VGP352 – Week 5

- Agenda:
  - Quiz #2!
  - Assigments:
    - Presentation of reading
    - Assignment #2, parts 1 and 2 due
  - Anisotropic reflection
    - What is it?
    - Anisotropic BRDFs
      - Ward
      - Ashikhmin
  - Implementing BRDFs in shaders

Isotropy

- From Wikipedia:

  “Isotropy is ... property of being independent of direction. Isotropic radiation has the same intensity regardless of the direction of measurement, and an isotropic field exerts the same action regardless of how the test particle is oriented”

Isotropy

- From Wikipedia:

  "Isotropy is ... property of being independent of direction. Isotropic radiation has the same intensity regardless of the direction of measurement, and an isotropic field exerts the same action regardless of how the test particle is oriented"

- For lighting, this means that as long as the angle between N, L, and V remain constant, the lighting will remain constant.

Anisotropy

- From Wikipedia:

  “Anisotropy ... is the property of being directionally dependent. Something which is anisotropic may appear different or have different characteristics in different directions. An example is the light coming through a polarising (sic) lens.”

Anisotropic Reflection

- Some materials reflect light differently based on the orientation of the light and viewer with respect to the surface
  - None of our current lighting models support this
- Consider this fuzzy blanket
  - The angle of the light and the camera is constant, but the material has rotated within the plane

Anisotropic Reflection

- What additional information is needed to implement an anisotropic lighting model?
  - All our current models only use N, L, and V
  - This gives no way to know the relative orientation of the surface w.r.t. the light or the viewer

Anisotropic Reflection

- What additional information is needed to implement an anisotropic lighting model?
  - All our current models only use N, L, and V
  - This gives no way to know the relative orientation of the surface w.r.t. the light or the viewer
- The surface tangent!
  - If V' is the projection of V onto the plane containing T and B, $arccos(V' \bullet T)$ is the relative orientation angle

Banks Lighting Model

- Simple variation of Phong lighting that introduces anisotropy:

$$L_o = \left(K_d (L \cdot L_N) + K_s ((L_T \cdot V_T) - |L_N||V_N'|)^n \right) \times (L \cdot N)$$

  - L_N and V_N are the projections of L and V onto N
  - L_T and V_T are the projections of L and V onto T

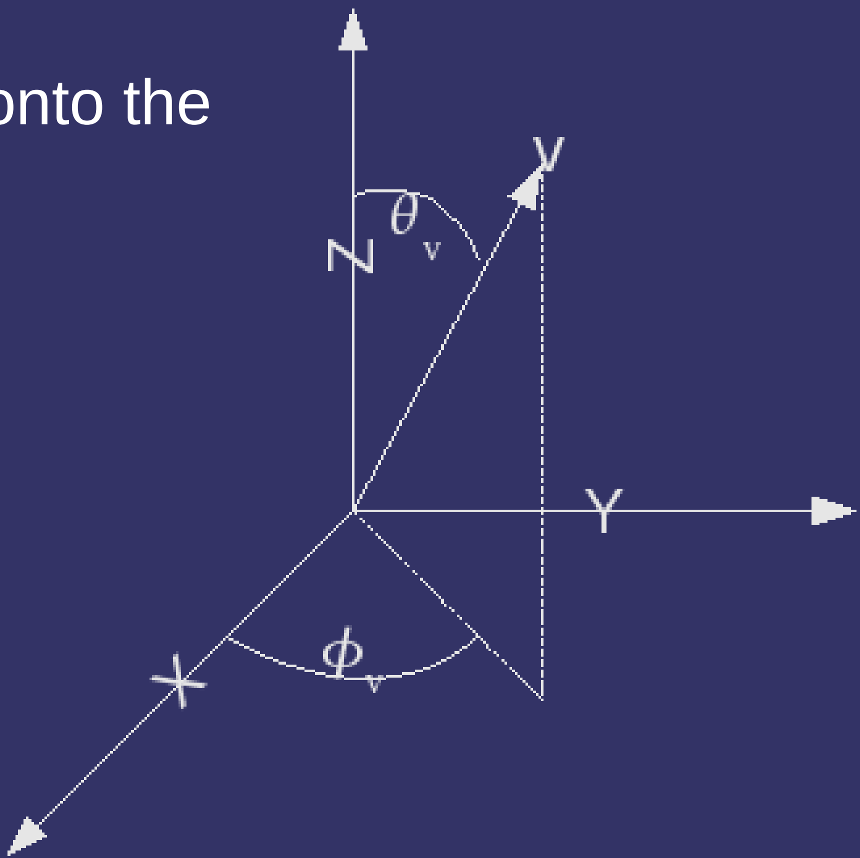
Ward's Anisotropic Model

- Ward defines a coordinate system with the surface normal as the positive Z-axis
  - The X-axis and Y-axis are mapped to the tangent and bitangent

θ_v is the angle between V and the Z-axis

ϕ_v is the angle between the projection of V onto the X/Y plane and the X-axis

The projection is easy: set $V_z = 0$



4-February-2008

Ward's Anisotropic Model

$$f(\omega_i, \omega_o) = \frac{K_d}{\pi} + \frac{K_s}{4\pi\alpha_x\alpha_y\sqrt{\cos\theta_i\cos\theta_o}} e^{-\tan^2\theta_H\left(\frac{\cos^2\phi_H}{\alpha_x^2}\frac{\sin^2\phi_H}{\alpha_y^2}\right)}$$

- α_x and α_y control the width of the highlight in the two principal directions
  - $\alpha_x = \alpha_y$ the reflection is isotropic
  - $\tan^2\theta = (1 - \cos^2\theta) / \cos^2\theta$
  - $\sin^2\theta = 1 - \cos^2\theta$

Ashikhmin Model

$$f_s(\omega_i,\omega_o)=\frac{\sqrt{(n_x+1)(n_y+1)}}{8\pi}\frac{(N\cdot H)^{n_x\cos^2\phi_H+n_y\sin^2\phi_H}}{(H\cdot\omega)\,max((N\cdot\omega_i),(N\cdot\omega_o))}F(\omega\cdot H)$$

- Most of the notation is the same as on the previous slides
  - This differs from the notation in Ashikhmin's paper
- n_x and n_y are Phong-like exponents that control the shape of the specular lobe
  - Roughly analogous to α_x and α_y in Ward's model
- $F(\theta)$ is the Fresnel term

Ashikhmin Model

$$f_d(\omega_i,\omega_o)=\frac{28K_d}{23\pi}(1-F(0))\left(1-\left(1-\frac{N\cdot\omega_i}{2}\right)^5\right)\left(1-\left(1-\frac{N\cdot\omega_o}{2}\right)^5\right)$$

- *F(0)* is the Fresnel term at an angle of 0°
- The strange constant factor is “designed to ensure energy conservation.”

References

Ashikhmin, M., Shirley, P. "An Anisotropic Phong BRDF Model" Journal of Graphics Tools, v.5, no. 2 (2000), pp.25-32. http://www.cs.utah.edu/~michael/brdfs/

Break

Implementing BRDFs in Shaders

- BRDF formulations assume integration over all incoming light in the positive hemisphere
  - This is clearly impractical in real-time

Implementing BRDFs in Shaders

- BRDF formulations assume integration over all incoming light in the positive hemisphere
  - This is clearly impractical in real-time
  - Not very practical on off-line rendering either!
- 4 high-level strategies:
  - Only implement point lights

Implementing BRDFs in Shaders

- BRDF formulations assume integration over all incoming light in the positive hemisphere
  - This is clearly impractical in real-time
  - Not very practical on off-line rendering either!
- 4 high-level strategies:
  - Only implement point lights
    - Direct implementation
    - Factorization

Implementing BRDFs in Shaders

- BRDF formulations assume integration over all incoming light in the positive hemisphere
  - This is clearly impractical in real-time
  - Not very practical on off-line rendering either!
- 4 high-level strategies:
  - Only implement point lights
    - Direct implementation
    - Factorization
  - Environment map lighting

Implementing BRDFs in Shaders

- BRDF formulations assume integration over all incoming light in the positive hemisphere
  - This is clearly impractical in real-time
  - Not very practical on off-line rendering either!
- 4 high-level strategies:
  - Only implement point lights
    - Direct implementation
    - Factorization
  - Environment map lighting
    - Prefiltering
    - Monte Carlo sampling

Point Light: Direct Implementation

- *Very* straightforward method

Point Light: Direct Implementation

- *Very* straightforward method.
- Use L for ω_i and V for ω_o and directly implement the math
  - We already do this for Phong lighting
  - More complex lighting equations can be prohibitively expensive
  - Since we're *not* integrating over the hemisphere, remove the $1/\pi$ from the equation!

Factorization

- Expensive equations can be broken down (factored) into sums or products of functions of fewer variables
  - Each input vector (i.e., V, L, H, N, etc.) becomes an input to one function
  - Each function is stored in an environment map
  - This technique works really well for sampled BRDFs
- Using two textures, the Poulin-Fournie anisotropic satin BRDF can be implemented as:

$$\alpha\, p(V)\, q(H)\, p(L)$$

  - $p()$ and $q()$ represent texture look-ups and α is a special scaling factor

References

University of Waterloo Factored BRDF Repository: http://www.cgl.uwaterloo.ca/Projects/rendering/Shading/database.html

Michael D. McCool, Jason Ang, Anis Ahmad, *Homomorphic Factorization of BRDFs for High-Performance Rendering*, SIGGAPH 2001, August 12-17, 2001. http://www.cgl.uwaterloo.ca/Projects/rendering/Papers/

Environment Maps

- Env. maps present additional challenges
  - Decent lighting require multiple samples

Environment Maps

- Env. maps present additional challenges
  - Decent lighting require multiple samples
- As with the Phong lighting model, env. maps can be prefiltered using complex BRDFs

Environment Maps

- Env. maps present additional challenges
  - Decent lighting require multiple samples
- As with the Phong lighting model, env. maps can be prefiltered using complex BRDFs
  - Doesn't work well with dynamic env. maps
  - Doesn't work *at all* with aniostropic BRDFs
    - The ideal reflection vector isn't enough information!

Grid Sampling

- Sample the environment map at multiple, prede-termined locations, use the sample vectors and the sampled values in the lighting equation

Grid Sampling

- Sample the environment map at multiple, predetermined locations, use the sample vectors and the sampled values in the lighting equation
  - Might not sample the most important vectors for the lighting equation
    - For most equations, samples closer R are more important

Grid Sampling

- Sample the environment map at multiple, predetermined locations, use the sample vectors and the sampled values in the lighting equation
  - Might not sample the most important vectors for the lighting equation
    - For most equations, samples closer R are more important
  - Might not sample the most important vectors for the environment map
    - If most of the environment map is dark with just a few bright spots, those bright spots are more important
    - This problem is especially difficult to solve

Monte Carlo Integration

- Instead of sampling at regular intervals, sample at pseudo-random locations

Monte Carlo Integration

- Instead of sampling at regular intervals, sample at pseudo-random locations
  - The random locations are determined with a BRDF-dependent probability density function (PDF)
    - Several of the papers from this term, including the reading for next week, include a PDF for the BRDF

Monte Carlo Integration

- Instead of sampling at regular intervals, sample at pseudo-random locations
  - The random locations are determined with a BRDF-dependent probability density function (PDF)
    - Several of the papers from this term, including the reading for next week, include a PDF for the BRDF
  - Still several problems:
    - Generating good random numbers on the GPU
    - Requires quite a few samples
      - Colbert and Křivánek found the around 40 looks good
    - Looks better on smaller polygons

References

Colbert, M. and Křivánek, J. 2007. *Real-time shading with filtered importance sampling.* In ACM SIGGRAPH 2007 Sketches (San Diego, California, August 05 - 09, 2007). SIGGRAPH '07. ACM, New York, NY, 71. http://graphics.cs.ucf.edu/gpusampling/

http://en.wikipedia.org/wiki/Monte_Carlo_integration

Next week...

- Fur rendering
  - Goldman's “Fake Fur Rendering”
    - *Please* read this paper for next week
  - Rendering larger hairs
  - More of the Banks model

Legal Statement

This work represents the view of the authors and does not necessarily represent the view of IBM or the Art Institute of Portland.

OpenGL is a trademark of Silicon Graphics, Inc. in the United States, other countries, or both.

Khronos and OpenGL ES are trademarks of the Khronos Group.

Other company, product, and service names may be trademarks or service marks of others.