A Critical Comparison of Human Face Rendering Techniques

by

Arturo Andrew Arizpe

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2, 2006

Author .....
Department of Electrical Engineering and Computer Science
June 2, 2006

Certified by .....
Fredo Durand
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Department Committee on Graduate Theses

A Critical Comparison of Human Face Rendering Techniques

by

Arturo Andrew Arizpe

Submitted to the Department of Electrical Engineering and Computer Science
on June 2, 2006, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Computer Science and Engineering

Abstract

Human skin exhibits complex light reflectance properties that make it difficult to render realistically. In recent years, many techniques have been introduced to render skin, with varying degrees of complexity and realism. In this thesis, I will implement several of these techniques, and use them to render scenes with various lighting and geometry parameters, in order to compare their strengths and weaknesses. My goal is to provide a clearer understanding of which rendering techniques are most effective in different scenarios.

Thesis Supervisor: Fredo Durand

Acknowledgments

First and foremost, I would like to thank my thesis supervisor, Fredo Durand, who always had helpful suggestions to keep me on the right path. Thanks also to Jan Kautz, Tom Mertens, and other members of the Computer Graphics Group, for providing assistance and answers to my questions. I would also like to thank my parents, for supporting and encouraging my educational endeavors throughout my entire life. Finally, thanks to my wonderful fiancée Arielle, for always being loving and supportive after a long day of work.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The aim of this research is to provide a critical comparison of techniques for rendering realistic-looking human faces. This chapter will provide some motivation for why skin is difficult to render realistically, as well as describe how I structured my comparison of skin rendering techniques.

1.1 Skin Local Illumination Properties

Skin is a fairly complex material. It is made up of several different translucent layers, each with different optical properties. Because of this translucency, skin exhibits subsurface scattering. Rather than simply being reflected off of the surface, light rays enter the skin, are reflected (or scattered) below the surface, and then re-emerge (see figure 1-1). Because each reflection that occurs beneath the surface scatters light in a random direction, light that exits the skin tends to be distributed equally in all directions, contributing to skin's diffuse appearance. Also, because light can enter the surface at one point and exit at another, it exhibits internal color bleeding.

Another feature that can affect the realism of rendered skin is the soft glow that can be seen when skin is illuminated at a grazing angle. This occurs because the skin is covered by a layer of tiny hairs. When light strikes the skin perpendicular to the surface, it does not interact with the hairs enough to make a noticeable difference. However, when the light is nearly parallel to the surface, it scatters through many

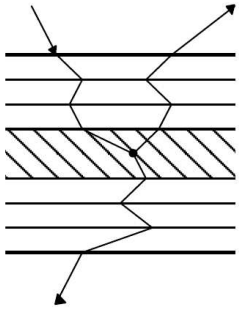

Figure 1-1: A light ray enters the surface of a translucent object. Below the surface, the ray splits into a component that is scattered back towards the surface, and a component that continues into a deeper layer of the material. (Taken from [4]).

of the hairs, leading to the previously mentioned glow. This is a somewhat subtle characteristic, but when done correctly it can make the skin appear more soft and realistic.

These two properties are quite useful in rendering realistic human skin, and both can be represented by the local illumination model used to simulate the skin. The local illumination model determines how incident light interacts with and reflects from a material. A major part of my comparison involves varying the local illumination model and comparing how well each model performs in different circumstances.

1.2 Comparison Approach

The basic approach of my comparison consists of rendering essentially the same scene many times while varying different parameters to get many different images. These images are compared by rendering time, and also qualitatively by realism. As mentioned above, perhaps the most important parameter that I vary is the local illumination model. I used four different models: a measured BRDF of skin, a Hanrahan-Krueger subsurface scattering BRDF, a multi-layer subsurface scattering BSSRDF, and an asperity scattering model. Each of these will be described in detail in the following chapters.

The next parameter that I varied was the type of light source used to illuminate

the skin. A local illumination model may behave differently depending on what type of light source is used, and I wanted to be able to determine whether some models were only effective under certain types of lighting. I used four different types of light sources: point, area, directional, and environment map. The differences between these light sources will be explained in a later chapter.

Finally, I varied the level of detail of the geometric mesh used to render the face. The high quality mesh I used was made up of one million triangles, and was detailed enough to distinguish pores in the skin. I also used a low quality mesh consisting of only 16750 triangles, plus a bump map to simulate the geometry of the high quality mesh. (A bump map is simply a texture that is used to modify the surface normals when rendering, to give the illusion that the surface is bumpier than the geometry actually is.) The purpose of varying this parameter was to determine whether very high quality input data is necessary to render realistic skin, or whether a bump map is sufficient.

Chapter 2

Background

In this chapter, I present the four local illumination models that I am comparing, and explain how each works. Only the theoretical concepts behind these models are described here. The actual implementation details are covered in the following chapter.

2.1 Measured BRDF

A common technique in rendering is to use a Bidirectional Reflectance Distribution Function (BRDF) to describe the reflectance properties of a material. A BRDF represents, for a given point on the surface of an object, the relationship between incoming and outgoing light. More specifically, the BRDF is a four-dimensional function which takes as input an incoming and an outgoing direction, and it outputs the amount of light that is reflected from the incoming to the outgoing direction by the material.

While some BRDFs are derived from theoretical models of reflection, for complex materials it is sometimes more effective to simply measure the BRDF by taking a sample of the material, applying light from many different directions, and measuring the reflected light in many different directions. One of the local illumination models that I evaluate is a BRDF that takes data measured from real human skin, fit to Lafortune et al.'s model [8] for parameterizing BRDFs.

2.2 Hanrahan-Krueger BRDF

Hanrahan and Krueger proposed a model for subsurface scattering based on one-dimensional linear transport theory [4]. In it, they model skin as a two-layer material, with each layer having different reflectance and transmittance properties. The total light reflected by the skin is determined by light reflected off the top layer, as well as transmitted through the top layer and reflected off the bottom.

The Hanrahan-Krueger model makes some simplifying assumptions that make the implementation simpler but less accurate. For example, because skin is made up of multiple layers, light that scatters below the surface could theoretically reflect back and forth between layers several times before exiting the surface. This model ignores that, and instead enforces the constraint that once light leaves a lower layer for a higher one, it does not return. Another source of inaccuracies in the Hanrahan-Krueger model is that it uses a BRDF to simulate skin. A BRDF only relates the amount of reflected light in the outgoing direction to the amount of incident light from the incoming direction at a single point. It provides no mechanism for light that is incident at one point to exit at another, which is an important feature of subsurface scattering.

2.3 Multi-layer BSSRDF

Jensen et al. [6] improved on this technique with a BSSRDF (Bidirectional Surface Scattering Reflectance Distribution Function)-based model. A BSSRDF is an eight-dimensional function similar to a BRDF, but with an additional dependence on the point of incidence and the point of exitance, thus allowing for light to enter the skin at one point and exit at another (see figure 2-1). This model uses a dipole to approximate the diffuse scattering of light below the surface. Each incident ray of light is converted into a dipole (a positive light source located just below the surface of the skin, and a negative light source located just above it - see Figure 2-2). The two light sources are placed such that the net inward diffuse radiance is zero along the

surface of the skin (i.e. any diffuse light that escapes the surface does not return). To render with this model, at each point, x, on the surface of the skin, nearby points are sampled to calculate the amount of incoming light. These light rays are converted to dipoles, and the contributions of these dipoles to the exitant light at x are summed. This produces a softer, more realistic looking skin than the BRDF model of Hanrahan and Krueger, but is very computationally expensive to compute. In order to reduce the noise in the rendering to an acceptable level, a very large number of samples must be taken for every point, and the dipole approximation must be calculated for each one.

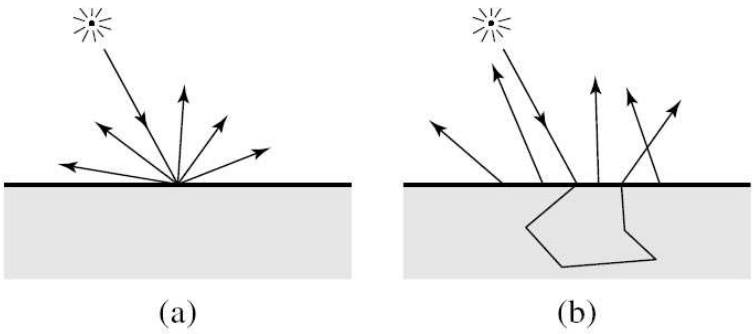

Figure 2-1: In a BRDF (a), light must enter and exit at the same point. In a BSSRDF (b), light can scatter beneath the surface and exit at a different point. (Taken from [6]).

Jensen and Buhler [5] improved the performance of the dipole technique by introducing a two-phase rendering scheme. The first phase is sampling the irradiance at points all over the surface of the object to be rendered. Each sample is represented as the point of incidence, the amount of irradiance, and the area associated with that point. (Turk's point repulsion algorithm [10] is used to distribute the sample points evenly over the mesh, so that each point covers the same amount of area.) The samples are stored in an octree, a data structure used for organizing values in three-dimensional space. In the next phase, the image is rendered as it was for the dipole model. However, rather than taking a large number of new samples for each point on the surface of the skin, the irradiance is simply looked up the in octree, and the dipole

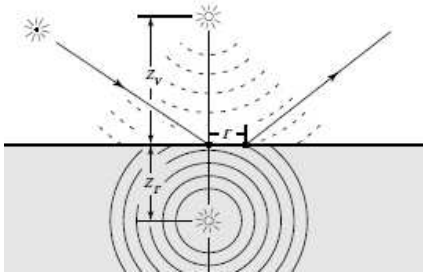

Figure 2-2: An incident light ray is converted to a dipole (taken from [6]).

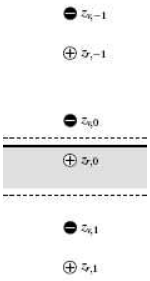

Figure 2-3: A series of dipoles above and below the surface are used to approximate diffuse reflectance in a thin slab (taken from [1]).

approximation is calculated from that. A significant performance improvement can be realized by aggregating many distant samples into a single dipole. Because these distant samples contribute only a small percentage to the total exitant radiance, this optimization can be done with a negligible loss of accuracy.

The dipole method described above simulates a scattering material that is composed of an infinitely thick single layer. Although this gives a fairly realistic looking image, real skin is composed of several thin layers, which is not possible to represent with the dipole approximation. Donner and Jensen [1] proposed a multipole model to simulate thin slabs, as well as a method of combining layers by convolving their transmittance and reflectance profiles. Just as the dipole model used the assumption that light that leaves through the top surface does not return, the multipole model assumes that any light transmitted through the bottom surface of a slab does not return. Enforcing this requires placing a second dipole below the slab, to balance the contribution of the top dipole at the bottom surface. However, this second dipole causes an imbalance at the top surface, which necessitates a third dipole above the first, and so on (Figure 2-3). To perfectly balance both the top and bottom surfaces, an infinite number of dipoles would be necessary. However, because each additional dipole contributes less and less to the total sum, a large number is sufficient in practice.

2.4 Asperity Scattering

Unlike the above models, which attempt to simulate the reflectance properties of the skin itself, asperity scattering [7] is a rendering technique that tries to simulate the effect of a thin layer of sparse scattering entities (such as dust or small hairs) on the surface of an object. It can be used to give a softer appearance to objects that are covered in small hairs or fibers, such as fabric, some plants, and human skin. Because the scatterers are sparsely distributed, the asperity scattering term does not contribute noticeably to the reflected light at angles nearly perpendicular to the surface. However, when either the direction of incoming light or the viewing direction is at a grazing angle to the surface, the effect of many scatters is compounded, and the result is a soft glow around the edges of the object. This glow is an important visual cue to suggest that the object is soft or "velvety".

Because the asperity scattering term simulates hair instead of skin, it should be combined with another local illumination model that simulates the properties of skin. The underlying skin model is used to calculate the shading for the skin itself, and the asperity scattering term adds a layer of glow on top of that. The angles of the incident and reflected light determine both the intensity and opacity of the glow, so that it only makes an appreciable difference at grazing angles.

Chapter 3

Implementation

This chapter describes the implementation details of the various components I wrote in order to render the images found in the results section. While the focus of my research was on the local illumination models used to simulate skin, I also needed to be able to vary the scene lighting and geometry to make a thorough comparison. The remainder of this chapter is broken up into sections for lighting, geometry, and local illumination models. Each section details the tools and techniques I used in my research to vary that particular parameter and, where applicable, how I implemented them.

Some of my work built upon code written by UROPs in the Computer Graphics Group during the summer of 2005. All of the rendering was done using pbrt, a ray-tracer developed by Matt Pharr and Greg Humphreys [9].

3.1 Light Sources

I used four different types of light source in my renderings: point, area, directional, and environment map. Because each of these light source types is already provided in pbrt, I did not have to implement any light source. However, I will give a brief description of each type of light.

A point light source is one in which all light originates from a single point. The intensity of the incident light at a point is proportional to the strength of the light

source and the inverse square of the distance from the source to the point. A point light source is a simple and reasonable abstraction, but does not produce the most realistic-looking images. I used a point light source positioned above and to the right of the face, to get a good mix of light and shadow in the scene.

An area light source is an improvement on the point light that is designed to act more like an actual light bulb. Rather than all of the light originating from a single point in space, an area light source defines a shape from which the light is emitted. While this may seem like a small distinction, it allows for the rendering of soft shadows, an important feature of realistic images. With a point light source, every point on an object is either completely illuminated by the light source or completely occluded by another object, resulting in a sharp boundary between the shadowed and illuminated region. With an area light source, it is possible for a point to be only partially shadowed, which allows for a soft shadow boundary. Because real-world light sources have some non-zero area, they cast soft shadows, so hard shadows make computer-generated images look fake. I used a disc-shaped area light source with a radius of 7 centimeters, positioned around the same location as the point source, in order to make a good comparison between the two. (For reference, the face mesh is around 30 centimeters from top to bottom.)

A directional light source does not originate from any particular point. Instead, it casts light rays in the same direction at all points in the scene. It could also be considered to be a point light source that is infinitely far away, so that the light rays that reach the scene are all parallel. A directional light source can be useful to represent the sun. Although the sun is actually more like an area light source, it is so far away that the rays that reach the earth all point in essentially the same direction. I used two directional light sources, one from the front-left and one from the front-right, so that I could have a scene in which the face is fully illuminated.

An environment map (or infinite area light) is like an area light source that surrounds the entire scene. It usually consists of an image where each pixel (u, v) corresponds to some direction (θ, ϕ). The color of each pixel is the color of the light coming from that direction. Environment maps can add a great deal of realism to a scene,

because they can be used to make an object appear as if it is actually illuminated by its surroundings, instead of by arbitrarily placed light sources. I used an environment map that simulates a sunset. The texture I used can be seen in Figure 3-1.

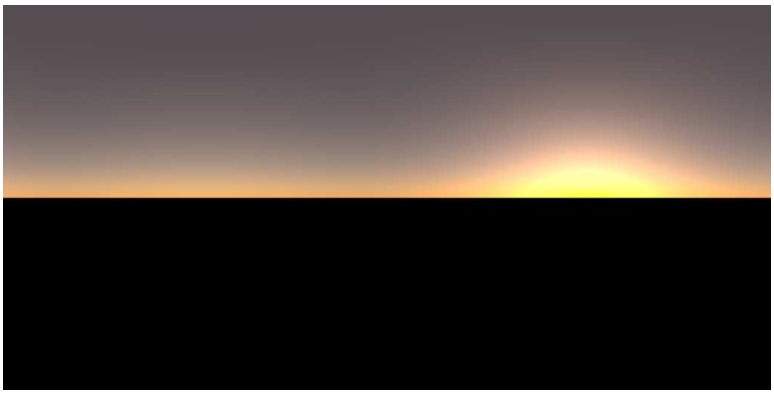

Figure 3-1: The environment map used to simulate the illumination of a sunset. This texture was provided as part of the pbrt distribution [9].

3.2 Geometry

One of the goals of my research was to determine whether some local illumination models were less effective at rendering realistic images if the quality of the geometric data used as input was reduced. To test this, I took a mesh with a high level of detail and used it to generate a mesh with a low level of detail with a bump map to approximate the fine geometry of the original mesh.

I used the qslim program [3], by Michael Garland to generate a mesh with a low level of detail. qslim takes as input a triangle mesh and a desired number of faces, and it produces a new triangle mesh that approximates the old one using the specified number of triangles. Unfortunately, qslim does not preserve the vertex normals or parameterization (texture coordinates) of the mesh, so I had to implement a way to calculate those on the new mesh.

3.2.1 Generating Vertex Normals

To generate the vertex normals for the new mesh, I use a simple method that works fairly well in practice. I first read in the mesh and calculate a normal vector for each face, by taking the cross product of two of the edges. Then, for each vertex I average the face normals from each face that touches that vertex, weighted by the angle of the face at that vertex, and renormalize. This method generates reasonably accurate vertex normals.

3.2.2 Generating Parameterization

To generate a parameterization for the new mesh, my basic strategy is to use the texture coordinates from nearby vertices in the old mesh. I first use the old mesh to create a mapping from vertices in the old mesh to texture coordinates. Because the texture is essentially made by cutting the mesh and "unfolding" it so that it is two-dimensional, there are some vertices that have more than a single texture coordinate pair. (Specifically, the vertices that are along the cut in the mesh will have two texture coordinate pairs – one for each side of the cut.) I keep track of all texture coordinates for each vertex, so that later I can choose which one to use.

After I generate the mapping of old vertices to texture coordinates, I read in the new mesh and, for each vertex in the new mesh, I find the closest vertex from the old mesh. This forms the basis for the new parameterization. For every face in the new mesh, I take the three new vertices, use those to find the three nearest vertices from the old mesh, and use the texture coordinates of those vertices as the texture coordinates for the new vertices. If one or more of the old vertices has multiple texture coordinates, I evaluate all possible combinations of texture coordinates and use the one that results in the triangle with the smallest perimeter. (The idea is that if two of the vertices are on one side of the cut, and the other vertex has two options, I want to choose the option that is on the correct side of the cut.)

This would be enough if every vertex along the cut in the new mesh corresponded to a vertex along the cut in the old mesh. However, it is sometimes the case that

a triangle in the new mesh will have two vertices on one side of the cut, and one on the other, and that none of these vertices will have multiple options for texture coordinates. I detect this case by looking for triangles in texture space with unusually large perimeters. When this happens, I take the two texture coordinates that are on the same side of the cut and I create a new texture coordinate near those two, so that I have a small triangle in texture space. When this process is all done, I have a parameterization that, while not as clean and symmetrical as the original, is still usable to render a good image.

3.2.3 Generating Albedo and Bump Maps

Now that I have a parameterization for the new mesh, I can generate albedo and bump maps that use that parameterization. For each triangle in the mesh, I rasterize that triangle into texture space by using the texture coordinates of the vertices to generate a bounding box, and then calculating the edge equations of the triangle for every pixel inside that box. If a pixel is covered by the triangle, I calculate the barycentric coordinates of that point within the triangle, and use them to interpolate the vertex normals to get a normal for that point.

Once I have the normal, I cast two rays from that point into the original (high triangle count) mesh. I cast the first ray in the direction of the normal, and the second in the exact opposite direction. Because this requires casting millions of rays for high resolution texture maps, I use a grid acceleration structure to reduce the amount of time that it takes. In practice, it takes around five to ten minutes to generate albedo and bump maps at a resolution of 4096 x 4096. Because this is only a one-time cost and is not incurred for each rendering, this amount of time is acceptable.

After casting those two rays, I have the nearest point in the original mesh along the normal for the current point. I save the signed distance between these points to be used in the bump map, and I calculate the barycentric coordinates of the intersection point and use them to interpolate texture coordinates. I look up those texture coordinates in the original mesh's albedo map, and I use that value for the current pixel in the new mesh's albedo map.

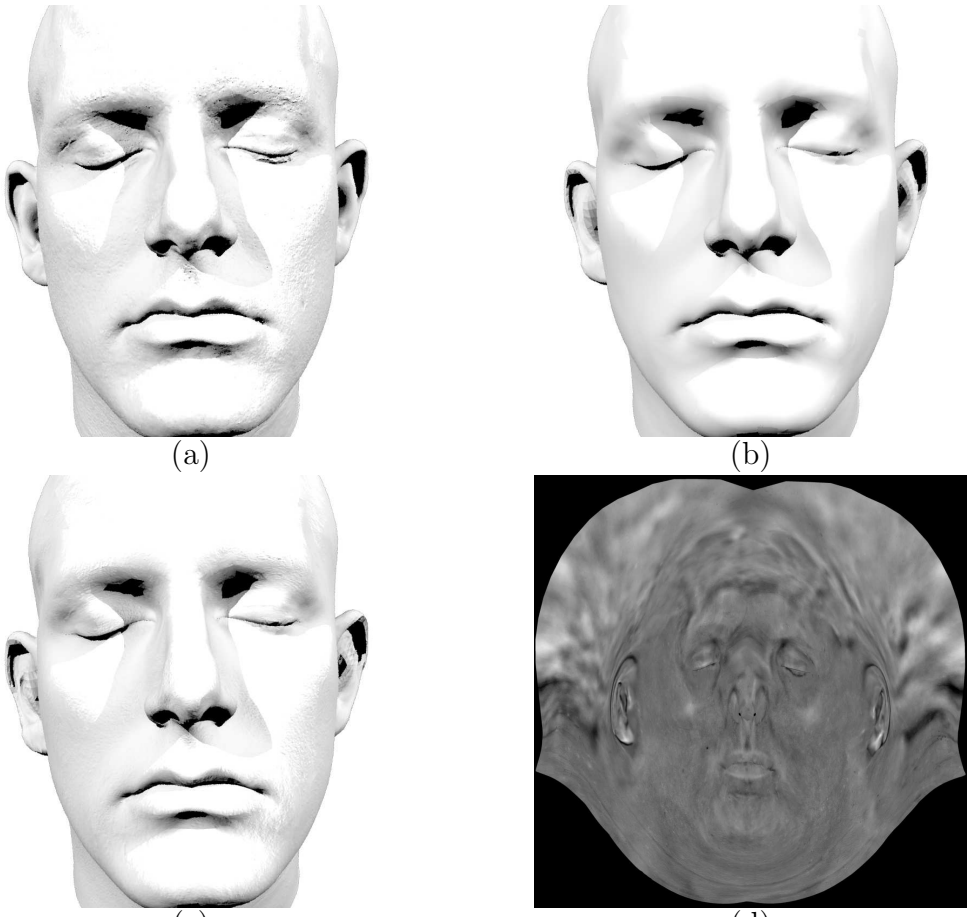

Figure 3-2: The 1000000-triangle mesh (a) has enough detail to show variation in the surface of the skin. The 16750-triangle mesh (b) produces a smooth surface. In (c), the same 16750-triangle mesh is rendered with a bump map (d). (Pixels in the bump map correspond to heights, with light colors being higher.)

Finally, after every face has been rasterized, I take the distances that were saved for the bump map and shift and scale them so they span the range 0 to 1. I use that normalized value as the height in the bump map. Because of discrepancies between the original and new mesh, especially around the edges of the mesh, sometimes the rays cast will miss the old mesh entirely, or hit it at an incorrect point. To counteract this problem, I simply ignore distances greater than some threshold value.

Figure 3-2 shows the effect of the bump map on the low level of detail mesh.

The method described above generates a fairly detailed bump map. One trick to make the surface of the skin appear more diffuse and softer is to blur this bump map, to make the normal vector vary more smoothly over the surface of the skin. After

computing the bump map, I blurred it with a Gaussian blur kernel and rendered images with both the blurred and non-blurred bump maps to compare the effects.

3.3 Local Illumination Models

The majority of my coding work spent on implementing local illumination models. This section will describe in detail how each of the models was implemented.

3.3.1 Measured BRDF

The measured skin BRDF uses Lafortune et al.'s model to parameterize measured BRDF data [8]. This BRDF is implemented as a material plugin that is included with the pbrt distribution. Therefore, I did not have to implement the reflectance model or do the actual measurement of the BRDF data. However, I did make a small modification of the code for this plugin, to allow it to take an albedo texture rather than use a constant color.

3.3.2 Hanrahan-Krueger BRDF

I implemented the Hanrahan-Krueger model as a new BRDF and material plugin for pbrt. The basic idea of this model is that the amount of light reflected by a material that exhibits subsurface scattering is calculated by summing the amount of light reflected by each layer times the percentage of light that actually reaches that layer. Skin is represented as a two-layer material, consisting of the epidermis and the dermis, each with different reflectance parameters that determine how light reflects from that layer.

In this model, each layer is parameterized by its index of refraction, η, the absorption cross section, σ_a, the scattering cross section, σ_s, the thickness of the layer, d, and the mean cosine of the phase function, g. The absorption and scattering cross sections represent the average number of absorption or scattering events per unit length, respectively. The mean cosine of the phase function is used as a parameter to

the Henyey-Greenstein equation, and essentially determines in which direction light is likely to scatter. In addition to these parameters, they define the total cross section as $\sigma_t = \sigma_a + \sigma_s$, the optical depth as $\tau_d = \sigma_t \cdot d$, and the albedo as $W = \frac{\sigma_s}{\sigma_t}$. The total cross section represents the expected number of either absorptions or scatterings per unit length. The optical depth is used to determine how much light is attenuated from the top to the bottom of the layer. Finally, the albedo represents the percentage of interactions that result in scattering. Basically, a higher albedo indicates that the material is more reflective, and a lower albedo indicates that the material absorbs most of the incident light.

Given these parameters, it is a straightforward calculation to determine the amount of reflected light in some direction, $L_r(\theta_r, \phi_r)$, given the incident light from some other direction $L_i(\theta_i, \phi_i)$. Hanrahan and Krueger derive the following equation for the case of a single scattering event below the surface:

$$L_r(\theta_r, \phi_r) = WT^{12}T^{21}p(\pi - \theta_r, \phi_r; \theta_i, \phi_i)\frac{\cos\theta_i}{\cos\theta_i + \cos\theta_r}(1 - e^{-\tau_d(1/\cos\theta_i + 1/\cos\theta_r)})L_i(\theta_i, \theta_i)$$

In this equation, T^{12} and T^{21} are the fresnel transmittance terms for entering and leaving the surface, respectively, and $p(\pi - \theta_r, \phi_r; \theta_i, \phi_i)$ is the Henyey-Greenstein equation mentioned above. This value is calculated for both layers, with L_i for the first layer being the the amount of incident light at the surface, and for the second layer being the reduced intensity, equal to $e^{-\tau_d/\cos\theta_i}$ times the incident light at the surface. These two values are summed, giving the reflected light from a single scattering event. Figure 3-3 illustrates this process. Hanrahan and Krueger note that the reflectance curve from multiple scattering event follows the same shape as the curve for a single scattering event, meaning that more scattering events tend to distribute light diffusely. Therefore, to approximate multiple scattering I simply add a constant term to the single scattering result.

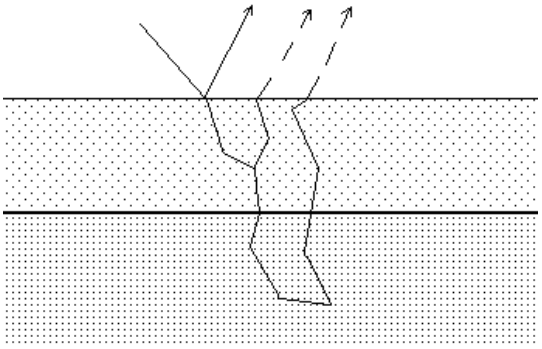

Figure 3-3: Some of the light that enters the skin scatters in the first layer and reflects out. Some of it enters the second layer, then scatters and reflects out. The dashed rays indicate where the light would actually exit, but because this is a BRDF, it is approximated as exiting at the point of incidence.

3.3.3 Multi-layer BSSRDF

The multi-layer BSSRDF could not be implemented as a simple material in pbrt, because materials are constrained to a BRDF interface (that is, they calculate the light reflected at a single point from an incident direction to an outgoing direction). Because the BSSRDF needs to consider incident light at points all around the point being shaded, I implemented it as a surface integrator. In pbrt, surface integrators perform the task of determining the color of a pixel, given the information about the scene and a ray from the camera that is cast into the scene.

There are two preprocessing steps that must be completed before rendering with the subsurface scattering integrator. The first is calculating the reflectance profile that comes from merging several layers of varying reflectance parameters. The second is calculating where to take irradiance samples that will be used to determine the light that reaches each point through subsurface scattering.

Calculating Reflectance Profile

The single-layer BSSRDF proposed by Jensen et al. [6] approximates each incident light ray as a diffuse dipole light source, with the positive light source beneath the surface and the negative light source above the surface. They are positioned so that

the net inward diffuse radiance is zero along the surface of the skin, which results in diffuse light being emanated from the surface at points near the incident ray.

This approximation ignores light that is transmitted all the way through the layer, so Donner and Jensen [1] improved it to handle thin slabs of translucent material. The improvement basically consists of adding a second dipole below the bottom boundary of the layer so the net inward diffuse radiance is also zero along the bottom surface. However, this causes a change at the top boundary, so a third dipole is added above the first, which necessitates a fourth dipole below the second, etc. Each additional dipole disturbs the balance at the opposite boundary. However, because each successive dipole is placed farther from the layer, their effects diminish, and with a sufficiently large number of dipoles, a good balance can be struck on both sides of the layer.

Donner and Jensen provide an equation for the reflectance and transmittance profiles ($R(r)$ and $T(r)$) of a layer, which describe for an incident light ray, how much is reflected or transmitted at a distance r from the point of incidence. Both of these profiles are calculated by summing the contribution of the dipoles at the appropriate point on either the top or bottom surface of the layer.

In order to combine multiple layers, the profiles must be convolved. For example, the transmittance of a two-layer material is given by $T_{12} \approx T_1 * T_2$. This is only an approximation because it ignores light that is reflected between the layers before being transmitted. That is, light can be transmitted through layer 1, then reflected back up by layer 2, then reflected back down by layer 1, and finally transmitted through layer 2. In fact, there is an infinite sum of possible interactions that can cause transmission through the two layers, taking the form $T_{12} = (T_1 * T_2) + (T_1 * R_2 * R_1 * T_2) + (T_1 * R_2 * R_1 * R_2 * R_1 * T_2) + ...$ If we take the Fourier transform, then we can think of this as an infinite sum of products, rather than an infinite sum of convolutions. In fact, the infinite sum is a geometric series, which reduces to $\mathcal{T}_{12} = \frac{\mathcal{T}_1\mathcal{T}_2}{1-\mathcal{R}_2\mathcal{R}_1}$. By a similar analysis, the reflectance profile of two layers can be merged to get $\mathcal{R}_{12} = \mathcal{R}_1 + \frac{\mathcal{T}_1\mathcal{R}_2\mathcal{T}_1}{1-\mathcal{R}_2\mathcal{R}_1}$. After this calculation is performed, you can simply take the inverse Fourier transform to get $R_{12}(r)$ and $T_{12}(r)$.

In order to render with the multi-layer BSSRDF model, I create the "layer file",

which is simply a file that holds the value of the reflectance profile $R(r)$ for discrete values of r from 0 to some configurable threshold. Generating the layer file is a one-time cost, and does not need to be done for each rendering. To calculate the reflectance profile for a multi-layer material, I first calculate $R(r)$ and $T(r)$ for each of the layers that make up the material individually, using the equation given by Donner and Jensen. I then compute the discrete Fourier transform of each reflectance and transmittance profile using fftw [2], a publically available C library for computing the discrete Fourier transform. Next, I calculate the reflectance and transmittance profiles of the merged layers, using the method described above. (For more than two layers, I combine the top two, then combine that one with the third, and so on.) Finally, I take the inverse Fourier transform, again using fftw, and write the resulting reflectance profile out to the layer file.

Calculating Irradiance Samples

In order to determine the color of a particular point, you need to take into account the incident light at many points around the point being shaded. While it would be possible to take many samples for every single point that is shaded, it is much more efficient to take a large number of samples at the outset, and cache them for later use.

In order to ensure that the sample points are well-distributed around the entire mesh, I used an implementation of Turk's point repulsion algorithm, in the form of the Uniform Surface Sampler, provided to me by Tom Mertens. I use this application to generate a collection of points that are evenly spread around the mesh and then write their locations to the "sample file" for use during rendering. Like the layer file, creating the sample file is a one-time cost for a particular mesh, and does not affect the rendering time.

After the points for the samples have been selected, the irradiance must actually be sampled at each of those points. This step must be done for each new rendering, because it depends upon elements of the scene besides the mesh, such as the position of lights. Before the actual rendering begins, I iterate through all of the points in the

sample file and measure the amount of light that reaches each point. I store this value in an octree data structure as described by Jensen and Buhler [5]. An octree is a tree data structure in which each node represents some bounding box in 3D space. Each interior node has 8 children, which represent the 8 octants of the space covered by the parent's bounding box. When a sample is added to the octree, it travels down the hierarchy of bounding boxes until it reaches a leaf node whose bounding box contains the point from which the sample was taken. It is then added to that leaf node. If the leaf node then contains more than some threshold number of samples, it is split into 8 octants, which become the 8 new leaf nodes below that node. This octree is used during rendering because it makes it fairly quick to locate all of the irradiance samples that are near any given point.

Calculating Shading for a Point

For each point that is shaded, I use the octree to find all of the irradiance samples that are within some distance of that point (where the distance is determined by the range of data from the layer file). For each of those samples, I sum the irradiance at the sample point times the reflectance from the layer file at the distance from the sample point to the point being shaded. Finally, I normalize this value by the total reflectance of all of the sample points within range. This ensures that the points very near irradiance samples are not too bright, and that points that are not very near any irradiance sample are not too dark. An illustration of this process is given in Figure 3-4.

3.3.4 Asperity Scattering

The asperity scattering BRDF does not represent skin per se, just a thin layer of sparse scatterers that cover the skin. Therefore, it can not be used as a material in and of itself, but rather as a term that can be added to another material. Rather than trying to actually represent tiny hairs that cause scattering of light near the surface of skin, the asperity scattering approximation suggested by Koenderink and Pont [7]

(a) (b)

Figure 3-4: Before rendering, the irradiance is sampled and cached at many points covering the surface of the skin (a). When a point is being shaded, the contributions of all of the points within range are summed (b).

uses a simple model to get the correct visual effect. The basic idea is that tiny surface scatterers do not do much individually, it is only when light passes through many of them that the soft glow is visible.

The asperity scattering model uses three parameters to represent the thin layer of scatterers: the thickness of the layer, Δ, the volume density of the scatterers, ϱ, and the total scattering cross section, σ_t. The mean free path of a photon through the scattering layer is defined as $\lambda = \frac{1}{\sigma_t \varrho}$. Assuming isotropic scatterers, the BRDF is given by $f(\theta_i, \phi_i, \theta_r, \phi_r) = \frac{\Delta}{4\pi\lambda \cos\theta_i \cos\theta_r}$. The inverse cosine factors suggest that whenever the light or viewing angles approach grazing, the light is reflected the most, whereas when the light and viewing angle are perpendicular to the surface, the effect is negligible. (See Figure 3-5.) In actuality, the approximation breaks down as the light or viewing angle approaches $\frac{\pi}{2}$, because the asperity scattering term approaches infinity. This is because the approximation doesn't take into account the fact that a curving surface will cause the light ray to only pass through a small number of scatterers, even if it is at a grazing angle. To correct this, I put a lower-bound on both cosine terms, so they cannot go lower than $\frac{\Delta}{\lambda}$.

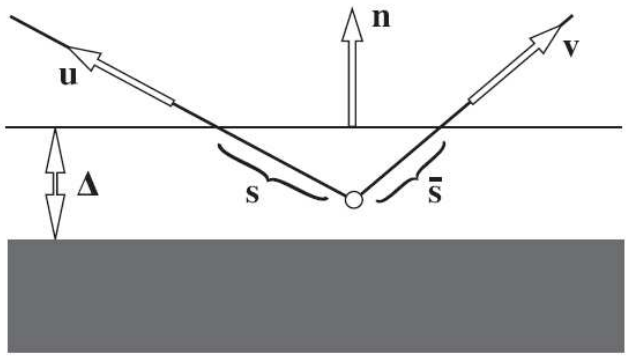

Figure 3-5: The degree of scattering that takes place is directly related to how long the light is travelling through the thin layer of scatterers, which in turn is affected by the angles of incidence and reflection. (Taken from [7]).

Chapter 4

Results

In this chapter, I will present the images I rendered, along with analysis of image quality and rendering time. All of these images were rendered at a resolution of 800x800, on a dual core Intel Xeon 2.4 GHz machine with 2 GB of RAM.

4.1 High Quality Mesh

4.1.1 Directional Light

Figure 4-1 shows the first four images, each using the 1000000-triangle mesh and directional lighting, but with different local illumination models. This provides a good baseline because the lighting is simple and illuminates the entire face, and because I used the high quality mesh, there are no artifacts from bump mapping.

The measured BRDF does not do particularly well here. Compared to the models that take subsurface scattering into account, it is too shiny, especially on the forehead and ears, which leads to a fake, waxy appearance. The asperity scattering model also suffers from some shininess, although around the edges of the face and the sides of the nose, there is a slight glow that suggests a thin layer of scatterers on top of the skin. However, the asperity scattering model has some difficulty with the shadows above the eyes. Because the asperity scattering term makes everywhere that the light touches a little brighter, there is a rather harsh transition into the shadowed region

Reflectance Model	Rendering Time
Measured BRDF	4:03
Asperity Scattering	3:46
Hanrahan-Krueger BRDF	6:05
Multi-layer BSSRDF	10:42

Table 4.1: Rendering times for the 1000000-triangle mesh with directional lighting.

that does not appear with the other models. The Hanrahan-Krueger image does not exhibit the same shininess as the measured BRDF. The light is spread more diffusely across the face, leading to a softer appearance. Finally, the multi-layer BSSRDF image is rather similar to the Hanrahan-Krueger one, but a difference is noticeable around the shadow boundaries, especially in the nostrils and ears. Because of the subsurface scattering taking place, some color bleeds into the shadows, creating a softer appearance.

Table 4.1 gives the rendering times for the four images in Figure 4-1. As could be expected, the simple BRDF models took the least amount of time, while the more complicated Hanrahan-Krueger BRDF took slightly more, and the BSSRDF took the most. The BSSRDF must take all of its irradiance samples before even beginning to calculate shading, making it significantly slower than the other models.

4.1.2 Point and Area Lights

Figures 4-2 and 4-3 show the high quality mesh rendered with point and area light sources, respectively. Much like in the images with the directional light sources, the measured BRDF image appears too shiny to be real. Especially in the area light source image, there are specular highlights on some parts of the face, while others parts are too dark. The Hanrahan-Krueger BRDF and multi-layer BSSRDF both diffuse the light more evenly across the surface, leading to a softer appearance. The asperity scattering model does not do very well with the point light source. Because of the position of the light within the scene, much of the face is illuminated at a slightly grazing angle, which causes the face to seem washed out. Also, once again,

Reflectance Model	Point Light	Area Light
Measured BRDF	3:01	28:56
Asperity Scattering	2:55	19:45
Hanrahan-Krueger BRDF	4:03	1:08:13
Multi-layer BSSRDF	10:32	19:17

Table 4.2: Rendering times for the 1000000-triangle mesh with a point and with an area light source.

the shadow boundaries are much too harsh. However, the asperity scattering model does a significantly better job with the area light source. The softening of the shadows by the area light source makes a real difference in the quality of the image, more so than with the other local illumination models.

The multi-layer BSSRDF suffers from some slight noise problems with the area light source. Normally, when rendering with an area light source, every point that is shaded casts many shadow rays towards different points on the surface of the light, to determine what percentage of the light source is illuminating that point. If some of the points happen to get an incorrent percentage, it does not make too big of a difference to the image as a whole. However, with the multi-layer BSSRDF, all of the irradiance samples are taken before rendering and cached, so any inaccuracies are not used just once, but for every point near that irradiance sample. Therefore, it is necessary to take more shadow samples when using the multi-layer BSSRDF model in order to get a good image. I used 64 shadow samples with the BSSRDF model, and only 16 with the other local illumination models, to achieve similar results.

Table 4.2 gives the rendering times for the images in Figures 4-2 and 4-3. The point light source times follow much the same pattern as the directional light. However, with the area light source, the multi-layer BSSRDF model is the fastest. This is because of the shadow sampling described above. Even though the BSSRDF uses four times as many shadow samples as the other models, this must only be done for the irradiance samples taken at the beginning. Those values are then cached and used when shading each pixel. With the simple light sources, the extra step of taking irradiance samples was a liability to rendering time, but with a light source that

Reflectance Model	Rendering Time
Measured BRDF	40:14
Asperity Scattering	34:19
Hanrahan-Krueger BRDF	1:12:27
Multi-layer BSSRDF	28:38

Table 4.3: Rendering times for the 1000000-triangle mesh with an environment map.

must be sampled many times, the caching actually leads to a performance increase. However, as noted above, the trade-off is greater noise in the image, which necessitates taking more shadow samples when using the multi-layer BSSRDF model.

4.1.3 Environment Map

Figure 4-4 shows the high quality mesh rendered with an environment map. Once again, the measured BRDF does not do a good job diffusing the light across the surface of the skin. It winds up looking too dark and flat. The asperity scattering model, on the other hand, does a fairly good job with the subtle light of the environment map. With the other light sources it was sometimes washed out, but with the environment map it has good color and exhibits highlights around the edges of the face and the side of the nose, just as you would expect. The Hanrahan-Krueger and multi-layer BSSRDF models perform similarly, both doing a fairly good job, but with the BSSRDF suffering from the same slight noise problems as with the area light source.

Table 4.3 gives the rendering times for the images in Figure 4-4. Just as with the area light source, I used 16 shadow samples for the three BRDF-based local illumination models, and 64 with the BSSRDF model. Once again, because of the cached irradiance samples, the BSSRDF model rendered faster than the other three, even with four times as many shadow samples.

4.2 Low Quality Mesh

Because the results for the low quality mesh do not differ too substantially from those for the high quality mesh, I will not present an analysis of each image, but rather only call attention to the ways in which these images differ from the results for the high quality mesh.

4.2.1 Sharp Bump Map

Figures 4-5, 4-6, 4-7, and 4-8 show the low quality mesh rendered with the non-blurred bump map and directional lights, a point light, an area light, and an environment map, respectively.

One quite noticeable difference between the low and high quality meshes is in the shadow boundaries. Because of the lower resolution of the geometry, the shadows are not smoothly shaped, but rather jagged. This is especially noticeable with the asperity scattering model, because of the harsh shadow boundaries. This shows up in the shadows above the eyes in the directional lighting scenes, as well as on the nose in the point and area light source scenes. The environment map, which does not have any completely shadowed regions except the nostils, does not have this problem.

The ears also show artifacts of the low triangle count. The edges of the ears have some jagged lines under every type of light, but the problem is most pronounced with the directional light source.

The bump map does a fair job of approximating more detailed geometry, except in the multi-layer BSSRDF. Because the subsurface scattering tends to blend and smooth colors, the effect of the bump map with this model is practically non-existant. However, the BSSRDF also helps to smooth out some of the geometry in the ears, and the shadows above the eyes in the directional lighting image.

The environment map images also do not show much surface variation as a result of the bump map. This is because the light is coming from all directions, so varying the surface normal does not significantly affect the angle between the normal and the light direction. The environment map images for the high quality mesh don't show

Reflectance Model	Directional	Point	Area	Environment Map
Measured BRDF	4:33	3:53	23:01	30:48
Asperity Scattering	4:17	3:45	13:29	23:54
Hanrahan-Krueger BRDF	6:31	4:52	1:02:14	1:11:41
Multi-layer BSSRDF	9:49	9:40	16:08	22:51

Table 4.4: Rendering times for the 16750-triangle mesh.

too much surface variation either, even though the geometry there actually is bumpy.

Table 4.4 gives the rendering times for all of the images using the low quality mesh. The times follow the same patterns as those for the high quality mesh, for the same reasons that were mentioned previously.

4.2.2 Blurred Bump Map

Figures 4-9, 4-10, 4-11, and 4-12 show the low quality mesh rendered with the blurred bump map and directional lights, a point light, an area light, and an environment map, respectively. Using a blurred bump map does not have any real affect on rendering time, so the rendering times for these images are approximately the same as those given in Table 4.4.

In general, the blurred bump map does exactly what you might expect – it makes the surface variation smoother than with the sharp bump map. This is especially noticeable on the left side of the face for the point and area light sources. The blurred bump map images still show the surface variation, but it is not so dramatic. This slightly fuzzier property can make the skin appear more soft.

On the other hand, the blurred bump map does not really make any difference for the images with environment map lighting or with the multi-layer BSSRDF local illumination model. Because these two features tend to reduce the impact of the bump map anyway, the difference between a sharp and blurred bump map is negligible.

4.3 Conclusions

In general, the multi-layer BSSRDF tends to produce the best images. This is not really surprising, as it is the model that most accurately simulates the physical properties of skin. However, it is interesting to note that because of the caching of irradiance samples, it can also be the most efficient model to use, if you are using a light source that requires many shadow samples. However, when using such a light source, it is important to use enough samples, because more are required than would be with a model that does not cache irradiance values.

Using a simple Lafortune-based measured BRDF does not seem to yield very good results. It simply does not reproduce the complex reflectance properties of skin. On the other hand, the Hanrahan-Krueger BRDF can produce some pretty good images. While it does not provide the color bleeding of true subsurface scattering, it does do a good job of diffusing the light across the surface of the skin.

The asperity scattering technique can produce some nice effects, but is sensitive to the type of light source used. It does not look good at all with hard shadow boundaries. However, it can yield rather nice results with environment map lighting.

While none of the local illumination models or light sources were completely effective at hiding artifacts from a low quality input mesh, the multi-layer BSSRDF and the environment map lighting both managed to smooth out some of the problems. Unfortunately, they both also had the unintended effect of smoothing out the surface variation from the bump map. Finally, blurring the bump map before rendering is a quick and easy way to make the surface appear softer in some cases, but does not make a noticeable difference when using an environment map or the multi-layer BSSRDF model.

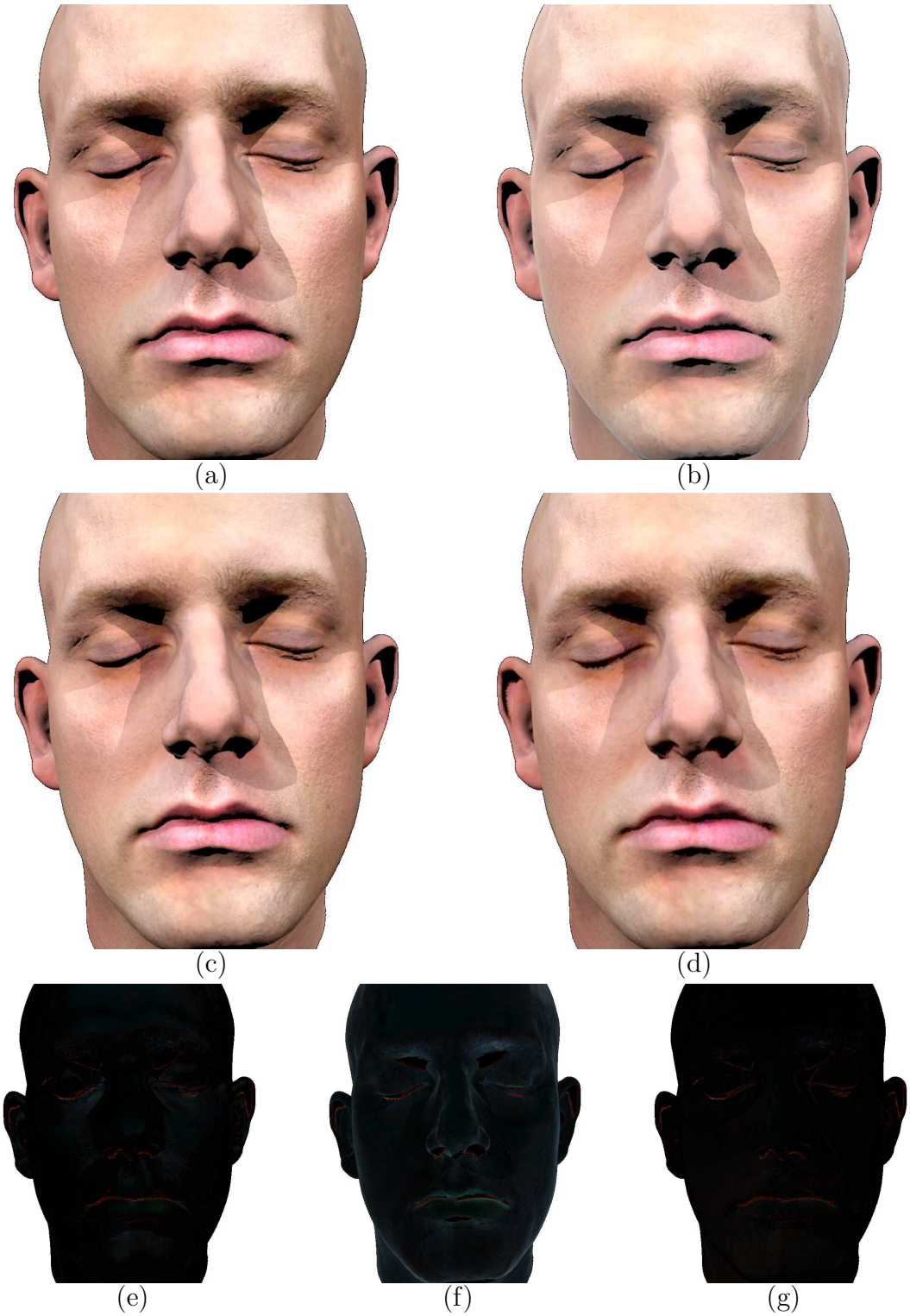

Figure 4-1: The high quality mesh rendered with directional lighting. (a) The measured BRDF. (b) The asperity scattering model. (c) The Hanrahan-Krueger BRDF. (d) The multi-layer BSSRDF. The bottom row shows difference images between the multi-layer BSSRDF and the measured BRDF (e), the asperity scattering model (f), and the Hanrahan-Krueger BRDF (g).

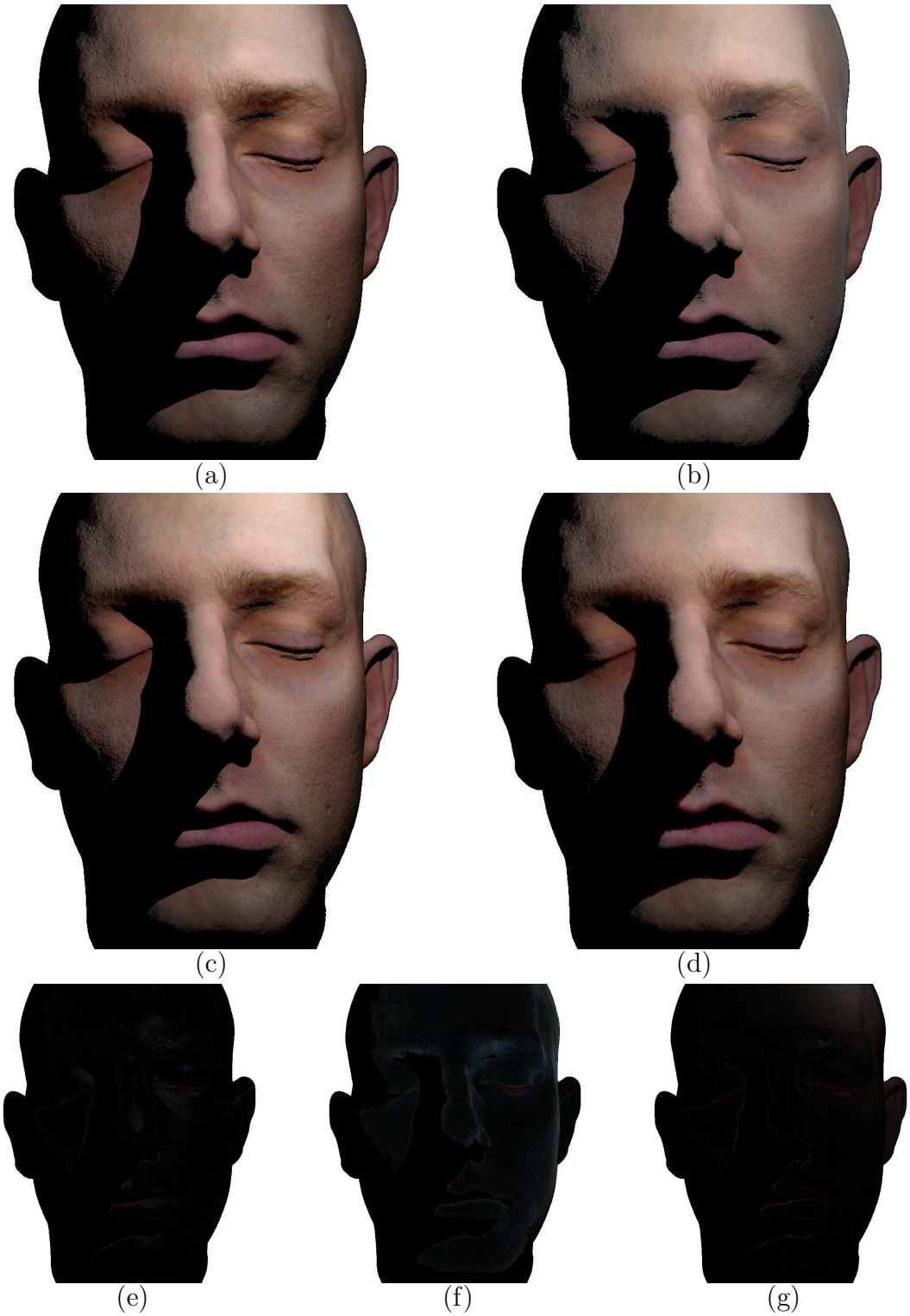

Figure 4-2: The high quality mesh rendered with a point light source. (a) The measured BRDF. (b) The asperity scattering model. (c) The Hanrahan-Krueger BRDF. (d) The multi-layer BSSRDF. The bottom row shows difference images between the multi-layer BSSRDF and the measured BRDF (e), the asperity scattering model (f), and the Hanrahan-Krueger BRDF (g).

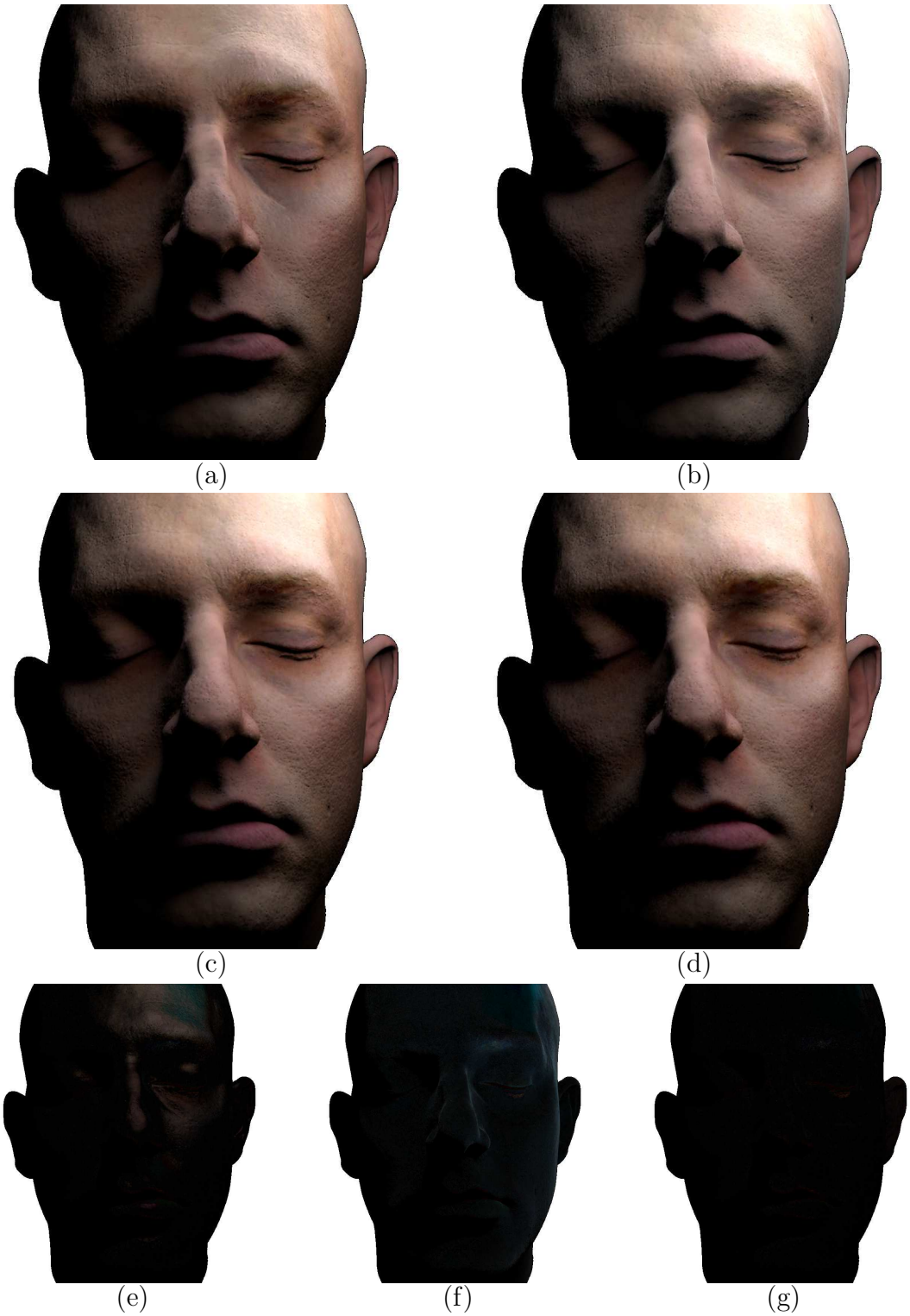

Figure 4-3: The high quality mesh rendered with an area light source. (a) The measured BRDF. (b) The asperity scattering model. (c) The Hanrahan-Krueger BRDF. (d) The multi-layer BSSRDF. The bottom row shows difference images between the multi-layer BSSRDF and the measured BRDF (e), the asperity scattering model (f), and the Hanrahan-Krueger BRDF (g).

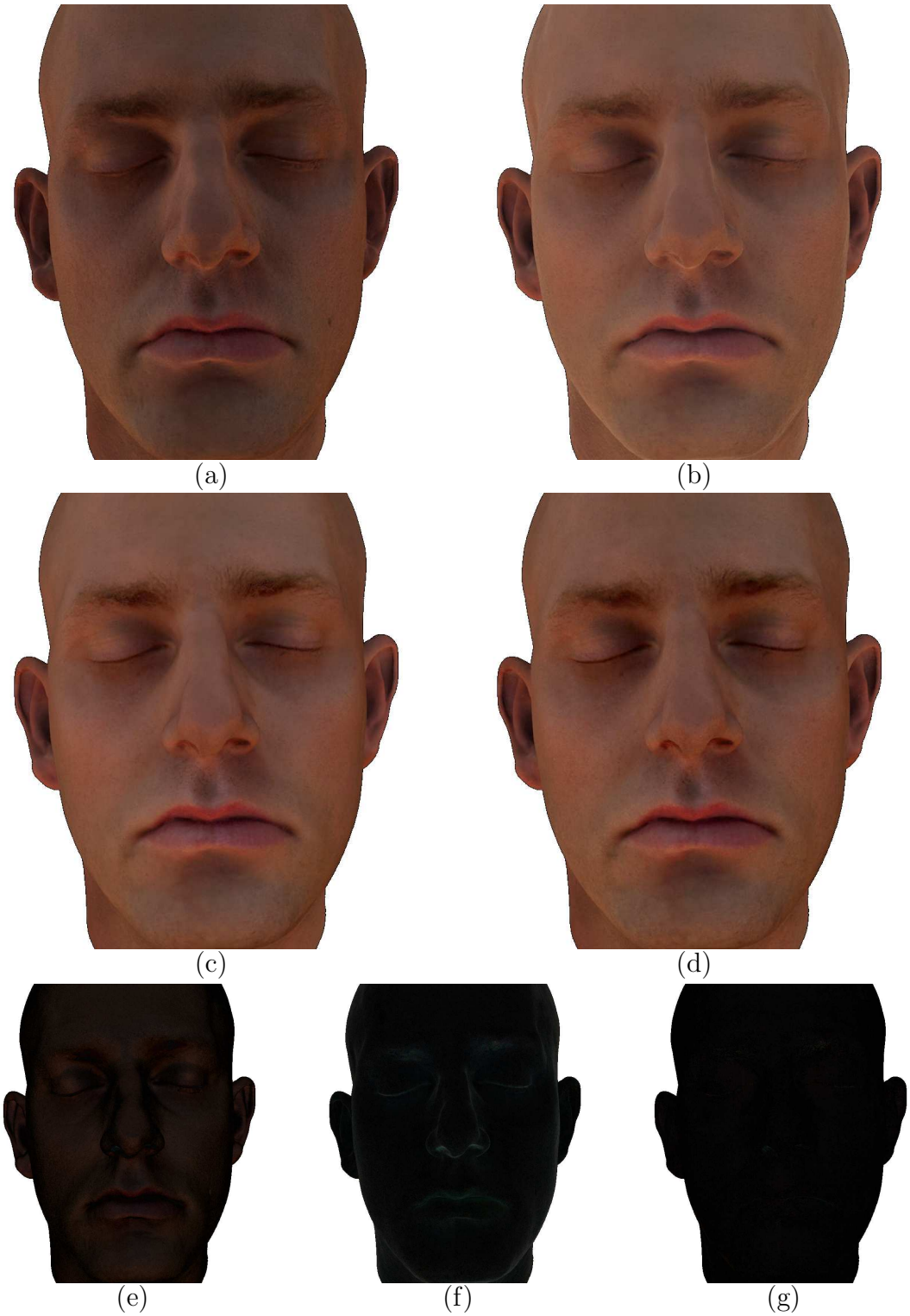

Figure 4-4: The high quality mesh rendered with an environment map simulating a sunset. (a) The measured BRDF. (b) The asperity scattering model. (c) The Hanrahan-Krueger BRDF. (d) The multi-layer BSSRDF. The bottom row shows difference images between the multi-layer BSSRDF and the measured BRDF (e), the asperity scattering model (f), and the Hanrahan-Krueger BRDF (g).

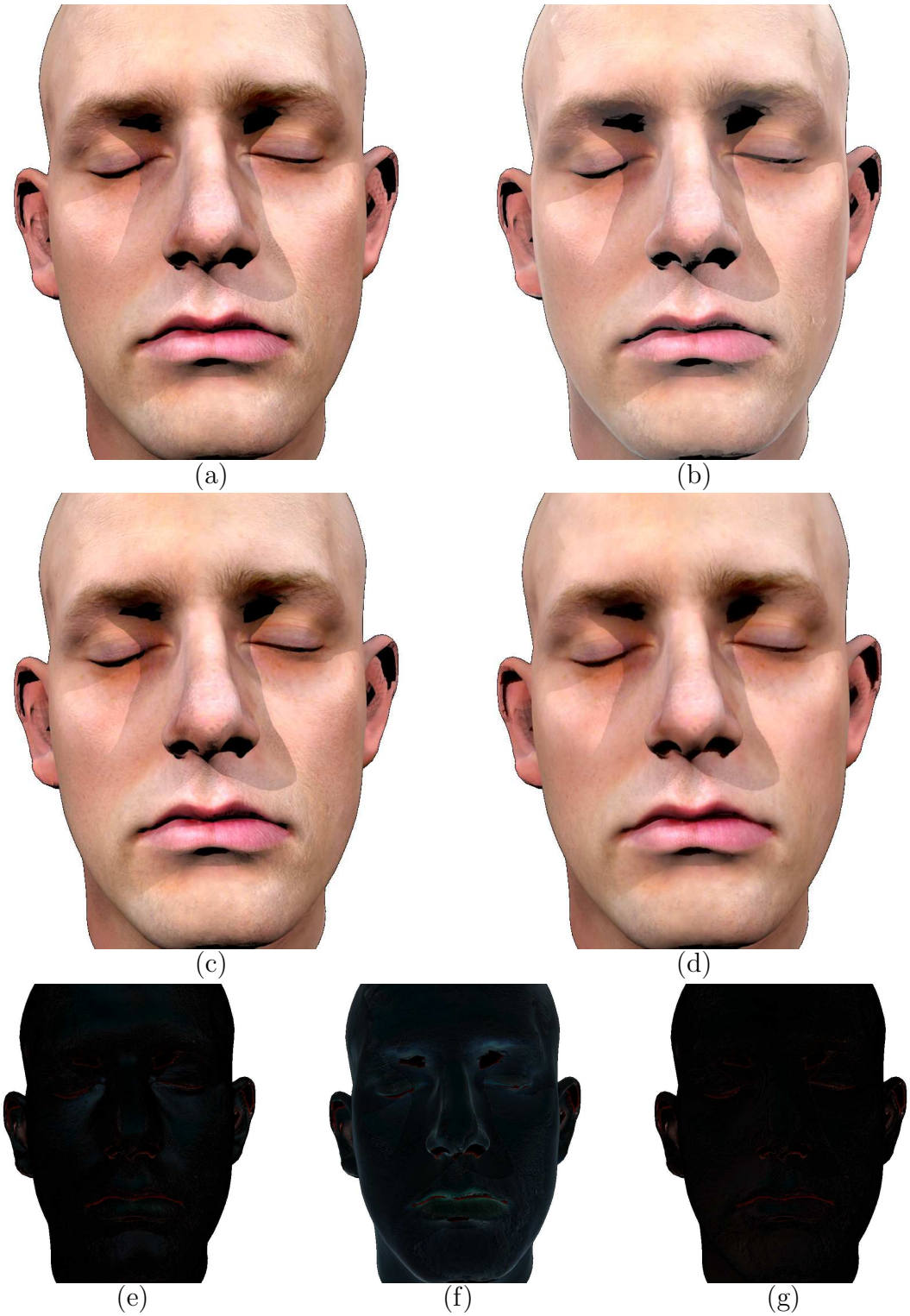

Figure 4-5: The low quality mesh rendered with the sharp bump map and directional lighting. (a) The measured BRDF. (b) The asperity scattering model. (c) The Hanrahan-Krueger BRDF. (d) The multi-layer BSSRDF. The bottom row shows difference images between the multi-layer BSSRDF and the measured BRDF (e), the asperity scattering model (f), and the Hanrahan-Krueger BRDF (g).

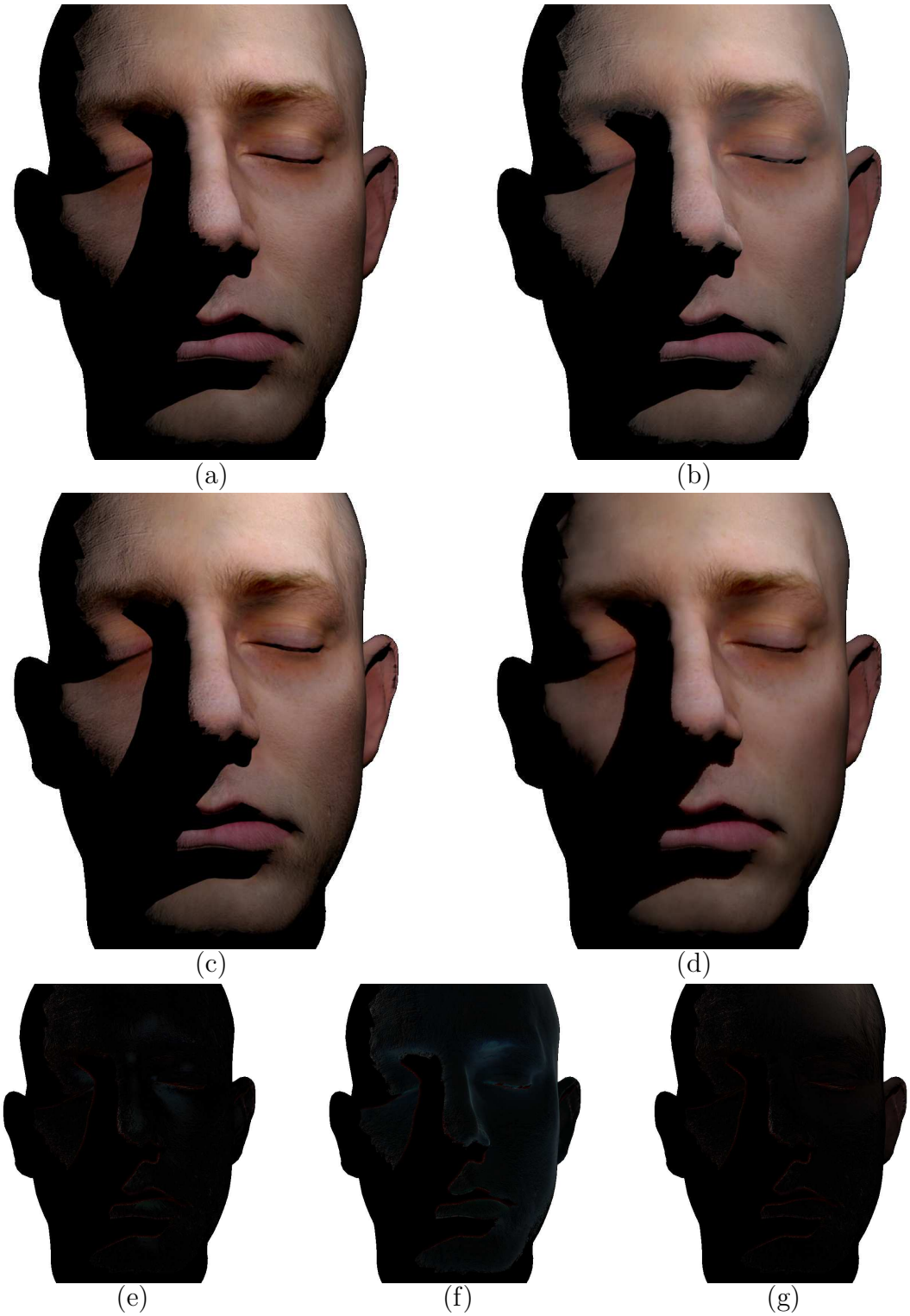

Figure 4-6: The low quality mesh rendered with the sharp bump map and a point light source. (a) The measured BRDF. (b) The asperity scattering model. (c) The Hanrahan-Krueger BRDF. (d) The multi-layer BSSRDF. The bottom row shows difference images between the multi-layer BSSRDF and the measured BRDF (e), the asperity scattering model (f), and the Hanrahan-Krueger BRDF (g).

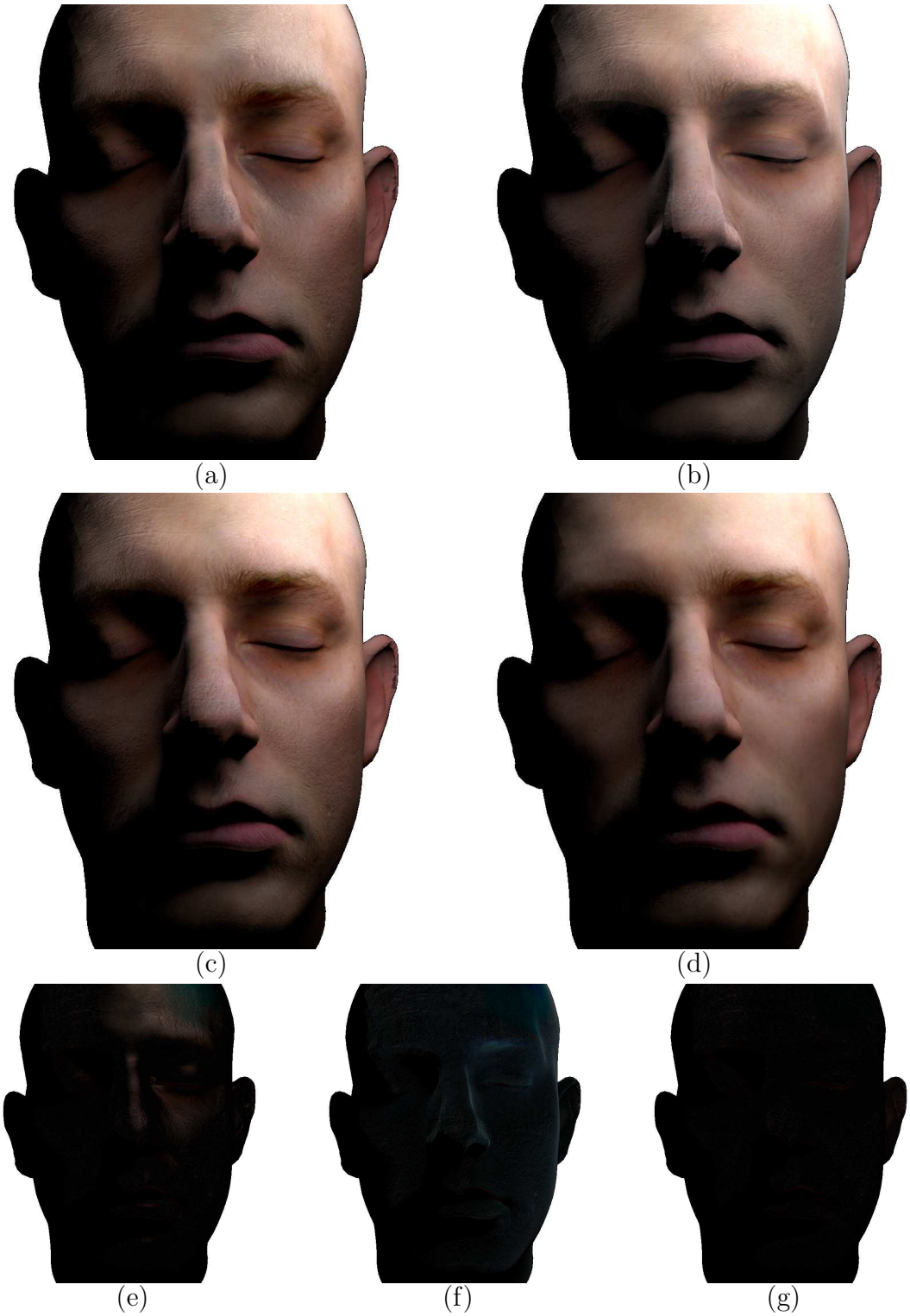

Figure 4-7: The low quality mesh rendered with the sharp bump map and an area light source. (a) The measured BRDF. (b) The asperity scattering model. (c) The Hanrahan-Krueger BRDF. (d) The multi-layer BSSRDF. The bottom row shows difference images between the multi-layer BSSRDF and the measured BRDF (e), the asperity scattering model (f), and the Hanrahan-Krueger BRDF (g).

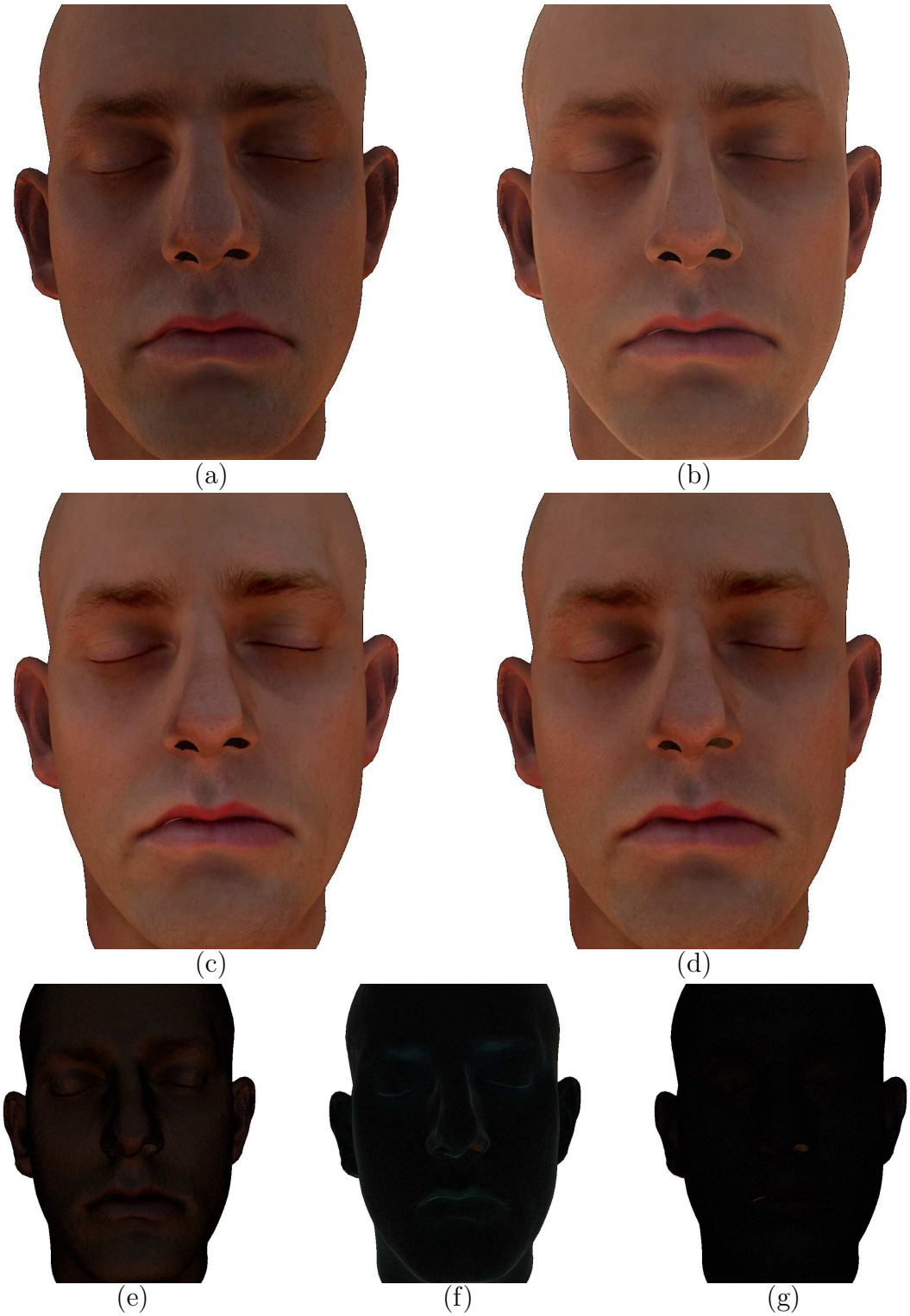

Figure 4-8: The low quality mesh rendered with the sharp bump map and an environment map simulating a sunset. (a) The measured BRDF. (b) The asperity scattering model. (c) The Hanrahan-Krueger BRDF. (d) The multi-layer BSSRDF. The bottom row shows difference images between the multi-layer BSSRDF and the measured BRDF (e), the asperity scattering model (f), and the Hanrahan-Krueger BRDF (g).

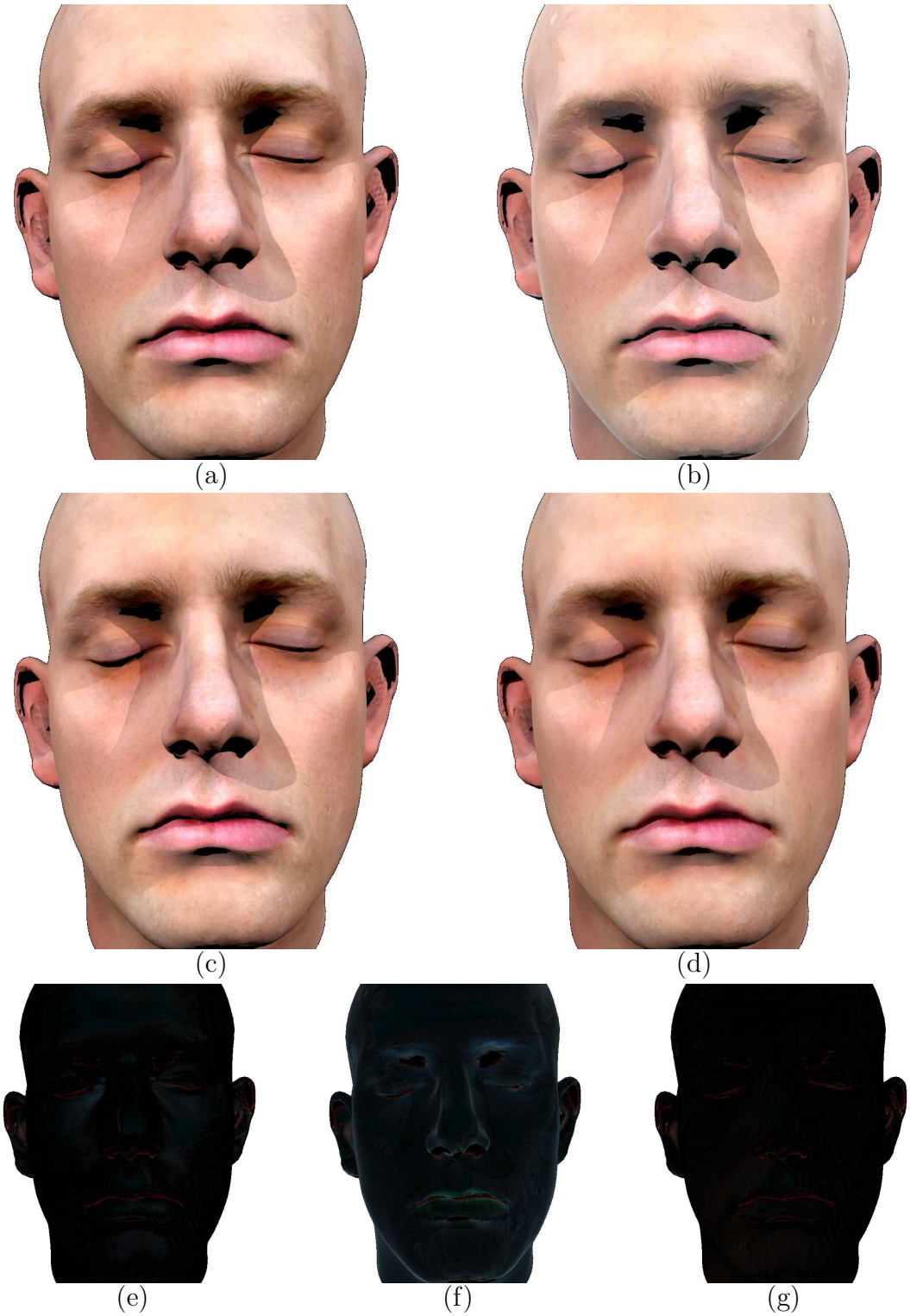

Figure 4-9: The low quality mesh rendered with the blurred bump map and directional lighting. (a) The measured BRDF. (b) The asperity scattering model. (c) The Hanrahan-Krueger BRDF. (d) The multi-layer BSSRDF. The bottom row shows difference images between the multi-layer BSSRDF and the measured BRDF (e), the asperity scattering model (f), and the Hanrahan-Krueger BRDF (g).

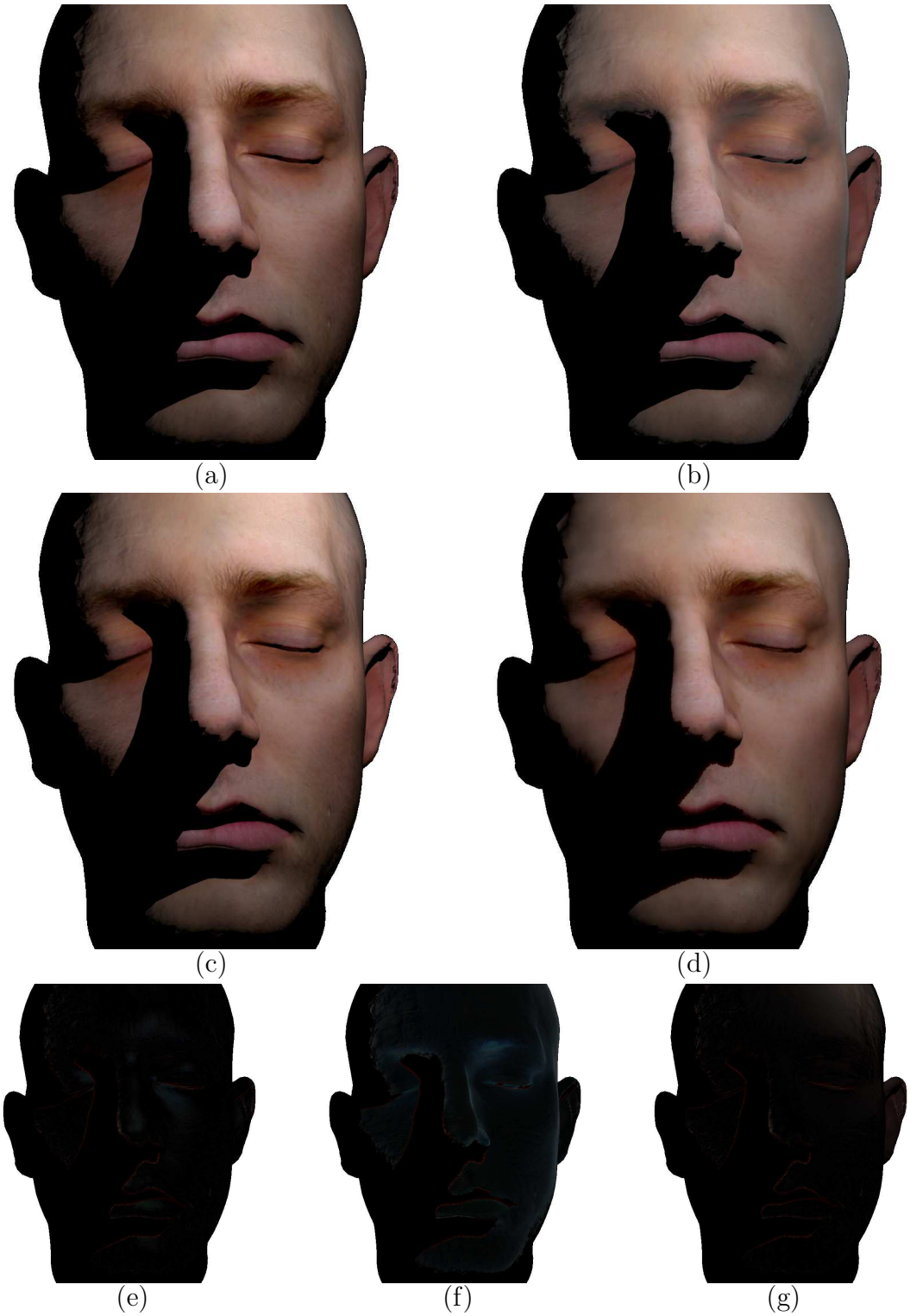

Figure 4-10: The low quality mesh rendered with the blurred bump map and a point light source. (a) The measured BRDF. (b) The asperity scattering model. (c) The Hanrahan-Krueger BRDF. (d) The multi-layer BSSRDF. The bottom row shows difference images between the multi-layer BSSRDF and the measured BRDF (e), the asperity scattering model (f), and the Hanrahan-Krueger BRDF (g).

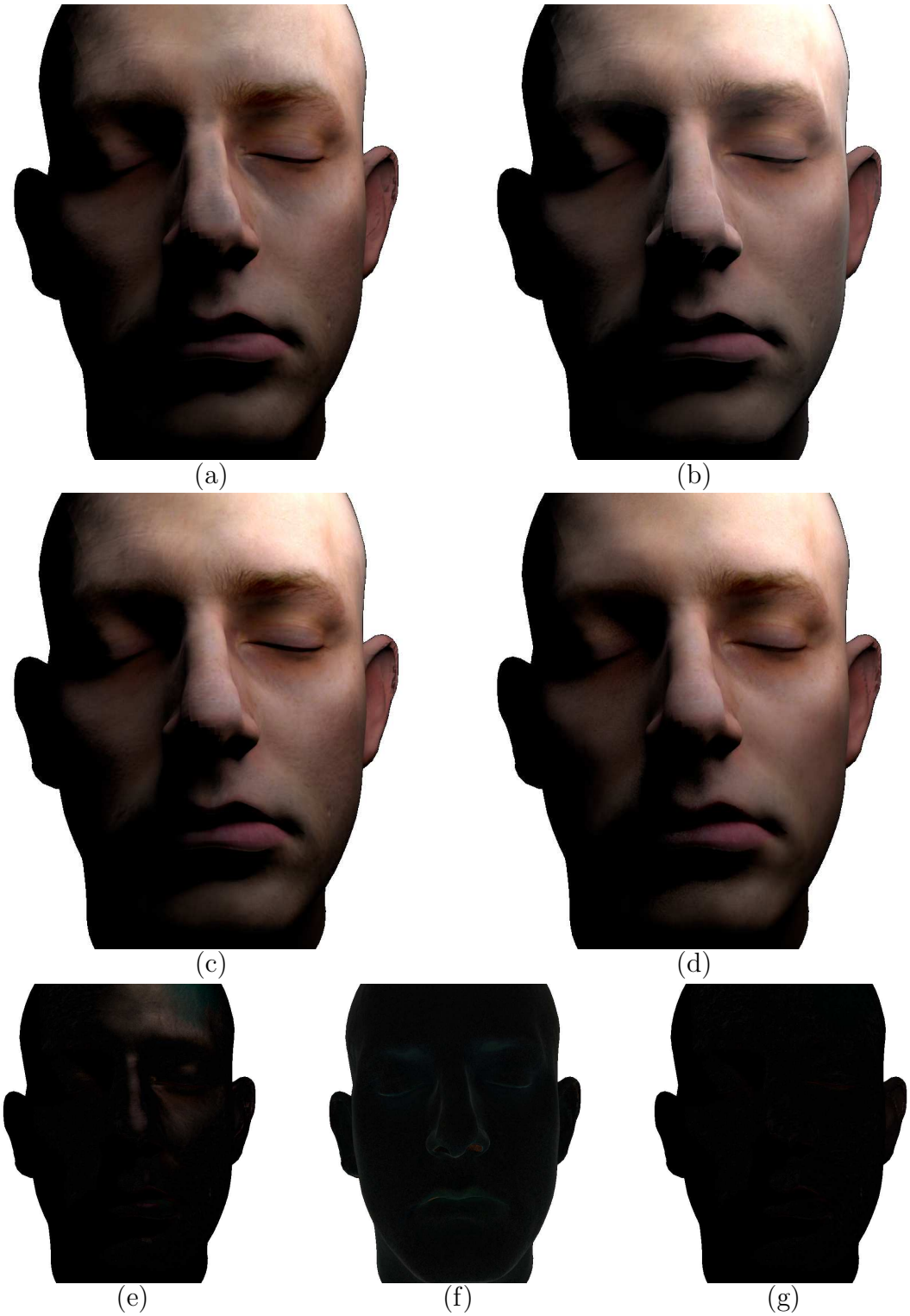

Figure 4-11: The low quality mesh rendered with the blurred bump map and an area light source. (a) The measured BRDF. (b) The asperity scattering model. (c) The Hanrahan-Krueger BRDF. (d) The multi-layer BSSRDF. The bottom row shows difference images between the multi-layer BSSRDF and the measured BRDF (e), the asperity scattering model (f), and the Hanrahan-Krueger BRDF (g).

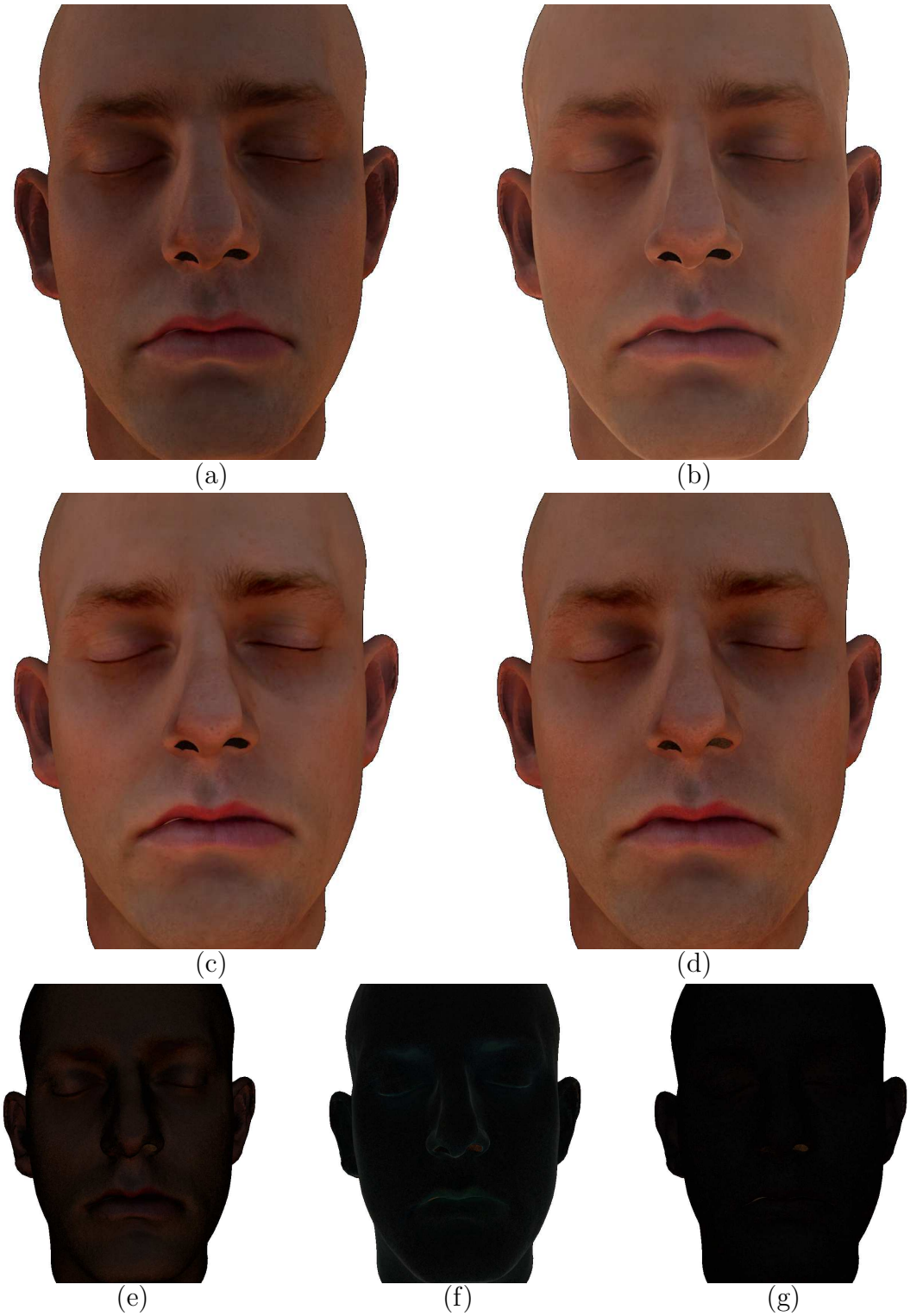

Figure 4-12: The low quality mesh rendered with the blurred bump map and an environment map simulating a sunset. (a) The measured BRDF. (b) The asperity scattering model. (c) The Hanrahan-Krueger BRDF. (d) The multi-layer BSSRDF. The bottom row shows difference images between the multi-layer BSSRDF and the measured BRDF (e), the asperity scattering model (f), and the Hanrahan-Krueger BRDF (g).

Bibliography

[1] Craig Donner and Henrik Wann Jensen. Light diffusion in multi-layered translucent materials. In *Proceedings of SIGGRAPH 2005*, 2005.

[2] Matteo Frigo and Steven G. Johnson. Fftw: Fastest fourier transform in the west. http://www.fftw.org.

[3] Michael Garland. *Quadric-Based Polygonal Surface Simplification*. PhD thesis, Carnegie Mellon University, 1999.

[4] Pat Hanrahan and Wolfgang Krueger. Reflection from layered surfaces due to subsurface scattering. In *Proceedings of SIGGRAPH 1993*, 1993.

[5] Henrik Wann Jensen and Juan Buhler. A rapid hierarchical rendering technique for translucent materials. In *Proceedings of SIGGRAPH 2002*, 2002.

[6] Henrik Wann Jensen, Stephen R. Marschner, Marc Levoy, and Pat Hanrahan. A practical model for subsurface light transport. In *Proceedings of SIGGRAPH 2001*, 2001.

[7] Jan Koenderink and Sylvia Pont. The secret of velvety skin. *Machine Vision and Applications*, 14(4):260–268, September 2003.

[8] Eric P. F. Lafortune, Sing-Choong Foo, Kenneth E. Torrance, and Donald P. Greenberg. Non-linear approximation of reflectance functions. In *Proceedings of SIGGRAPH 1997*, 1997.

[9] Matt Pharr and Greg Humphreys. *Physically Based Rendering from Theory to Implementation*. Morgan Kaufmann Publishers, 2004.

[10] Greg Turk. Re-tiling polygonal surfaces. In *Proceedings of SIGGRAPH 1992*, 1992.