/*jupiter jazz*/

Jupiter SGI Skin

Jupiter Jazz Spherical Gradient Illumination Skin Shader, for 3Delight RenderMan

Version 1.0, April 2010

OpenSourced at: http://code.google.com/p/jupiterskin/

Sponsor:

This work has been sponsored by Srinivas M. Mohan (Indian Artists Computer Graphics Pvt. Ltd.) for the movie "Endhiran".

Authors:

Luis B. Barrancos, Moritz Moeller, Paolo Berto

Contact:

software@jupiter-jazz.com

Based on:

"Rapid Acquisition of Specular and Diffuse Normal Maps from Polarized Spherical Gradient Illumination".
By Wan Chun Ma, Tim Hawkins, Pieter Peers, Charles-Fe lix Chabert, Malte Weiss, Paul Debevec.
University of Southern California Institute for Creative Technologies.
Eurographics Symposium on Rendering 2007.

"Estimating Specular Roughness and Anisotropy from Second Order Spherical Gradient Illumination".
By Abhijeet Ghosh , Tongbo Chen, Pieters Peers, Cyrus A. Wilson, Paul Debevec.
University of Southern California Institute for Creative Technologies.
Eurographics Symposium on Rendering 2009.

OPENSOURCE LICENSE

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name ofJupiter Jazz Ltd. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL JUPITER JAZZ LTD BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Jupiter Skin

Jupiter SGI Skin is a shader for 3Delight RenderMan that reads scanned RGB world space normal maps for 3 different wavelengths (red, green, blue), and a fourth normal map to be used in the specular term.

This technique, presented at:

http://gl.ict.usc.edu/Research/FaceScanning/ ,

allows the user to re-create the appear of the subtle subsurface effects of skin, together with self-shadowing and inter-reflections, since the three normal maps for the Lambertian diffuse term, contain increasing detail as the light wavelength decreases - the normal map for the blue wavelength contains more detail than the normal map for the green wavelength, which in turn contains more detail than the red wavelength map.

The specular normal map is used to drive the specular contribution (and the reflections), with the specular term being comprised of two specular lobes based on the Torrance-Sparrow model. The user has the possibility of selecting between two microfacet distribution functions used, either the Beckmann microfacet distribution as suggested in the publication by Robert L. Cook and Kenneth E. Torrance, "A reflectance model for computer graphics", or a Gaussian distribution, which is cheaper to compute and produces slightly softer results. The weight of each of the specular lobe's contribution can also be controlled directly by the user, or it can be made automatically via the extracted data in the specular intensity image.

Notice also that this shader provides optionally a full two pass (point cloud based) subsurface term as well for greater flexibility.

Usage Notes

The shader comes with a good set of default values, but some user intervention is required to allow the best results to be achieved. There are some particular points worth considering. First, some information about the type of data that the shader expects, or the type of the data that the user might have. The scanned data can be acquired with the USC ICT Light Stage scanning system. Typically this might consist of four world space normal maps, one for each of the red, green and blue wavelengths, and one for the specular contribution, besides a diffuse albedo map and a specular intensity map.

The specular intensity map can be in one of two variants. In the 2007 publication, the apparent surface roughness parameters wasn't extracted, and the specular intensity map was used to control exclusively the intensity of the specular highlights. In the 2009 publication however, the apparent surface roughness for both specular lobes was provided, together with surface tangents, and a specular albedo image.

Note that in the example dataset available at

http://gl.ict.usc.edu/Research/FaceScanning

the specular intensity image contains in the red channel, the apparent surface roughness of the first specular lobe, in the green channel the apparent surface roughness of the second specular lobe, and in the blue channel the weighting factor for the two specular lobes, and that no specular albedo image was provided.

Fig.1: Example data set from "Rapid Acquisition of Specular and Diffuse Normal Maps from Polarized Spherical Gradient Illumination", University of Southern California, Institute of Creative Technologies, Eurographics Symposium on Rendering, 2007.

Notice also that the specular roughness isn't tied to any particular analytical BRDF, although several publications refer to the Torrance-Sparrow model, and a valid range for roughness between 0.2 and 0.4, with a value of 0.35 being a good starting point, but in any case, the user can ignore the specular intensity map and its encoded roughness parameters and supply the roughness parameter directly, or remap the range to what best fits the needs.

For some information on this see also "A Spectral BSSRDF for Human Skin", by Craig Donner and Henrik Wann Jensen. Further bibliography is suggested at the end.

Notes on importing geometry, shadows

The normal maps, as mentioned earlier, are in "world" space. When importing the geometry the user should after importing, create a texture reference object, and enable "Use Pref" and "Use Nref" in 3DfM, since these will provide the reference surface point and normals to allow correct rigid and soft deformations, otherwise the normals when the mesh is manipulated, won't be in the correct space, since they won't have a frame of reference, and won't be converted properly to allow deformations.

Regarding the usage of shadows, the best results are achieved when using slightly blurred, and properly sampled, shadow maps, or deep shadow maps (DSMs). The use of very sharp raytraced shadows looks somewhat displaced, since the scanned geometry seems at some points not to match the idea of shape that the normals convey to the user. In of these cases though, raytraced, shadow maps, or deep shadow maps, there might be the need for some bias adjustment, and/or shadow softening to achieve a visually consistent and pleasing result.

Shader Parameters

Normal Maps

Rmap, Gmap, Bmap, Smap

- The parameters Rmap, Gmap, Bmap, Smap,should have the filenames of the normal map files for the red, green and blue wavelenghts, and the specular normal maps (notice that the images aren't shifted to [-1,1] range.

diffalb

- The diffuse albedo map (diffuse color). If not provided, Cs will be used instead.

specalb

- The specular albedo map (specular color), if extracted. If not provided, Cs will be used instead.

specmap

- The specular intensity map. This map has encoded in its R and G channels, the specular roughness for the first and second specular lobes, and in the B channel, the mixing factor for both lobes.

Texture Maps

blendmask

- a greyscale mask to blend between mapped normals and actual surface normals in case the user needs to blend a transition between different geometry.

transmask

- the transparency value, in the range [0,1]

gammain

- the input gamma of the diffuse and specular albedo, and specular intensity images. The normal map images should be in linear space so they're not corrected by this value. If the textures are all already in linear space, use a value of 1.0 (which is the default).

BRDF

Ka

- Ka: controls the intensity of the ambient term, and is in the [0,1] range, 0 being no contribution at all, and 1 full contribution.

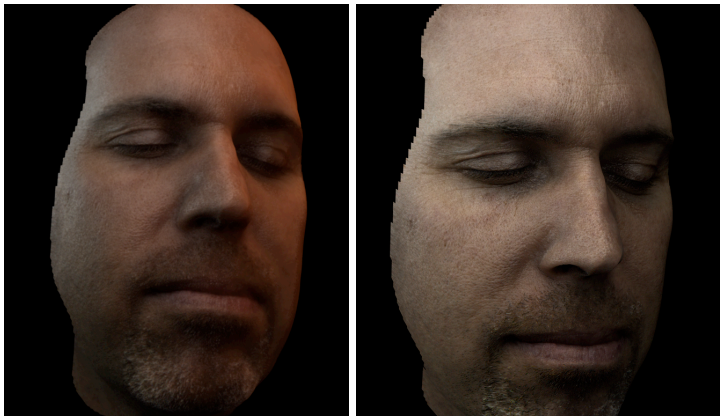
ambcolor

- Ambient color. The color to use as a factor for the ambient term. Usually it's the same as the diffuse color. See diffuse albedo and specular albedo parameters further below.

Kd

- Kd controls the intensity of the diffuse term, and is in the [0,1] range, 0 being no contribution at all, and 1 full contribution.

Lobe1 & Lobe2



The specular term is made of 2 specular lobes, with predominance given to the first or the second specular lobe according to a weighting factor. When this factor is 0, the first specular lobe has full contribution, when this factor is 1, the second specular lobe has full contribution, and values in between blend between the contributions of both lobes.

Ks1, and Ks2

- Control the intensity of the first and second specular lobes respectively, and are in the [0,1] range. Notice however that although the parameter exposed in the UI is in [0,1] range, this value gets doubled in the code to [0,2] range, this way the user can

increase the specular intensity if he/she deems it too "soft", while for all purposes, the UI parameter still is in [0,1] range and can be mapped to a greyscale image.

Ksscale

- an overall scaling factor for the intensity of the specular highlights.

roughness1, roughness2

- control the apparent surface roughness for the first and second specular lobes respectively, and while this should be in [0,1] range, in reality the human skin's roughness value for the Torrance-Sparrow model is between [0.2, 0.4], with a value of 0.35 being suggested in "A Spectral BSSRDF for Shading Human Skin", by Craig Donner, and Henrik Wann Jensen (Eurographics Symposium on Rendering 2006).
- Lower values create sharper specular highlights and off-specular peaks, which increase the appearance of "wetness", but too low a value increases artifacts specially at grazing angles or in very small areas with somewhat random normals, such as eyelashes, or beard - this increases dramatically as roughness approaches 0. Values higher than 0.5 increase the appearance of "velvet" like tissue.

weight

- Controls the blending between the first and the second specular lobes, and is in the [0,1] range, with 0 giving full weight to the first specular lobe and none to the second, and 0 giving full weight to the second specular lobe and none to the first, with in-between values blending between the two lobes, so a value of 0.5 would have a 50%/50% mix of the first and second specular lobes.

ior

- The index of refraction, with a value of 1.3 corresponding to the IOR of human skin (approximately). This parameter is used for for the subsurface call, and as a factor in the specular model and the reflection amount.

Remap

This section serves to remap the overall roughness values (both roughness1 and roughness2) to the range defined by its variables, and is similar to Maya's remap node, with:

romin

- the old range minimum value

romax

- the old range maximum value

rnmin

- the new range minimum value

rnmax

- the new range maximum value

So if we wanted to fit the values encoded in the images (in [0,1] range), to a new range of [0.2, 0.4], the old range minimum and maximum would be 0 and 1 respectively, while the new range minimum and maximum would be 0.2 and 0.4 respectively.

usespecmap

- If this checkbox is checked, then the values used to drive the roughness parameters for the specular lobes are extracted from the specular intensity image.
- The roughness for the first specular lobe is extracted from the image's red channel; the roughness for the second specular lobe is extracted from the image's green channel; and the weight factor for both specular lobes is extracted from the image's blue channel.
- If this checkbox isn't checked, the values used are the values defined in the roughness1, roughness2, and weight parameters respectively.

useTSlobes

- If this checkbox is checked, the specular term will use a Gaussian distribution, as used in the Torrance-Sparrow model, if not, it'll use the Beckmann distribution, as used in the Cook-Torrance model. The specular highlights created by the Beckmann distribution seem somewhat sharper, and it's included here as an option, since some publications and articles suggest this distribution.
- See also GPU Gems 3 Chapter 14, "Advanced Techniques for Realistic Real-Time Skin Rendering", by Eugene d'Eon, and David Luebke (NVidia Corporation).

BSSSRDF

The RGBS normals technique allows for subtle subsurface effects. In case the user wants to add a full SSS term, the parameters to control a standard subsurface() call are included below. Notice that this requires a pointcloud file from a previously baked file. Parameters to control baking for this are provided further below as well.

Ksss

- This parameters ontrols the intensity of the subsurface scattering effect, in [0,1] range, 0 being no effect, 1 full effect.

dmfp

- Diffuse mean free path. This value can be computed from the scattering and absorption coefficients data that can be found in "A Practical Model for Subsurface Light Transport", by Henrik Wann Jensen, Steve Marschner, Marc Levoy, Pat Hanrahan.
- The default value is the value corresponding to the first skin type included in the above mentioned paper. If the user has scattering and absorption coefficient data to use, the diffuse mean free path can be computed with the following equation: being sigma_a the absorption coefficient, and sigma_s the scattering coefficient, then:

$$dmfp = \frac{1}{\sqrt{3\left(\sigma_s + \sigma_a\right)\sigma_a}}$$

albedo

- The ratio of absorbed to reflected light on the surface. Also provided from literature on the topic. The default value is the value corresponding to the first skin type defined in the above mentioned paper. Notice that in recent 3Delight versions, the subsurfac() shadeop now supports varying albedo. For greater flexibility, the "albedo" parameter is provided, defaulting to the first skin type defined in the jjskin_utils.h header. If the "albedo" parameter has a value of 0, the subsurface() shadeop uses instead the diffuse albedo image. Also note that in the single pass subsurface approach one can pass the

absorption and scattering coefficients directly, while on the two pass approach as used in this shader, the albedo and diffuse mean free path should be provided instead.

scalesss

Controls the object scale when doing the SSS calculations on the point cloud. The diffuse mean free path (or the absorption and scattering coefficients used to compute it) measures are provided in millimeters. This scale factor is provided so that the scale of the geometry can be consistent with the scale of the data passed, so if the scale of the scene is in centimeters, a factor of 0.1 would need to be passed (a ten fold increase).

balancesss

This parameter controls the mixing between the Lambertian diffuse term, and the SSS term. A weight of 0 gives full contribution to the diffuse term and none to the SSS term, a weight of 1 gives full contribution to the SSS term and none to the diffuse term, with in-between values providing a blend between the two terms.

dobake

A toggle to enable/disable baking. If this checkbox is enabled, and a ptc file is provided, a pointcloud with the _radiance_t and _area channels is created.

bakefile

The point cloud file to bake the data to, or to read the data from, for the SSS term.

ptccoordsys

the coordinate system used in the point cloud file. Defaults to "world".

Reflection

This section serves to control reflection properties.

Kr

- This parameter controls the intensity of the reflection, and is in [0,1] range.

reflcolor

- The reflection color, defaults to (1,1,1) (white).

envmap

- The environment map to look when the ray travels the allowed distance and hits no
- geometry.

envspace

- The space to use for the environment texture lookup, defaults to "world".

envsamples

The number of samples to use for the environment lookup. While texture() only uses this value for filters of type "box", environment() uses this for all filter types, "box", "triangle", and "gaussian".

label

- The ray label to use, so that reflections are restricted to rays with the provided label.

subset

- The geometry subset to consider for ray intersections. Defaults to 3DfM's "-_3dfm_not_visible_in_reflections", which considers everything except sets that belong to the 3DfM's "not visible to reflections" set.

hitsides

. The side of surface to consider for ray hit. Allowed values are "both", "front", and "back", the shader default is "front".

raysamples

. The number of rays to use when raytracing.

maxdist

. The maximum distance a ray is allowed to travel.

bias

. The bias amount for a ray's starting point in order to control self-intersection.

envblur

. The amount of blur to apply to the environment texture

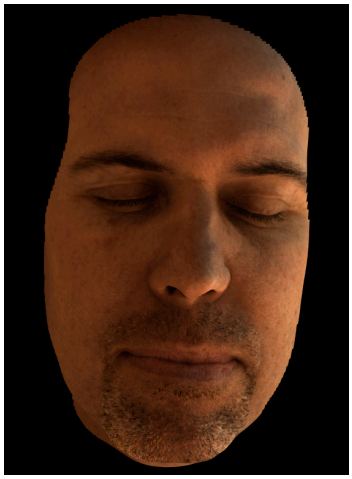
filter

. The filter to use for the environment texture lookup, defaults to "gaussian".

width

. The filter width to use for the environment texture lookup, defaults to 1.

Extra shader paramaters



This section contains generic parameters which cannot be grouped in previous sections.

category

The light category to use, restricts the evaluation of the illuminance() constructs to lights corresponding to the specified categories.

useenvlight

Use this checkbox is you're using the modified 3Delight's "envlight2.sl" shader "envlight2_jjskin.sl". When this is active, the illuminance() construct in the main shader body will send the scanned normals to the light shader via forward message passing. This is slightly more expensive, but there is a visual difference between using the surface normal N, or the scanned normals provided by the normal maps. This has no effect with the standard 3Delight envlight2 shader.

Arbitrary Output Variables (AOVs)

This section contains AOVs parameters, when defined in a Display the output of each AOV will be stored in a separate image/pass. The shader comes with a preset range of arbitrary output variables (AOVs). Although this is quite specific to the user's needs and the way it structured the workflow, some passes were thought to be useful suggestions, and their description follows.

varying color aov_ambient_color

varying color aov_diffuse_color

varying color aov_specular_color

varying color aov_reflection_color

Which contain the surface color considered for the ambient and diffuse term, and the specular highlights and reflection color.

varying color aov_ambient

varying color aov_diffuse

varying color aov_specular

The overall ambient, diffuse, and specular contribution.

varying float aov_specularlobe1

varying float aov_specularlobe2

varying float aov_lobebalance

These passes further decompose the specular term and contain the overall contribution of the first specular lobe, the second specular lobe, and the weighting factor for both lobes.

varying color aov_reflection

varying color aov_reflection_raytraced

varying color aov_reflection_environment

varying float aov_reflection_fresnel

These AOVs contain the overall reflection contribution, and then the separate raytraced and environment lookup colors, as well as the Fresnel reflection amount to factor into these passes. Note that the aov_reflection pass already takes into consideration the Fresnel reflection amount.

varying color aov_subsurfacescattering

varying float aov_sssmixfactor

varying color aov_diffusesssblend

- These passes contain the overall sub-surface scattering contribution, the blending weight between the Lambertian diffuse and the sub-surface scattering terms, and the resulting blend.

varying color aov_indirectdiffuse

varying float aov_occlusion

varying float aov_shadowing

- These passes contain the indirect diffuse contribution, the occlusion contribution, and the shadowing amount as retrieved from 3DfM's (output varying float) __3dfm_shadowing variable, filled in the light shaders. The user might need to tweak this to fit his/her light shaders.

varying color aov_transparency

varying float aov_blendfactor

- These passes contain the transparency, and a blending factor to mix between local surface normals and scanned normals being used in the shader.

varying normal aov_surfacenormal

varying normal aov_Rmap_normal

varying normal aov_Gmap_normal

varying normal aov_Bmap_normal

varying normal aov_Smap_normal

- These passes contain the actual surface normal, and the scanned normals for the R, G, B wavelengths and the S(pecular) normals. One possibility could be using this in some sort of relighting tool in post.

varying color aov_Oi

varying color aov_Ci

- Finally the overall global output variables values, Oi, and Ci.

Bibliography and References

[1] Robert L. Cook and Kenneth E. Torrance. A reflectance model for computer graphics. In SIGGRAPH '81: Proceedings of the 8th annual conference on Computer graphics and interactive techniques, pages 307-316, New York, NY, USA, 1981. ACM Press.

[2] Craig Donner and Henrik W. Jensen. Light diffusion in multi-layered translucent materials. ACM Trans. Graph., 24(3):1032-1039, July 2005.

[3] Craig Donner and Henrik W. Jensen. A spectral bssrdf for shading human skin. In Rendering Techniques 2006: 17th Eurographics Workshop on Rendering, pages 409-418, June 2006.

[4] Craig Donner, Tim Weyrich, Eugene d'Eon, Ravi Ramamoorthi, and Szymon Rusinkiewicz. A layered, heterogeneous reflectance model for acquiring and rendering human skin. ACM Trans. Graph., 27(5): 1-12, 2008.

[5] Abhijeet Ghosh, Tongbo Chen, Pieter Peers, Cyrus A. Wilson, and Paul Debevec. Estimating specular roughness and anisotropy from second order spherical gradient illumination. Computer Graphics Forum, 28(4):1161-1170.

[6] Henrik W. Jensen and Juan Buhler. A rapid hierarchical rendering technique for translucent materials. In SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques, pages 576 581, New York, NY, USA, 2002. ACM Press.

[7] Henrik W. Jensen, Stephen R. Marschner, Marc Levoy, and Pat Hanrahan. A practical model for subsurface light transport. In SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques, pages 511 518, New York, NY, USA, 2001. ACM Press.

[8] Wan C. Ma, Tim Hawkins, Charles F. Chabert, Mark Bolas, Pieter Peers, and Paul Debevec. A system for high-resolution face scanning based on polarized spherical illumination. In SIGGRAPH '07: ACM SIGGRAPH 2007 sketches, pages 61+, New York, NY, USA, 2007. ACM.

[9] Wan C. Ma, Tim Hawkins, Pieter Peers, Charles F. Chabert, Malte Weiss, and Paul Debevec. Rapid acquisition of specular and diffuse normal maps from polarizaed spherical gradient illumination. pages 183-194, 2007.

[10] Christophe Schlick. A customizable Reflectance Model for Everyday
Rendering. In Fourth Eurographics Workshop on Rendering, number Series EG 93 RW, pages 73-84, Paris, France, 1993.

[11] K. E. Torrance and E. M. Sparrow. Theory for off-specular reflection from roughened surfaces. pages 32-41, 1922.

[12] Tim Weyrich, Wojciech Matusik, Hanspeter Pfister, Bernd Bickel, Craig Donner, Chien Tu, Janet Mcandless, Jinho Lee, Addy Ngan, Henrik W. Jensen, and Markus Gross. Analysis of human faces using a measurement-based skin reflectance model. ACM Trans. Graph., 25(3):1013-1024, July 2006.

