Advanced Material Rendering : Human Skin

Rajaditya Mukherjee*
The Ohio State University

Micah D Lamb†
The Ohio State University

Abstract

Skin has always been difficult to render: it has many subtle visual characteristics, and human viewers are acutely sensitive to the appearance of skin in general and faces in particular. The sheer amount of detail in human skin presents one barrier. A realistic model of skin must include wrinkles, pores, freckles, hair follicles, scars, and so on. Fortunately, modern 3D scanning technology allows us to capture even this extreme level of detail. However, naively rendering the resulting model gives an unrealistic, hard, dry-looking appearance.

For most materials, the reflectance of light is usually separated into two components that are handled independently: (1) surface reflectance, typically approximated with a simple specular calculation; and (2) subsurface scattering, typically approximated with a simple diffuse calculation. However, both of these components require more advanced models to generate realistic imagery for skin. Even the highly detailed diffuse, specular, and normal maps available with modern scanning techniques will not make skin look real without accurate specular reflection and subsurface scattering.

In this report, we will review and collect the existing algorithms for realistic skin rendering based on the principles of subsurface scattering on biological materials. We will use the paper by [dEon et al. 2007] for our algorithm. Then we will render a model of the human face using the algorithms based on the paper.

CR Categories: I.3.3 [Computer Graphics]: Three-Dimensional Graphics and Realism—Material Rendering I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation;

Keywords: animation, material rendering, subsurface scattering

Links: DL PDF

1 Introduction

The human skin varies in thickness ranging from 0.1 to more than 0.5 cms. It is primarily composed of three distinct layers, each with its own optical properties. The **epidermis** is the thin outer layer, whihc includes the exterior layer of dead cells (the *stratum cornuem*). The **dermis** is thicker, and includes the vessels that carry blood. **Hypodermis** connects the skin to the rest of the body. [A.Walters 2002] There is also an outer oil layer that accounts for the specular reflection properties of the skin.

*e-mail:raj@rajaditya.com
†e-mail:micah.d.lamb@gmail.com

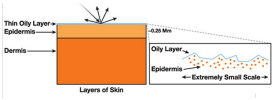

Figure 1: *Multilayered Structure of Skin*

Because of this oil layer, roughly 6 percent of all light incident on the skin reflects directly without being absorbed. This reflection is obviously not perfect because of the roughness of the skin which causes a single incident ray to reflect into a range of exitant angles. This roughness is descibed by the specular *Bidirectional Reflection Distribution Function* or BRDF.

Simple empirical specular models, such as the familiar Blinn-Phong model long supported by OpenGL and Direct3D, do not accurately approximate the specular reflectance of skin. As a result, if we want realistic rendering of human skin, we need to take physically based specular model into account.

The remaining light that is not reflected is absorbed by the skin and enters the subsurface layers. The scattering and absorbtion of light in these layers give the skin its characteristic color and appearance. This process is further complicated by the fact that multiple layers in the skin absorb and scatter light differently as shown below.

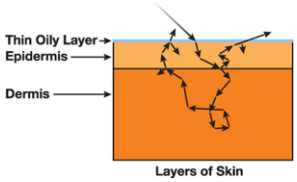

Figure 2: *Subsurface Scattering in Skin*

A realistic skin shader must model this scattering process otherwise the result appears hard and dry. [Krishnaswamy and Baronoski 2004] uses a complex optical model comprising of five layers but for most practical rendering purposes using a two layered model in [Donner and Jensen 2006] suffices our purpose. In order to simplify this process, we note that beneath the surface of the skin, the light quickly becomes diffuse as it scatters. This allows us to use a physical model known as *diffusion model*. Unfortunately, the mathematics of diffusion model is quite sophisticated to render and hence we will approximate this model using a weighted sum of gaussians.

2 Algorithmic Overview

2.1 Specular Reflection

A small percentage of the light that hits out skin specularly reflects off the top oily layer. This specular reflection is not modelled efficiently by the Phong Reflection as we explained before. To model this we will develop a BRDF which takes the fresnel effect into account. It will also represent the fact that light is more likely to reflect at grazing angles. For this purpose, we will use the Kelemen/Szirmay-Kalos specular BRDF function to model the skin. [Kelemen and Szirmay-Kalos 2001]

We will have to prefactor the BRDF for efficient computation. Our implementation will use the Beckmann distribution function and the Schlick Fresnel approximation to keep the calculation efficient. We can precompute the Beckmann Distribution into a texture for easy access. Another important aspect of the specular reflection is that it will be white (the same color as the light) since the oil is a dielectric and the light is not altered when it reflects like happens with several metals.

2.2 Subsurface Scattering

As mentioned before, we will use the Diffusion Model to simulate the subsurface scattering. A diffusion profile provides an approximation for the manner in which light scatters underneath the surface of a highly scattering translucent material. The diffusion profile tells us how much light emerges as a function of the angle and distance from the point where the ray strikes. If we consider only uniform materials, the scattering is the same in all directions and the angle is irrelevant. Each color has its own profile, which we can represent as a 1D curve, as illustrated below.

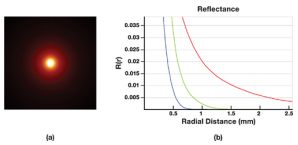

Figure 3: *Sample Diffusion Profile*

The profiles plotted above closely resemble the well-known Gaussian function e^{-r^2} . We can approximate the diffusion profile by adding together several gaussian functions. Gaussians have several useful property that make them extremely attractive for rendering such as they are simultaneously separable and radially symmetric, and they convolve with each other to produce new Gaussians. We use the exact same principle in our rendering. For each diffusion profile $R(r)$, we find k Gaussians with weights w_i and variances v_i such that

$$R(r) \approx \sum_{i=1}^{k} w_i G(v_i, r)$$

where we define the Gaussian Variance v as

$$G(v, r) = \frac{1}{2\pi v} e^{\frac{-r^2}{2v}}$$

Accurate rendering requires knowing the exact shapes of the diffusion profiles for the material we want to simulate. For multilayer materials with different scattering properties, we can use the multipole model of [Donner and Jensen 2005]. In order to model this via sum of gaussians, we will require six Gaussians to accurately model the skin. The parameters and the resulting diffusion profiles is as shown.

	Variance (mm^2)	Blur Weights Red	Green	Blue
·	0.0064	0.233	0.455	0.649
·	0.0484	0.100	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

Figure 4: *Sum of Gaussian Parameters for Multipole Skin Model*

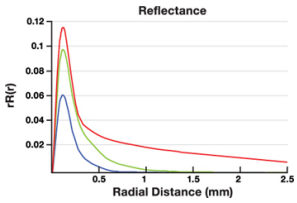

Figure 5: *Diffusion Profiles for Skin*

Given that we have computed the appropriate diffusion profiles for the material, simulating subsurface scattering involves collecting all the light rays that fall on the incoming light and then spreading it around the neighbourhood based on this profile. So we sum up all the light incident at a point in the skin. Then this diffuse light scatters into the neighbouring regions (as determined by the diffusion profiles) and then exits the surface flowing equally in all directions. This is our two part rendering algorithm for the human skin.

3 Method

We have implemented the final solution using a GPU shader. It uses multiple GLSL shading units to get the final result. We describe each of the components in parts.

3.1 Specular Lighting

To compute the specular lighting parameters, we have used the Kelemen/Szirmay-Kalos specular BRDF. We have used the Beckmann distribution function and precomputed the results in a texture. This is done by the shader unit called `beckmann.frag`.

After this the final specular color is computed in the final pass in the shader entitles `face~~t~~rag`. The Fresnel relectance is computed by the function `fresnelReflectance`. The computation of the final specular color is done in the function `KS_Skin_Specular` in the same file.

3.2 Computing the Irradiance Map

The next part is the computing the irradiance maps from the disparate elements. We already have the diffuse color and the normal maps from the texture pack that we had obtained before.

First we will have to compute the shadow maps. We have used a single light source. Hence we first compute the soft shadow map. Then we will use thsi together with the diffuse color maps and the normal map to generate the final irradiance map. This is done in the GLSL shader entitled `irradiance.frag`.

We will implement the above algorithm using a GPU shader first. If constraints permit, we will port the shader code to `pbrt` as a ray tracing solution. Our algorithm requires the parameterized mesh as well as several maps as inputs. Principle of them are the ambient occlusion maps, bump map, diffuse maps and specular brightness maps.

3.3 Stretch Map Computation

Since we will be computing the gaussian blurs in the texture space independently in U and V directions, we may introduce distortions in the final image if we are not careful.

Hence to correct such distortions in the local convolution kernel, we have computed the two channel stretch maps in U and V direction.

This is done in the shader entitled `stretch.frag`. These maps are used during the blurring phase described next.

3.4 Subsurface Scattering

The subsurface scattering part uses the `blur.frag` to generate the six textures. Each of the textures is generated by blurring the irradiance map successively with different values of the variance. First we blur it in the U direction, apply the stretch correction maps for U direction. Then we apply the blur in the V direction and again apply the stretch maps in the V direction. This produces 6 individual textures which we will combine in the final phase.

3.5 Perturbation/Noise Texture

The actual human face has quite some irregularities which unfortunately our textures couldn't reproduce correctly owing to quality issues. A part of it was modelled by the normal perturbation maps. However we added some perlin noise to the final image to make the image look a little more realistic in terms of the texture roughness. Though the effect is not noticable in normal lighting, upclose views can show us how this subtly effects the visual impact of our realistic skin shader.

3.6 Aggregation

Everything was aggregated in the final solution in the final pass to the shader entitled `face.frag`. This is a pseudocode representation of the entire shader.

Algorithm 1 Final Face Shader

```
Load all the textures computed before
funcDef fresnelReflectance()
funcDef KS_Skin_Specular()
Compute the diffuse color weight as per table
Compute light direction Information
Compute Ambient Color
Add diffused Color Component calculated above
Add Specular component by calling KS_Skin_Specular()
Output final result as fragment color
```

4 Source Code

The source code for the project can be found in the following *GitHub* repository : Human Skin Rendering Source Code

5 Results

Figure 6: *Final Result*

As it can be seen from the figure above, we obtained a decent image of the human face rendering. Admittedly, it was not as photorealistic as the original source. However we attribute this difference to the quality of the textures that we had obtained over the internet. They were not HDR textures that were probably used in the original source.

6 The Final Word

This was undoubtedly one of the most challenging projects in Graphics that we have worked. The implementation, looking for quality textures and ofcourse processing the underlying maths made this project an intellectually simulating one. We thank our course instructor Prof. Raghu Machiraju for his guidance and support through the entire duration of the project.

The area of human skin rendering is one which holds a lot of relevance in the present consumer driven industry of games and animation. This years GPU Technology Conference showcased a wonderful rendering of the human face. They used the state of the art Kepler GPU for this purpose. The interested reader is directed to the following link : GTC 2013 Face Works

References

A.WALTERS, K. 2002. *Dermatological and Transdermal Formulations*. Marcel Dekker Incorporated.

DONNER, C., AND JENSEN, H. W. 2005. Light diffusion in multilayered translucenet materials. In *Proceedings of SIGGRAPH 2005*.

DONNER, C., AND JENSEN, H. W. 2006. A spectral shading model for human skin. In *Proceedings of SIGGRAPH 2006*.

DEON, E., LUEBKE, D., AND ENDERTON, E. 2007. Efcient rendering of human skin. In *EUROGRAPHICS Symposium on Rendering*.

KELEMEN, C., AND SZIRMAY-KALOS, L. 2001. A microfacet based coupled specular-matte brdf model with importance sampling. In *EUROGRAPHICS*.

KRISHNASWAMY, A., AND BARONOSKI, G. 2004. An introduction to light interaction with human skin. *Revista de Informatica Teorica e Aplicada 11*, 1, 33–62.