SIGGRAPH2008

Lighting and Material of Halo 3

SIGGRAPH2008

Hao Chen haochen@bungie.com

Xinguo Liu xgliu@cad.zju.edu.cn

BUNGIE

Microsoft Research Asia/Zhejiang University

Talk Overview

- Introduction
- Halo3 Lighting
- Halo3 Material Model
- Results
- Conclusions
- Future Work

Motivation

- global illumination
- handle variety of environments
- consistent lighting everywhere
- render bump maps “correctly”
- complex materials under complex lighting
- HDR

Related Work

- Irradiance Volume [GSHG98][Oat05]
- SH Irradiance Environment Map [RamamoorthiHanrahan01]
- Pre-computed Radiance Transfer [SKS02]
- SH Light Maps [GoodTaylor05]
- Sky and Atmosphere [PSS99][HoffmanPreetham02]
- Reflectance Models [CookTorrance81][Schlick94]
- Low Frequency Glossy Material [KSS02][SHHS03]
- Frequency Space Environment Map [RamamoorthiHanrahan02]

Rendering Equation

$$I(V) = \int f(V, L)\,\ell(\omega)\cos(\theta)\,d\omega$$

BRDF Distant Lighting Incident angle

Spherical Harmonics

- A complete set of orthogonal basis for L^2 functions on a unit sphere

$$Y_\ell^m(\theta,\varphi)=\sqrt{\frac{(2\ell+1)}{4\pi}\frac{\ell-|m|}{\ell+|m|}}P_\ell^m(\cos\theta)e^{im\varphi}$$

Orthogonality:

$$\int_{\varphi=0}^{2\pi}\int_{\theta=0}^{\pi}Y_\ell^m(\theta,\varphi)Y_{\ell'}^{m'*}(\theta,\varphi)\sin\theta d\theta d\varphi=\delta_{\ell\ell'}\delta_{mm'}$$

Spherical Harmonics

- Real Spherical Harmonics $y_\ell^m(\theta,\varphi)$

$$y_\ell^m(\theta,\varphi)=\begin{cases}\sqrt{2}\,\mathrm{Re}\left[Y_\ell^m(\theta,\varphi)\right] & , \quad m>0\\ Y_\ell^0(\theta,\varphi) & , \quad m=0\\ \sqrt{2}\,\mathrm{Im}\left[Y_\ell^m(\theta,\varphi)\right] & , \quad m<0\end{cases}$$

Spherical Harmonics

- SH expansion

$$f(\theta,\varphi) = \sum_{\ell,m} f_\ell^m y_\ell^m(\theta,\varphi)$$

where

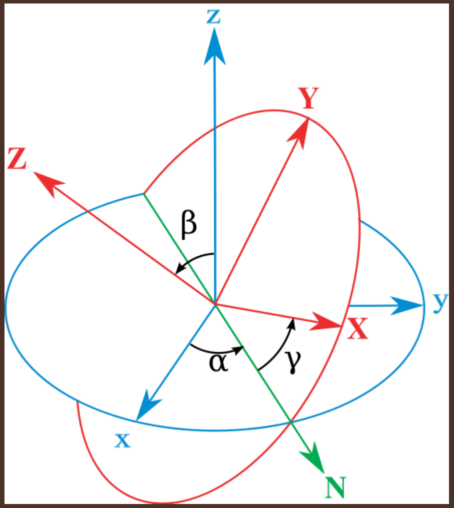
$$f_\ell^m = \int_{\theta=0}^{\pi}\int_{\varphi=0}^{2\pi} f(\theta,\varphi)\, y_\ell^m(\theta,\varphi)\sin\theta\, d\theta\, d\varphi$$

Spherical Harmonics

- SH Rotation by euler angles: α, β, γ

[Green03]

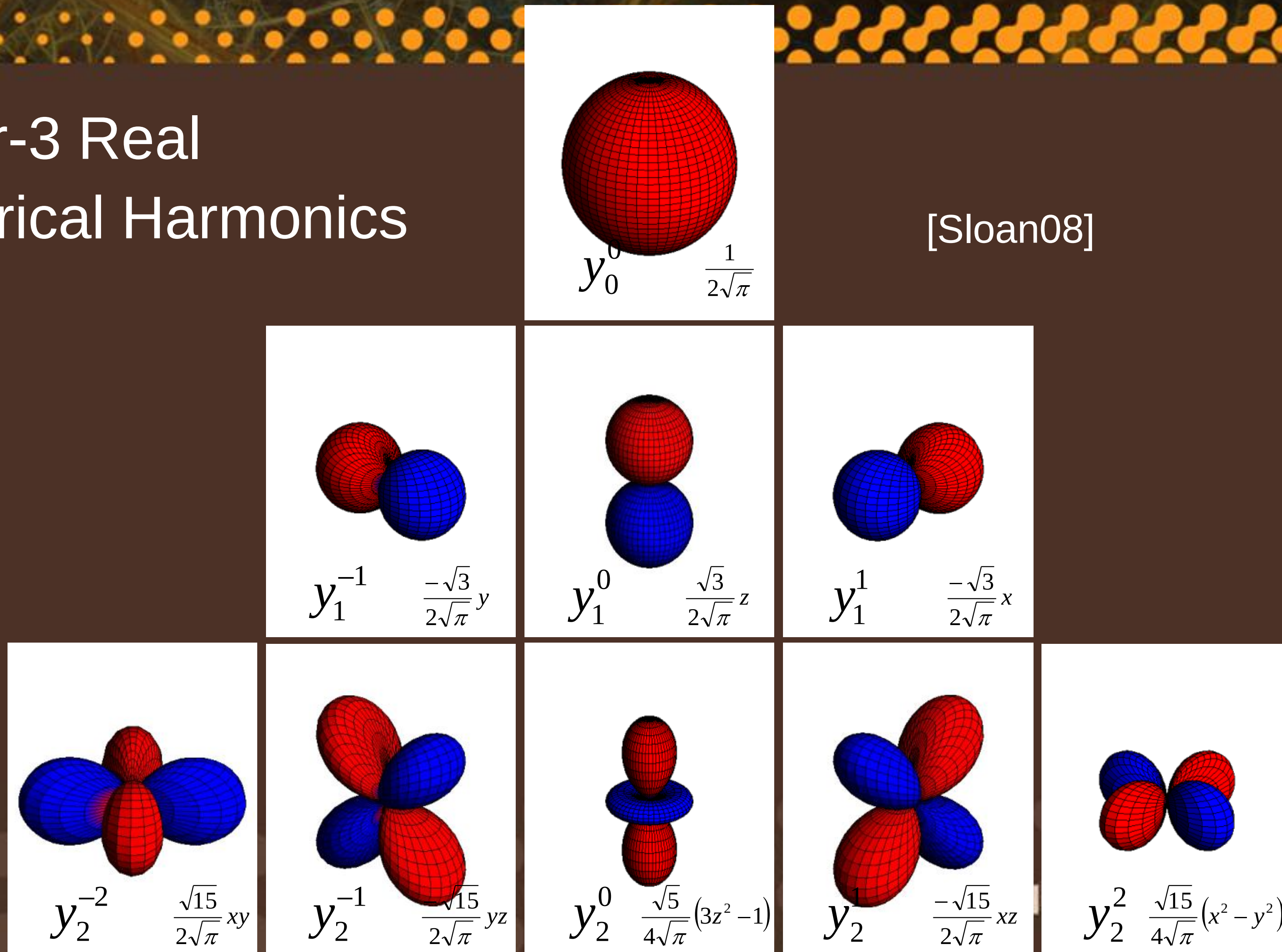
$$\begin{pmatrix} f'^{0}_{0} \\ f'^{-1}_{1} \\ f'^{0}_{1} \\ f'^{1}_{1} \\ \vdots \end{pmatrix} = R_{SH}(\alpha, \beta, \gamma) \begin{pmatrix} f^{0}_{0} \\ f^{-1}_{1} \\ f^{0}_{1} \\ f^{1}_{1} \\ \vdots \end{pmatrix}$$



$$R_{SH}(\alpha, \beta, \gamma) = \mathbf{Z}_{\gamma} \mathbf{X}_{-90} \mathbf{Z}_{\beta} \mathbf{X}_{+90} \mathbf{Z}_{\alpha}$$

Spherical Harmonics

- Order-3 Real Spherical Harmonics



[Sloan08]

Spherical Harmonics

- Order-3 Real SH
  - Rotation

$$\mathbf{Z}_{+90} = \begin{pmatrix} 1 & & & & & & & & \\ & c & 0 & s & & & & & \\ & 0 & 1 & 0 & & & & & \\ & -s & 0 & c & & & & & \\ & & & & c_2 & 0 & 0 & 0 & s_2 \\ & & & & 0 & c & 0 & s & 0 \\ & & & & 0 & 0 & 1 & 0 & 0 \\ & & & & 0 & -s & 0 & c & 0 \\ & & & & -s_2 & 0 & 0 & 0 & c_2 \end{pmatrix}$$

$$c \equiv \cos(\alpha) \qquad s \equiv \sin(\alpha)$$

$$c_2 \equiv \cos(2\alpha) \quad s_2 \equiv \sin(2\alpha)$$

SIGGRAPH2008

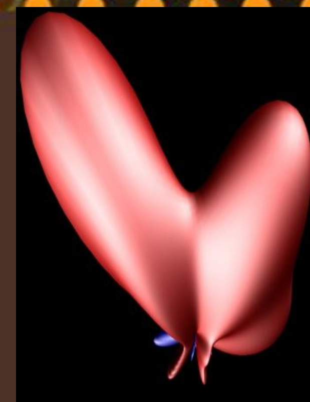
Spherical Harmonics

- A fast rotation code for pixel shader

```
/*
  rotation   [ r00  r01  r02 ]    ->  x-tagent of the local frame
  matrix R = [ r10  r11  r12 ]    ->  y-tagent of the local frame
             [ r20  r21  r22 ]    ->  z/normal of the local frame

     r[] = { r00, r01, r02, r10, r11, r12, r20, r21, r22 };
*/
void sh_rot( const double r[], const double pIn[9], double pOut[9] )
{
        // DC
        pOut[0] = pIn[0];

        // Linear
        pOut[1] =   r[3]*pIn[3] + r[4]*pIn[1] + r[5]*(-pIn[2]) ;
        pOut[2] = -(r[6]*pIn[3] + r[7]*pIn[1] + r[8]*(-pIn[2]));
        pOut[3] =   r[0]*pIn[3] + r[1]*pIn[1] + r[2]*(-pIn[2]) ;

        // Quadratics
        pOut[4] = -(
                ( r[0]*r[4] + r[1]*r[3] ) * ( -pIn[4] ) +
                ( r[1]*r[5] + r[2]*r[4] ) * (  pIn[5] ) +
                ( r[2]*r[3] + r[0]*r[5] ) * (  pIn[7] ) +
                ( r[0]*r[3]             ) * (  -pIn[8] ) +
                ( r[1]*r[4]             ) * (   pIn[8] ) +
                ( r[2]*r[5]             ) * ( -v173*pIn[6] ) );

        pOut[5] =   ( r[3]*r[7] + r[4]*r[6] ) * ( -pIn[4] ) +
                    ( r[4]*r[8] + r[5]*r[7] ) * (  pIn[5] ) +
                    ( r[5]*r[6] + r[3]*r[8] ) * (  pIn[7] ) +
                    ( r[3]*r[6]             ) * (  -pIn[8] ) +
                    ( r[4]*r[7]             ) * (   pIn[8] ) +
                    ( r[5]*r[8]             ) * ( -v173*pIn[6] ) ;

        pOut[7] =   ( r[0]*r[7] + r[1]*r[6] ) * ( -pIn[4] ) +
                    ( r[1]*r[8] + r[2]*r[7] ) * (  pIn[5] ) +
                    ( r[2]*r[6] + r[0]*r[8] ) * (  pIn[7] ) +
                    ( r[0]*r[6]             ) * (  -pIn[8] ) +
                    ( r[1]*r[7]             ) * (   pIn[8] ) +
                    ( r[2]*r[8]             ) * ( -v173*pIn[6] ) ;

        pOut[6] =-v173*(( r[7]*r[6]         ) * ( -pIn[4] ) +
                    ( r[8]*r[7]             ) * (  pIn[5] ) +
                    ( r[6]*r[8]             ) * (  pIn[7] ) +
                    0.5f*( r[6]*r[6]   ) * (   -pIn[8]) +
                    0.5f*( r[7]*r[7]   ) * (    pIn[8]) +
                    0.5f*( r[8]*r[8]   ) * ( -v173*pIn[6]) +
                    0.5f*( 1.f/3.f     ) * (  v173*pIn[6]) );
        pOut[8] = -(  ( r[1]*r[0] - r[4]*r[3] ) * ( -pIn[4] ) +
                    ( r[2]*r[1] - r[5]*r[4] ) * (  pIn[5] ) +
                    ( r[0]*r[2] - r[3]*r[5] ) * (  pIn[7] ) +
                    0.5f*( r[0]*r[0] - r[3]*r[3] ) * (   -pIn[8] ) +
                    0.5f*( r[1]*r[1] - r[4]*r[4] ) * (    pIn[8] ) +
                    0.5f*( r[2]*r[2] - r[5]*r[5] ) * ( -v173*pIn[6] ) );
}
```

2008

SH Irradiance Env Map



[Ramamoorthi00]

$$L_{lm} = \iint_{\theta,\phi} \boxed{L(\theta,\phi)}\boxed{Y_{lm}(\theta,\phi)}\boxed{sin(\theta)d\theta d\phi}$$

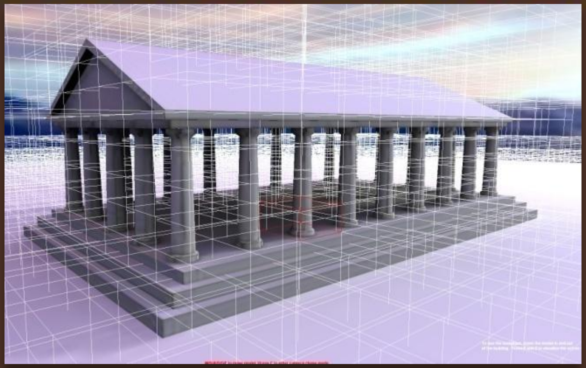
distant radiance at given direction

SH basis evaluated at given direction

diff solid angle

Irradiance Volumes

[GSHG98]



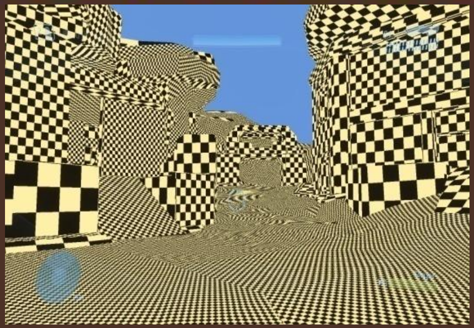
[Oat05]

- Spatially divide volume into cells.
- irradiance volume per cell.
- Interpolate between samples.
- Sharp shadow boundaries?
- Bump maps?

Spherical Harmonics Light Map

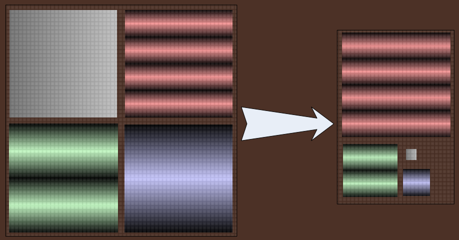
- Parameterize over geometry surface.
- Each texel is a SH Vector.
- 9 textures for quadratic SH.
- Highly compressed.

Halo3 Lighting Pipeline



Parameterize

GI Solver

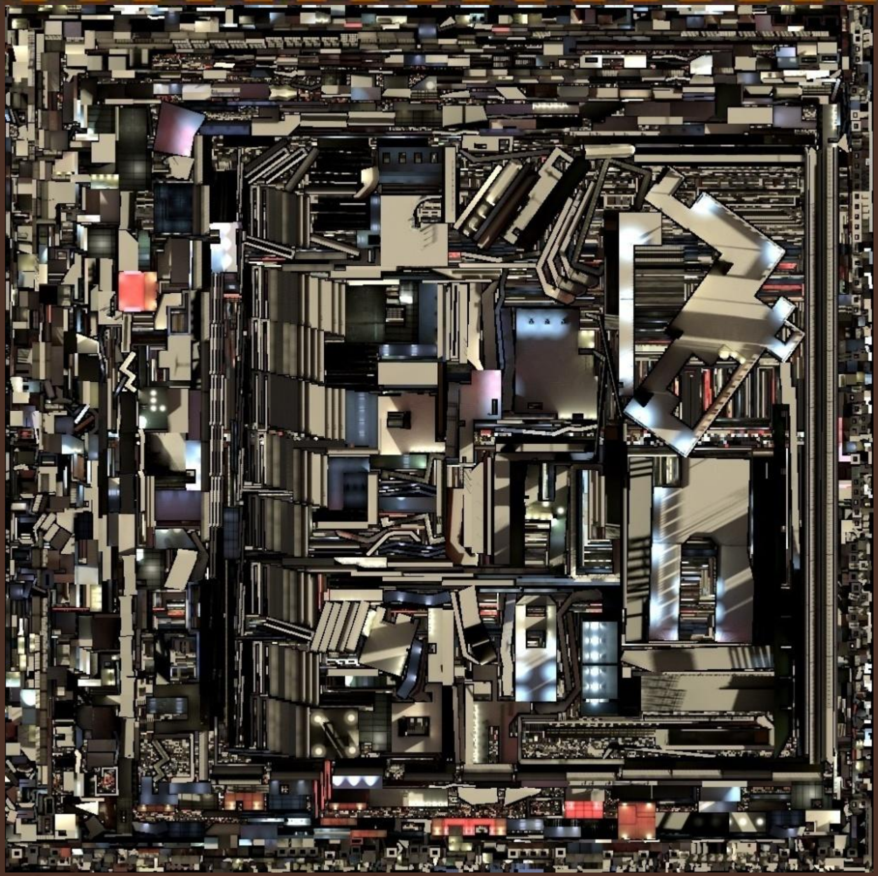


Compression

Rendering

Parameterization

- UVAtlas (MSRA)
- Small Charts
- Long and thin charts
- > 80% utilization.
- Vastly improved over Halo2.



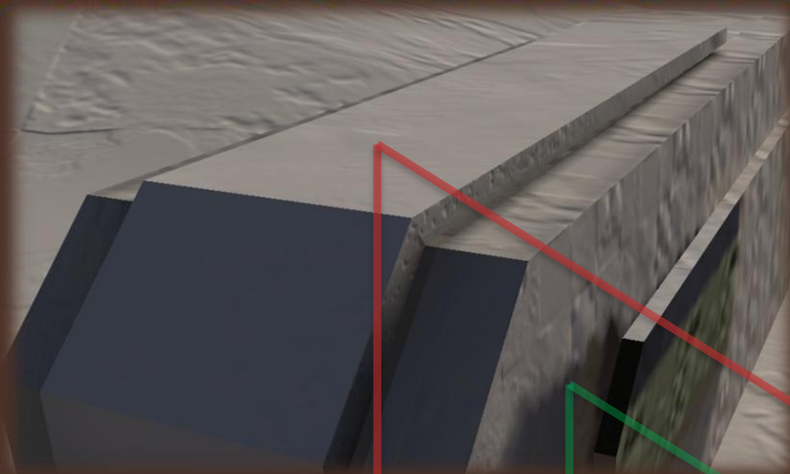
Photon Mapping

Initialization → Direct Illumination → Photon Cast → Radiance Estimate → Exit Illumination → Final Gather → Signal Compression → DXT Compression

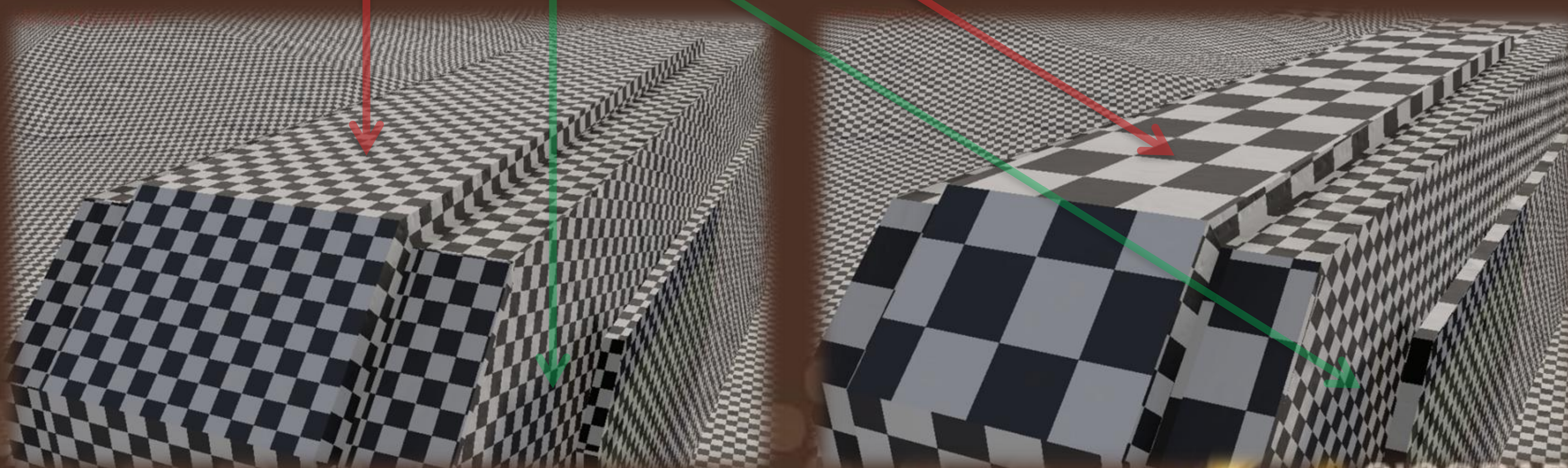
Compression

- Two Stage Process.
- Signal Based Optimization.
  - Sub-chart optimization.
- DXT HDR compression.
  - Use 2 DXT5 to compress each floating point texture.

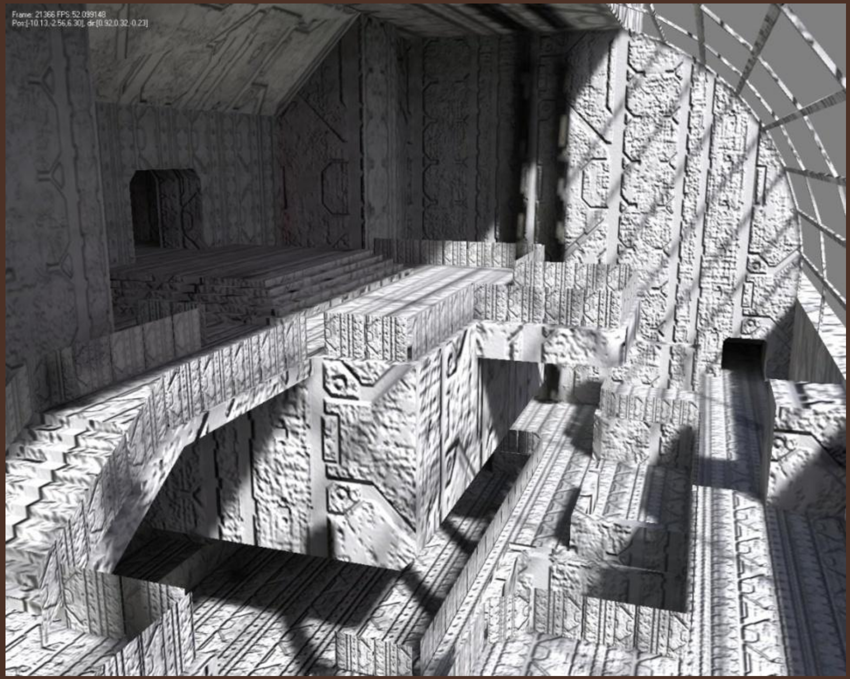
Signal Based Optimization



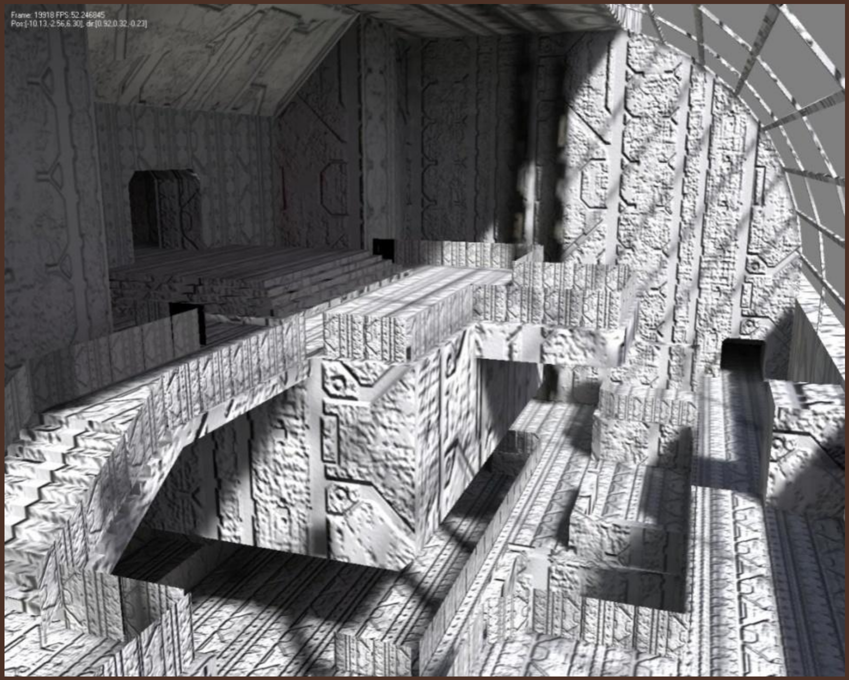
- Compute gradient of pixels in log space.
- Resize charts based on gradients.



Signal Optimization Result



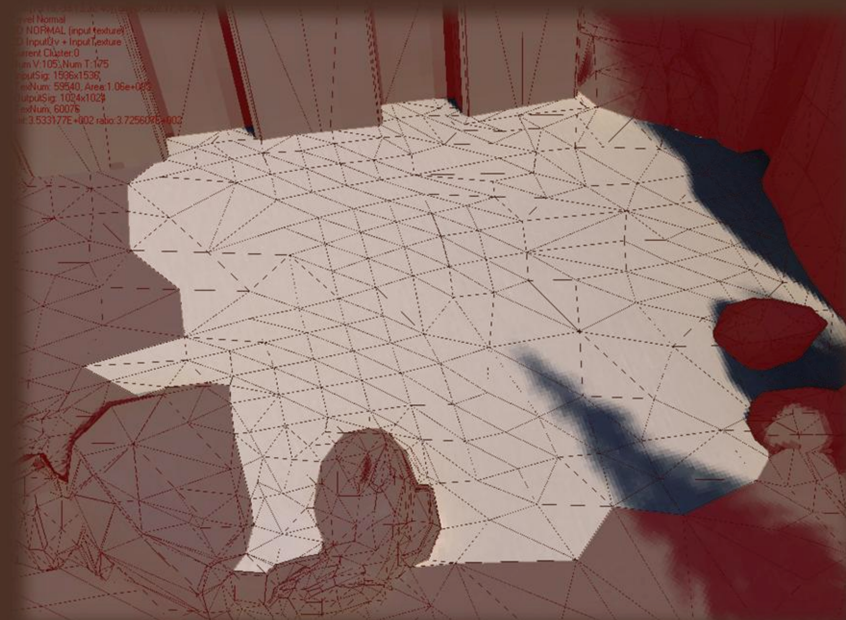
Before: 1024x1024



After: 512x512

Subchart Optimization

- Charts with only a few high freq pixels can still get large area.
- Solution: cut out the high frequency area into a separate chart.



DXT HDR Compression

- Use two DXT5 textures to compress the SH coefficients (HDR, positive/negative)

4 x 4 block:

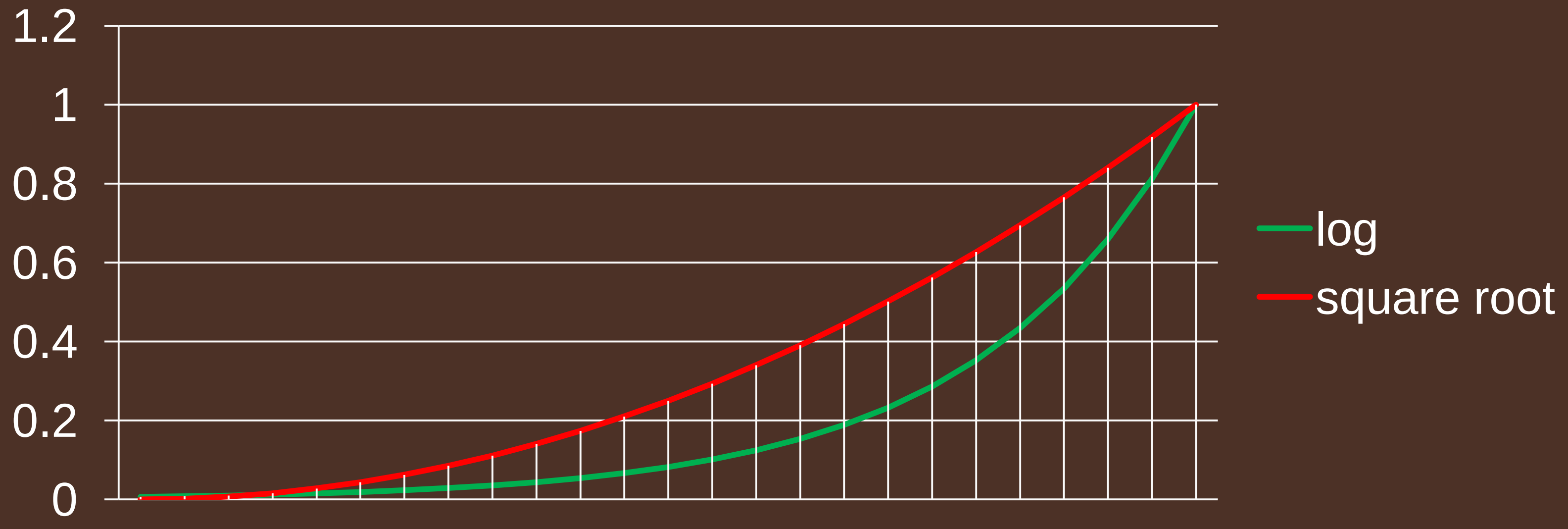
DXT[0]:	Alpha (Luminance) / 64 bits	RGB / 64 bits
DXT[1]:	Alpha (Luminance) / 64 bits	RGB / 64 bits

"Luvw" Color Space

- Each SH coefficient is a RGB vector and is converted to Luvw color space.
- L: magnitude of the RGB vector.
  - Non negative.
  - Stored in DXT5 x 2's alpha channels for higher precision.
- uvw: normalized vector.
  - Good coherence, we store in the rgb channels of DXT5.

Luminance

- Store square root of L for higher dynamic range.
- Similar to log space, but cheaper to decode.



Luminance Compression

- Use 2nd DXT alpha to compensate for the error.

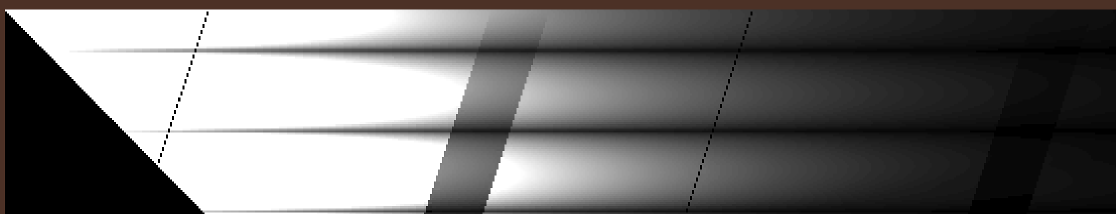
Alpha channel of a 4 x 4 block

DXT[0]: Block[0] = DXT_compress (sqrt (L_block))

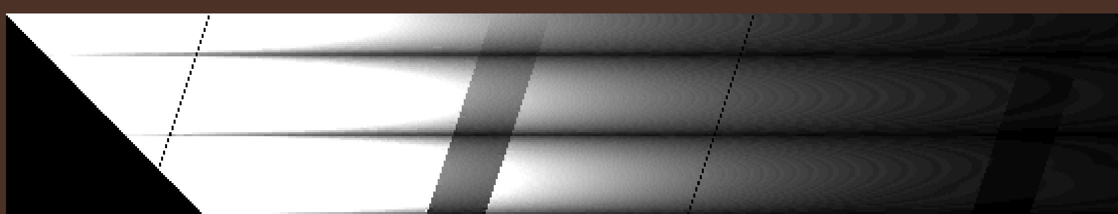
DXT[1]: Block[1] = DXT_compress (L_block / Decompress (Block[0])))

Luminance Decoding

- L= Alpha0 * Alpha1 * Max_Luminance



2 DXT 5 alpha blocks



Just 1 alpha block

UVW compression

- Similar, use 2nd DXT5 to minimize error.

RGB channel of a 4 x 4 block

DXT[0]:	Block[0] = DXT_compress (RGB_block / 2 +0.5)
DXT[1]:	Block[1] = DXT_compress ((RGB_block – Decompress (Block[0])) / 2 + 0.5)

- Decode: UVW= (rgb0 + rgb1) * 2 - 2

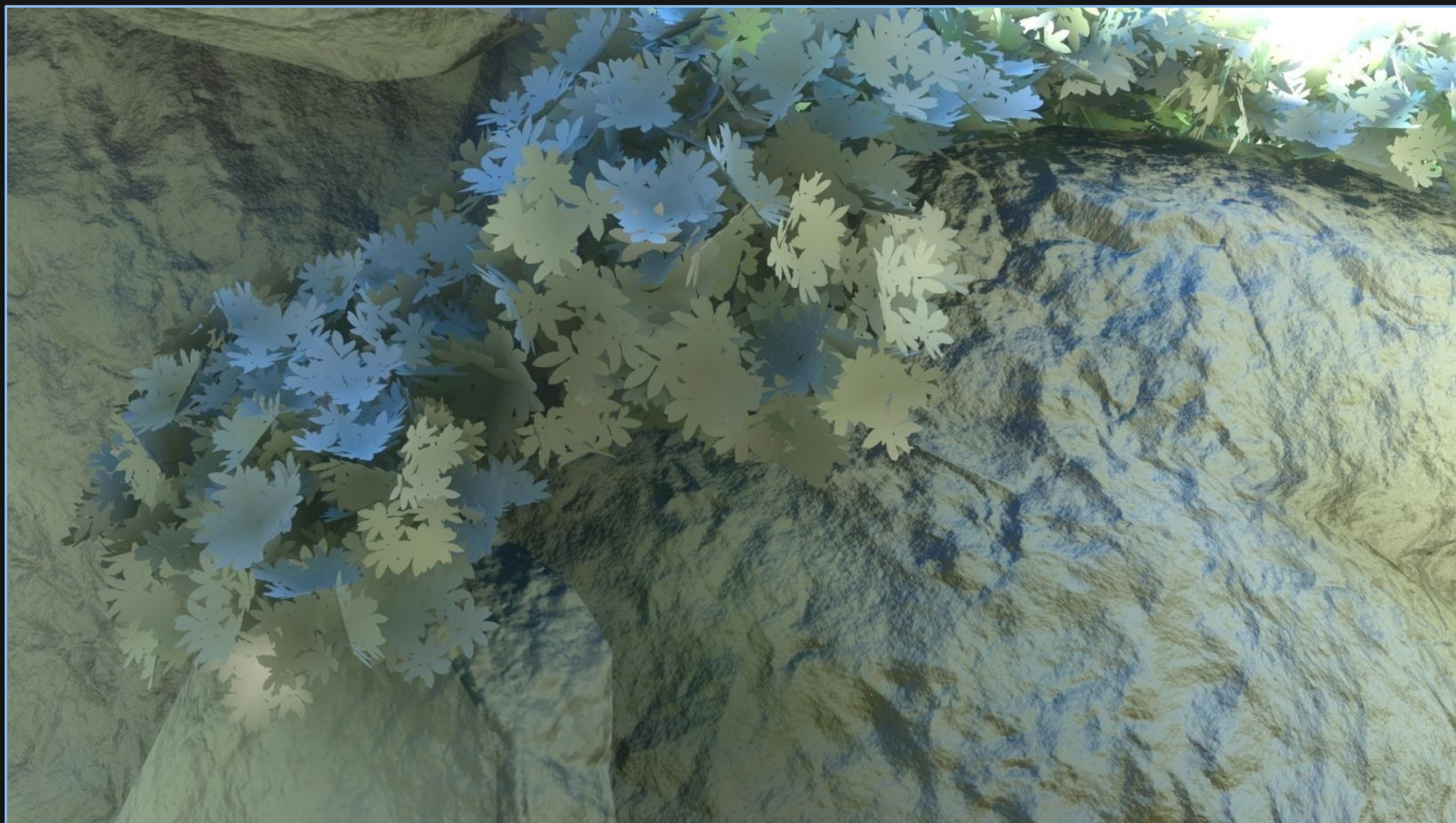
Compression Summary

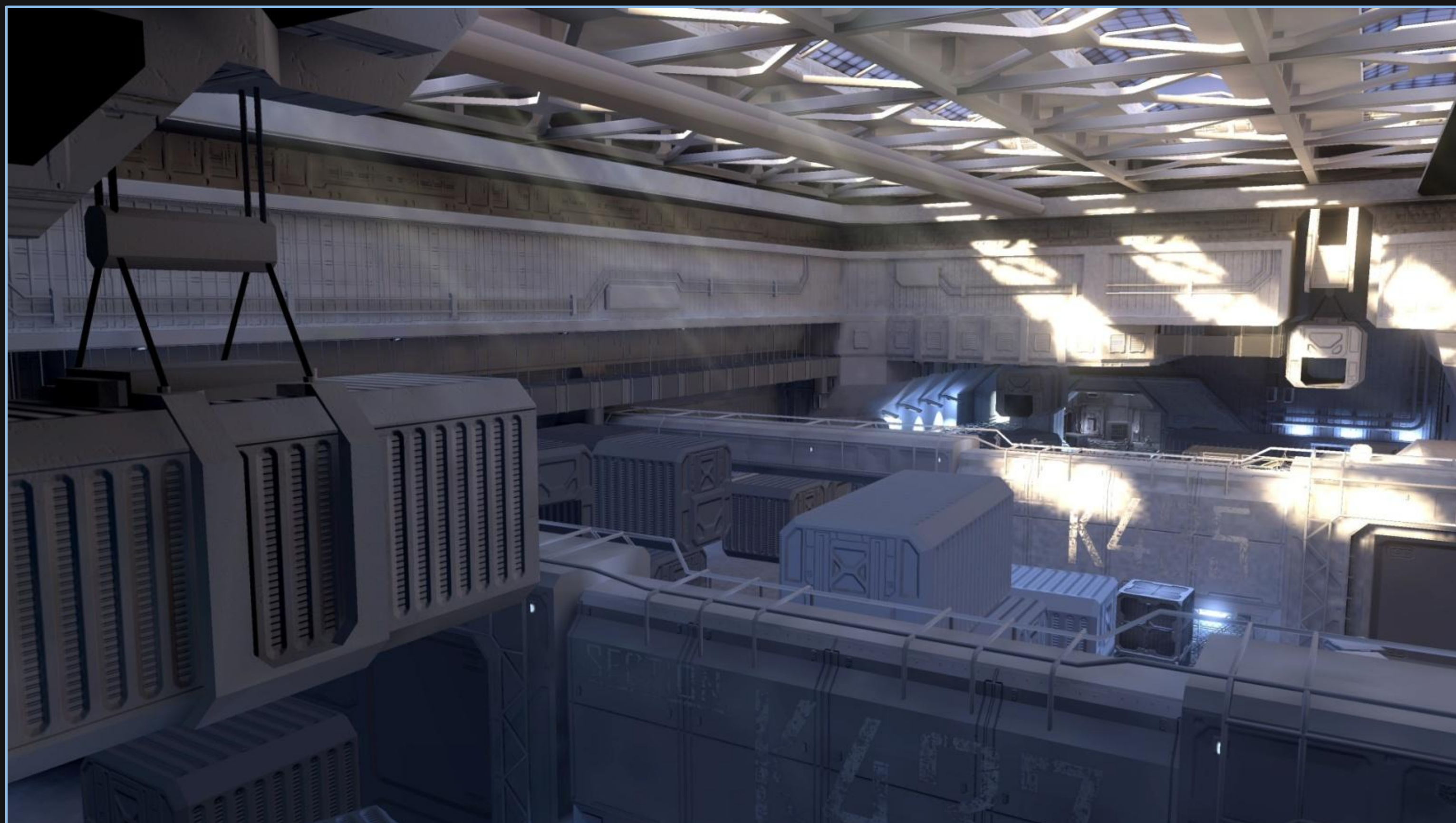
- Signal compression re-packs charts based on signal frequency. (4: 1 compression ratio).
  - Sub-chart optimization break up charts if desired.
- DXT HDR compression compresses the raw floating point value. (3: 1 compression ratio)
- Overall 12 : 1 compression ratio, quality loss is perceivably small.

Rendering Diffuse Lighting

- Static Geometry.
  - SH light map is bound as a surface textures.
  - Per pixel: evaluate normal with the SH vector.
  - Exactly like in Spherical Harmonics Irradiance Env Map.
- Dynamic Objects.
  - Sample the SH light maps based on object position.
  - Render object using PRT and the SH vector.

Diffuse Lighting Examples





Optimization

- Reduce storage and ALU count.
- Pull out dominant light, store intensity.

$$E = \sum_{i=0,\dots 8} (\lambda_i - cY_i(d))^2, E' = 0$$

$$c = \sum_{i=0,\dots 8} (\lambda_i Y_i(d)) / \sum_{i=0,\dots 8} Y_i(d)^2$$

- Store linear SH instead of quadratic.
- In shader, do (N * L + sh_eval(sh[] – c * Y(d), N).

Comparing “SH 2.5” and SH 3

SH quadratic

N . L + SH linear

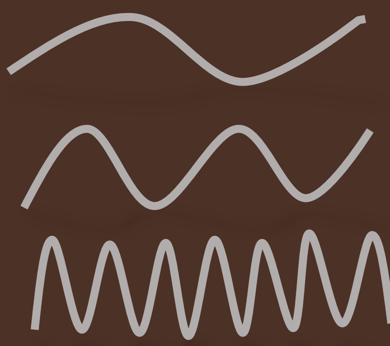
Material Motivation

- BRDF expressiveness.
- Real time performance.
- Compatible with Halo 3 lighting model.
- Requires low storage.

Halo3 Material Model

- Separate reflectance into separate, low to high frequency parts.
  - Diffuse.

  - Low frequency glossy (area specular).
  - Mid frequency glossy (environment map).
  - High frequency glossy (analytical specular).

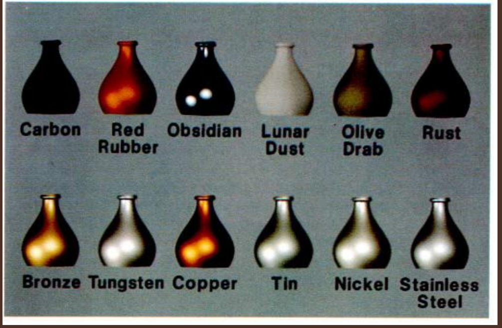


- Handle each with different techniques.

Cook Torrance BRDF

$$f(V, L) = k_d R_d + k_s \cdot F \cdot R_m$$

view direction light direction diffuse & specular scalar diffuse lobe Fresnel specular lobe



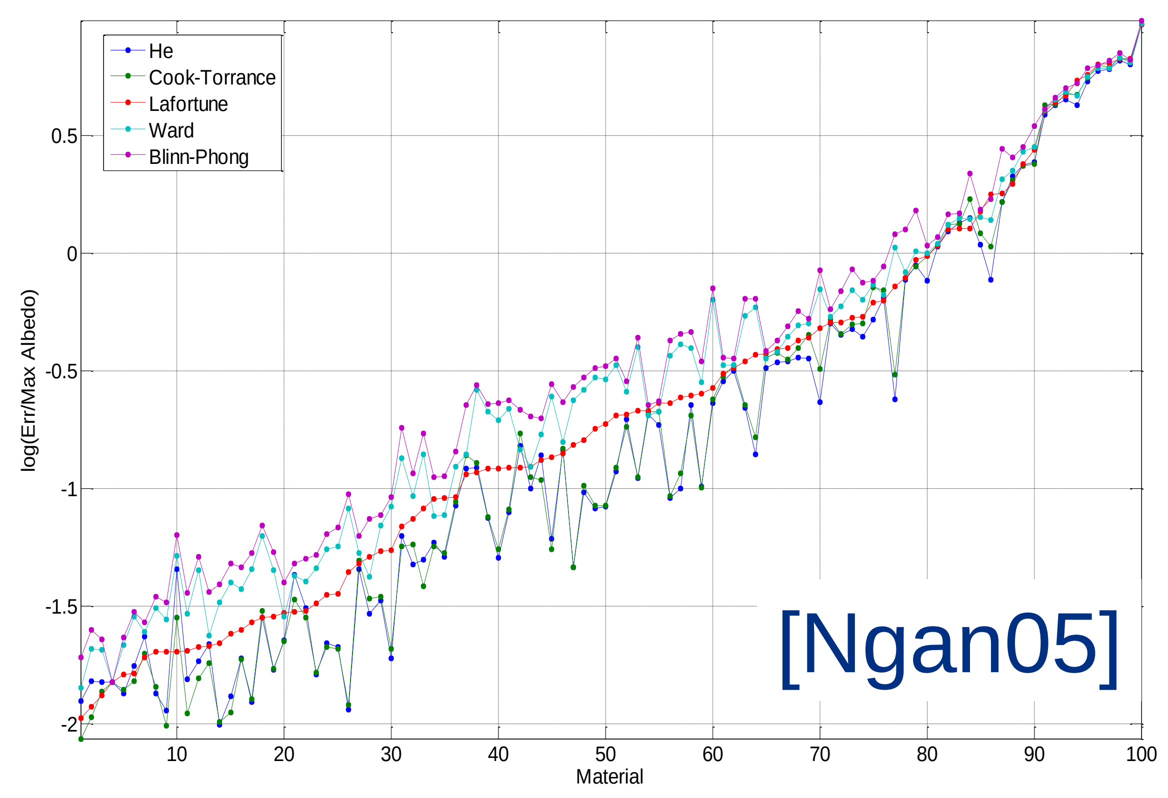
$$R_m(V, L) = \frac{DG}{\pi (N \cdot L)(N \cdot V)}$$

D: microfacet distribution function

G: geomery term

[CookTorrance81]

Cook Torrance BRDF



SIGGRAPH2008

Analytical Specular

- Evaluate BRDF directly in pixel shader (point light)
- Use the dominant light direction and intensity.
- HLSL listing in paper.

Environment Map

- Render cube map at discrete locations.
- Filter out high frequency.
- Pre-divide by area specular.
- Multiply back in shader.

Area Specular

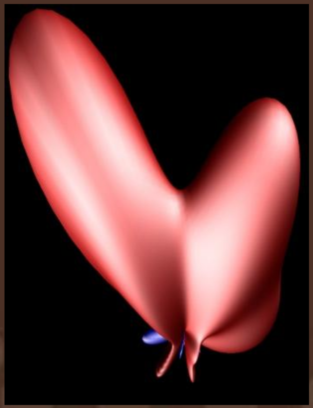
- Main idea:
  - Express BRDF model itself in SH.
  - Pre-integrate key terms into 2D textures.
  - Evaluate BRDF in shader with SH light.
  - Low frequency only.

Light Integration

$$I(V) = \oint \boxed{f(V,L)} \cos(\theta) L(\omega) d\omega$$

$$\boxed{\boxed{k_d}\boxed{R_d}\oint \cos(\theta) L(\omega) d\omega} + \boxed{\boxed{k_s}\oint \boxed{F}\boxed{R_m(V,L)} \cos(\theta) L(\omega) d\omega}$$

↓



SH irradiance env. map

??

Light Integration in SH

$$I_s(V) = k_s \oint \boxed{FR_m(V,L)cos(\theta)}\,\boxed{L(\omega)}\,d\omega$$

Project light into SH basis.

$$L(\omega) = \sum_{i=0}^{8} \lambda_i Y_i(\omega)$$

Project BRDF and cosine term in SH basis

$$B_{m,i}(V) = \oint \frac{F}{F_0} R_m(V,L)cos(\theta)Y_i(\omega)d\omega$$

$$I_s(V) = K_s F_0 \sum_{i=0}^{8} \lambda_i B_{m,i}(V)$$

Dot product to convolve

Light Integration in SH Cont.

$$F \approx F_0 + (1 - F_0)(1 - (L \cdot H))^5$$ [Schilick94]

$$B_{m,i}(V) = \oint \frac{F_0 + (1 - F_0)(1 - (L \cdot H))^5}{F_0} R_m(V, L)\cos(\theta) Y_i(\omega) d\omega$$

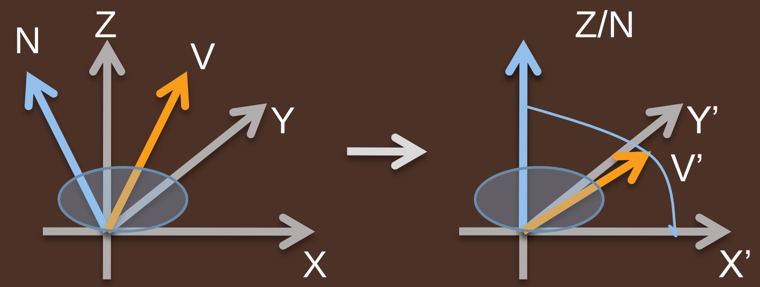
$$C_{m,i}(V) = \oint R_m(V, L)\cos(\theta) Y_i(\omega) d\omega$$

$$D_{m,i}(V) = \oint (1 - (L \cdot H)^5) R_m(V, L)\cos(\theta) Y_i(\omega) d\omega$$

$$B_{m,i}(V) = \boxed{C_{m,i}(V)} + \frac{1 - F_0}{F_0} \boxed{D_{m,i}(V)}$$

Preintegration

Pre-integration

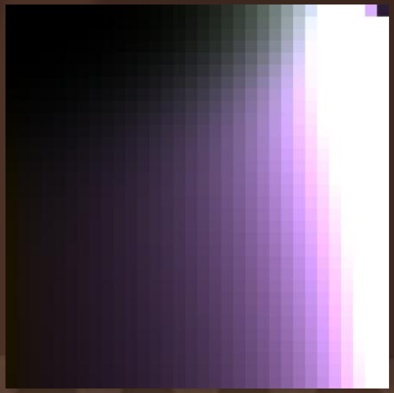


Reflective symmetry means:

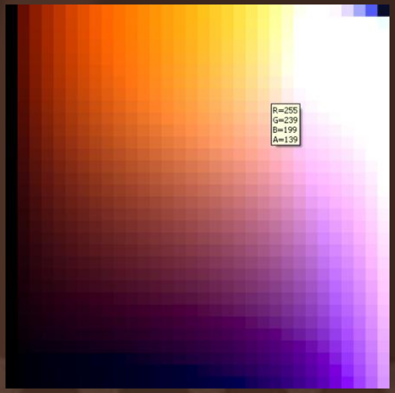
$$C_{m,i}(V) = D_{m,i}(V) = 0, i = 1,4,5.$$

Isotropic BRDF = any coordinate frame16 m values, and 8 V directions is enough.

C (i=0,2,3,6)



D (i=0,2,3,6)



C,D (i=7,8)

Rendering Area Specular

- Build a local frame.
- Look up C and D texture.
- Rotate SH light vector into local frame.
- Do SH Dot product.
- HLSL Listing in course note.

Results


TRAXUS

BUNGIE


TRAXUS

BUNGIE

Conclusions

- SH light map is a natural extension to the traditional light-mapping pipeline.
- Separating material into layers is a good approximation for all frequency reflectance.
- Area specular is critical for achieving seamless lighting and material integration.
- ALU is cheap, and will get cheaper, take maximum advantage of it.

Future Work

- Global Illumination with local, moving lights.
- GI for dynamic and semi-dynamic scenes.
- Better lighting basis (less ringing, higher frequency).
- Area specular model with complex transport.
- Measured BRDF.
- Non photo-realistic rendering.

Acknowledgement

- Authors
  - Bungie: Hao Chen, Ben Wallace, Chris Tchou, David Cook, Xi Wang, Yaohua Hu.
  - MSRA: Xinguo Liu, Zhipeng Hu, Xin Huang, Minmin Gong, Kun Zhou.
- Special Thanks
  - Bungie artists.
  - Peter Pike Sloan, Baining Guo, Harry Shum
  - Kutta Srinivasan, Matt Lee, Mikey Wetzel.

Want to team up with the Master Chief?

Bungie is hiring.

http://www.bungie.net/jobs