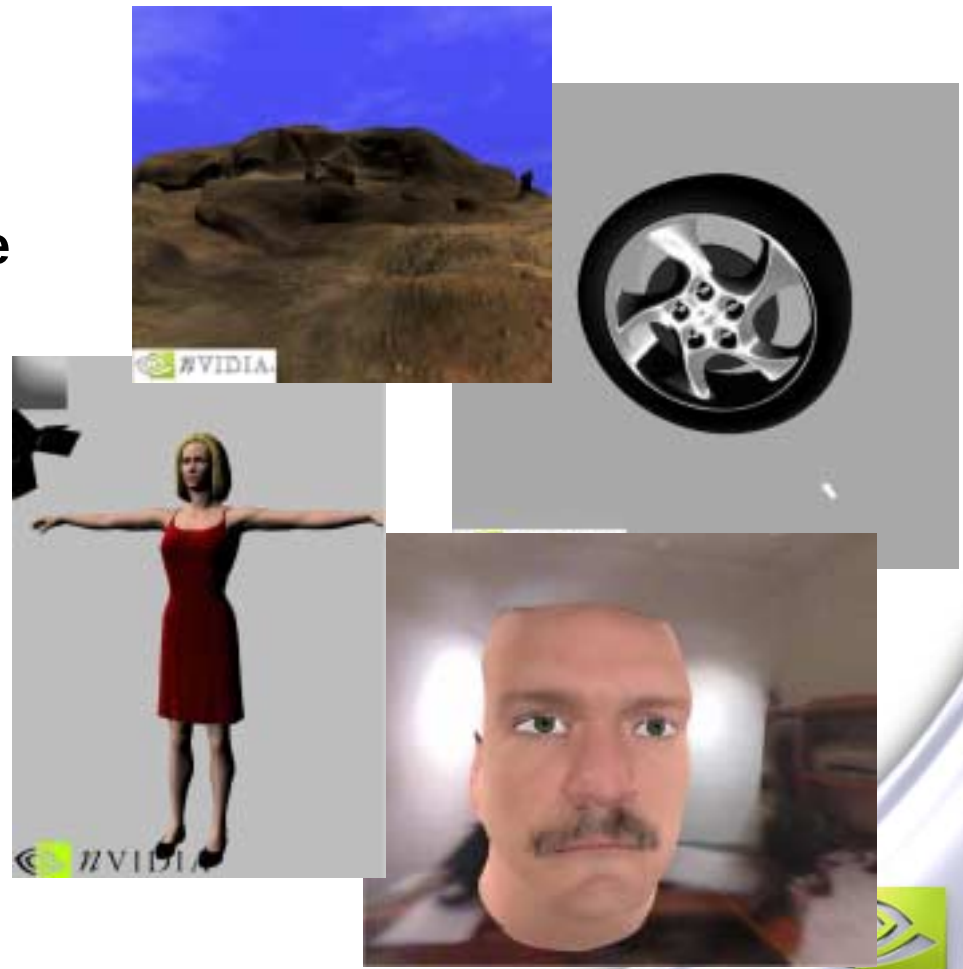
NVIDIA®

Extending Textures Beyond Simple Pixels

Kenneth L. Hurley

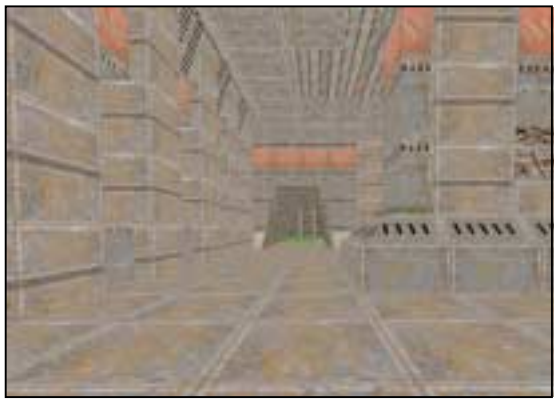
Agenda

- Traditional Texture Mapping
- Cube Maps
- Encoding Data into Texture Maps
- Minnaert Lighting
- Brushed Metal
- Cloud Cover
- Low Dynamic Range Images
- High Dynamic Range Images



Traditional Texture Mapping

Base Texture (Diffuse)



×

(modulate)

Light Maps

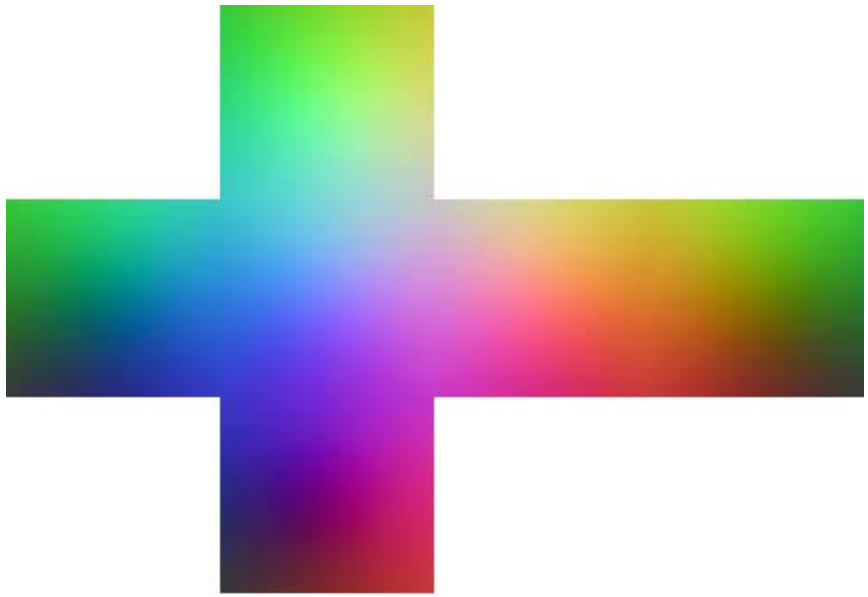
=

Other uses for texture maps

- 2D Texture maps – Look up tables
- Cube Maps - Look up tables
- Masks
- Mathematical functions
- Temporary Storage
- Not just painted pixels

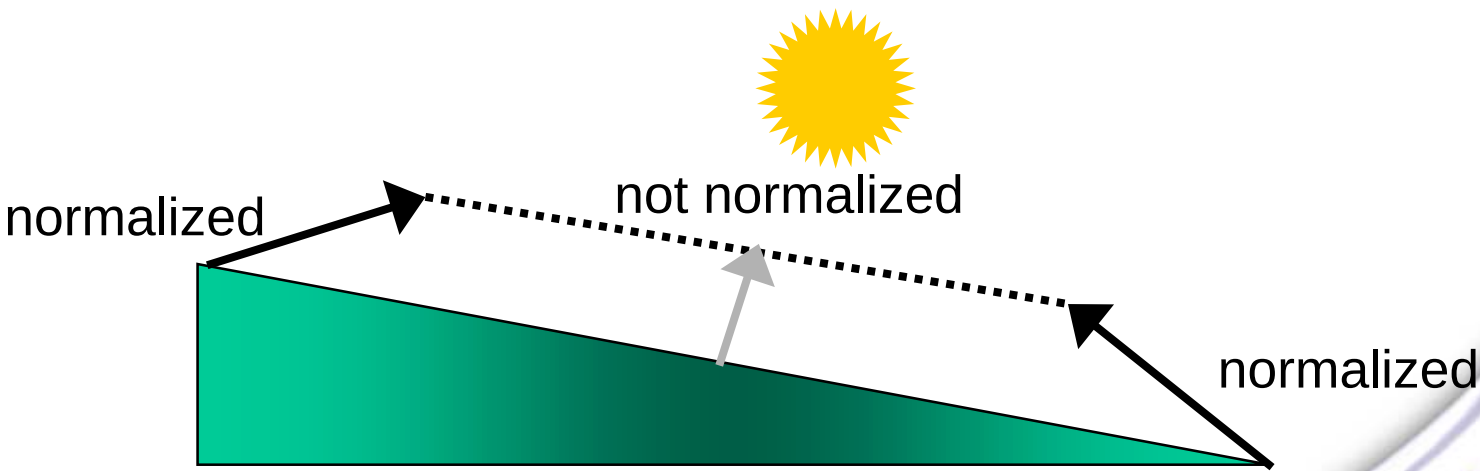
Cube Maps

- Can be used for a wide range of things
  - Reflections – Similar to diffuse texture mapping
    - Pixel shader instruction - texm3x3vspec
  - Normalization inside pixel shader / fixed function pipeline
    - why do we need this?



Why we need normalization map

- We’re interpolating between vectors linearly
- Interpolated vector is not normalized
- It can be shorter than unit length
- Only noticeable when light is close to object



Cube Maps (Cont)

- Can be used for a wide range of things (continued)
  - Projective shadows
    - Why Cubemaps?
  - You decide. Use your imagination

Encoding Data into Texture Maps

- Several Tools are available
  - NVIDIA'S Normal map generator
  - DCM – Diffuse Cube map generator – ShaderX book and www.shaderx.com soon
  - HDR Shop – Encodes IBR lighting in cubemap
  - Photoshop
- CPU Generated
  - Sample function inside code
  - write to texture map
  - Run Time
  - Off-Line – Use DevIL package to read/write textures out to disk

Encoding Data into Texture Maps

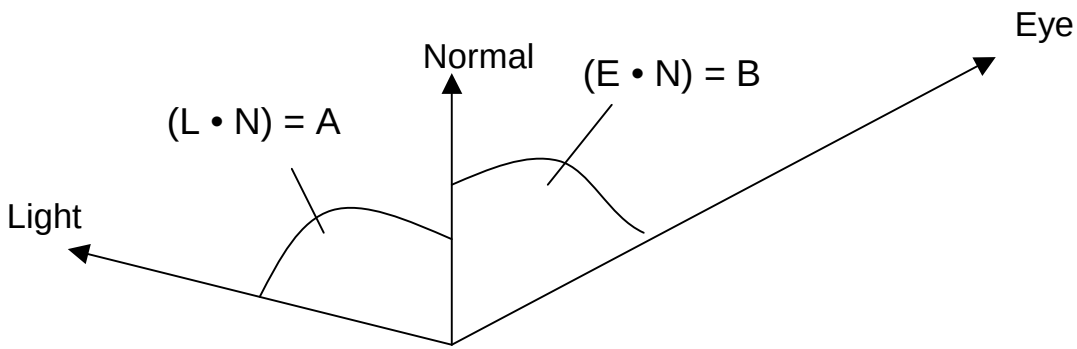
- GPU Generated
  - Greg James – Dynamic Normals maps

Minnaert Lighting

- Minnaert, M., 1941. The reciprocity principle in lunar photometry. *Astrophysical Journal*, Volume 93, pp. 403-410.
- Subtle shading technique for Isotropic lighting effects
- Darkening limbs (edges, WRT eye/light)
- Portion of BRDF (Bidirectional Reflectance Distribution Function) calculations.

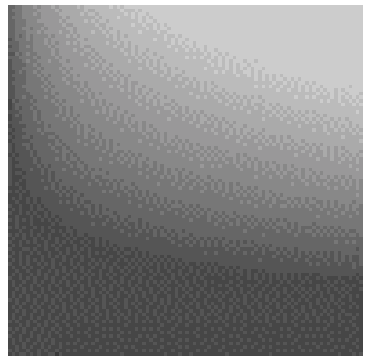
Minnaert Map

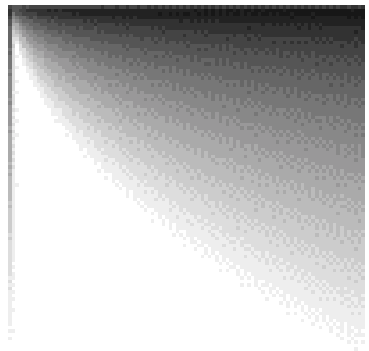
- **Map that is used to look up**
  - **Color * (cos(A)^{k} * cos(B)$^{1-k}$)**
  - A **= Angle between Light and Normal**
  - B **= Angle between Eye and Normal**



Minnaert Map Creation

- Done on CPU
- Traditional way
  - Lock Texture
  - Write Pixels
  - Unlock Texture

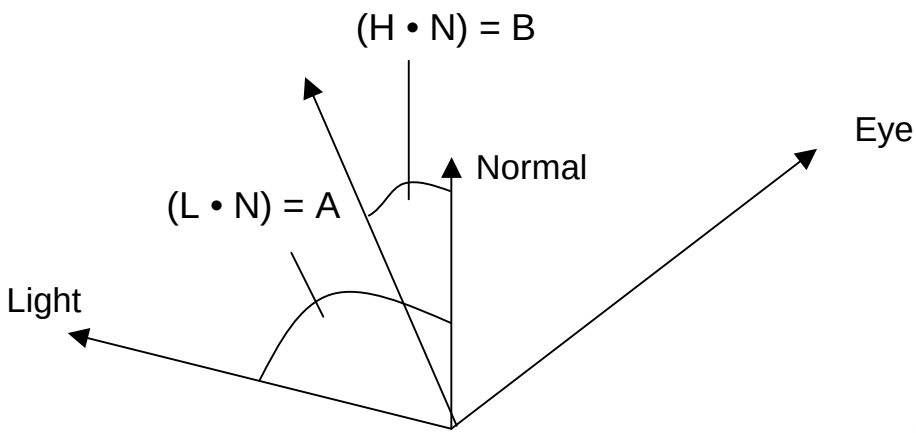




Minnaert Lighting Demo

Brushed Metal

- **Map that is used to look up**
  - **Color * (L • N) Diffuse**
  - **Color * (H • N) Specular**
  - A **= Angle between Light and Normal**
  - B **= Angle between Half Angle and Normal**



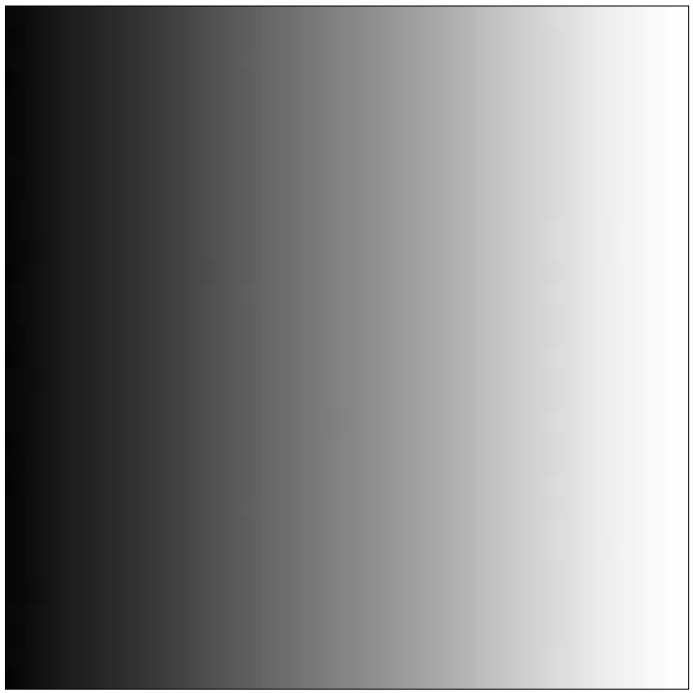
Brushed Metal (Cont)

- Light look ups can be encode in the RGBA values of a texture
  - Build 2 Ramp textures in Photoshop
    - N • L (Encode in RGB)
    - N • H (Encode in Alpha)
  - Probably shouldn’t be linear ramp as eye is more sensitive to changes in lower luminance values
  - N • H doesn’t need to be linear
- There are no traditional texture maps

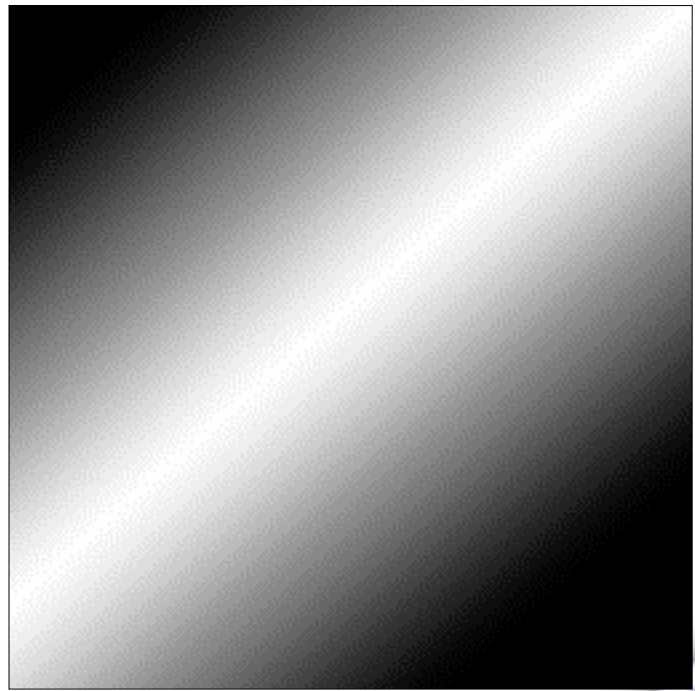
Brushed Metal (Cont)

- **N • L lookup in RGB, N • H in alpha**

RGB portion of bitmap

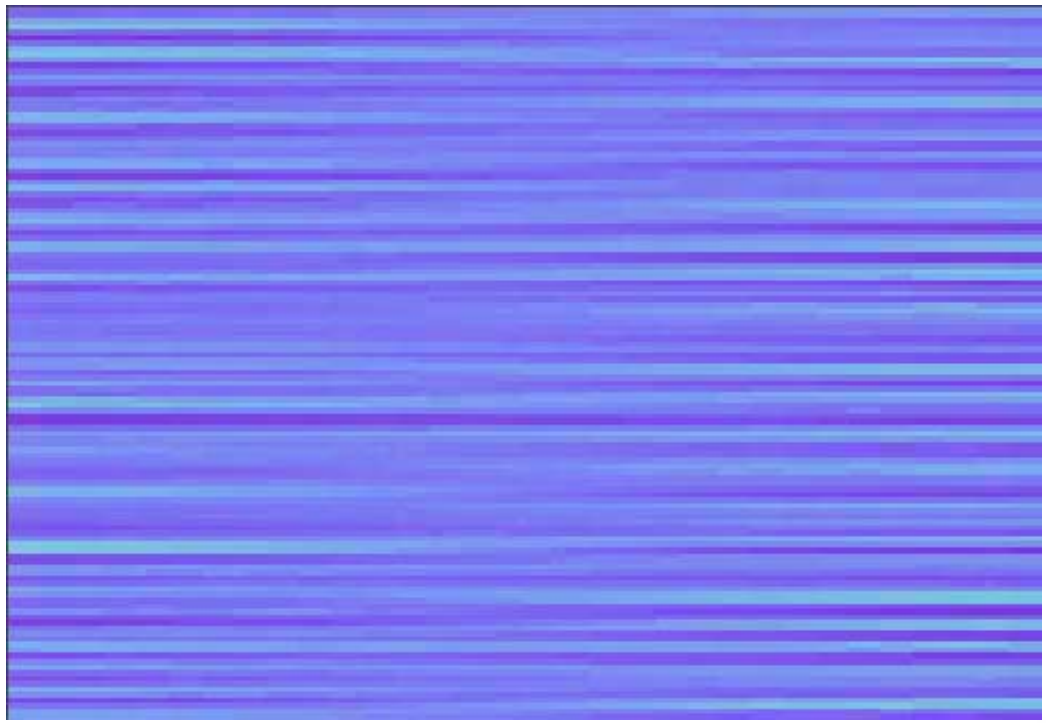


Alpha Portion of bitmap



Brushed Metal (Cont)

- Combine with faked high resolution bump map
  - What is meant by fake?



Code Sample (DX8)

```
tex  t0                        // fetch base texture
tex  t1                        // fetch normal map
texm3x2pad  t2, t1_bx2         //  u = ( t1=N ) dot ( t2=L )
texm3x2tex  t3, t1_bx2         //  v = ( t1=N ) dot ( t3=H )
                               //  fetch texture 4 at (u,v)
mov r1, t3                     // RGBA diffuse,alpha into r1

mul r0, r1, t0                 // Diffuse * base texture

mul r1, t3.a, t3.a             // spec * spec

mad r0, r1, c1, r0             // (spec * constant) + diffuse
```

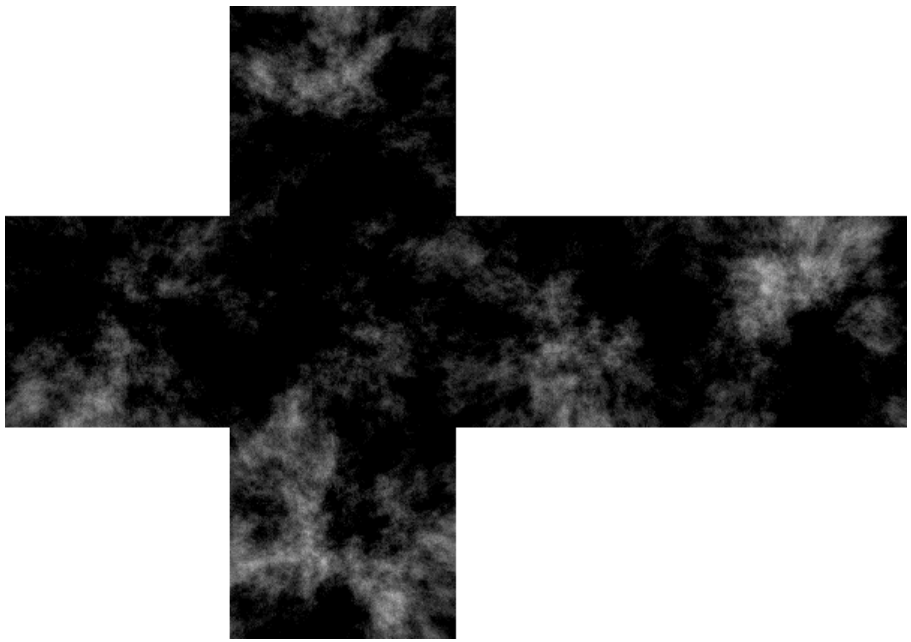
Brushed Metal Demo

Cloud Cover

- Uses only one texture and one cube map for entire scene
- Why not just use projective texture?
- Simple vertex shader calculation using position of vertex

Cloud Cover (Cont.)

- Cloud Texture created using fBM
- Encoded into cube map all at once



Cloud Cover (Cont.)

- Terrain uses cloud cube map to look up shadows
- Darkens diffuse texture map
- Sky sphere also uses same cube map
  - Normals are inverted and sphere is rendered inside out.

Sky Sphere Vertex Shader

```
;transform position
   dp4 oPos.x, srcPosition, c[CV_WORLDVIEWPROJ_0]
   dp4 oPos.y, srcPosition, c[CV_WORLDVIEWPROJ_1]
   dp4 oPos.z, srcPosition, c[CV_WORLDVIEWPROJ_2]
   dp4 oPos.w, srcPosition, c[CV_WORLDVIEWPROJ_3]

   dp3  r0, srcNormal, c[CV_LIGHT_DIRECTION]

   slt  r1, r0, CV_HALF
   mul  r2, r1, CV_HALF
   add  r3, CV_ONE, -r1
   mad  destColor, r3, r0, r2

;  mov destColor, CV_ONE

; Output texture coordinates
   mov destTexCoord, srcPosition
```

Sky Sphere Pixel Shader

```
   ps.1.1

   tex t0          // grab base texture

; multiply in sky color
   mad r1, c[CP_SKY_COLOR], 1-t0, t0
; and now lighting color
   mul_sat r0, v0, r1
```

Terrain Pixel Shader

```
   tex t0          // grab base texture
   tex t1          // grab cube map sky sphere texture

;  mov_x2  t1, t1 // uncomment this line to make shadows darker
; and now lighting color
   mul_x2  r1, v0, t0
; multiply in sky clouds shadow
   mul r0, r1, 1-t1
```

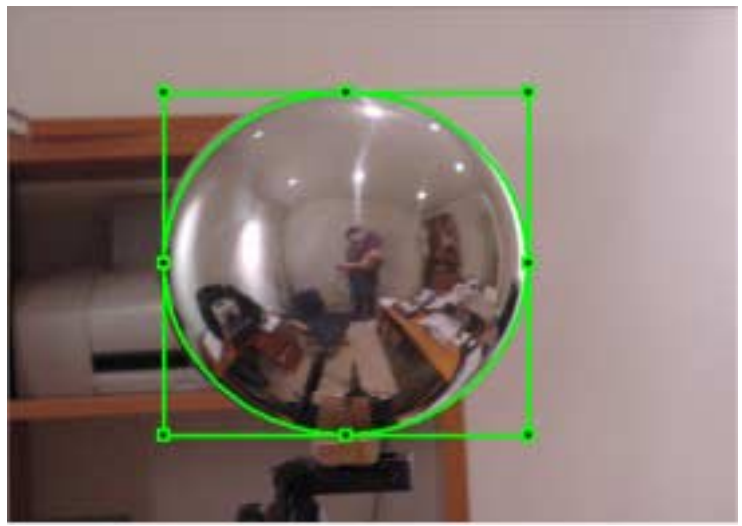
Cloud Cover Demo

More Information

- Games Programming GEMS III Chapter

Low Dynamic Range Images

- Why do I call these low dynamic range?
- AKA Image Base Lighting
- Lighting encoded in cubemap
- Low precision but can be effective for Diffuse lighting



- Take high resolution photographs of mirrored ball from as many as 6 angles

Low Dynamic Range Images

- **Align images into cubemap faces.**

Low Dynamic Range Images

- Run though diffuse convolution filter

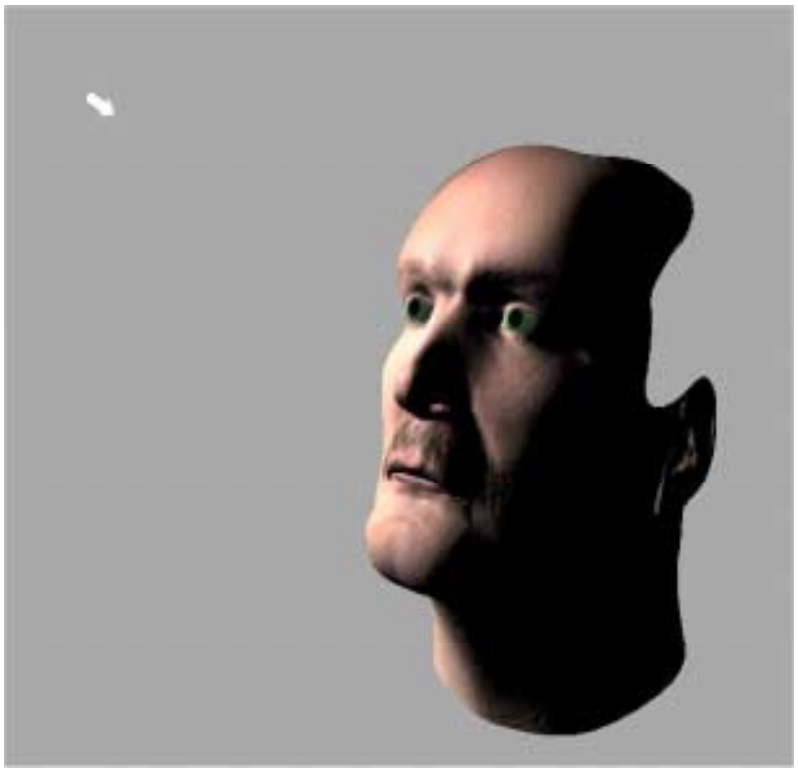
IBR Pixel Shader

```
ps.1.1

tex     t0                  // fetch base texture
tex     t1                  // fetch bump map
tex     t2                  // fetch diffusion map using normal
tex     t3                  // fetch specular using reflection
                            //vector
mul     r0, t0, t2          // base map * diffusion map
mad     r1, t0.a, t3, r0    // specular environment cubemap *
                            // specular base map + previous
mad     r0, r0, t1, r1      // now use diffusion bump map *
                            // diffuse + previous
```

Results / Demo

Traditional Rendering



IBR Rendering

IBR from High Dynamic Range Image

Another IBR Rendering

More Information

- Photo Realistic faces with Vertex and Pixel Shaders – ShaderX book – May 2002 – Wordware Publishing
- DCM Diffuse cube map generator program on ShaderX CD or contact khurley@nvidia.com, if you must have it.

What is High Dynamic Range?

- The human visual system adapts automatically to changes in brightness
- In photography, shutter speed and lens aperture are used to control the amount of light that reaches the film
- HDR imagery attempts to capture the full dynamic range of light in real world scenes
- Measures *radiance = amount of energy per unit time per unit solid angle per unit area* $W / (sr.m^{2})$
- 8 bits is not enough!

Why Do We Need HDR?

- It effectively allows us to change the exposure *after* we've taken/rendered the picture
- Dynamic adaptation effects – e.g. moving from a bright outdoor environment to indoors
- Allows physically plausible image-based lighting
- BRDFs may need high dynamic range
- Enables realistic optical effects – glows around bright light sources, more accurate motion blurs

High Dynamic Range Images

- Eyes sensitivity to luminance suggests we must encode 9,900 values if we use linear steps for luminance
- If not linear then only 460 values are requires (9 bits)
- Eye is very sensitive to luminance changes
- Less sensitive to color changes
- Currently working on idea using pixel shaders in one pass

High Dynamic Range Images

- **Simon Green's talk gives information on OpenGL Implementation**

Conclusion

- Why are pushing this?
  - Movie renders use a combination of procedural and painted textures
  - The more procedural textures, the less time taken for artists
    - Now they can concentrate on the necessary painted textures
  - Re-use – Build a material library that can be used over and over again.

References

- Charles Poynton, *A Technical Introduction to Digital Video*. (New York:Wiley, 1996). Chapter 6, “Gamma” is available online at http://www.inforamp.net/~poynton/PDFs/TIDV/Gamma.pdf (Acrobat PDF format).
- Recovering High Dynamic Range Radiance Maps from Photographs", Debevec, Malik, Siggraph 1997
- http://www.debevec.org/
- ShaderX Book – http://www.shaderx.com
- Games Programming Gems III

Questions?

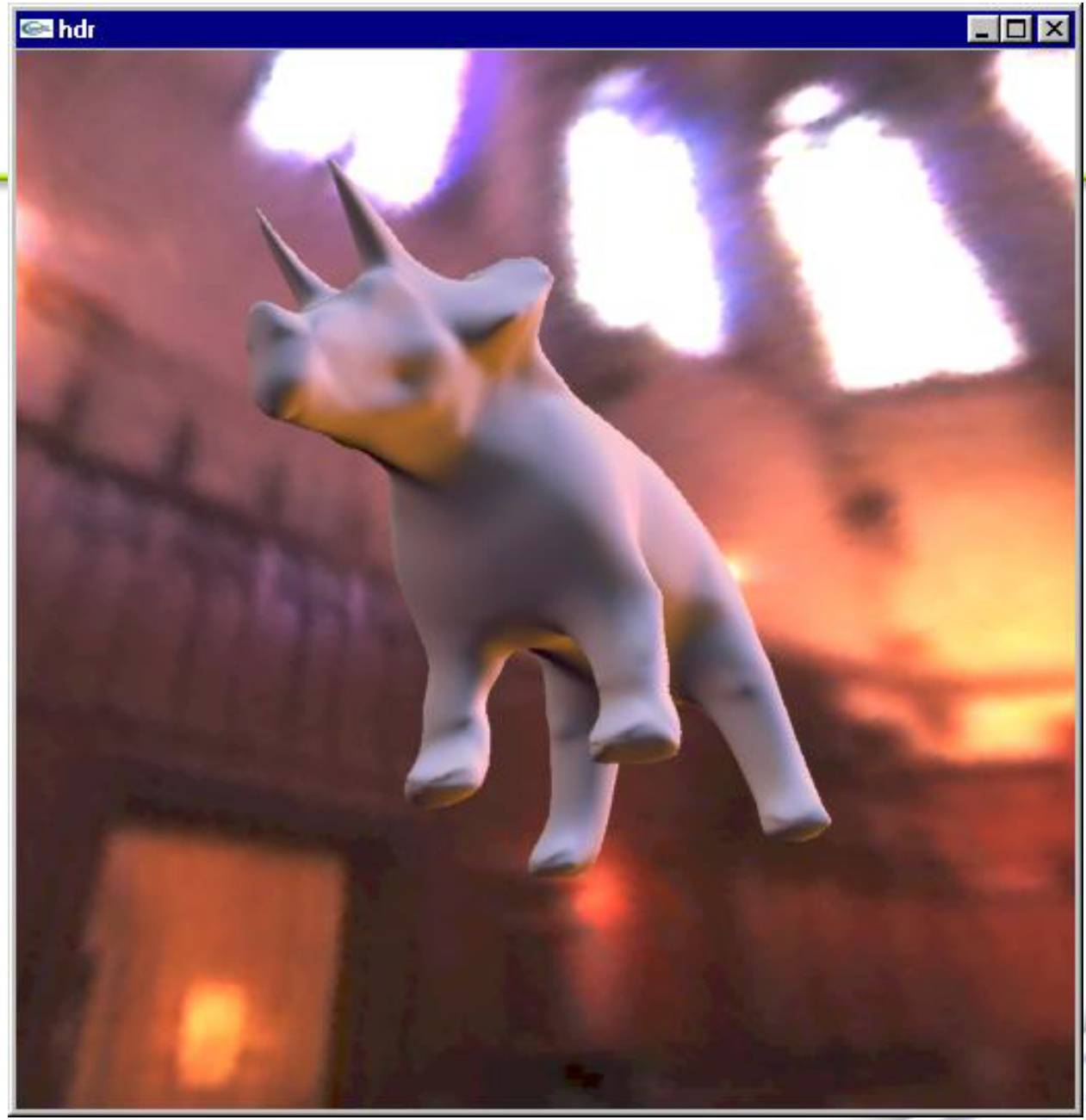
- **E-mail:** khurley@nvidia.com

Extra Slides

- Simon Greene's Slides on Image Based Lighting

Image Based Lighting

- Lighting synthetic objects with “real” light
- An environment map represents all light arriving at a point for each incoming direction
- By convolving (blurring) an environment map with the diffuse reflection function (N.L) we can create a diffuse reflection map
- Indexed by surface normal N, this gives the sum of N.L for all light sources in the hemisphere
- Very slow to create
- Low freq - cube map can be small - e.g. 32x32x6
- HDRShop will do this for you



References

- "Recovering High Dynamic Range Radiance Maps from Photographs", Debevec, Malik, Siggraph 1997
- "Real-time High Dynamic Range Texture Mapping", Cohen, Tchou, Hawkins, Debevec, Eurographics Rendering Workshop 2001
- “Illumination and Reflection Maps: Simulated Objects in Simulated and Real Environments”, Gene S. Miller and C. Robert Hoffman, Siggraph 1984 Course Notes for Advanced Computer Graphics Animation
- "Real Pixels", Greg Ward, Graphics Gems II P.80-83
- http://www.debevec.org/