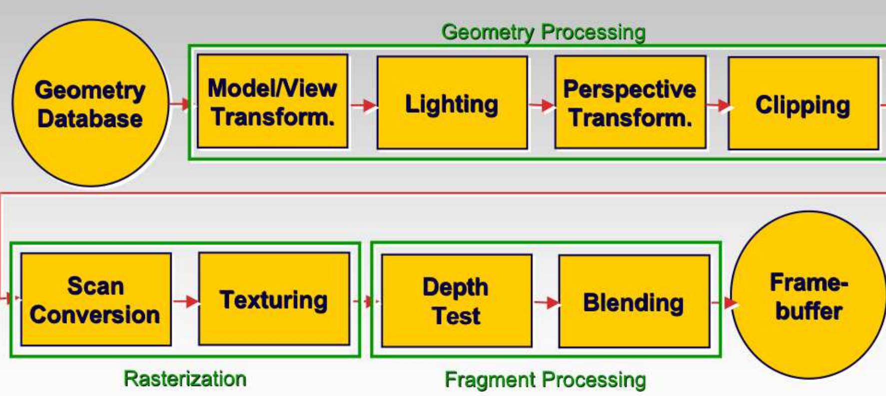
Lighting, Illumination, and Shading

CPSC 314

The Rendering Pipeline



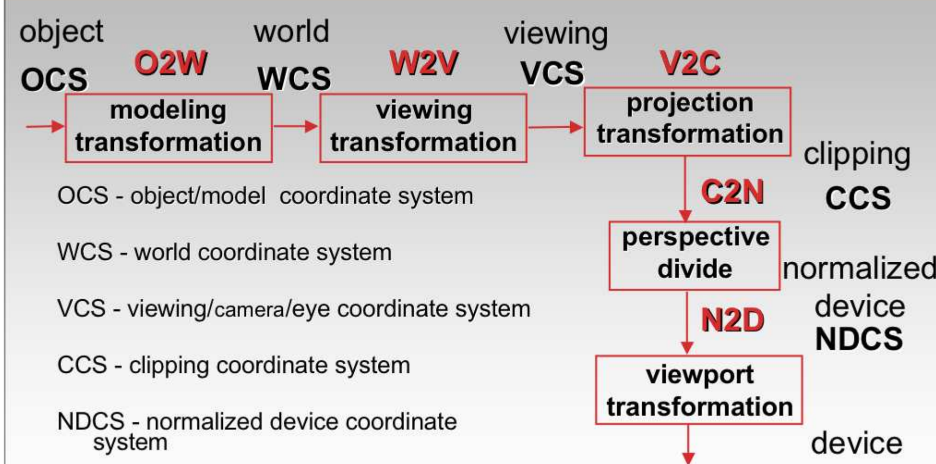
Homogeneous Coordinates

Homogeneous representation of points:

- Add an additional component $w=1$ to all *points*
- All multiples of this vector are considered to represent the same 3D point
- All points are represented as *column vectors*

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \equiv \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \equiv \begin{bmatrix} x \cdot w \\ y \cdot w \\ z \cdot w \\ w \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ z' \\ w \end{bmatrix}, \forall w \neq 0$$

Projective Rendering Pipeline



OCS - object/model coordinate system

WCS - world coordinate system

VCS - viewing/camera/eye coordinate system

CCS - clipping coordinate system

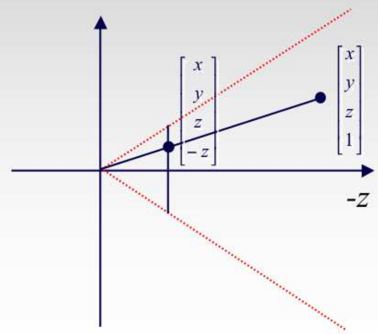
NDCS - normalized device coordinate system

DCS - device/display/screen coordinate system

Perspective Projection

Example:

- Assume image plane at z=-1
- A point $[x,y,z,1]^T$ projects to
  $[-x/z,-y/z,-z/z,1]^T \equiv [x,y,z,-z]^T$



Perspective Projection

Analysis:

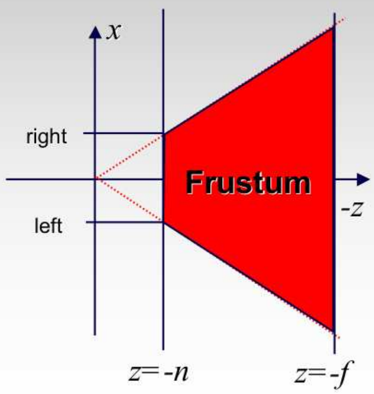
- This is a special case of a general family of transformations called *projective transformations*
- These can be expressed as 4x4 homogeneous matrices!
  - *E.g. in the example:*

$$T\left(\begin{bmatrix}x\\y\\z\\1\end{bmatrix}\right)=\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&-1&0\end{bmatrix}\cdot\begin{bmatrix}x\\y\\z\\1\end{bmatrix}=\begin{bmatrix}x\\y\\z\\-z\end{bmatrix}\equiv\begin{bmatrix}-x/z\\-y/z\\-1\\1\end{bmatrix}$$

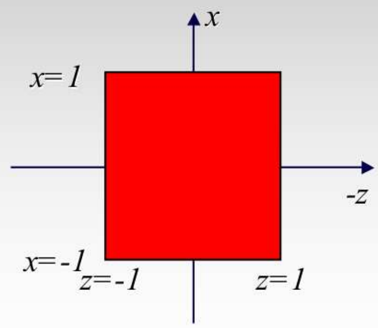
Projective Transformations

OpenGL Convention

Camera coordinates



NDC



Perspective Matrices in OpenGL

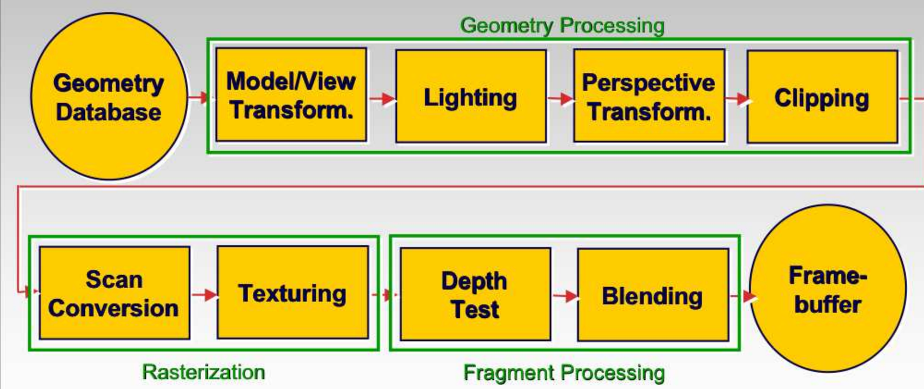
Perspective Matrices:

- glFrustum(left, right, bottom, top, near, far)
  - *Specifies perspective xform (near, far are always positive)*
- glOrtho(left, right, bottom, top, near, far)

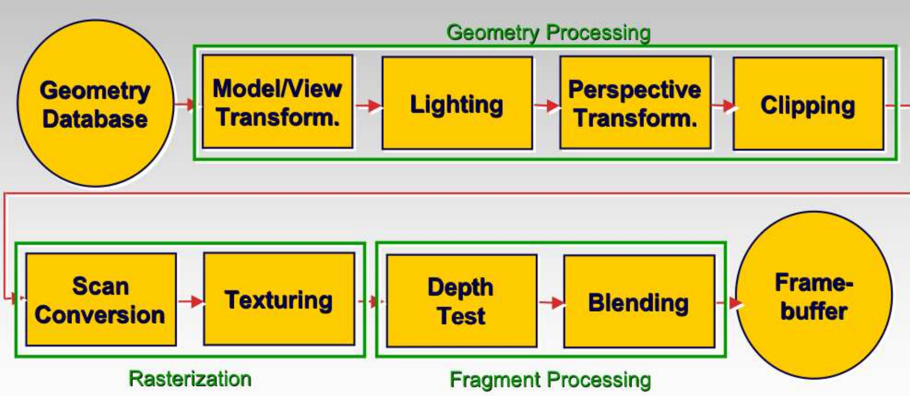
Convenience Functions:

- gluPerspective(fovy, aspect, near, far)
  - *Another way to do perspective*
- gluLookAt(eyeX, eyeY, eyeZ,
centerX, centerY, centerZ,
upX, upY, upZ)
  - *Useful for viewing transform*

The Rendering Pipeline



The Rendering Pipeline



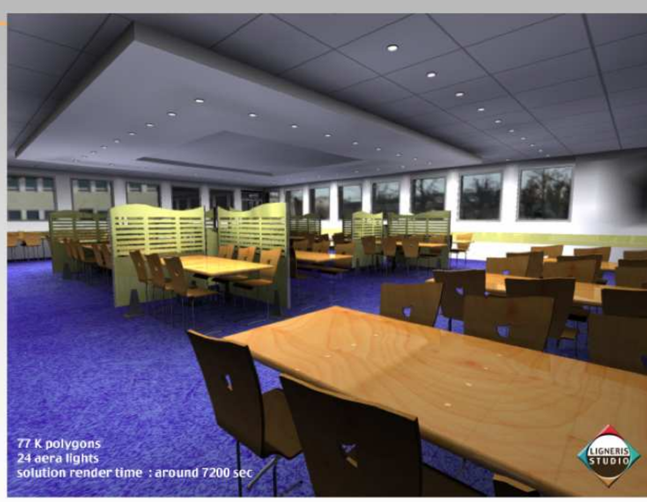
Illumination

Goal

Model interaction of light with matter in a way that appears realistic and is fast

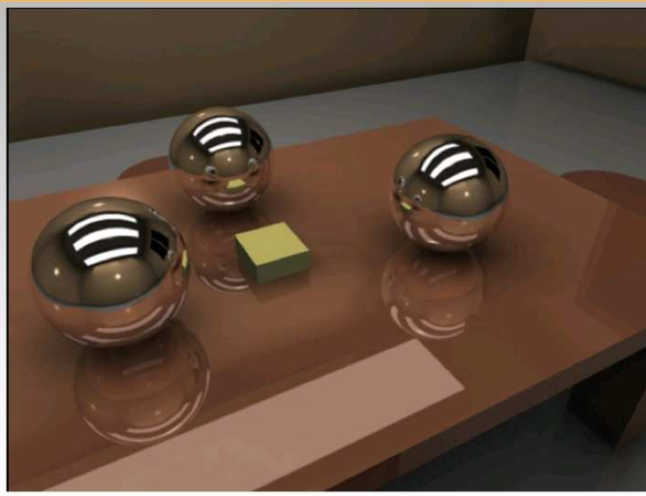
- Phenomenological reflection models
  - *Ignore real physics, approximate the look*
  - *Simple, non-physical*
  - *Phong, Blinn-Phong*
- Physically based reflection models
  - *Simulate physics*
  - *BRDFs: Bidirectional Reflection Distribution Functions*

Photorealistic Illumination



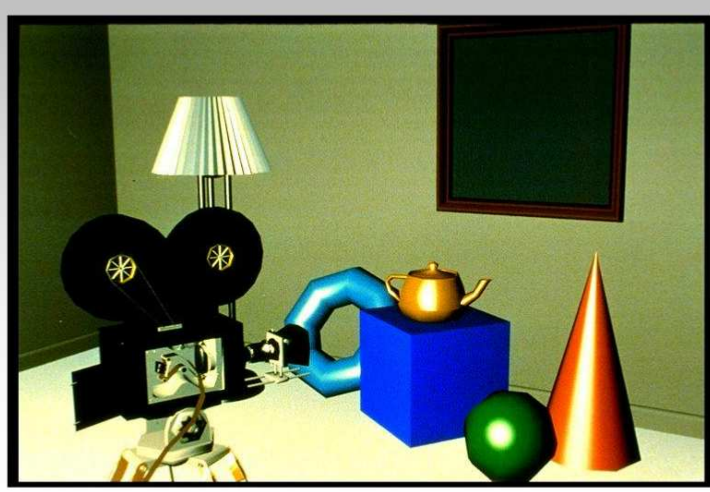
[electricimage.com]

Photorealistic Illumination



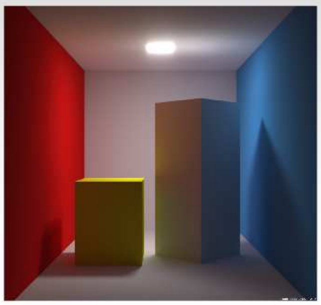
[electricimage.com]

Fast Local Illumination



Illumination

- Transport of energy from light sources to surfaces & points
  - *Includes* **direct** *and* **indirect illumination**



Images by Henrik Wann Jensen

Components of Illumination

Two components

- Light sources and surface properties

Light sources (or emitters)

- Spectrum of emittance (i.e., color of the light)
- Geometric attributes
  - *Position*
  - *Direction*
  - *Shape*
- Directional attenuation
- Polarization

Components of Illumination

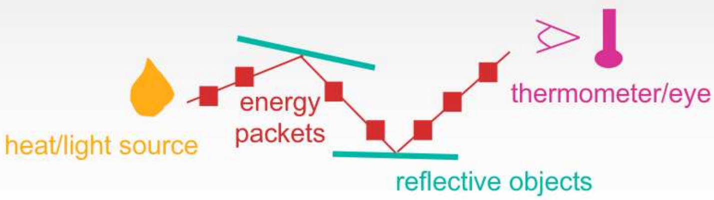
Surface properties

- Reflectance spectrum (i.e., color of the surface)
- Subsurface reflectance
- Geometric attributes
  - *Position*
  - *Orientation*
  - *Micro-structure*

Illumination as Radiative Transfer

Radiative heat transfer analogon

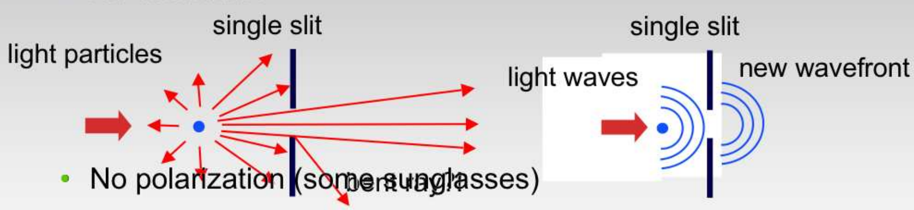
- Substitute light for heat
- Light as packets of energy (photons)
  - *Particles not waves*
- Model light transport as packet flow



Light Transport Assumptions

Geometrical optics:

- Light is photons not waves)
- No diffraction



- No polarization (some sunglasses)
  - *Light of all orientations gets through*
- No interference (packets don't interact)
  - *Which visual effects does this preclude?*

Light Transport Assumptions II

Color approximated by discrete wavelengths

- Quantized approx of dispersion (rainbows)
- Quantized approx of fluorescence (cycling vests)

No propagation media (surfaces in vacuum)

- No
  - *Atmospheric scattering (fog, clouds)*
  - *Refraction (mirages)*
  - *Gravity lenses*
- But methods exist for all these effects

Superposition (lights can be added)

- No nonlinear reflection models
  - *Pretty good assumption (only few non-linear materials)*

Light Sources and Materials

Appearance depends on

- Light sources, locations, properties
- Material (surface) properties
- Viewer position

Local illumination

- Compute at material, from light to viewer

Global illumination (later in course)

- Ray tracing: from viewer into scene
- Radiosity: between surface patches

Illumination in the Rendering Pipeline

Local illumination

- Only models light arriving directly from light source
- No interreflections and shadows
  - *Can be added through tricks, multiple rendering passes*

Light sources

- Simple shapes

Materials

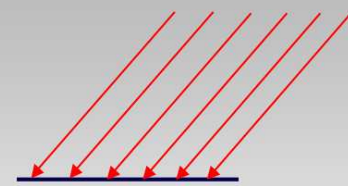
- Simple, non-physical reflection models

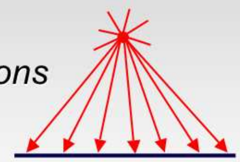
© Wolfgang Heidrich

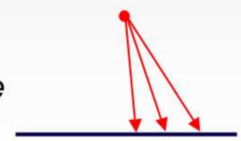
Light Sources

Types of light sources

- Directional/parallel lights
  - *E.g.sun*
  - *Homogeneous vector*
- (Homogeneous) point lights
  - *Same intensity in all directions*
  - *Homogeneous point*
- Spot lights
  - *Limited set of directions*
  - *Point+direction+cutoff angle*



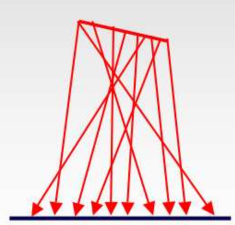




Light Sources

Area lights:

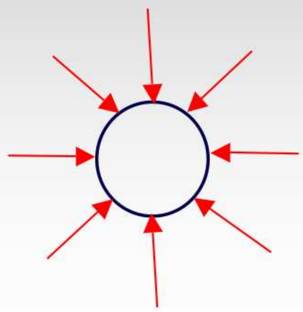
- Light sources with a finite area
- Can be considered a continuum of point lights
- Not available in many rendering systems



Light Sources

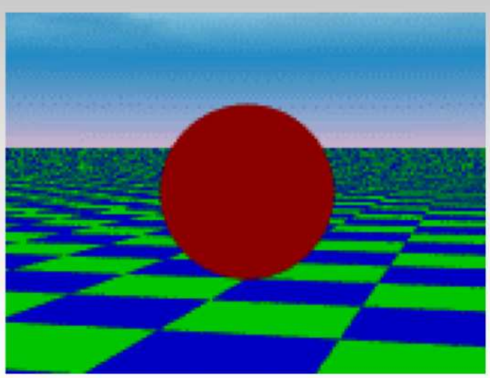
ambient lights

- no identifiable source or direction
- hack for replacing true global illumination
  - *(light bouncing off from other objects)*



Ambient Light Sources

- Scene lit only with an ambient light source



Light Position Not Important

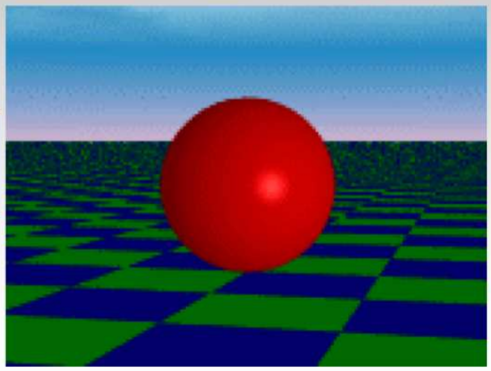
Viewer Position Not Important

Surface Angle Not Important

Directional Light Sources

- Scene lit with directional and ambient light

Surface Angle Important



Light Position Not Important

Viewer Position Not Important

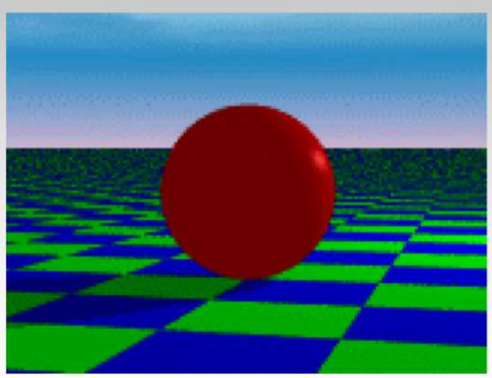
Point Light Sources

- Scene lit with ambient and point light source

Light Position Important

Viewer Position Important

Surface Angle Important



Light Sources

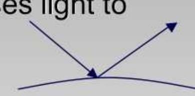
Geometry: positions and directions

- Standard: world coordinate system
  - *Effect: lights fixed wrt world geometry*
  - *Demo: http://www.xmission.com/~nate/tutors.html*
- Alternative: camera coordinate system
  - *Effect: lights attached to camera (car headlights)*
- Points and directions undergo normal model/view transformation

illumination calculations: camera coords

Types of Reflection

- *Specular* (a.k.a. *mirror* or *regular*) reflection causes light to propagate without scattering.

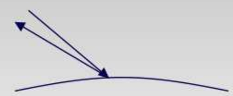


- *Diffuse* reflection sends light in all directions with equal energy.

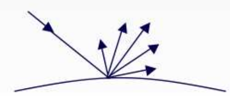
- *Mixed* reflection is a weighted combination of specular and diffuse.

Types of Reflection

- *retro-reflection* occurs when incident energy reflects in directions close to the incident direction, for a wide range of incident directions.



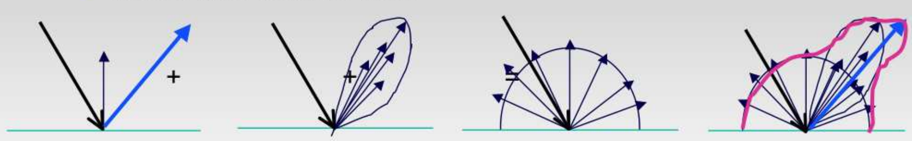
- *gloss* is the property of a material surface that involves mixed reflection and is responsible for the mirror like appearance of rough surfaces.



Reflectance Distribution Model

Most surfaces exhibit complex reflectances

- Vary with incident and reflected directions.
- Model with combination



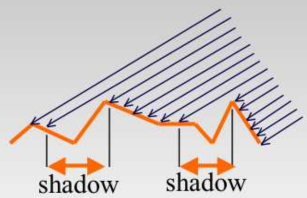
specular + glossy + diffuse =

reflectance distribution

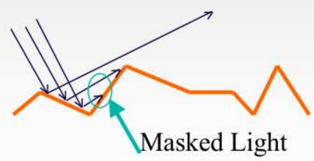
Surface Roughness

- at a microscopic scale, all real surfaces are rough

- cast shadows on themselves



- "mask" reflected light:



Surface Roughness

Notice another effect of roughness:

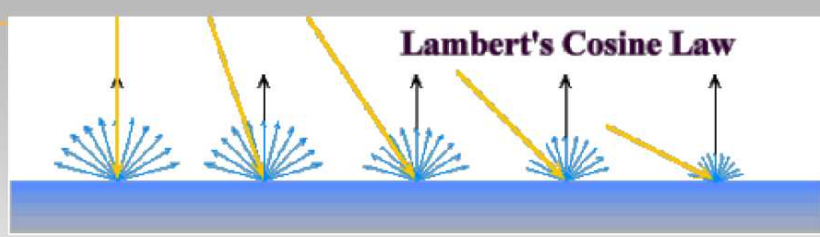
- Each "microfacet" is treated as a perfect mirror.
- Incident light reflected in different directions by different facets.
- End result is mixed reflectance.
  - *Smoother surfaces are more specular or glossy.*
  - *Random distribution of facet normals results in diffuse reflectance.*

Physics of Diffuse Reflection

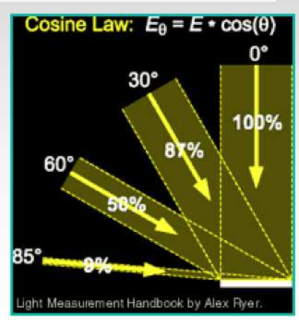
Ideal diffuse reflection

- Very rough surface at the microscopic level
  - *Real-world example: chalk*
- Microscopic variations mean incoming ray of light equally likely to be reflected in any direction over the hemisphere
- Reflected intensity only depends on light direction!

Lambert's "Law"

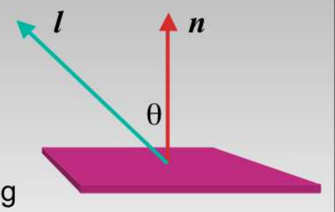


Intuitively: cross-sectional area of the "beam" intersecting an element of surface area is smaller for greater angles with the normal.



Computing Diffuse Reflection

- Depends on angle of incidence: angle between surface normal and incoming light
  - $I_{diffuse} = k_d I_{light} \cos \theta$
- In practice use vector arithmetic
  - $I_{diffuse} = k_d I_{light} (\mathbf{n} \bullet \mathbf{l})$
- Always normalize vectors used in lighting
  - $\mathbf{n}$, $\mathbf{l}$ *should be unit vectors*
- Scalar (B/W intensity) or 3-tuple or 4-tuple (color)
  - k_d: *diffuse coefficient, surface color*
  - I_{light}: *incoming light intensity*
  - $I_{diffuse}$: *outgoing light intensity (for diffuse reflection)*



Diffuse Lighting Examples

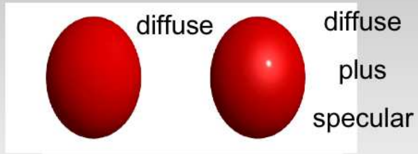
- Lambertian sphere from several lighting angles:

- need only consider angles from 0° to 90°

Specular Reflection

Shiny surfaces exhibit specular reflection

- Polished metal
- Glossy car finish



Specular highlight

- Bright spot from light shining on a specular surface

View dependent

- Highlight position is function of the viewer's position

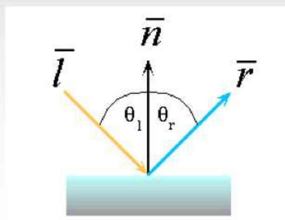
Physics of Specular Reflection

- At the microscopic level a specular reflecting surface is very smooth
- Thus rays of light are likely to bounce off the microgeometry in a mirror-like fashion
- the smoother the surface, the closer it becomes to a perfect mirror

Optics of Reflection

Reflection follows **Snell's Law:**

- Incoming ray and reflected ray lie in a plane with the surface normal
- Angle the reflected ray forms with surface normal equals angle formed by incoming ray and surface normal

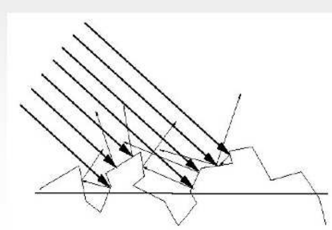


$$\theta_{(l)ight} = \theta_{(r)eflection}$$

Non-Ideal Specular Reflectance

- Snell's law applies to perfect mirror-like surfaces, but aside from mirrors (and chrome) few surfaces exhibit perfect specularity
- How can we capture the "softer" reflections of surface that are glossy, not mirror-like?
- One option: model the microgeometry of the surface and explicitly bounce rays off of it

or…

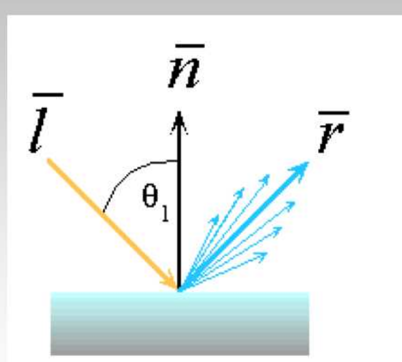


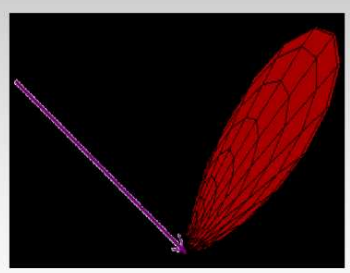
Empirical Approximation

- We expect most reflected light to travel in direction predicted by Snell's Law
- But because of microscopic surface variations, some light may be reflected in a direction slightly off the ideal reflected ray
- As angle from ideal reflected ray increases, we expect less light to be reflected

Empirical Approximation

Angular falloff





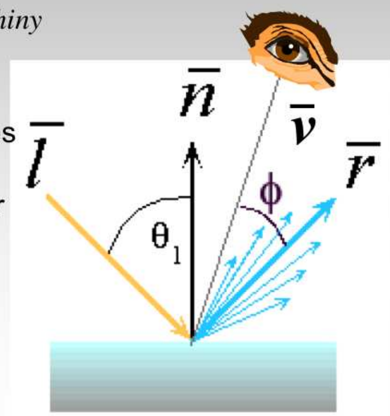
how might we model this falloff?

Phong Lighting

Most common lighting model in computer graphics

– *(Phong Bui-Tuong, 1975)*

$$\mathbf{I}_{\mathbf{specular}} = \mathbf{k}_s \mathbf{I}_{\mathbf{light}} (\cos\phi)^{n_{shiny}}$$



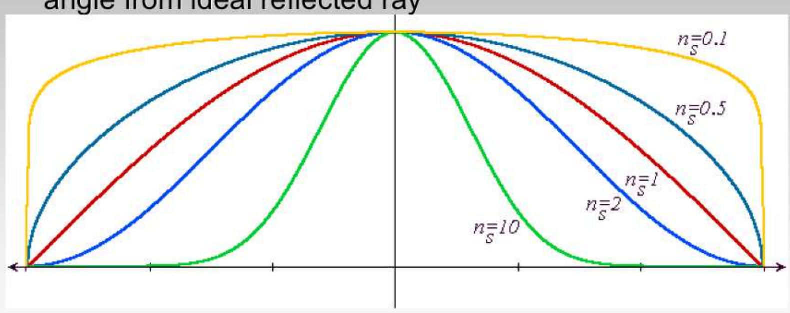
n_{shiny} : purely empirical constant, varies rate of falloff

k_s: specular coefficient, highlight color

no physical basis, works ok in practice

Phong Lighting: The n_{shiny} Term

- Phong reflectance term drops off with divergence of viewing angle from ideal reflected ray

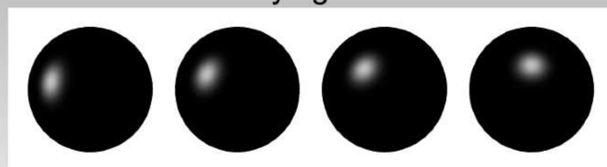


what does this term control, visually?

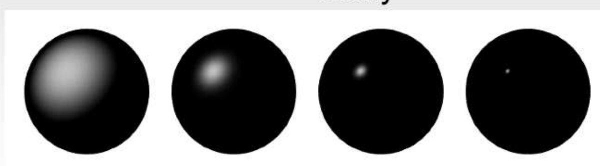
Viewing angle – reflected angle

Phong Examples

varying l



varying n_{shiny}

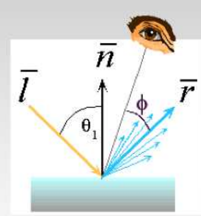


Calculating Phong Lighting

compute cosine term of Phong lighting with vectors

$$\mathbf{I}_{\mathbf{specular}} = \mathbf{k}_s \mathbf{I}_{\mathbf{light}} (\mathbf{v} \bullet \mathbf{r})^{n_{shiny}}$$

- v: unit vector towards viewer/eye
- r: ideal reflectance direction (unit vector)
- k_s: specular component
  - *highlight color*
- I_{light}: incoming light intensity

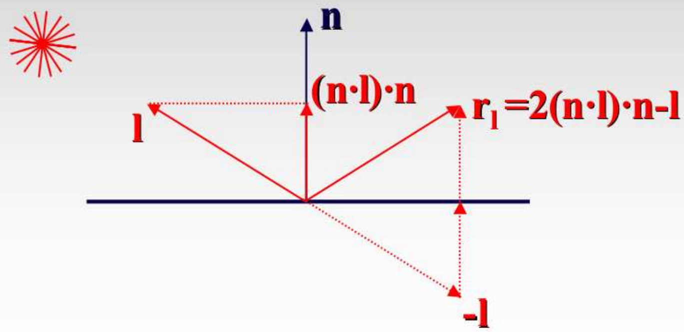


how to efficiently calculate r ?

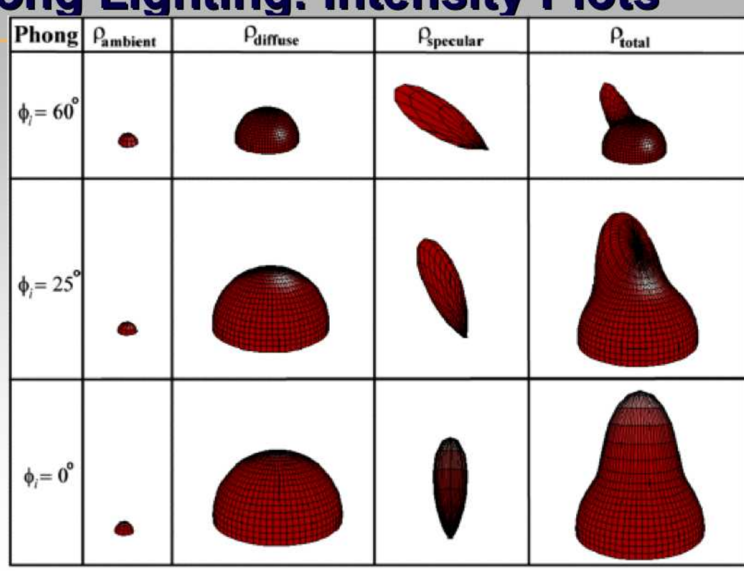
Computing the Reflected Direction

Specular/Glossy

- Computing reflection direction $\mathbf{r_l}$ of $\mathbf{l}$
  - $\mathbf{n}$ *and* $\mathbf{l}$ *are unit length!*



Phong Lighting: Intensity Plots

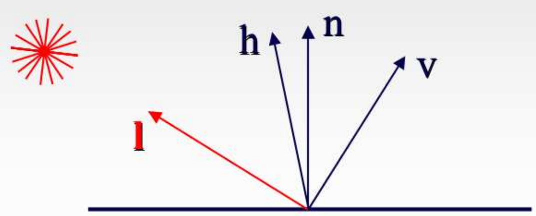


Alternative Model

Blinn-Phong model (Jim Blinn, 1977)

- Variation with better physical interpretation
  - **h**: *halfway vector;* r*: roughness*

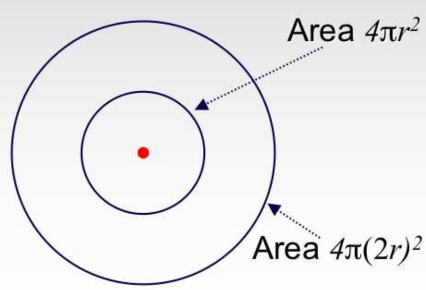
$$I_{out}(\mathbf{x}) = k_s \cdot (\mathbf{h} \cdot \mathbf{n})^{1/r} \cdot I_{in}(\mathbf{x}); \text{ with } \mathbf{h} = (\mathbf{l} + \mathbf{v})/2$$



Light Source Falloff

Quadratic falloff

- Brightness of objects depends on power per unit area that hits the object
- The power per unit area for a point or spot light decreases quadratically with distance



Light Source Falloff

Non-quadratic falloff

- Many systems allow for other falloffs
- Allows for faking effect of area light sources
- OpenGL / graphics hardware
  - I_o: *intensity of light source*
  - $\mathbf{x}$: *object point*
  - r: *distance of light from* $\mathbf{x}$

$$I_{in}(\mathbf{x}) = \frac{1}{ar^2 + br + c} \cdot I_0$$

Lighting Review

Lighting models

- Ambient
  - *Normals don't matter*
- Lambert/diffuse
  - *Angle between surface normal and light*
- Phong/specular
  - *Surface normal, light, and viewpoint*

Lighting in OpenGL

Light source: amount of RGB light emitted

- Value represents percentage of full intensity
  E.g., (1.0,0.5,0.5)
- Every light source emits ambient, diffuse, and specular light

Materials: amount of RGB light reflected

- Value represents percentage reflected
  e.g., (0.0,1.0,0.5)

Interaction: multiply components

- Red light (1,0,0) x green surface (0,1,0) = black (0,0,0)

Lighting in OpenGL

```
glLightfv(GL_LIGHT0, GL_AMBIENT, amb_light_rgba );
glLightfv(GL_LIGHT0, GL_DIFFUSE, dif_light_rgba );
glLightfv(GL_LIGHT0, GL_SPECULAR, spec_light_rgba );
glLightfv(GL_LIGHT0, GL_POSITION, position);
glEnable(GL_LIGHT0);

glMaterialfv( GL_FRONT, GL_AMBIENT, ambient_rgba );
glMaterialfv( GL_FRONT, GL_DIFFUSE, diffuse_rgba );
glMaterialfv( GL_FRONT, GL_SPECULAR, specular_rgba );
glMaterialfv( GL_FRONT, GL_SHININESS, n );
```

Coming Up

Thursday:

- Shading (Part 1)

Tuesday:

- Shading (Part 2)
- Quiz 1