acceptable based on our production needs. However, an exact implementation is possible if it is required. The goal is to find $\prod_{k=1}^{n} \bar{a}_f(\theta_d^k)$ and $\sum_{k=1}^{n} \bar{\beta}_f^2(\theta_d^k)$ where n is the number of hairs in front of the shading point and θ_d^k is the longitudinal angle of the hair strand with respect to the light source. Here, $\bar{a}_f$ and $\bar{\beta}_f^2$ are precomputed functions.

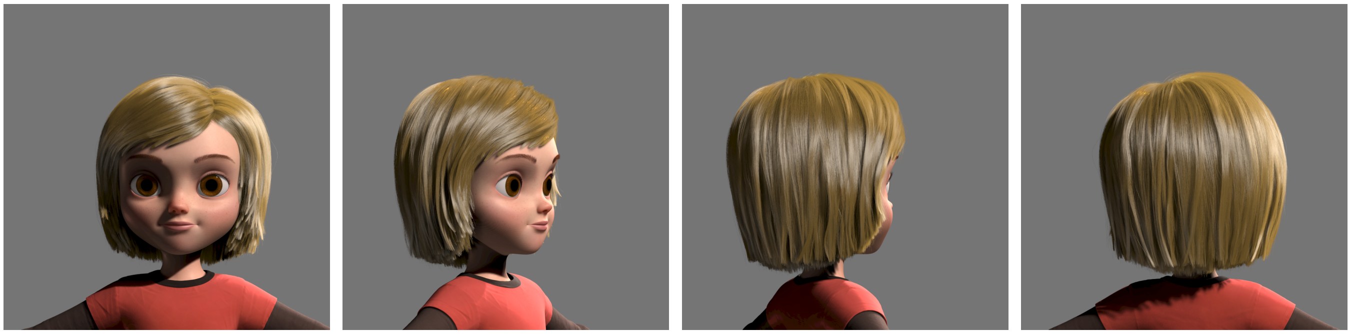

Figure 3: *Applying blond hair to one of our existing characters. This was another early test done without the help of artists.*

The main idea is to use deep shadow maps to calculate the list of all θ_d^k angles. Then we can use these values to compute all the values of $\bar{\alpha}_f(\theta_d^k)$ and $\bar{\beta}_f^2(\theta_d^k)$. For the exact solution, we introduce a virtual opacity function inside the hair surface shader and let it return opacity values κ as a function of its longitudinal orientation (Any one-to-one mapping form $-\pi/2 \le \theta_d \le \pi/2$ to $0 < \kappa < 1$ works). In the first step we compute the deep shadow map of the hair volume using these opacity functions. In the second step we do a post process on the deep texture file generated for the deep shadow maps. We can retrieve the actual opacity values from the stored shadow function inside the deep texture. Also, from those opacity values we can compute the actual orientations of the hair fibers. This will provide us with list of θ_d^k values.

2.2 Local Multiple Scattering

The local multiple scattering function Ψ^L approximates the light scattering events within the neighborhood of the shading point. Light paths contributing to Ψ^L must include at least one backward scattering event since other light paths are already accounted for in Ψ^G. For simplification, we assume that all the hairs surrounding the shading region have the same orientation and that there is an infinite number of them. Therefore Ψ^L is only dependent on the longitudinal inclination of the hair strand at the shading point. This enables us to pre-compute Ψ^L for all the longitudinal inclination angles :

$$\Psi^L(x,\omega_d,\omega_i)f_s(\omega_i,\omega_o) \approx d_b f_{back}(\omega_i,\omega_o) \tag{8}$$

Here d_b is the backward scattering factor and similar to d_f it is set to 0.7. f_{back} is the backscattered light which is the product of an average backscattering attenuation function $\bar{A}_b$ and an average backscattering spread function $\bar{S}_b$:

$$f_{back}(\omega_i,\omega_o) = 2\bar{A}_b(\theta_d)\bar{S}_b(\omega_i,\omega_o)/\cos\theta_d \tag{9}$$

The average backscattering attenuation function $\bar{A}_b$ is given by

$$\bar{A}_b(\theta_d) = \frac{\bar{a}_b\bar{a}_f^2}{1-\bar{a}_f^2} + \frac{\bar{a}_b^3\bar{a}_f^2}{(1-\bar{a}_f^2)^2} \tag{10}$$

where $\bar{a}_f$ and $\bar{a}_b$ are the average forward and backward scattering attenuations. Similar to S_f (Equation 4), $\bar{S}_b$ is computed as

$$\bar{S}_b(x,\omega_d,\omega_i) = g(\theta_h - \bar{\Delta}_b(\theta_d), \bar{\sigma}_b^2(\theta_d))/(\pi\cos\theta_d)$$

where $\bar{\Delta}_b$ is the average longitudinal shift given by:

$$\bar{\Delta}_b \approx \bar{\alpha}_b(1 - \frac{2\bar{a}_b^2}{(1-\bar{a}_f^2)^2}) + \bar{\alpha}_f(\frac{2(1-\bar{a}_f^2)^2 + 4\bar{a}_f^2\bar{a}_b^2}{(1-\bar{a}_f^2)^3}) \tag{11}$$

and $\bar{\sigma}_b$ is the average backscattering standard deviation given by:

$$\bar{\sigma}_b \approx (1 + d_b\,\bar{a}_f^2)\frac{\bar{a}_b\sqrt{2\bar{\beta}_f^2 + \bar{\beta}_b^2} + \bar{a}_b^3\sqrt{2\bar{\beta}_f^2 + \bar{\beta}_b^2}}{\bar{a}_b + \bar{a}_b^3(2\bar{\beta}_f + 3\bar{\beta}_b)} \tag{12}$$

Here $\bar{\alpha}_f(\theta_d)$ and $\bar{\alpha}_b(\theta_d)$ are the average forward and backward scattering shifts respectively. They are basically weighted averages of longitudinal shifts, α_R, α_{TT}, and α_{TRT} for the front and back scattering hemispheres. They can be computed from the following equations:

$$\bar{\alpha}_f(\theta_d) = \frac{\int_{\Omega_f} f_R\alpha_R + f_{TT}\alpha_{TT} + f_{TRT}\alpha_{TRT}(\theta_d)}{\int_{\Omega_f} f_R + f_{TT} + f_{TRT}(\theta_d)} \tag{13}$$

$$\bar{\alpha}_b(\theta_d) = \frac{\int_{\Omega_b} f_R\alpha_R + f_{TT}\alpha_{TT} + f_{TRT}\alpha_{TRT}(\theta_d)}{\int_{\Omega_b} f_R + f_{TT} + f_{TRT}(\theta_d)} \tag{14}$$

where Ω_f and Ω_b represent the forward and backward scattering hemispheres. f_X represents the X portion of the single scattering component f_s for $X \in \{R, TT, TRT\}$.

Similarly, for the average forward and backward scattering variances $\bar{\beta}_f^2(\theta)$ and $\bar{\beta}_b^2(\theta)$ we have:

$$\bar{\beta}_f(\theta_d) = \frac{\int_{\Omega_f} f_R\beta_R + f_{TT}\beta_{TT} + f_{TRT}\beta_{TRT}(\theta_d)}{\int_{\Omega_f} f_R + f_{TT} + f_{TRT}(\theta_d)} \tag{15}$$

$$\bar{\beta}_b(\theta_d) = \frac{\int_{\Omega_b} f_R\beta_R + f_{TT}\beta_{TT} + f_{TRT}\beta_{TRT}(\theta_d)}{\int_{\Omega_b} f_R + f_{TT} + f_{TRT}(\theta_d)} \tag{16}$$

Please note that $\bar{\alpha}_{f/b}$, $\bar{\beta}_{f/b}$, $\bar{a}_{f/b}$, $\bar{\Delta}_b$ and $\bar{\sigma}_b^2$ are color variables and have to be computed for different color channels (in general for different wavelengths). This means that different wavelengths will be shifted and spread differently.

For efficient computation, Zinke et al. computed the shading differently depending on whether or not the shading point receives illumination directly F^{direct} or through other hairs (points in shadow) $F^{scatter}$. For indirect versions they introduced $f_s^{scatter}$ which is a modified version of single scattering component f_s^{direct}.

Similar to the single scattering BRDF, $f_s^{scatter}$ has three longitudinal function called M_X^G and three azimuthal functions N_X^G for $X \in \{R, TT, TRT\}$. These functions are computed directly from the longitudinal and azimuthal functions of the hair BRDF as follows:

$$M_X^G(\theta_h) = M_X(\theta_h - \alpha_X, \bar{\beta}_X^2 + \bar{\sigma}_f^2) \tag{17}$$

$$N_X^G(\theta_d,\phi) = \frac{2}{\pi}\int_{\pi/2}^{\pi} N_X(\theta_d,\phi')d\phi' \tag{18}$$

Note that N_X^G should be the averaged version of N_X over the front scattering hemisphere but the authors of the paper considered a simpler approximation to be the average over the front semi-circle.

The front hemisphere is dominated by the TT component. There are some contributions from the R component as well, but the contributions from the TRT component in the front scattering hemisphere are very small. Therefore, N_{TT}^G is larger than N_R^G and both of them are larger than N_{TRT} which means that the color of the $f_s^{scatter}$ component is being dominated by the color of the TT component.

Similarly, f_{back} should be computed separately for direct and indirect lighting. We have decomposed the f_{back} term form the original paper into two terms f_{back}^{direct} and $f_{back}^{scatter}$. The latter term accounts for the change in variance of the forward scattered light in the longitudinal direction and will be used in the calculation of $F^{scatter}$. See Figure 4 for a visualization of different terms in the dual scattering model.

Figure 4: *Sub-components of the dual scattering method. (a) Dual scattering model. (b) Single scattering component (f_s). (c) Average backscattering attenuation ($\bar{A}_b$). (d) Local multiple scattering for direct light lighting (f_{back}^{direct}). (e) Local multiple scattering for indirected lighting ($f_{back}^{scatter}$). (f) Average backscattering spread ($\bar{S}_b$). (g) Single scattering for indirect lighting ($f_s^{scatter}$) (h) $F^{scatter}$ (i) F^{direct}*

$$f_{back}^{direct}(\omega_i, \omega_o) = \frac{2\bar{A}_b(\theta)g(\theta_h - \bar{\Delta}_b(\theta), \bar{\sigma}_b^2(\theta) + \bar{\sigma}_f^2(x, \omega_d))}{\pi \cos^2 \theta_d} \tag{19}$$

$$f_{back}^{scatter}(\omega_i, \omega_o) = \frac{2\bar{A}_b(\theta)g(\theta_h - \bar{\Delta}_b(\theta), \bar{\sigma}_b^2(\theta))}{\pi \cos^2 \theta_d} \tag{20}$$

2.3 Pseudo Code

The following pseudo code is our revised version of the pseudo code presented in Figure 5 of the original paper [Zinke et al. 2008]. The corresponding RenderMan pseudo code is shown in Appendix A.

// Pre-compute $\bar{A}_b(\theta_d)$, $\bar{\Delta}_b(\theta_d)$ and $\bar{\sigma}_b^2(\theta_d)$ from f_s for $0 < \theta_d < \pi$

// Compute the amount of direct lighting $directFraction$
// Compute the number of hairs in front of the shading point n

$T_f(x, \omega_d) = d_f\ \bar{a}_f(\theta_d)^n$
$\bar{\sigma}_f^2(x, \omega_d) = \bar{\beta}_f^2(\theta_d) \times n$

F($T_f, \bar{\sigma}_f^2, directFraction$)

// Backscattering for direct/indirect lighting
$f_{back}^{direct} \Leftarrow 2\bar{A}_b(\theta_d)g(\theta_h - \bar{\Delta}_b(\theta_d), \bar{\sigma}_b^2(\theta_d))/(\pi \cos^2 \theta_d)$
$f_{back}^{scatter} \Leftarrow 2\bar{A}_b(\theta_d)g(\theta_h - \bar{\Delta}_b(\theta_d), \bar{\sigma}_b^2(\theta_d) + \bar{\sigma}_f^2(\theta_d))/(\pi \cos^2 \theta_d)$

// Longitudinal functions for direct/indirect lighting
$M_X \Leftarrow g(\theta_h - \alpha_X, \bar{\beta}_X^2)$
$M_X^G \Leftarrow g(\theta_h - \alpha_X, \bar{\beta}_X^2 + \bar{\sigma}_f^2)$

// Azimuthal functions for indirect lighting
$N_X^G(\theta_d, \phi) \Leftarrow \frac{2}{\pi} \int_{\pi/2}^{\pi} N_X(\theta_d, \phi') d\phi'$

// Single scattering for direct/indirect lighting
$f_s^{direct} \Leftarrow \sum M_X(\theta_h) N_X(\theta_d, \phi)$
$f_s^{scatter} \Leftarrow \sum M_X^G(\theta_h) N_X^G(\theta_d, \phi)$

$F^{direct} \Leftarrow directFraction(f_s^{direct} + d_b f_{back}^{direct})$
$F^{scatter} \Leftarrow (T_f - directFraction) d_f (f_s^{scatter} + \pi d_b f_{back}^{scatter})$

// Combine the direct and indirect scattering components
return $(F^{direct} + F^{scatter}) \cos \theta_i$

3 Results

Figure 1 shows some rendering results from different viewing angles and lighting directions. Figure 4 illustrates the contribution of different terms involved in the computations of the dual scattering model. Finally, Figure 3 shows some rendered frames of an animation using our dual scattering implementation.

References

MARSCHNER, S. R., JENSEN, H. W., CAMMARANO, M., WORLEY, S., AND HANRAHAN, P. 2003. Light Scattering from Human Hair Fibers. *ACM Transactions on Graphics 22*, 3, 780–791.

MOON, J. T., AND MARSCHNER, S. R. 2006. Simulating Multiple Scattering in Hair Using a Photon Mapping Approach. *ACM Transactions on Graphics 25*, 3, 1067–1074.

MOON, J. T., WALTER, B., AND MARSCHNER, S. 2008. Efficient Multiple Scattering in Hair Using Spherical Harmonics. *ACM Transactions on Graphics 27*, 3, 1–7.

SADEGHI, I., PRITCHETT, H., JENSEN, H. W., AND TAMSTORF, R. 2010. An Artist Friendly Hair Shading System. *To appear in ACM Transactions on Graphics 29*.

ZINKE, A., AND WEBER, A. 2006. Global Illumination for Fiber Based Geometries. In *Electronic proceedings of the Ibero American Symposium on Computer Graphics (SIACG)*.

ZINKE, A., AND WEBER, A. 2007. Light Scattering from Filaments. *IEEE Transactions on Visualization and Computer Graphics 13*, 2, 342–356.

ZINKE, A., YUKSEL, C., WEBER, A., AND KEYSER, J. 2008. Dual Scattering Approximation for Fast Multiple Scattering in Hair. *ACM Transactions on Graphics 27*, 3, 1–10.

A RenderMan Shading Language Code

Here is a summary of the RenderMan Shading Language required for implementing the Dual Scattering model.

```
class dual_scattering(
    // Defining the shader parameters
    float enable = 1;
    /*
        type switch
        name {Enable}
        desc {Enables the hair shader.}
    */
    color absorption_coefficients = color (0.2, 0.3, 0.5);
    /*
        name {Absorption Coefficients}
        desc {Absorption coefficients of hair medium for R, G and B channels.}
    */
    float cuticle_angle = 5;
    /*
        name {Cuticle Angle}
        desc {The angle of cuticle scales on hair fiber.}
    */
    ...
)
{
    //Defining the uniform parameters used in the pre-computation step
    float d_b, d_f = 0.7;

    color ā_f[θ_h], ā_b[θ_h], ᾱ_f[θ_h], ᾱ_b[θ_h], β̄_f^2[θ_h], β̄_b^2[θ_h];
    color σ̄_b^2[θ_h], Ā_b[θ_h], Δ̄_b[θ_h], N_R^G[θ_h], N_TT^G[θ_h], N_TRT^G[θ_h];

    //Defining the varying parameters which have different values for each shading point
    varying float hairs_in_front;
    varying color σ̄_f^2;
    varying color T_f;

    //Defining the auxiliary functions like normalized Gaussian function etc.
    float g(float variance; float x;)
    {
        return exp(-x*x/(2*variance))/sqrt(2*π*variance);
    }
    ...

    //Pre-computing and tabulating the uniform variables in the Constructor
    public void construct ()
    {
        //Pre-computations should be done in the following order:
        ā_f[θ_h], ā_b[θ_h] = ...
        ᾱ_f[θ_h], ᾱ_b[θ_h] = ...
        β̄_f[θ_h], β̄_b[θ_h] = ...
        Δ̄_b[θ_h] = ...
        σ̄_b^2[θ_h] = ...
        Ā_b[θ_h] = ...
        N_R^G[θ_h], N_TT^G[θ_h], N_TRT^G[θ_h] = ...
    }

    //Main body of the hair surface shader.
    public void surface( output color Ci, Oi;)
    {
        //Compute Hair tangent U, and viewing direction ω_r, θ_r and φ_r
        vector U = - normalize (dPdv);
        ω_r = -normalize(I);
        ...

        // Loop over all the lights in the scene
        illuminance (P )
        {
            //Compute light direction ω_i, θ_i, θ_h, θ_d, φ_i and φ
            ω_i = normalize(L);
            ...

            //Compute the amount of shadow from the deep shadow maps
            float shadowed = 0;
            lightsource ("out_shadow", shadowed);
            float illuminated = 1 - shadowed;

            //Estimate the number of hairs in front of the shading point
            hairs_in_front = shadowed * hairs_that_cast_full_shadow;

            //Use the number of hairs in front of the shading point to approximate σ̄_f^2
            σ̄_f^2 = hairs_in_front * β̄_f^2[θ_h];

            //Use the number of hairs in front of the shading point to approximate T_f
            T_f = d_f * pow(ā_f[θ_h], hairs_in_front);

            //Computing f_s^direct and f_s^scatter
            f_s^direct =
                M_R(θ_h) * N_R(θ_d, φ) +
                M_TT(θ_h) * N_TT(θ_d, φ) +
                M_TRT(θ_h) * N_TRT(θ_d, φ);
            f_s^scatter =
                M_R^G(θ_h) * N_R^G(θ_d, φ) +
                M_TT^G(θ_h) * N_TT^G(θ_d, φ) +
                M_TRT^G(θ_h) * N_TRT^G(θ_d, φ);

            //Computing f_back^direct and f_back^scatter
            f_back^direct =
                2 *Ā_b[θ_d] * g(θ_h-Δ̄_b[θ_d], σ̄_b^2[θ_d])/(π * cos^2(θ_d));
            f_back^scatter =
                2 *Ā_b[θ_d] * g(θ_h-Δ̄_b[θ_d], σ̄_b^2[θ_d]+σ̄_f^2[θ_d])/(π * cos^2(θ_d));

            // Computing F^direct
            color F^direct = (illuminated) * (f_s^direct + d_b * f_back^direct);

            // Computing F^scatter
            color F^scatter = (T_f - illuminated) * d_f * (f_s^scatter + π * d_b * f_back^scatter);

            // Computing the final result
            Ci += (F^direct + F^scatter) * cos(θ_i);
        }
        Oi = Os;
        Ci *= Oi;
    }
}
```