Computer Graphics

Cours 2:

- Couleurs
- Illumination/éclairage
- Textures

- Ray tracing...

Global Illumination

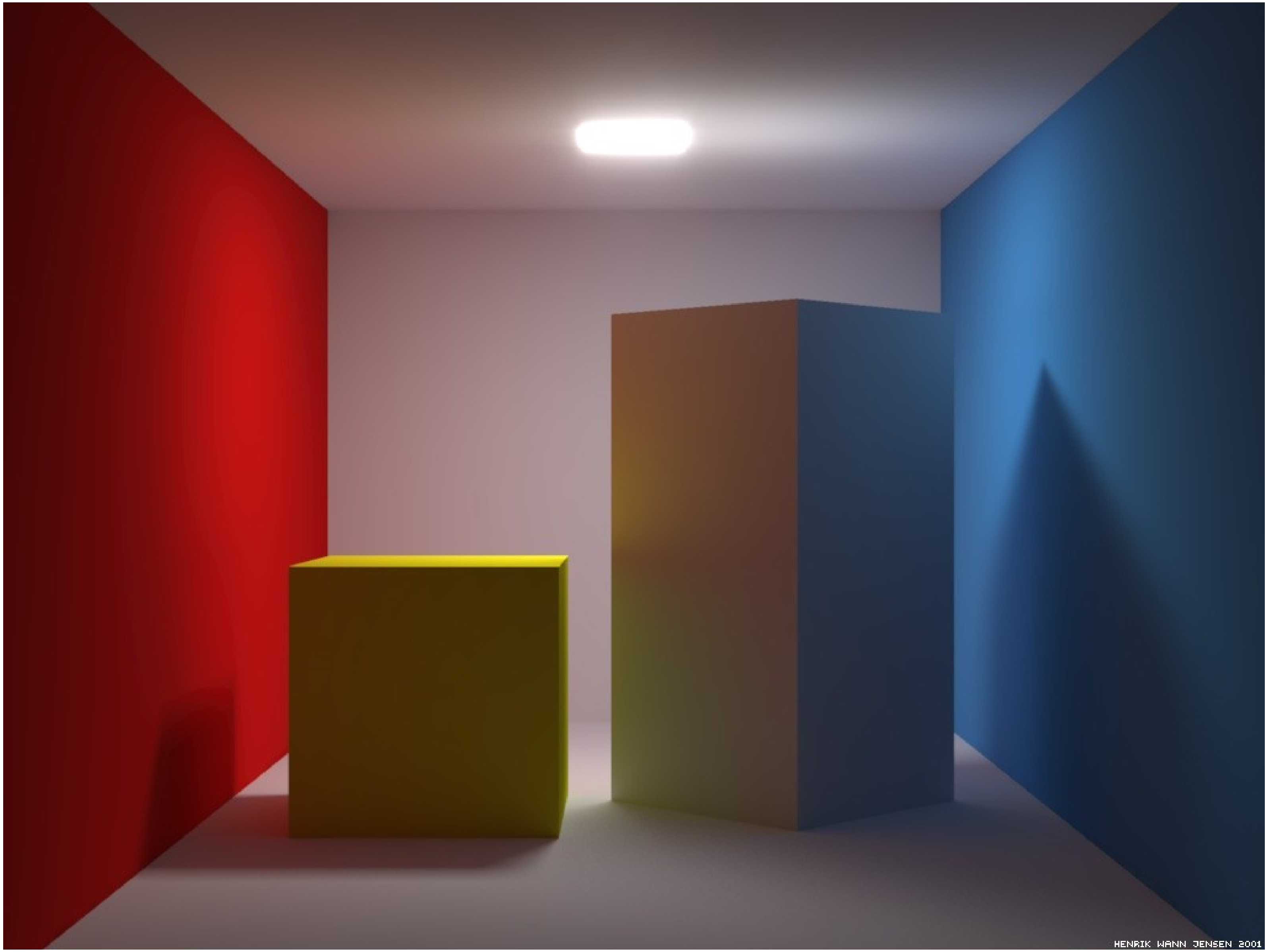
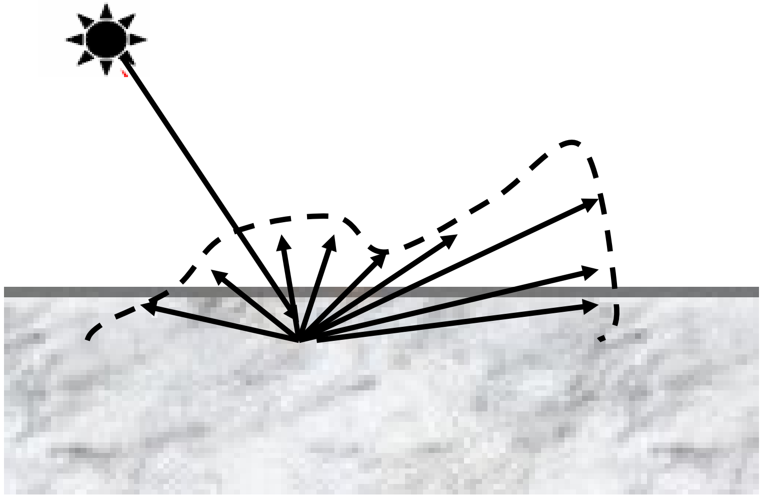


Image taken from http://graphics.ucsd.edu/~henrik/images/cbox.html

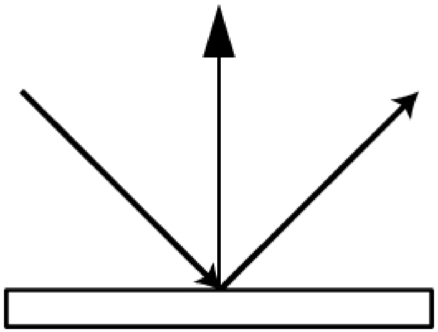
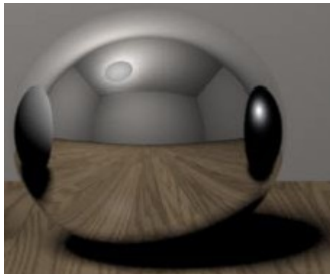
Reflection Models

- Definition: Reflection is the process by which light incident on a surface interacts with the surface such that it leaves on the incident side without change in frequency.

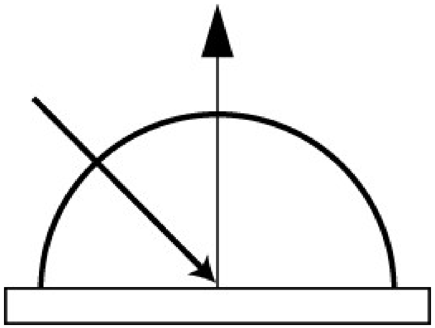
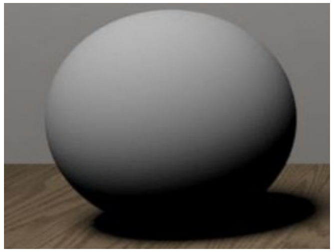


Types of Reflection Functions

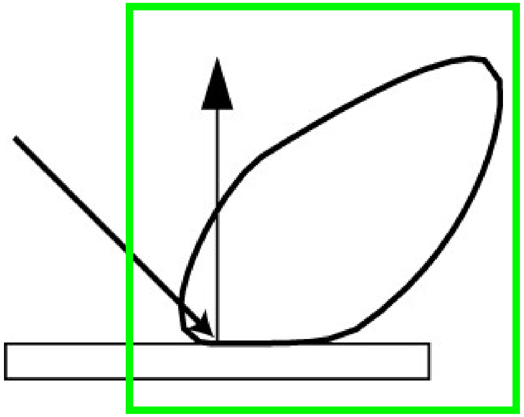
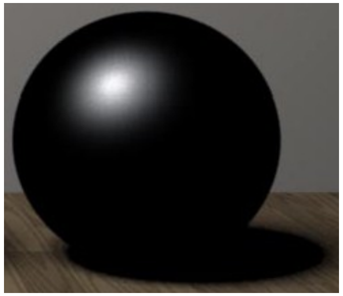
- Ideal Specular
  - Reflection Law
  - Mirror

- Ideal Diffuse
  - Lambert's Law
  - Matte

- Specular
  - Glossy
  - Directional diffuse

Illumination Model

- Ambient Light
  - Uniform light caused by secondary reflections
- Diffuse Light
  - Light scattered equally in all directions
- Specular Light
  - Highlights on shiny surfaces

Ambient Light

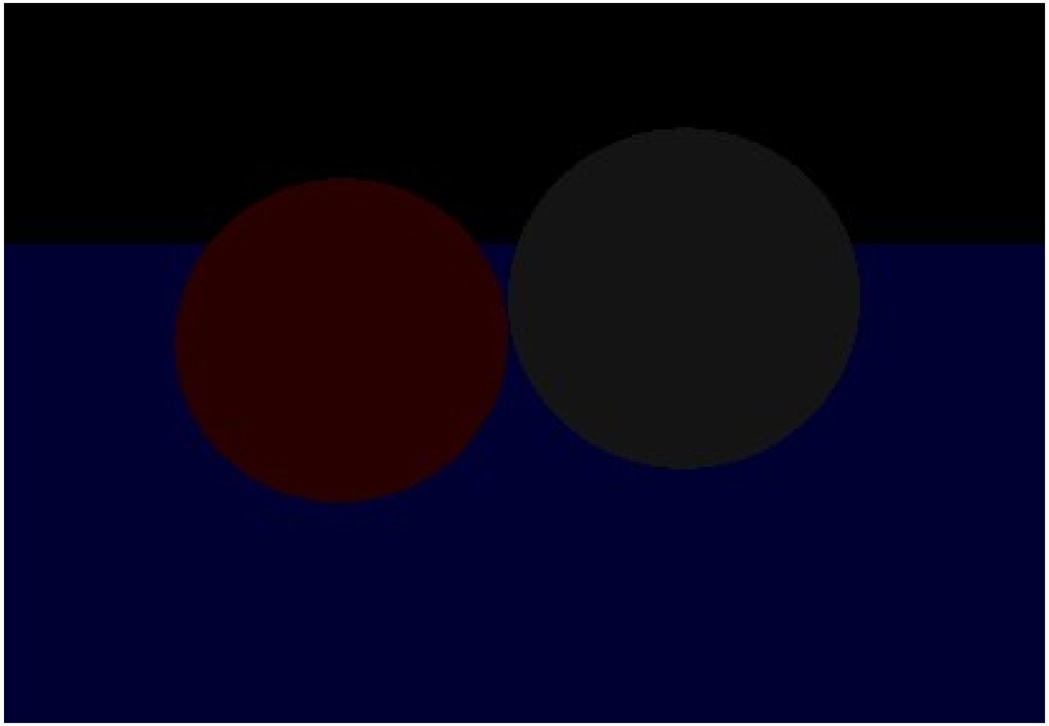
$$I = k_a A$$

- A =intensity of ambient light
- k_a=ambient reflection coefficient

- Really 3 equations! (Red, Green, Blue)
- Accounts for indirect illumination

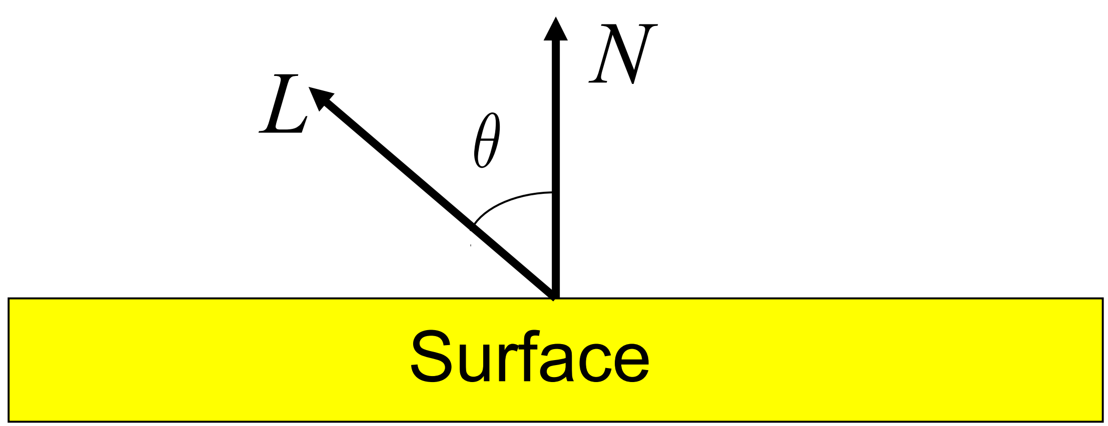
Total Illumination

$$I = k_a A$$



Diffuse Light

- Assumes that light is reflected equally in all directions
- Handles both local and infinite light sources
  - Infinite distance: L doesn't change
  - Finite distance: must calculate L for each point on surface

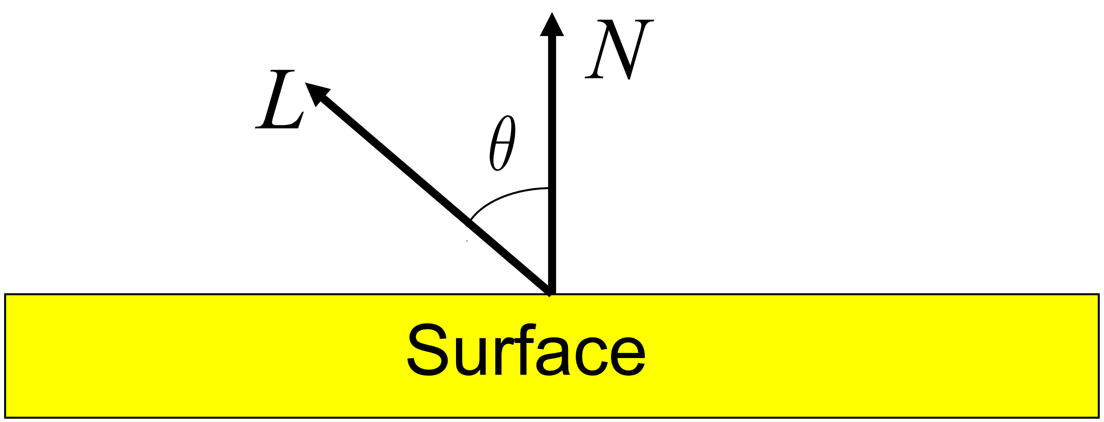


Diffuse Light

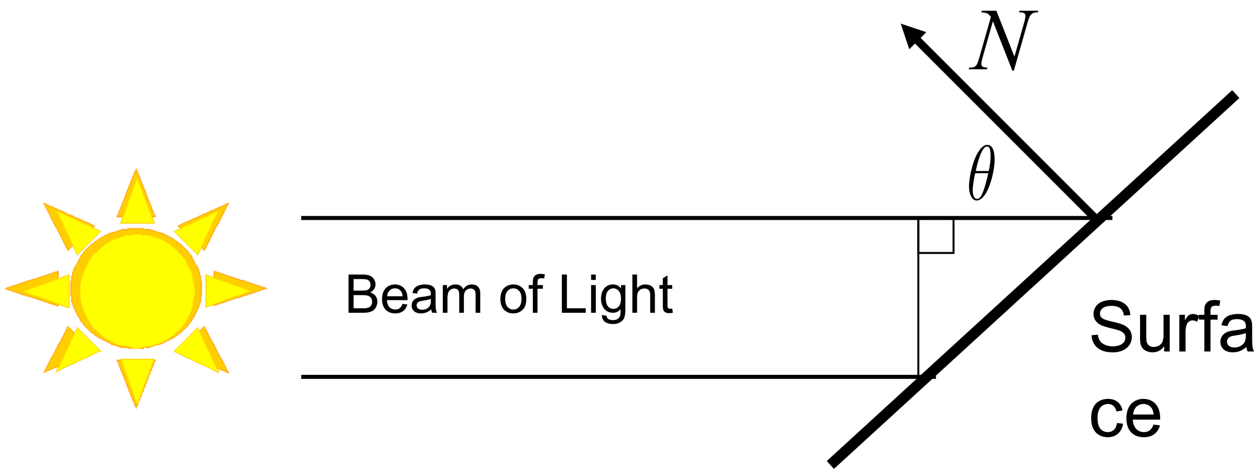
$$I = C\, k_d \cos(\theta) = C\, k_d (L \cdot N)$$

- C = intensity of point light source
- k_d = diffuse reflection coefficient
- θ = angle between normal and direction to light

$$\cos(\theta) = L \cdot N$$



Lambert's Law

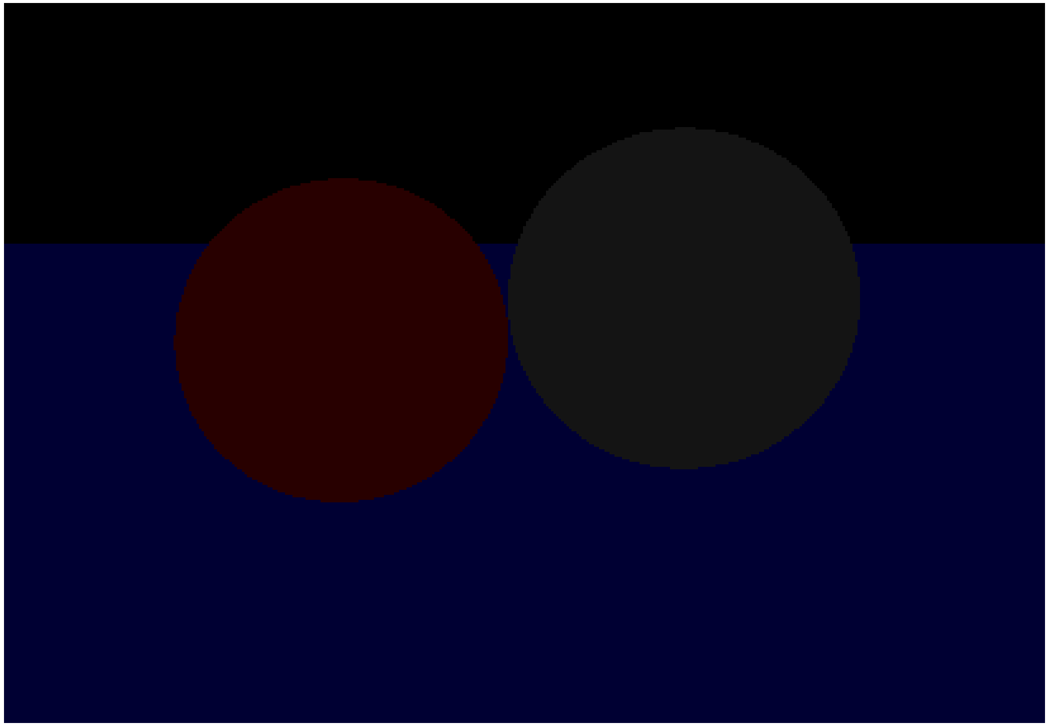


$$I = \frac{Light}{Area} = \frac{Beam\ \ Width \times I_{source}}{Surface\ \ Area} = I_{source}\left(L \cdot N\right)$$

$$\frac{Beam\ \ Width}{Surface\ \ Area} = \cos\left(\theta\right)$$

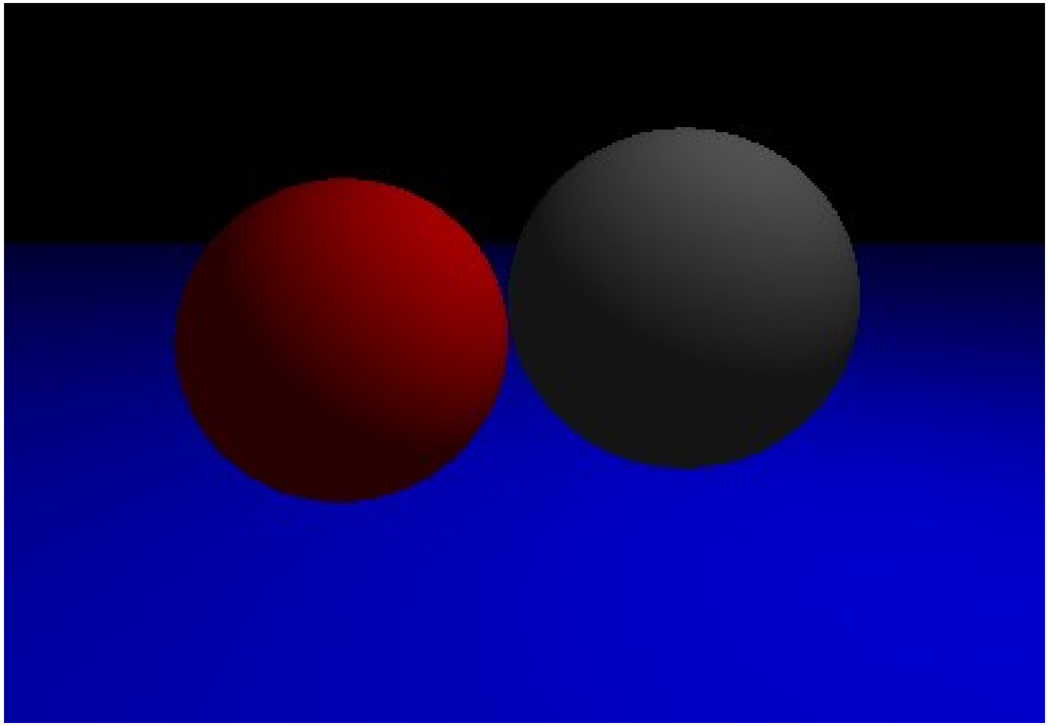
Total Illumination

$$I = k_a A$$



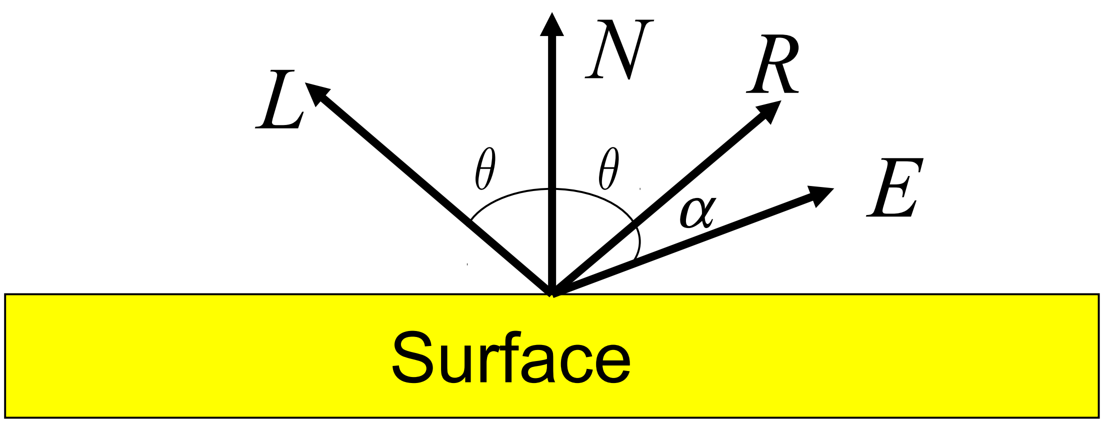
Total Illumination

$$I = k_a A + k_d C\left(L \cdot N\right)$$



Specular Light

- Perfect, mirror-like reflection of light from surface
- Forms highlights on shiny objects (metal, plastic)

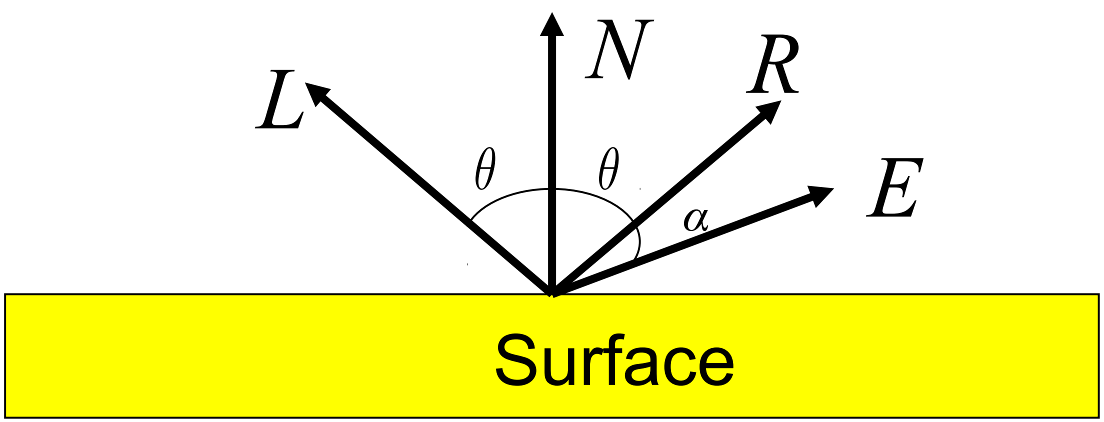


Specular Light: Phong model

$$I = C\ k_s \cos^n(\alpha) = C\ k_s (R \cdot E)^n$$

- C = intensity of point light source
- k_s = specular reflection coefficient
- α =angle between reflected vector (R) and eye (E)
- n =specular coefficient

$$\cos(\alpha) = R \cdot E$$

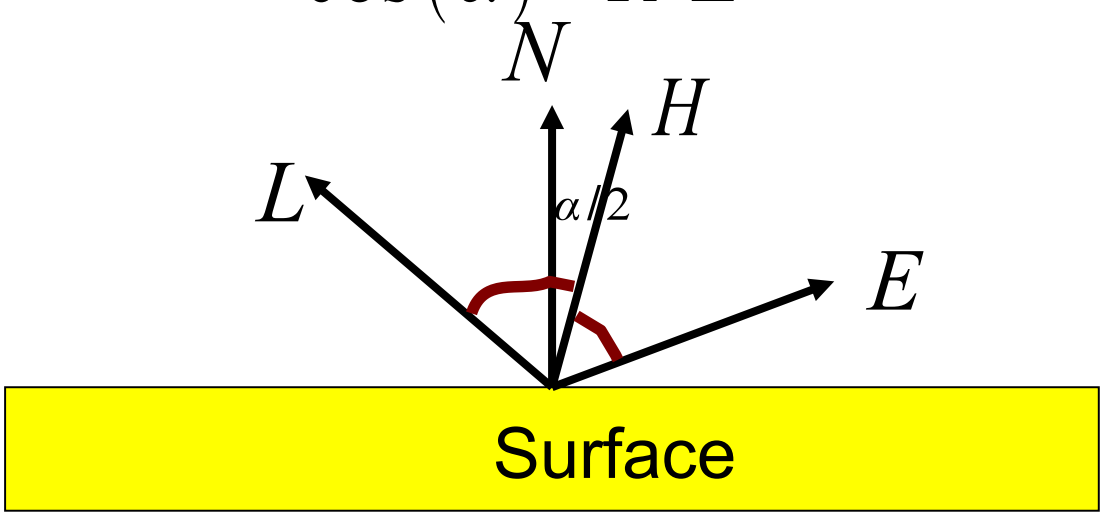


Specular Light: Blinn-Phong

$$I = C\, k_s \cos^n(\alpha/2) = C\, k_s (N \cdot H)^n$$

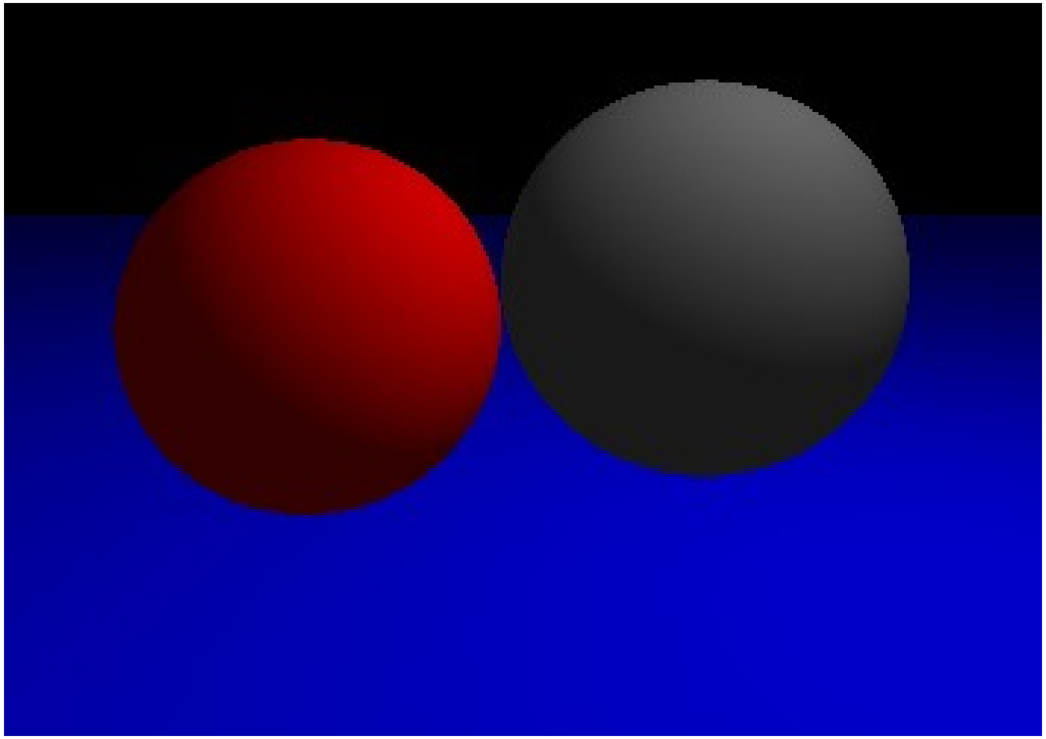
- C = intensity of point light source
- k_s = specular reflection coefficient
- $\alpha/2$ = angle between bissector vector (H) and normal (N)
- n = specular coefficient

$$\cos(\alpha) = R \cdot E$$



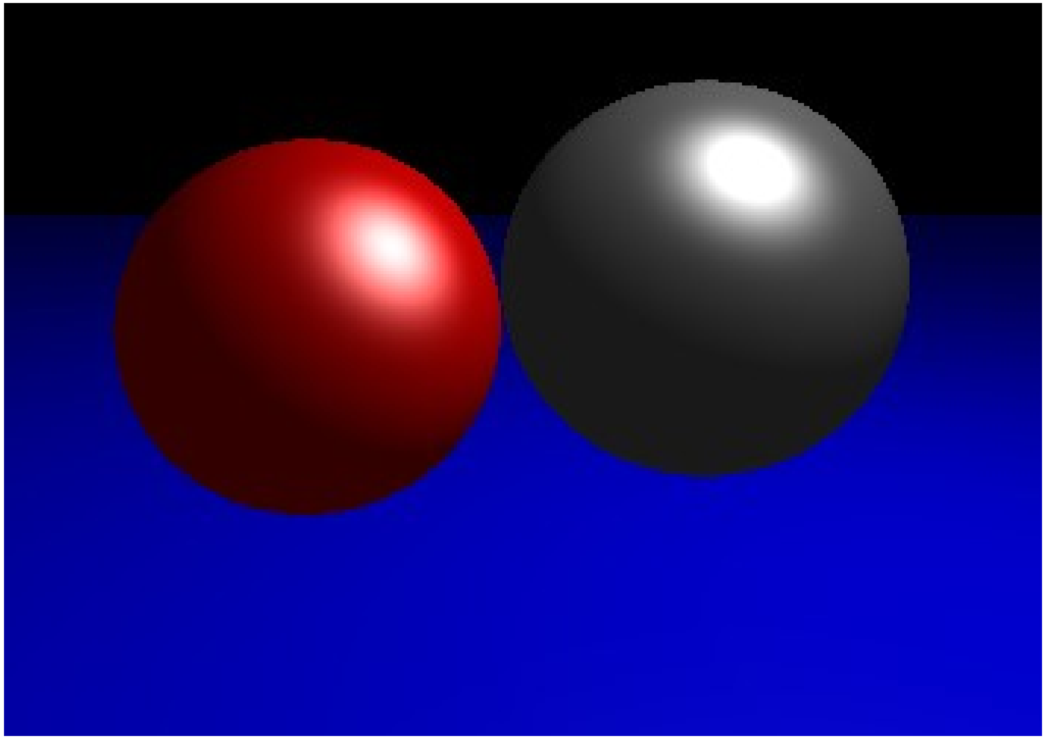
Total Illumination

$$I = k_a A + k_d C\left(L \cdot N\right)$$



Total Illumination

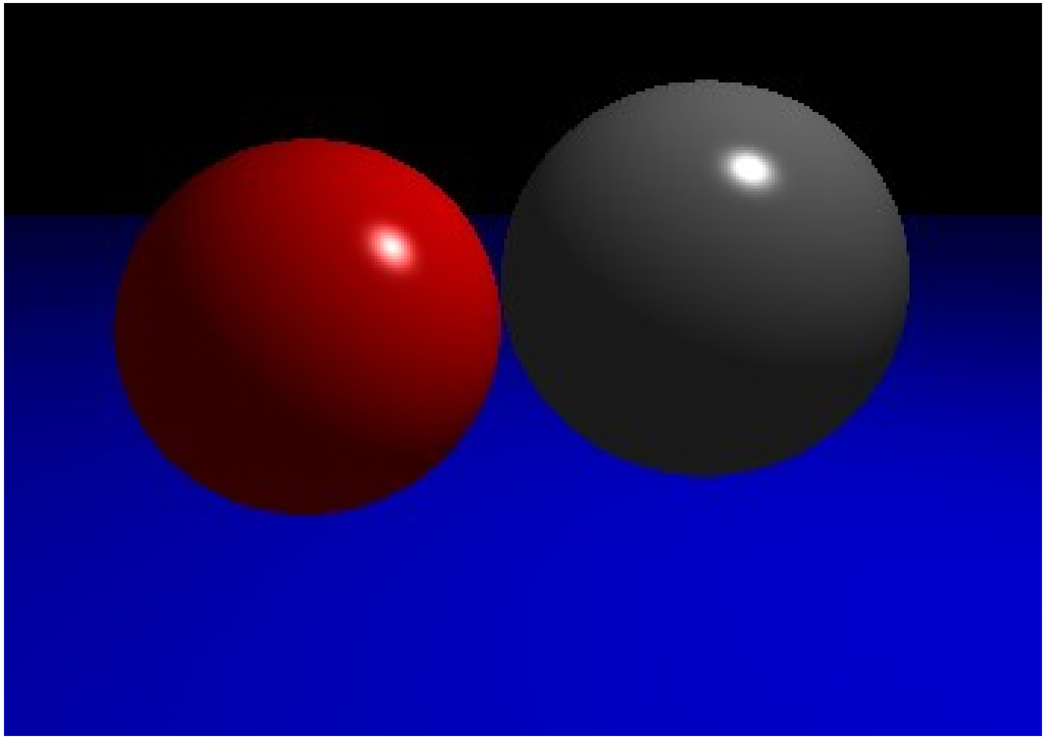
$$I = k_a A + C\left(k_d (L \cdot N) + k_s (N \cdot H)^n\right)$$



$$n = 5$$

Total Illumination

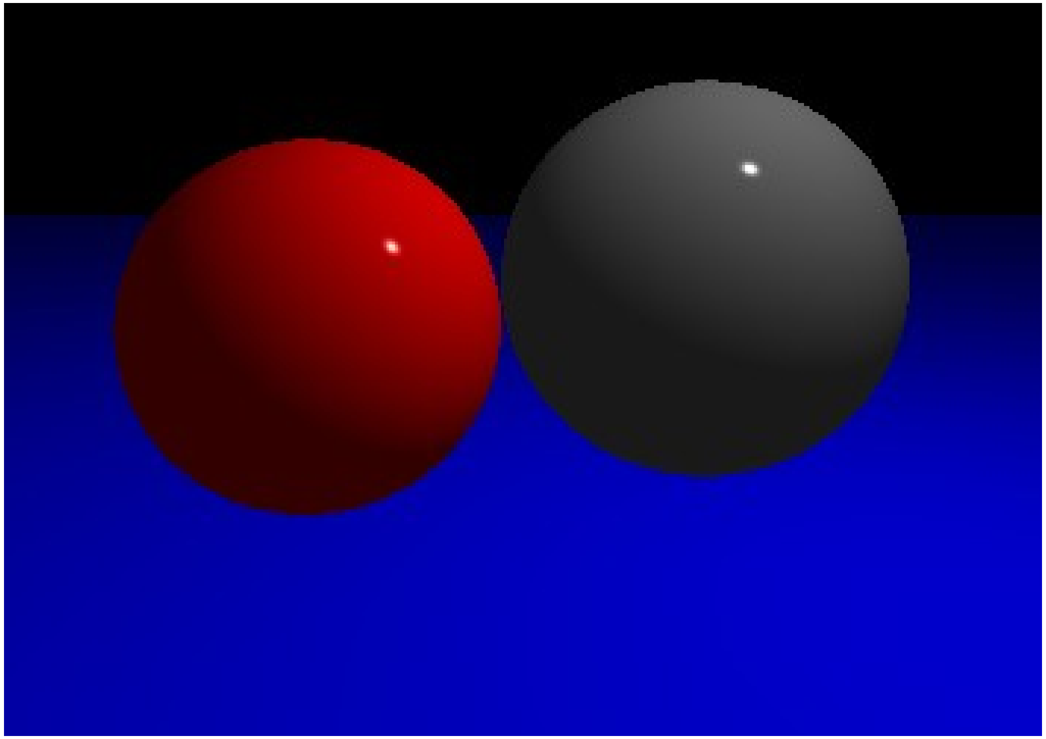
$$I = k_a A + C\left(k_d\left(L \cdot N\right) + k_s\left(N \cdot H\right)^n\right)$$



$$n = 50$$

Total Illumination

$$I = k_a A + C\left(k_d (L \cdot N) + k_s (N \cdot H)^n\right)$$



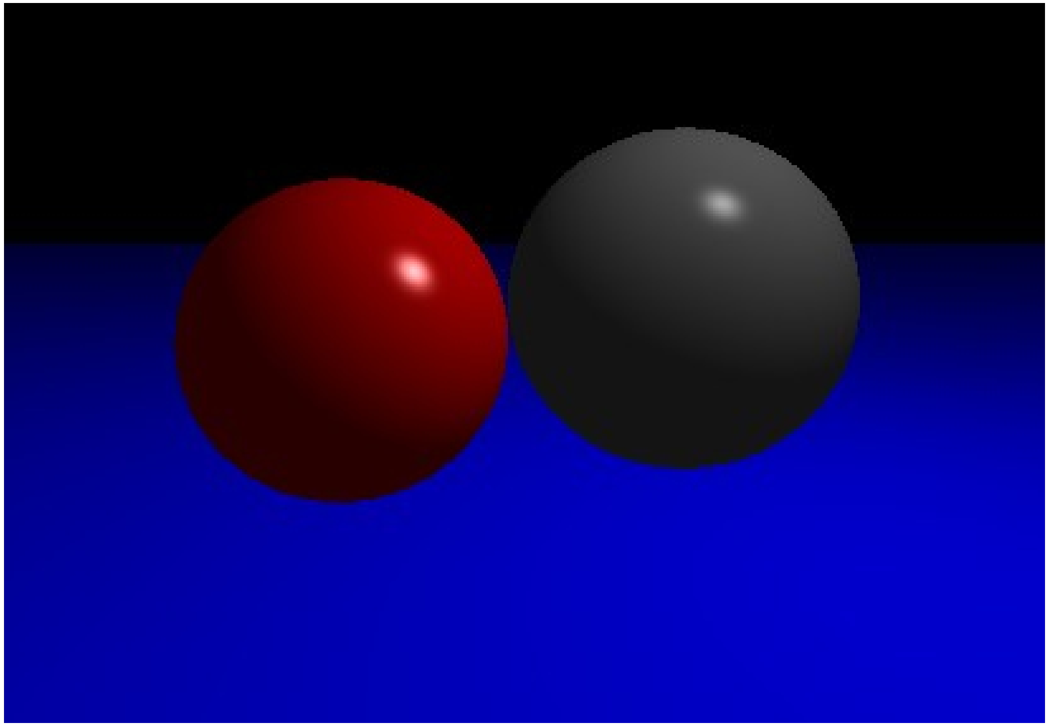
$$n = 500$$

Multiple Light Sources

- Only one ambient term no matter how many lights
- Light is additive; add contribution of multiple lights (diffuse/specular components)

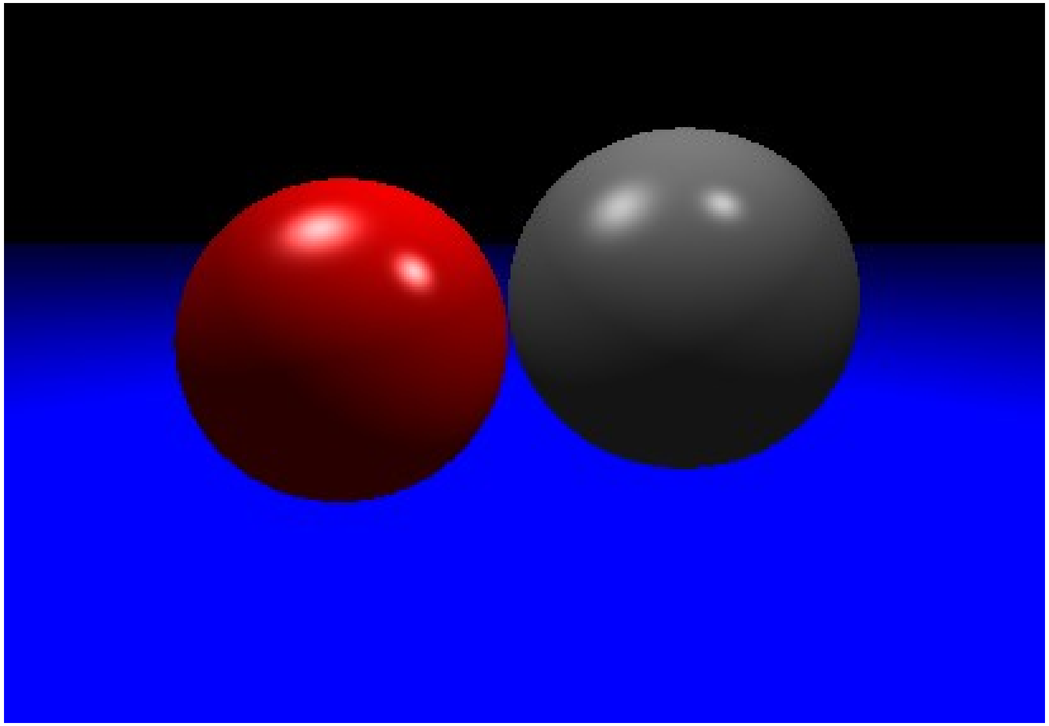
Total Illumination

$$I = k_a A + C\left(k_d (L \cdot N) + k_s (R \cdot E)^n\right)$$



Total Illumination

$$I = k_a A + \sum_i C_i \left(k_d \left(L_i \cdot N \right) + k_s \left(R_i \cdot E \right)^n \right)$$



Attenuation

- Decrease intensity with distance from light

- d = distance to light
- r = radius of attenuation for light

$$att(d,r) = \max(0, 1 - d/r)$$

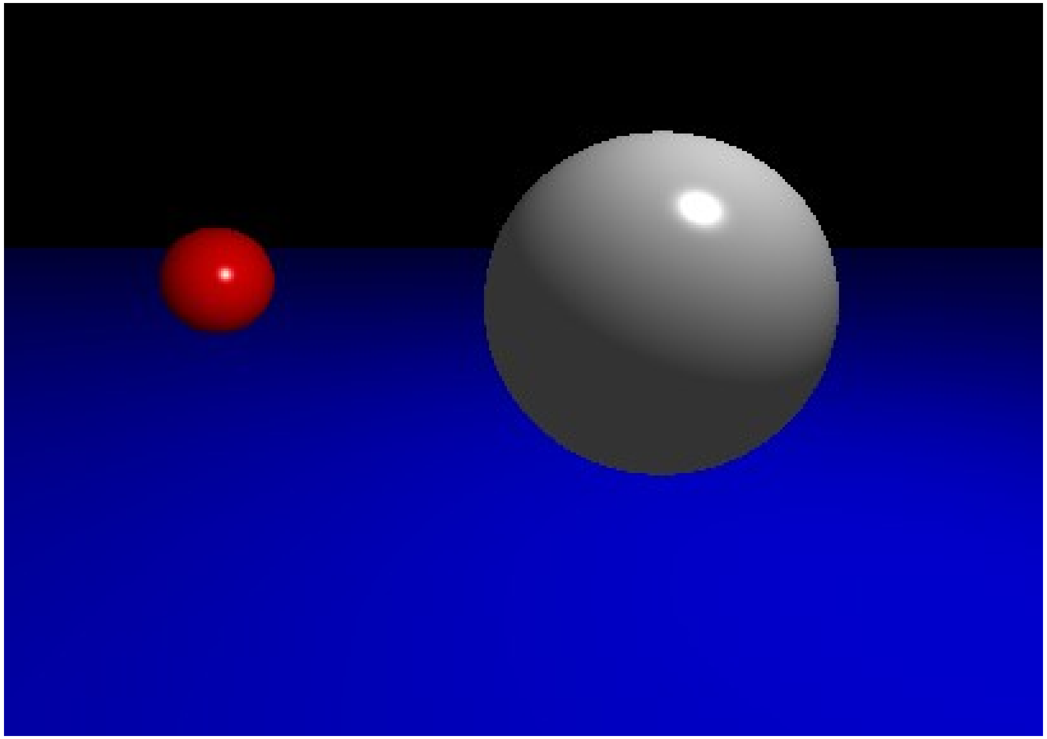
$$att(d,r) = \max(0, 1 - d^2/r^2)$$

$$att(d,r) = \max\left(0, \left(1 - d^2/r^2\right)^2\right)$$

$$att(d,r) = e^{-d^2/r^2}$$

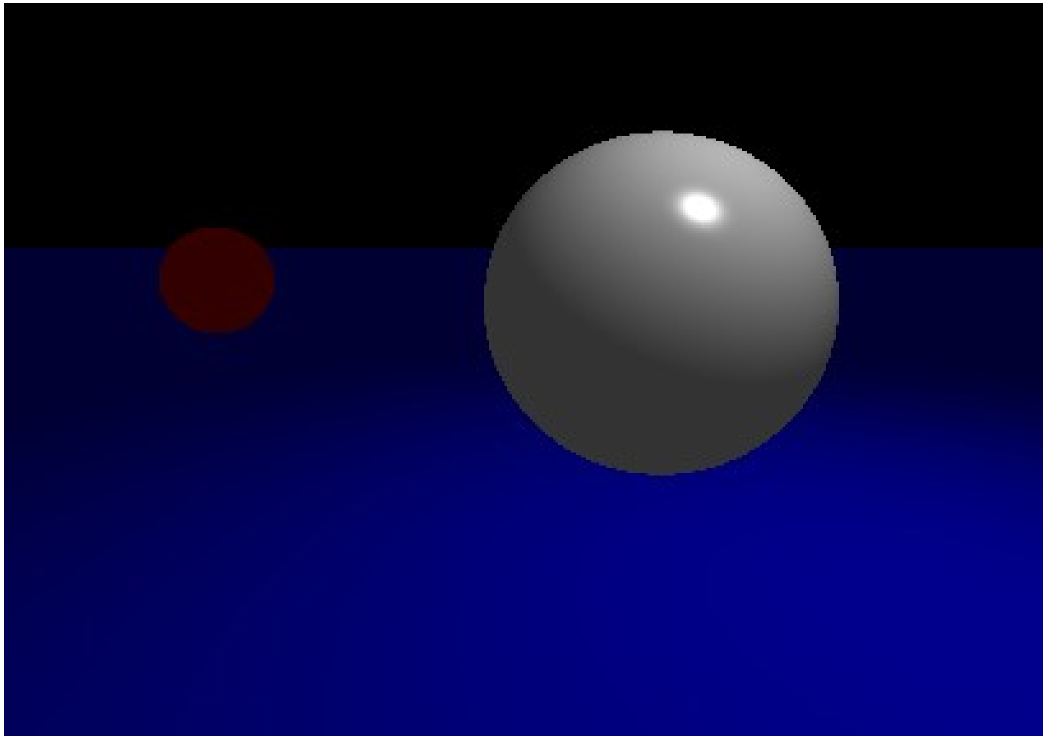
Attenuation

$$I = k_a A + \sum_i C_i \left(k_d (L_i \cdot N) + k_s (R_i \cdot E)^n \right) att(d, r_i)$$



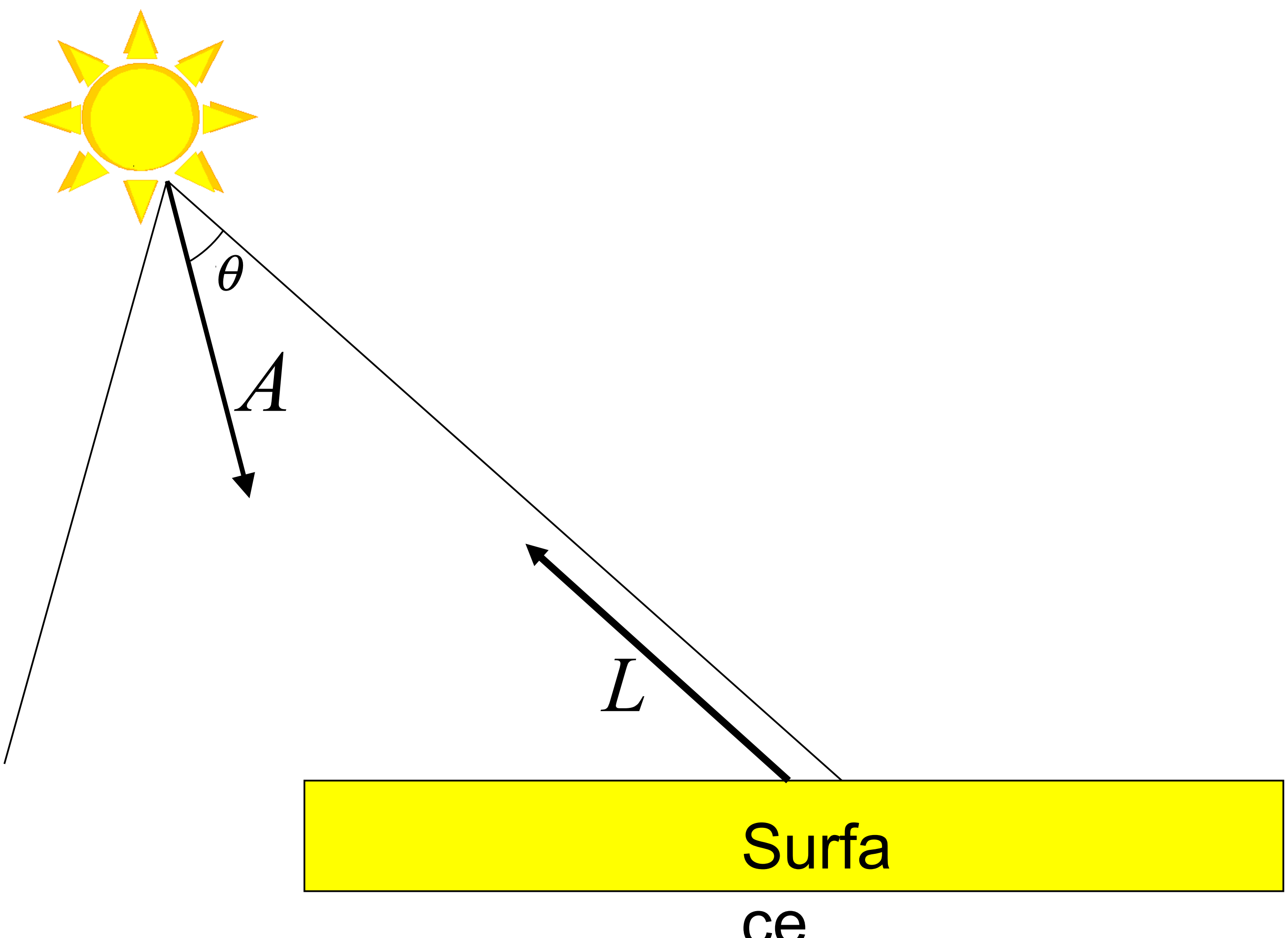
Attenuation

$$I = k_a A + \sum_i C_i \left(k_d (L_i \cdot N) + k_s (R_i \cdot E)^n \right) att(d, r_i)$$



Spot Lights

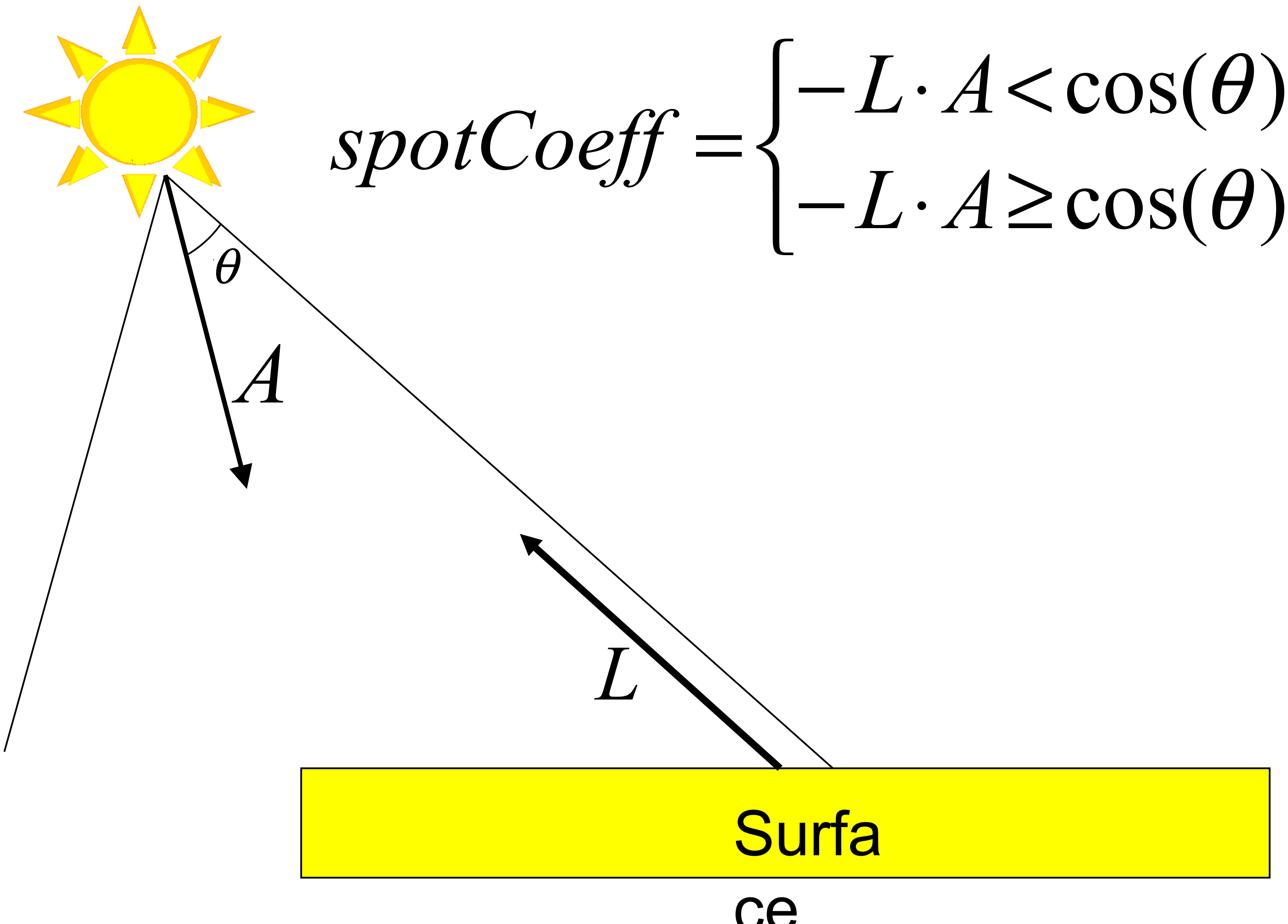
- Eliminate light contribution outside of a cone



Spot Lights

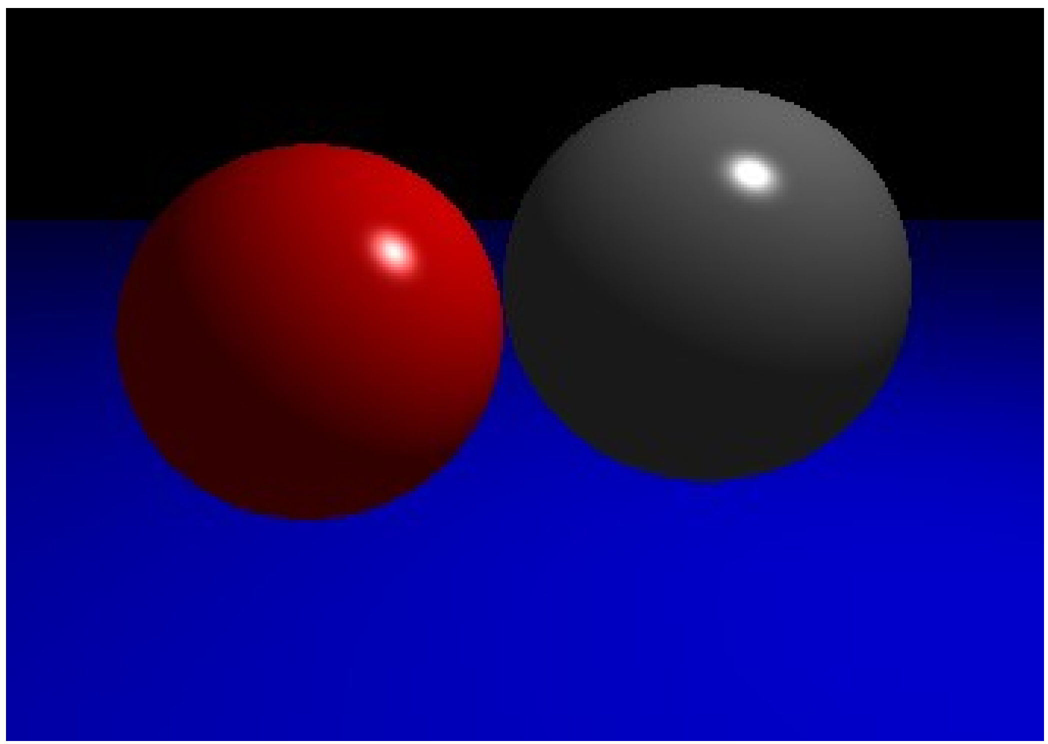
- Eliminate light contribution outside of a cone

$$spotCoeff = \begin{cases} -L \cdot A < \cos(\theta), & 0 \\ -L \cdot A \geq \cos(\theta), & \left(-L \cdot A\right)^{\alpha} \end{cases}$$



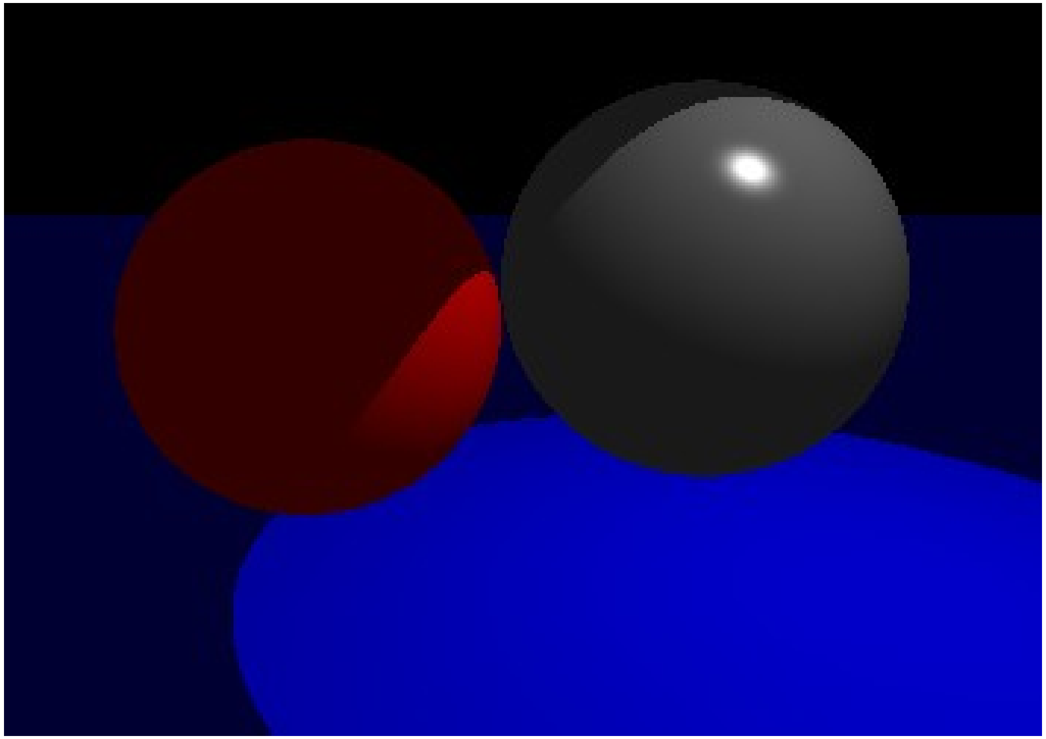
Spot Lights

$$I = k_a A + \sum_i C_i \left(k_d (L_i \cdot N) + k_s (R_i \cdot E)^n \right) spotCoeff_i$$



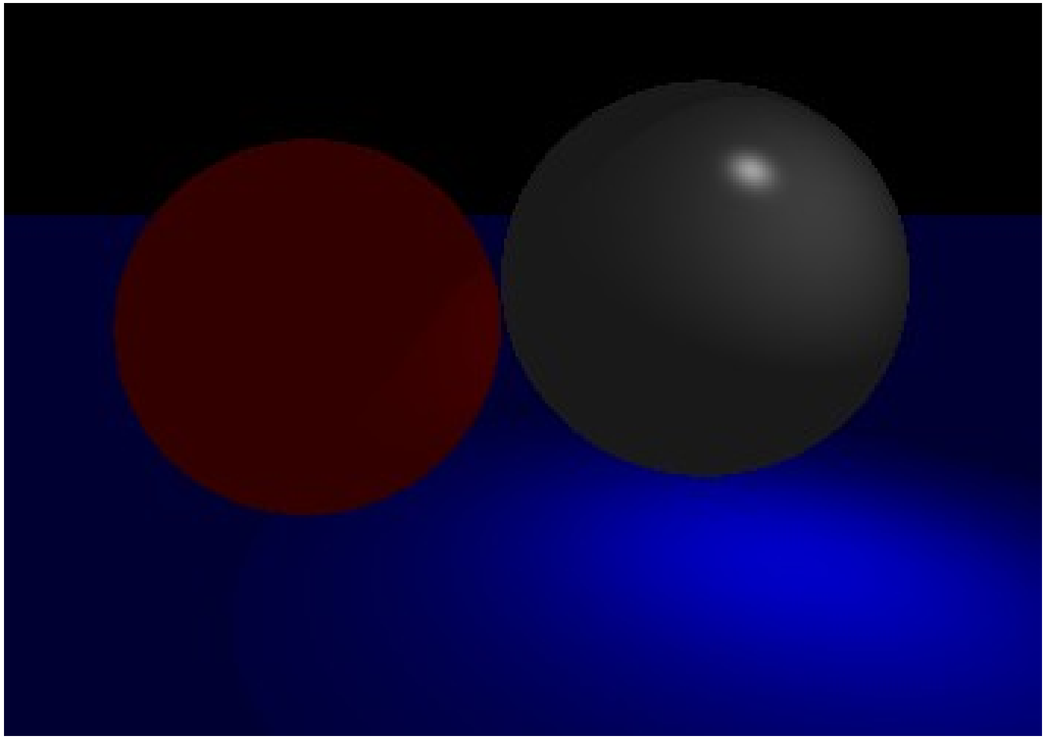
Spot Lights

$$I = k_a A + \sum_i C_i \left(k_d (L_i \cdot N) + k_s (R_i \cdot E)^n \right) spotCoeff_i$$

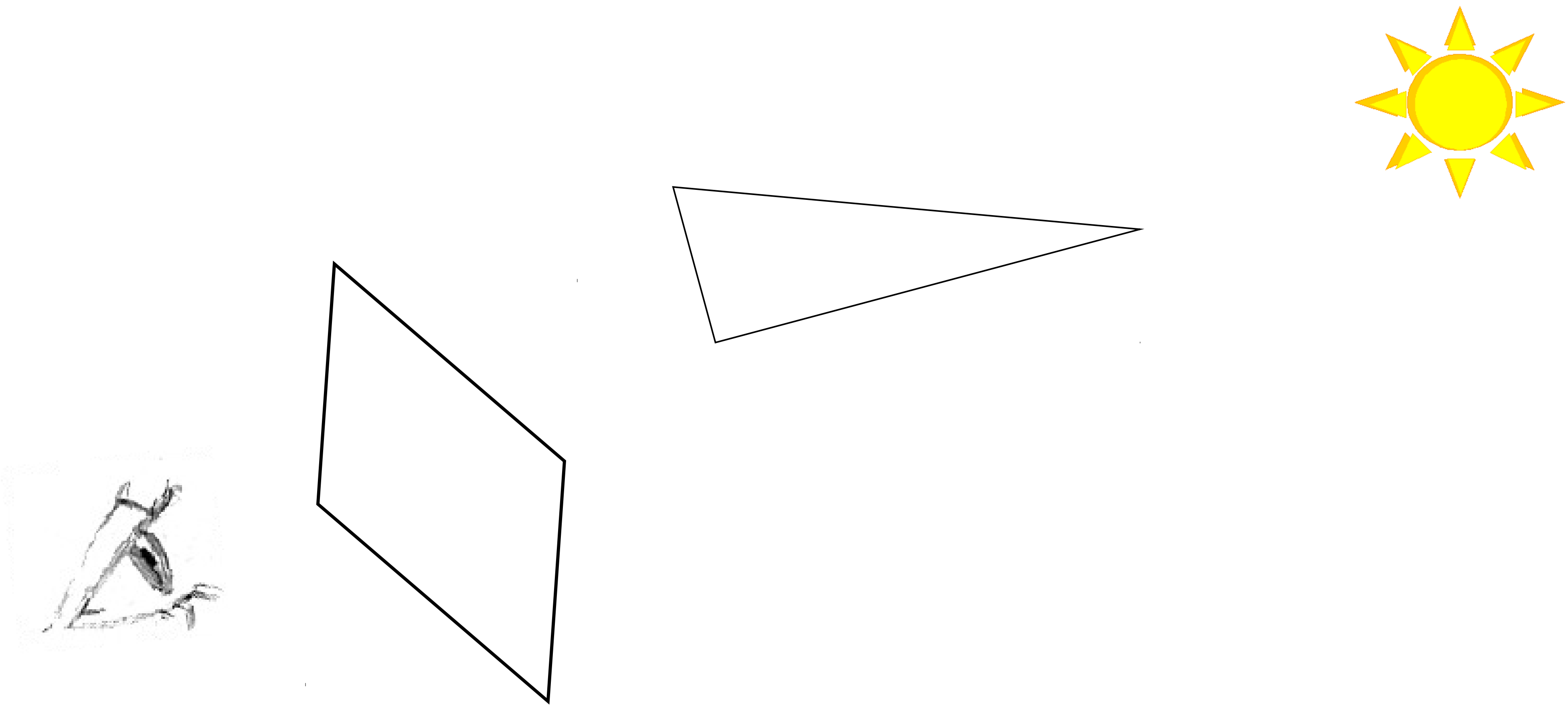


Spot Lights

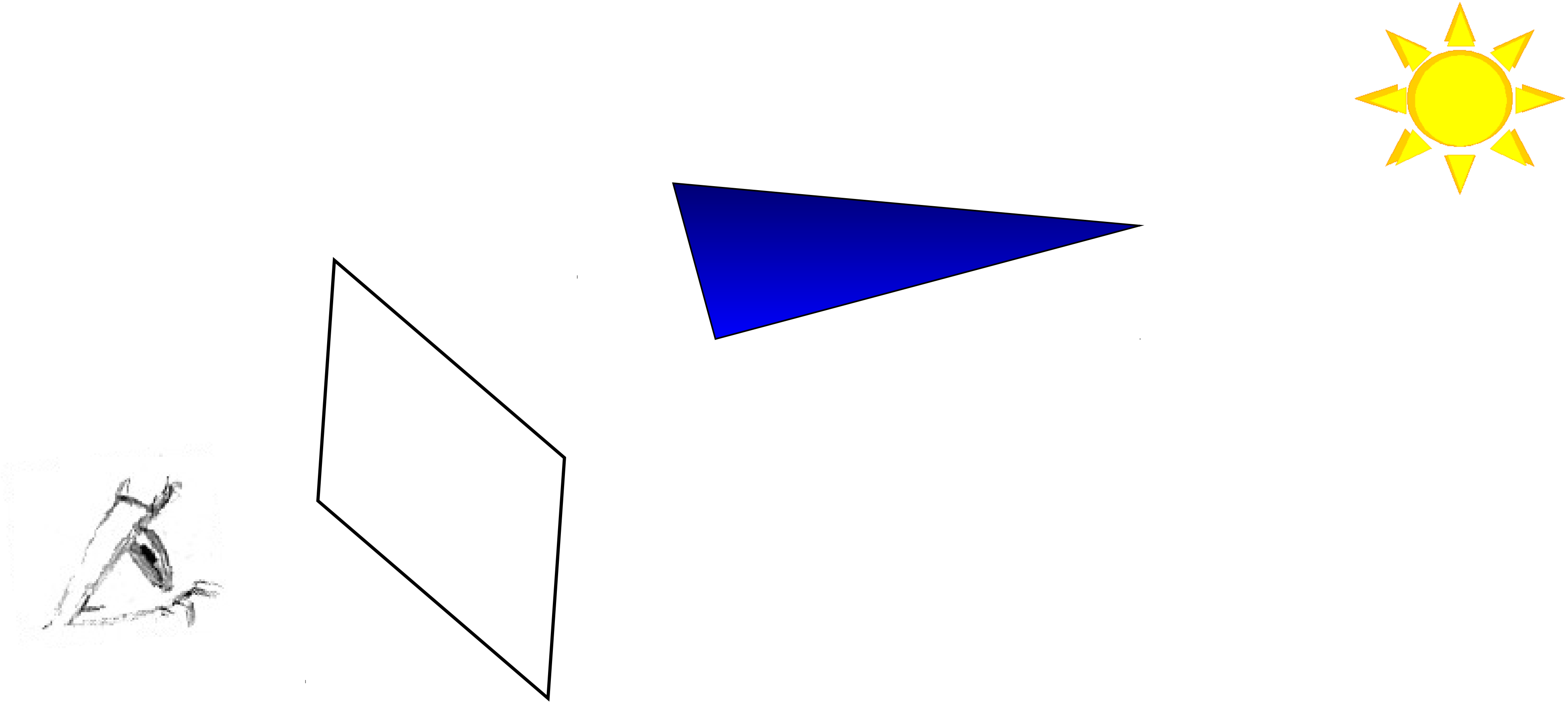
$$I = k_a A + \sum_i C_i \left(k_d \left(L_i \cdot N \right) + k_s \left(R_i \cdot E \right)^n \right) spotCoeff_i$$



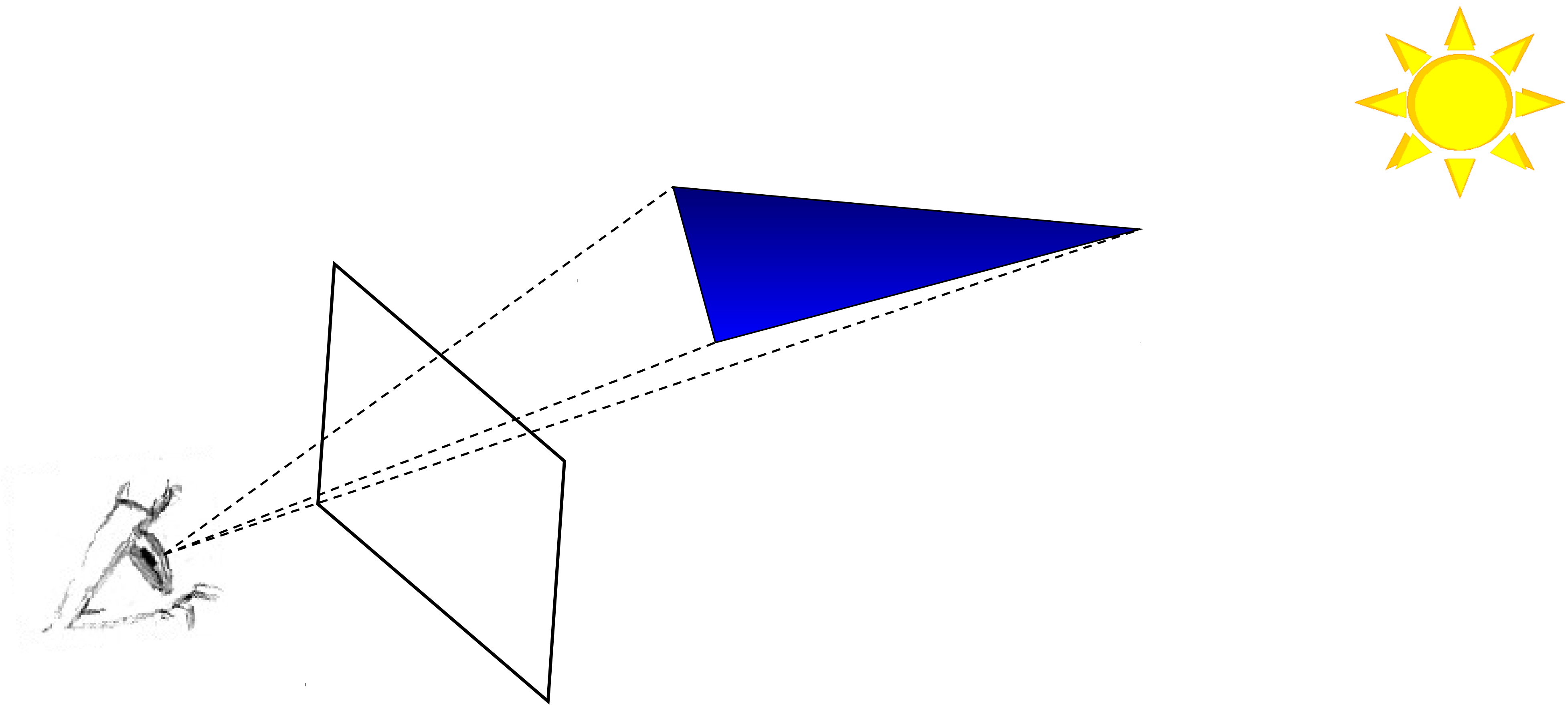
Problem



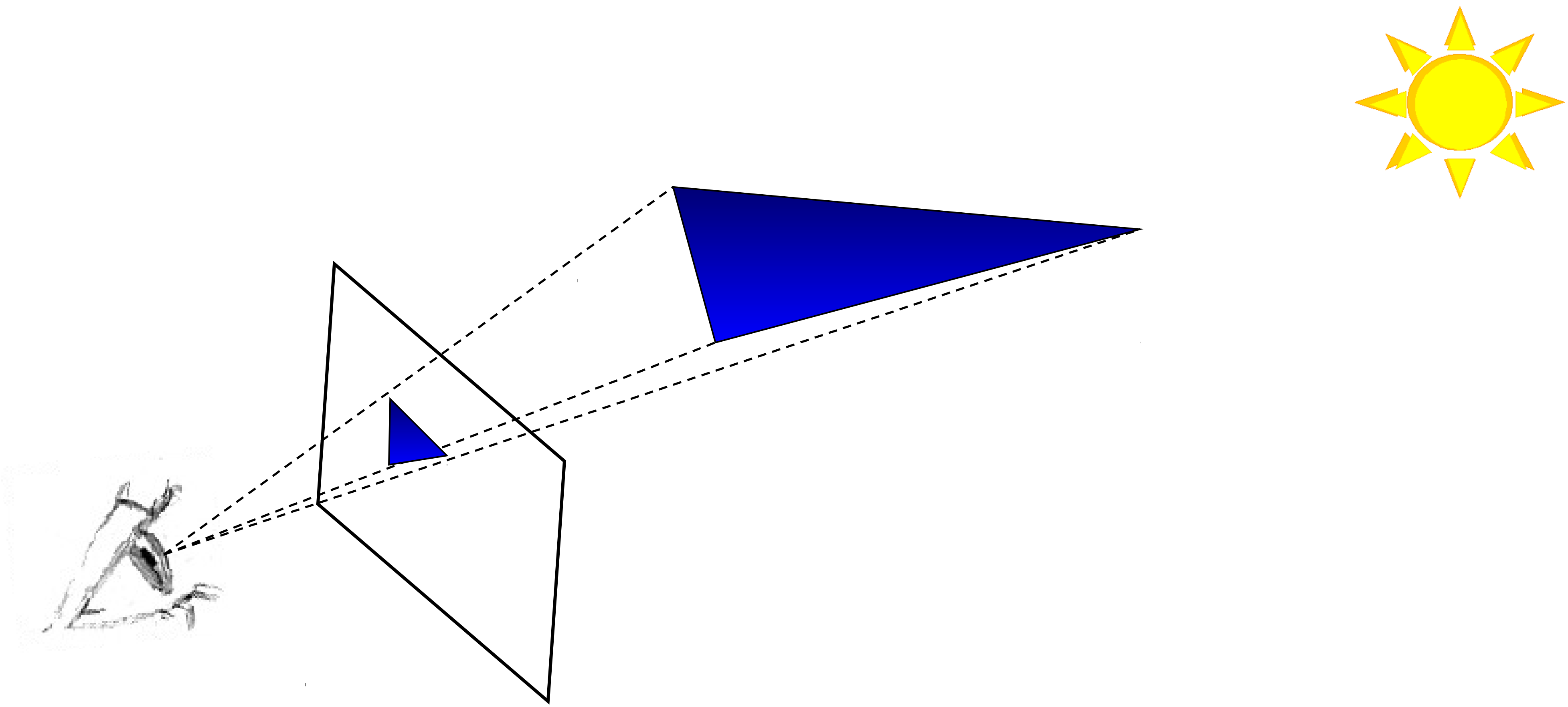
Problem



Problem



Problem



Shading Algorithms

- Flat Shading
- Gouraud Shading
- Phong Shading

Flat Shading

- Apply same color across entire polygon
- Calculate color once per polygon
  - Typically use center of polygon

- Fast, but not very desirable for smooth shapes

Flat Shading

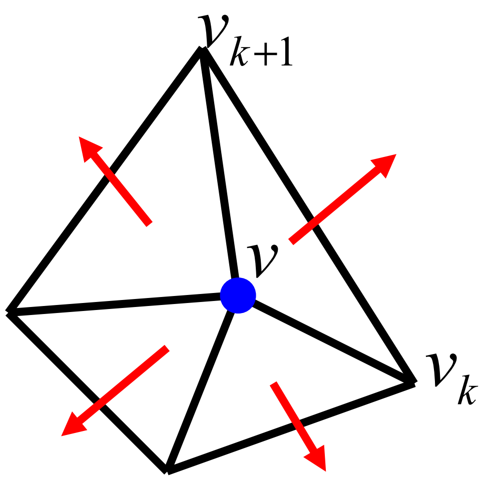
Image taken from http://www.cs.cmu.edu/~ph/nyit/

Gouraud (Per-Vertex) Shading

- Assume normals at vertices of polygon
  - If all normals the same, then the result is the same as flat shading
- Determine color at each vertex
- Interpolate colors from vertices across polygon

Gouraud (Per-Vertex) Shading

- Assume normals at vertices of polygon
  - If all normals the same, then the result is the same as flat shading
- Determine color at each vertex
- Interpolate colors from vertices across polygon



$$N_k = \left(v_{k+1} - v\right) \times \left(v_k - v\right)$$

$$N_V = \frac{\sum_{k=1}^{n} N_k}{\left|\sum_{k=1}^{n} N_k\right|}$$

Flat Shading

Image taken from http://www.cs.cmu.edu/~ph/nyit/

Gouraud Shading

Image taken from http://www.cs.cmu.edu/~ph/nyit/

Phong (Per-Pixel) Shading

- Assume normals at vertices of polygon
- Interpolate *normals* from vertices across polygon
- Determine color at each pixel in polygon

- Captures highlights better

Gouraud Shading

Image taken from http://www.cs.cmu.edu/~ph/nyit/

Phong Shading

Image taken from http://www.cs.cmu.edu/~ph/nyit/

Texture Mapping

- Geometry and lighting alone do not provide sufficient visible detail
- “Paste” 2D image onto 3D surface

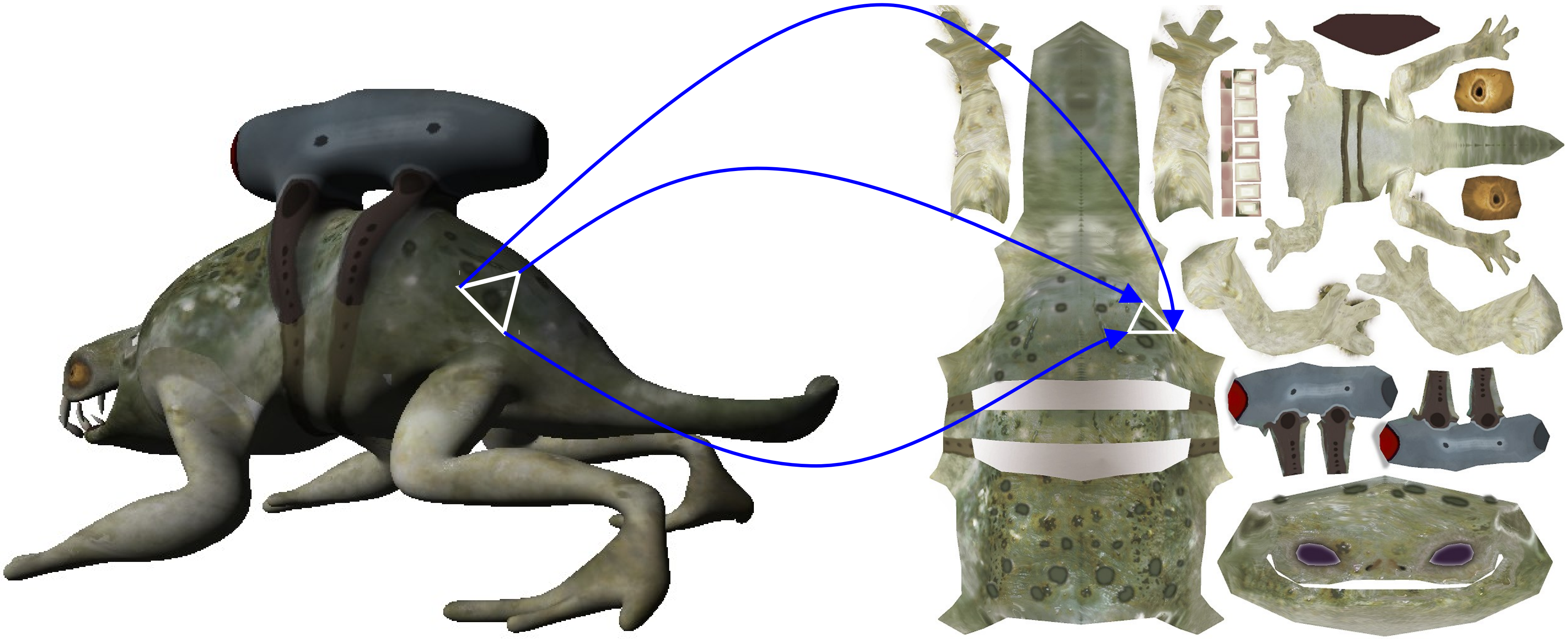
- Surface appears much more complex than reality

Texture Mapping

Texture Mapping

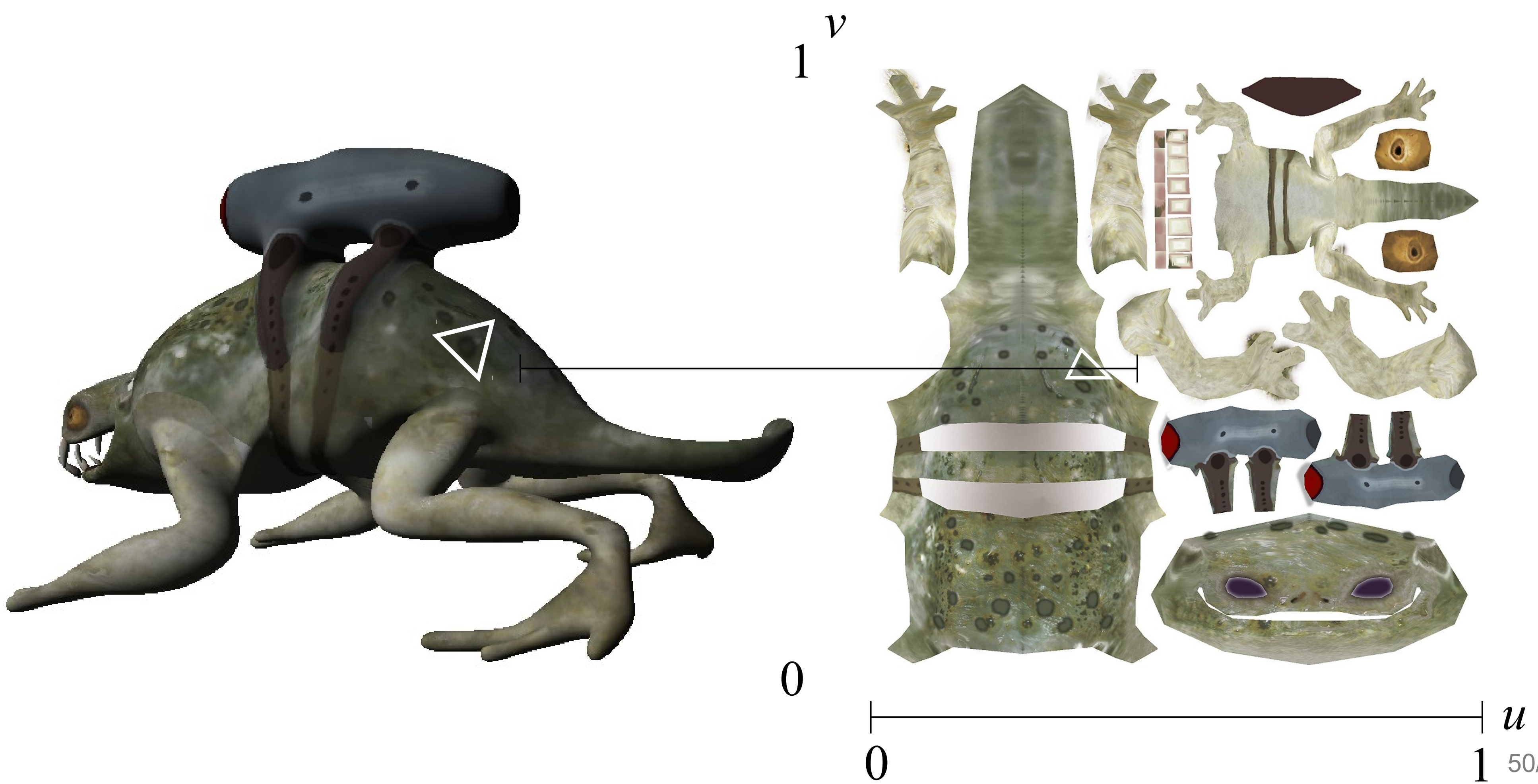
Texture Mapping

Texture Mapping



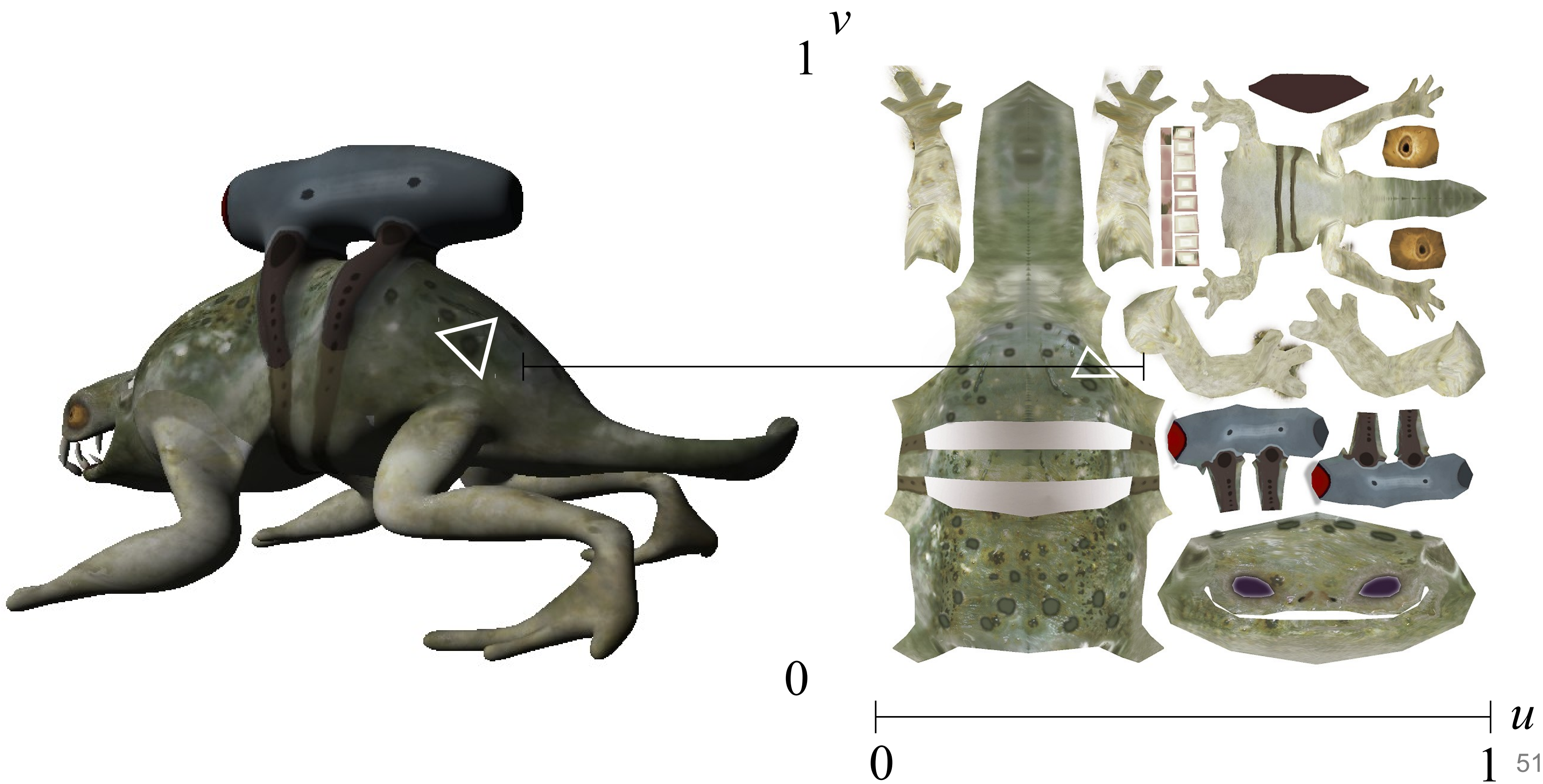
Texture Mapping

- Assume texture parameterized by u, v



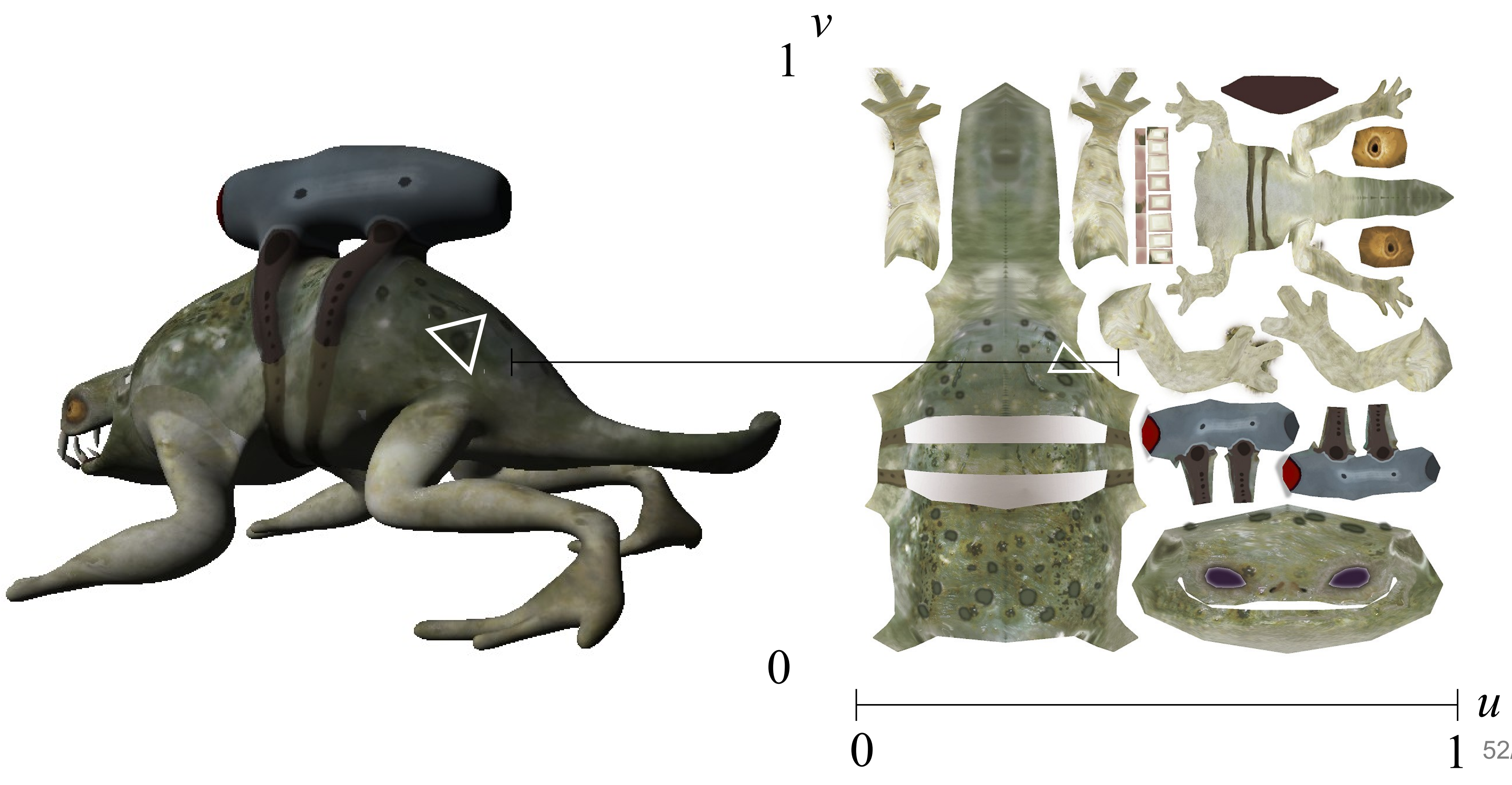
Texture Mapping

- Any u, v coordinate maps to a point on the image



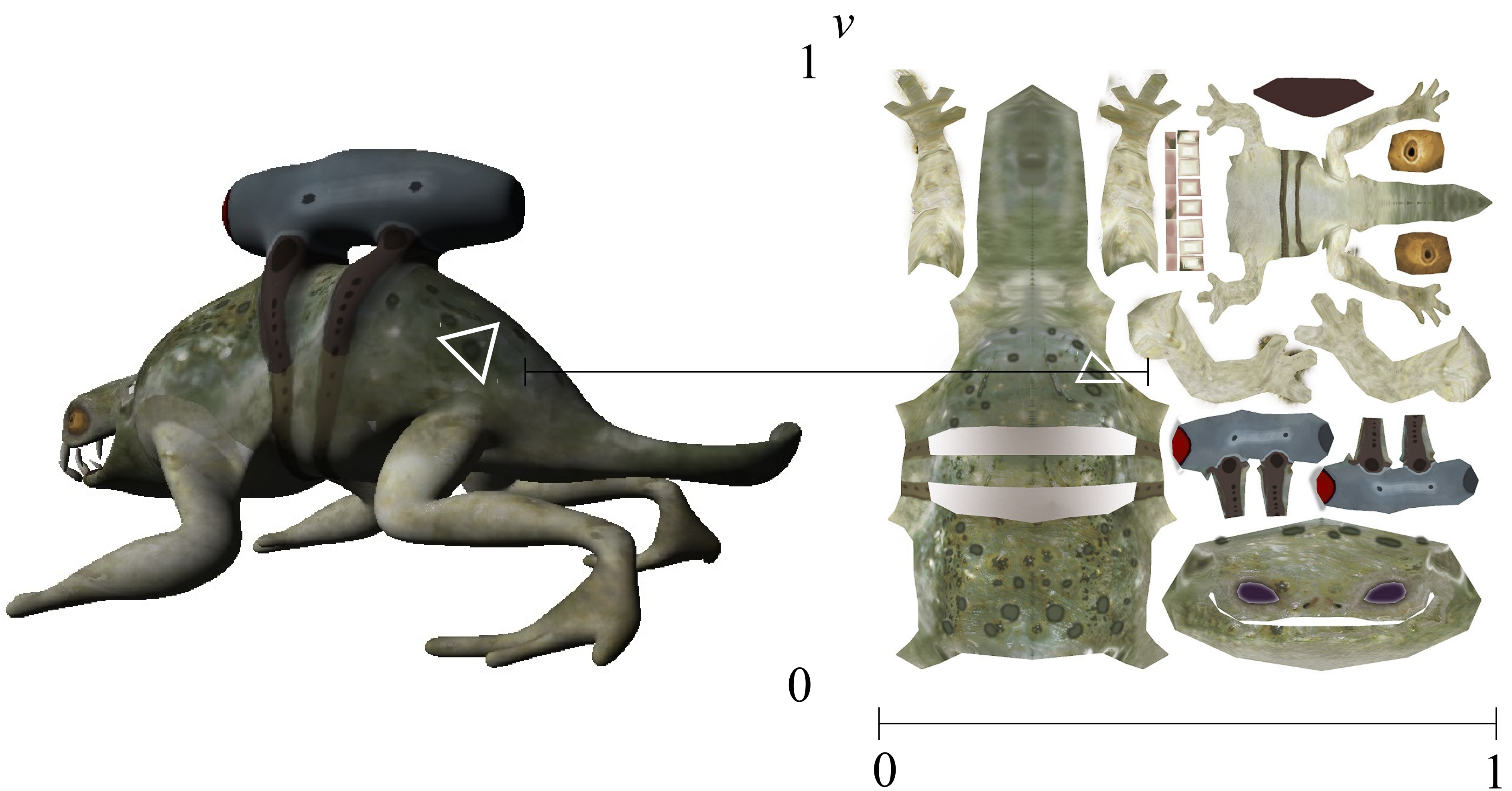
Texture Mapping

- Associate *texture coordinates* with each vertex on the surface

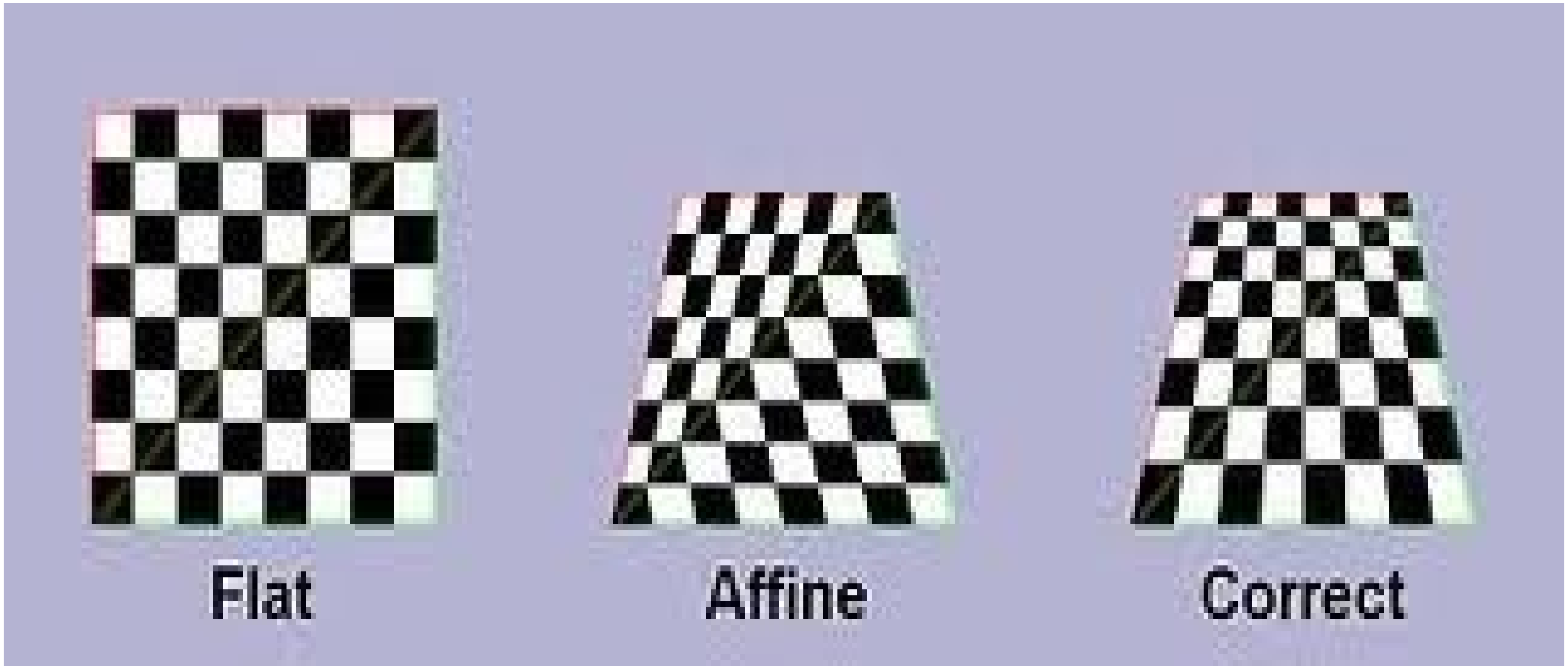


Texture Mapping

- During polygon drawing, lookup color from texture using interpolated texture coordinates



Interpolation in image space



Perspective correct
rasterizer...

Other Uses of Texture Mapping

- Environment Mapping
- Bump/Normal Mapping
- Displacement Mapping
- ….

- Any attribute of the surface position, normal, color, etc… can be placed in a texture

Other Uses of Texture Mapping

- Environment Mapping
- Bump/Normal Mapping
- Displacement Mapping
- ….

- Any attribute of the surface position, normal, color, etc… can be placed in a texture

Environment Mapping

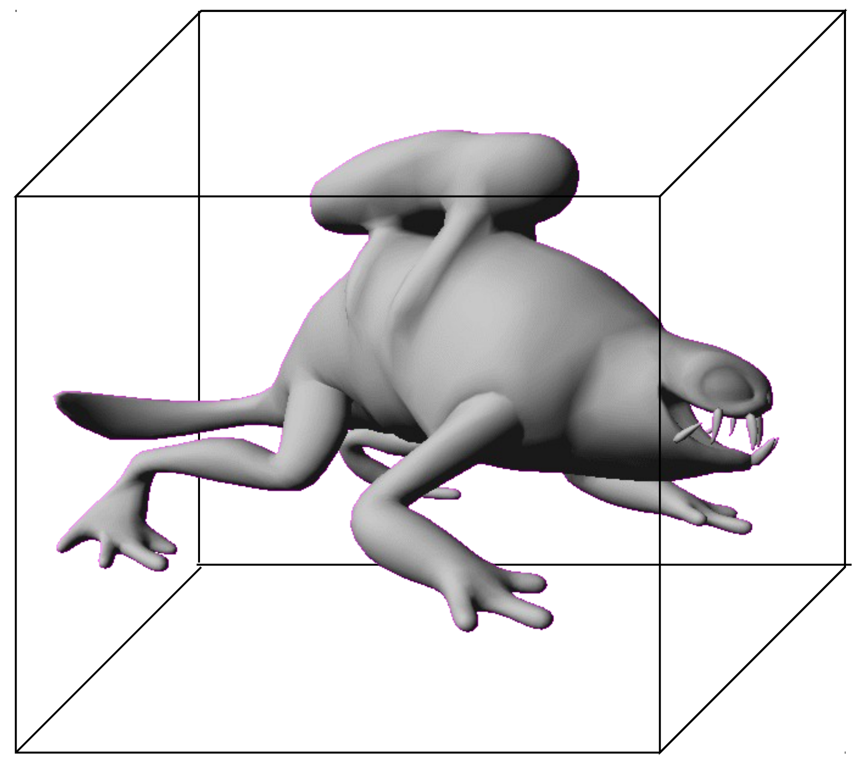
- Cheap attempt at modeling reflections
- Makes surfaces look metallic

- Use six textures to model faces of a cube
- Assume cube faces infinitely far away
- The reflected eye vector is used to find which of the textures to use and what texture coordinate

Environment Mapping

Environment Mapping

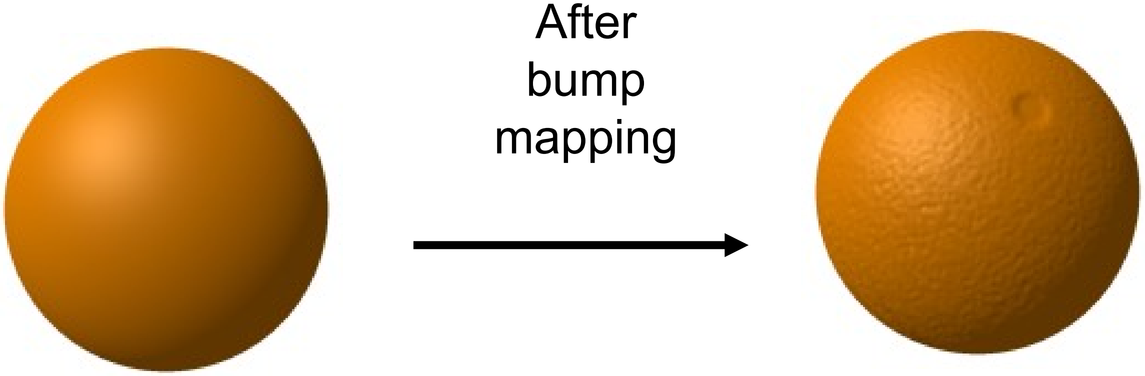
Environment Mapping



Environment Mapping

Bump/Normal Mapping

- Replace colors R,G,B with coordinates X,Y,Z
- Interpret pixels as normal vectors
- Makes the shading look more complicated than geometry really is



Bump/Normal Mapping Example

Bump/Normal Mapping Example

Bump/Normal Mapping Example


Displacement Mapping

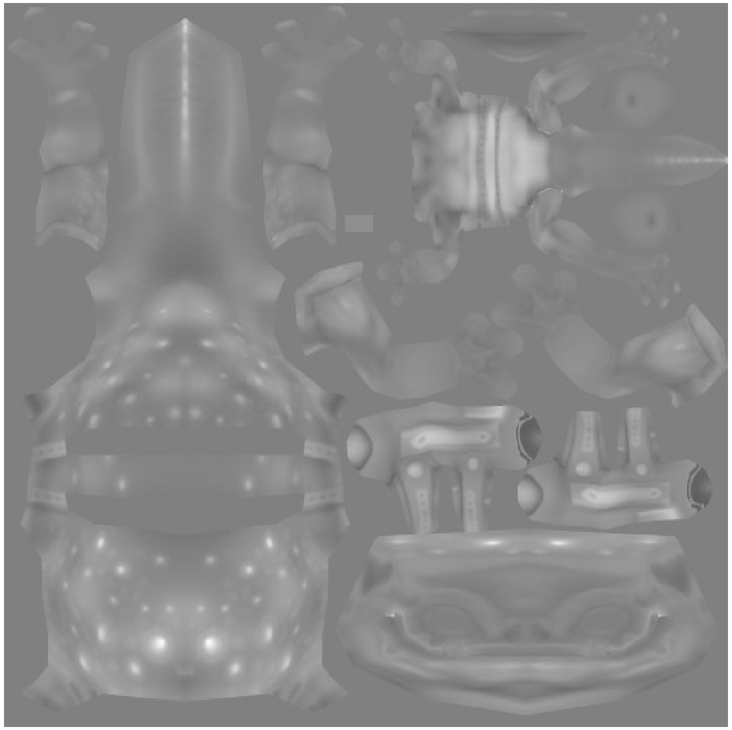
- Offset geometry in direction of normal
- Encode offset inside texture
- Used to actually change the geometry and provide more detail (especially silhouette)

- Difficult/expensive to perform with current hardware

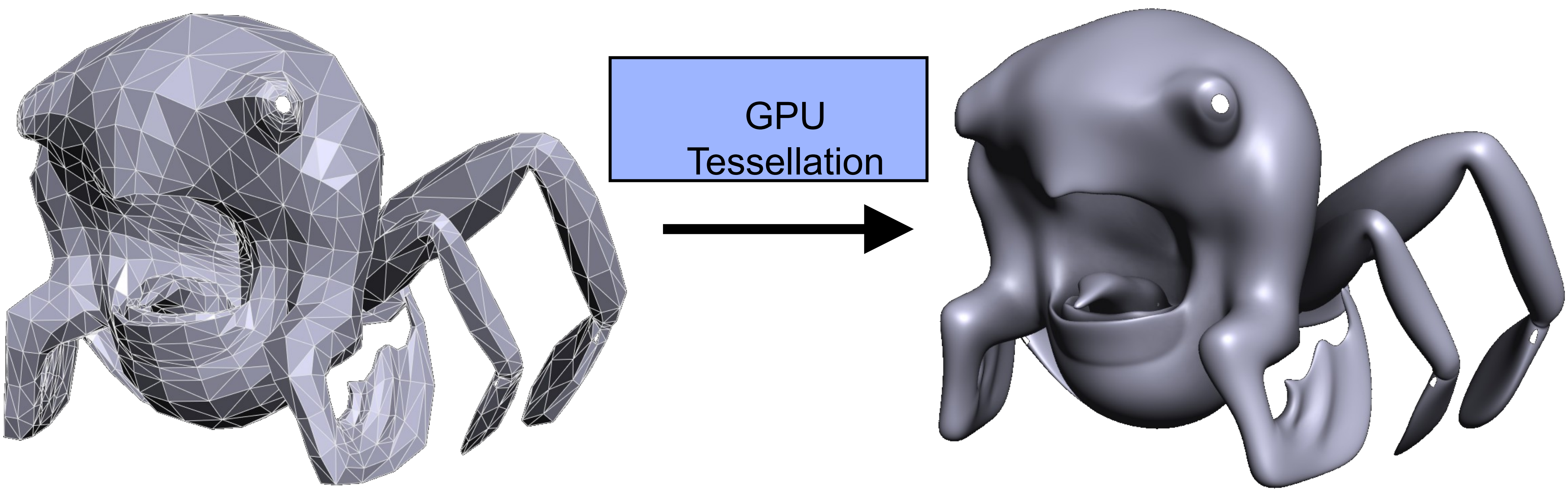
Bump/Normal Mapping Example

Displacement Mapping Example

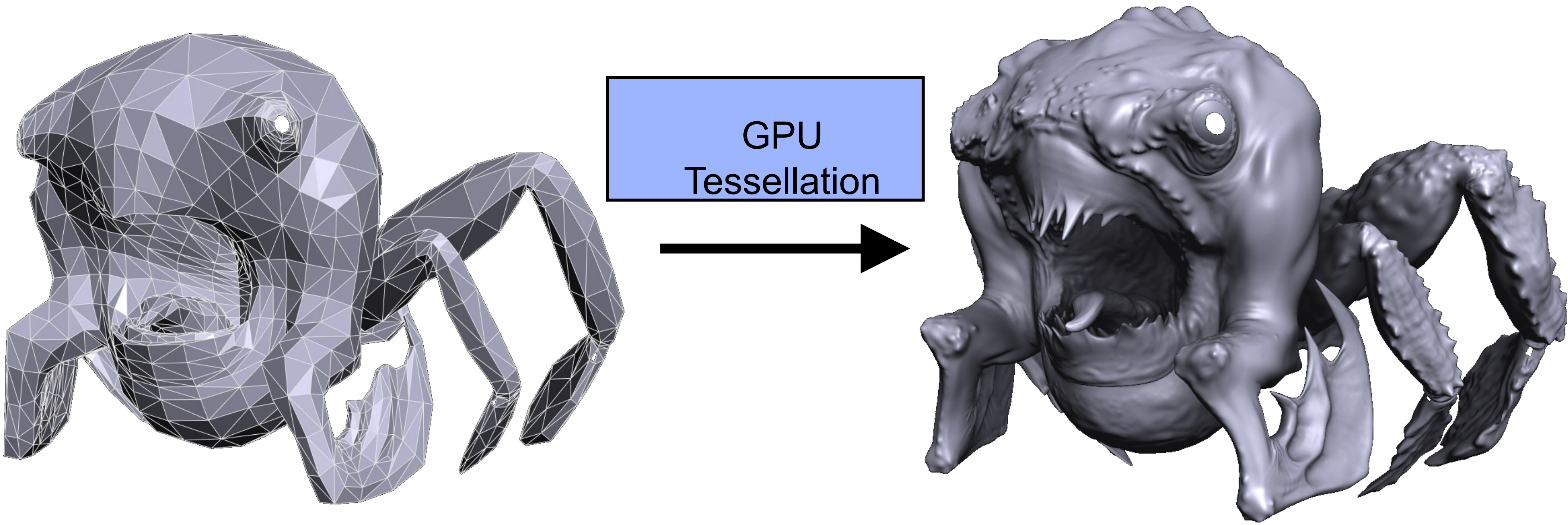
Displacement Mapping Example



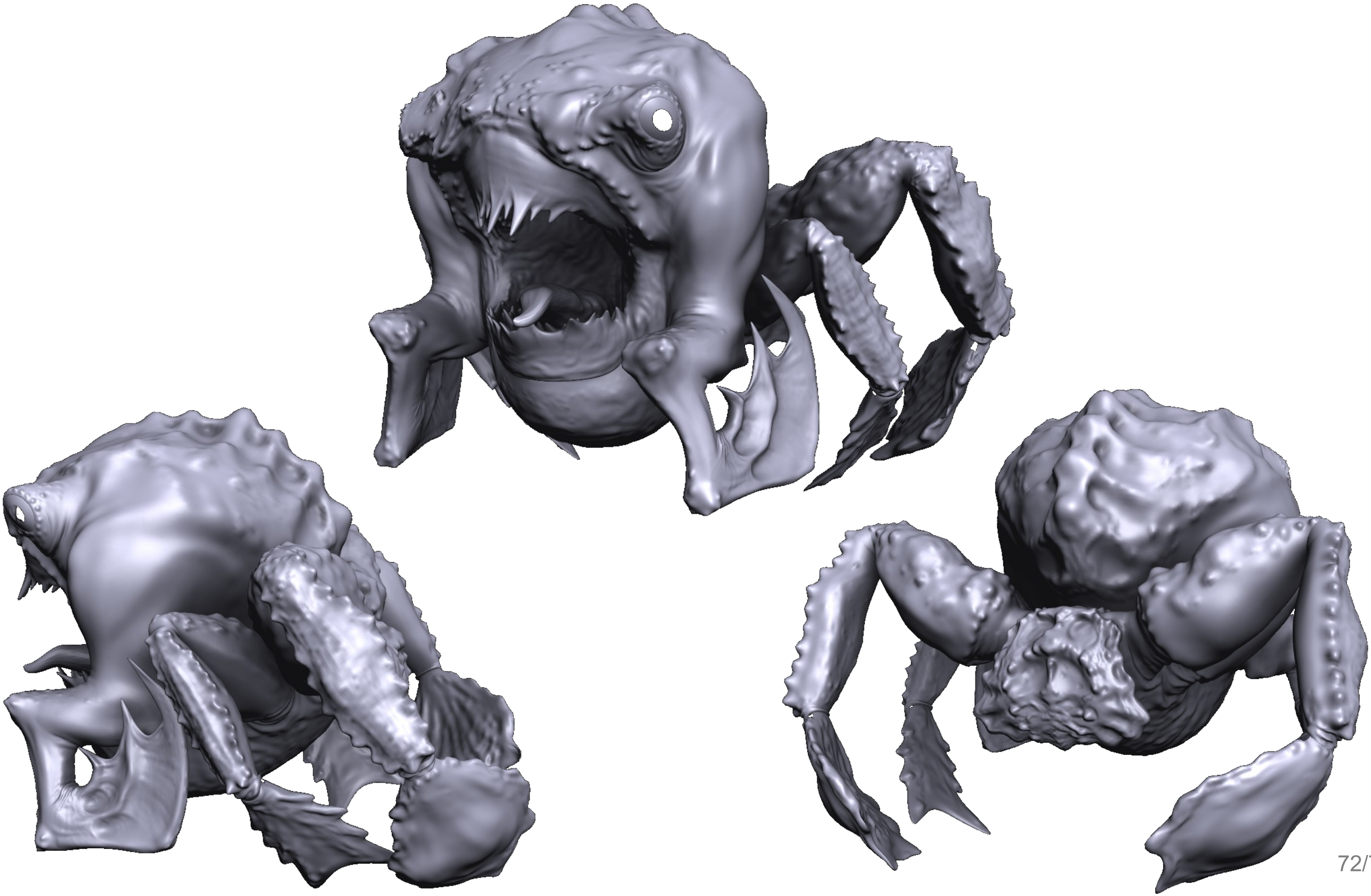
More Examples



More Examples



More Examples



Ray Tracing

- Provides rendering method with
  - Refraction/Transparent surfaces
  - Reflective surfaces
  - Shadows

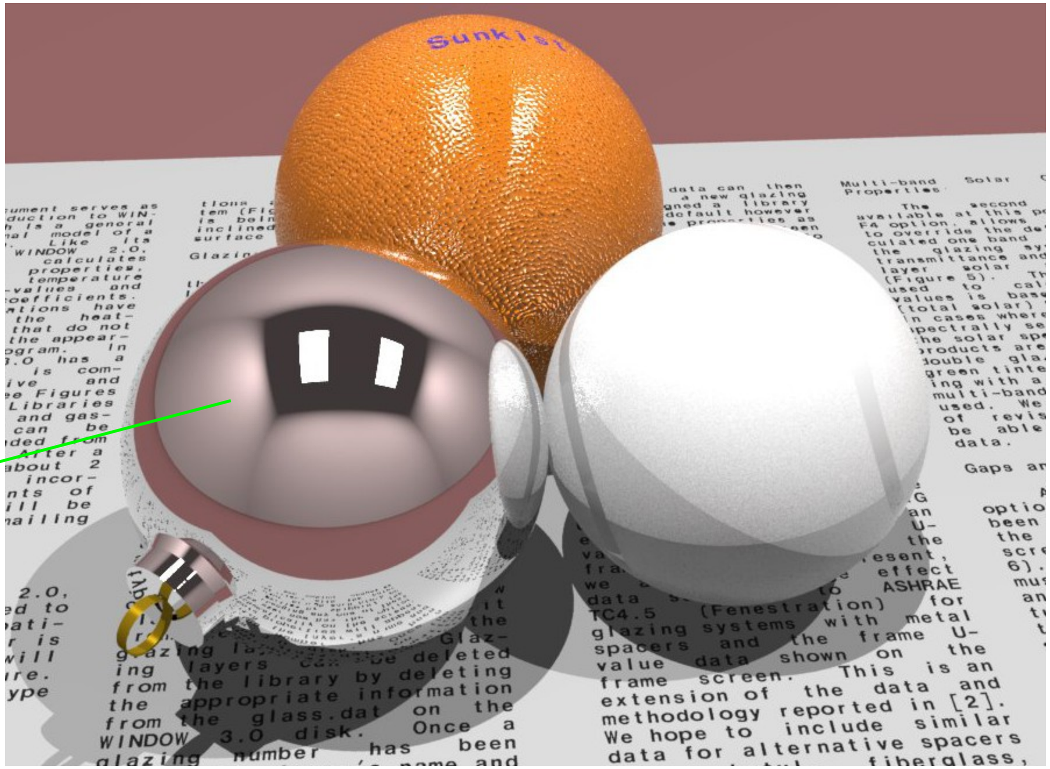


Image taken from http://radsite.lbl.gov/radiance/book/img/plate10.jpg

Ray Tracing

- Provides rendering method with
  - Refraction/Transparent surfaces
  - Reflective surfaces
  - Shadows

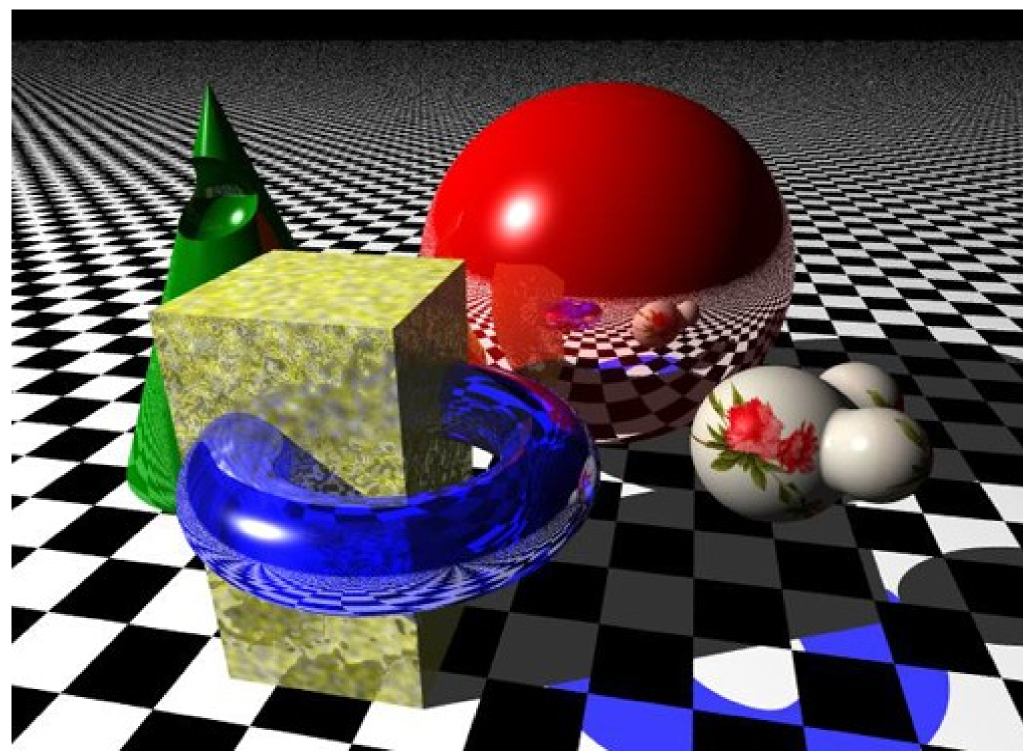


Image taken from http://www.tjhsst.edu/~dhyatt/superap/samplex.jpg

Ray Tracing

- Provides rendering method with
  - Refraction/Transparent surfaces
  - Reflective surfaces
  - Shadows