Graphics Pipeline: Modeling, Illumination, and Rendering

Fall 2026

Daniel G. Aliaga
Department of Computer Science
Purdue University

Computer Graphics Pipeline

Geometry

Modeling Transformation	Transform into 3D *world* coordinate system
Lighting	Simulate illumination and reflectance
Viewing Transformation	Transform into 3D *camera* coordinate system
Clipping	Clip primitives outside camera's view
Projection	Transform into 2D camera coordinate system
Scan Conversion	Draw pixels (incl. texturing, hidden surface…)

Image

Computer Graphics Pipeline

Geometry

Modeling	Modeling Transformation	Transform into 3D *world* coordinate system
Illum.	Lighting	Simulate illumination and reflectance
Rendering	Viewing Transformation	Transform into 3D *camera* coordinate system
	Clipping	Clip primitives outside camera's view
	Projection	Transform into 2D camera coordinate system
	Scan Conversion	Draw pixels (incl. texturing, hidden surface…)

Image

Computer Graphics Pipeline

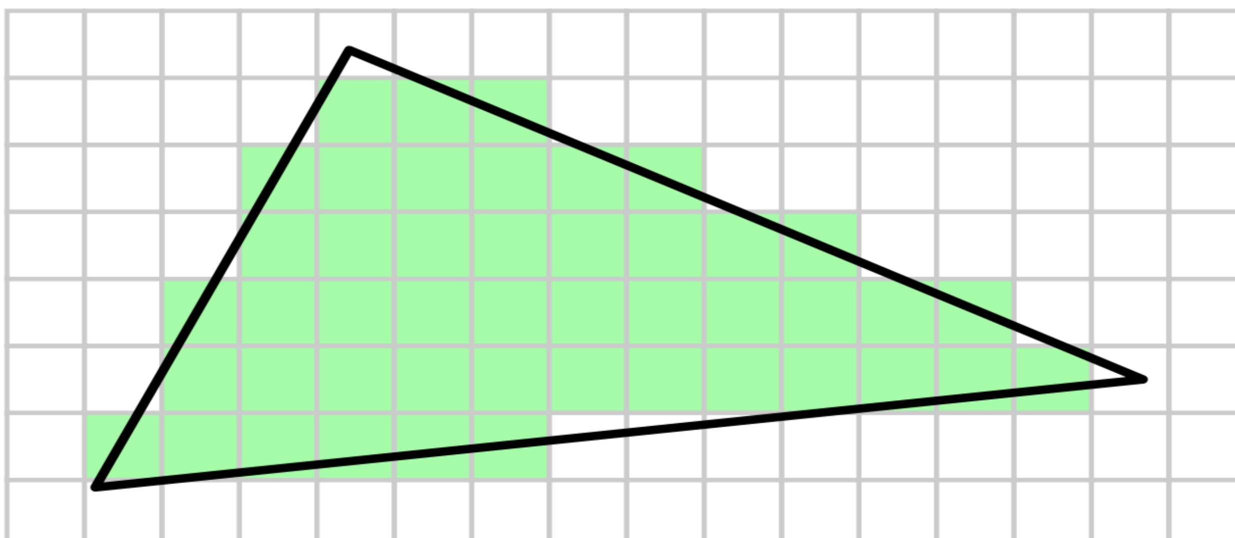
Geometry

Modeling	Modeling Transformation	Transform into 3D *world* coordinate system
Illum.	Lighting	Simulate illumination and reflectance
Rendering	Viewing Transformation	Transform into 3D *camera* coordinate system
	Clipping	Clip primitives outside camera's view
	Projection	Transform into 2D camera coordinate system
	Scan Conversion	Draw pixels (incl. texturing, hidden surface…)

Image

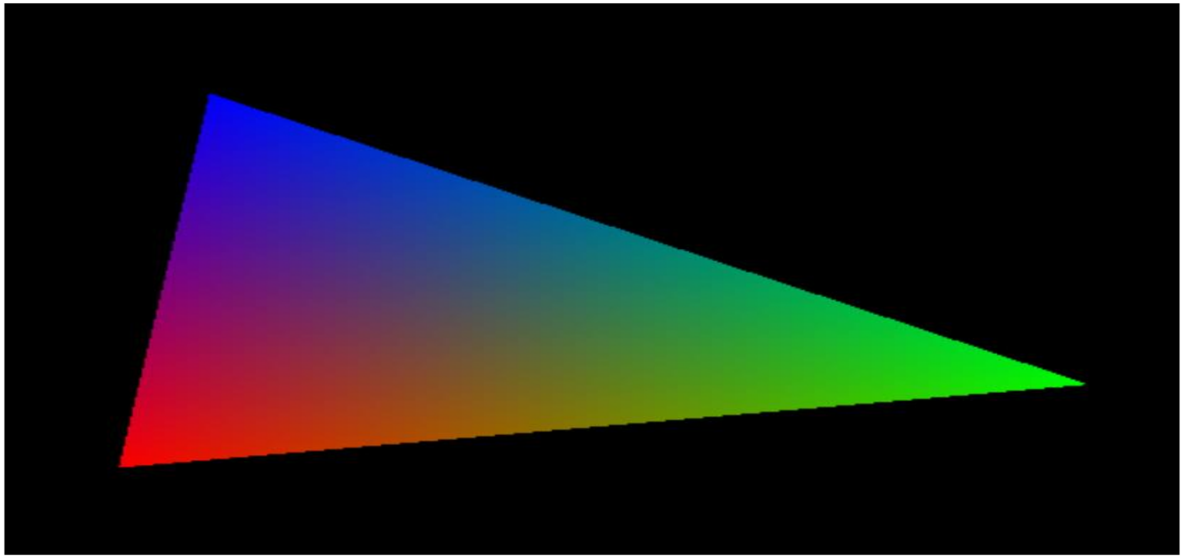
Scan Conversion/Rasterization

- Determine which fragments get generated
- Interpolate parameters (colors, textures, normals, etc.)



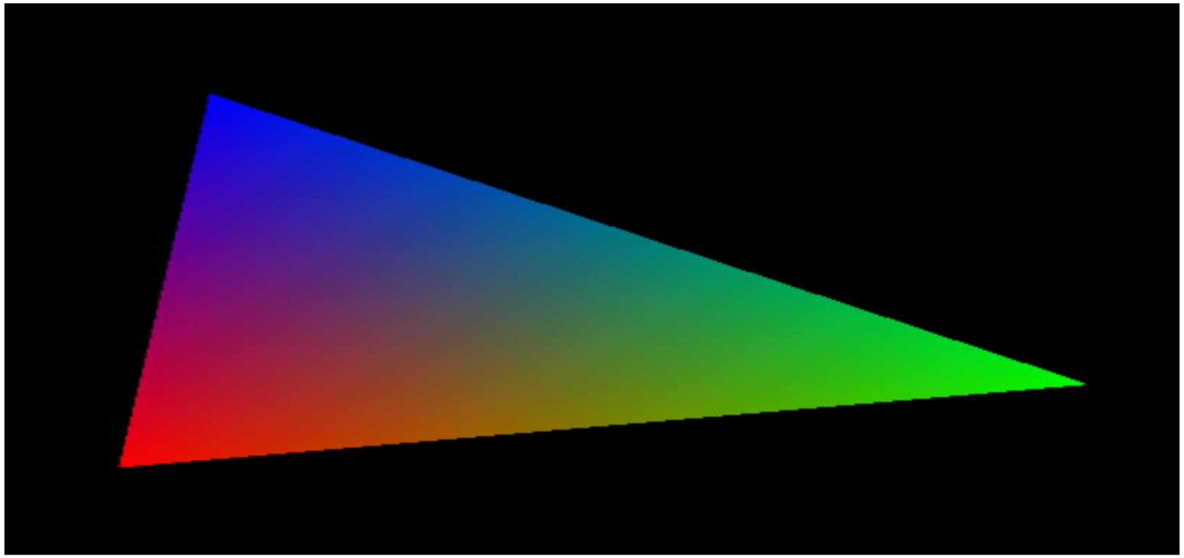
Scan Conversion/Rasterization

- Determine which fragments get generated
- Interpolate parameters (colors, textures, normals, etc.)



Scan Conversion/Rasterization

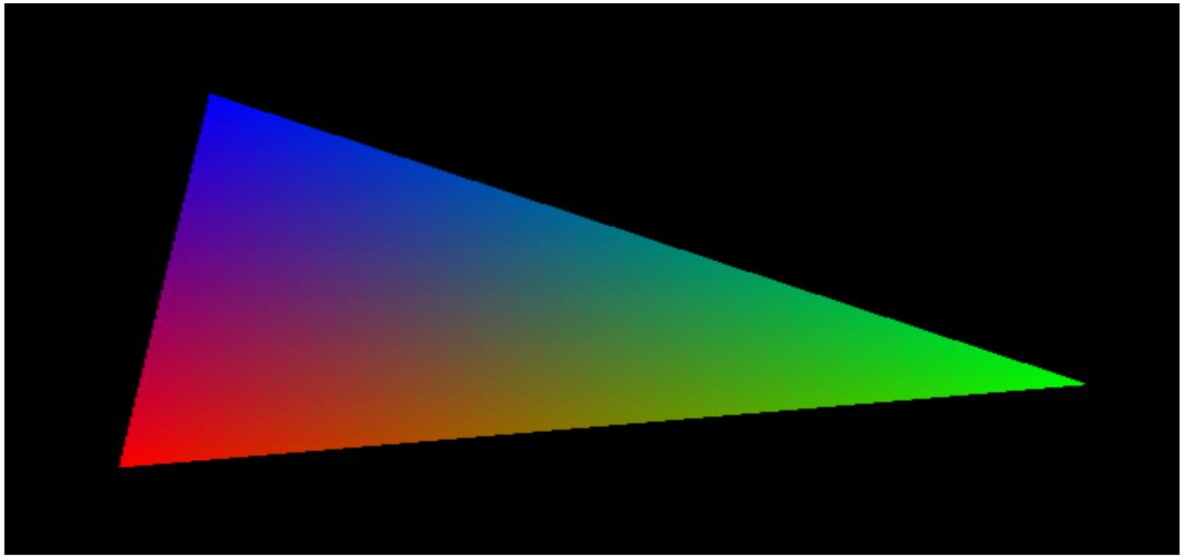
- Determine which fragments get generated
- Interpolate parameters (colors, textures, normals, etc.)



- How?

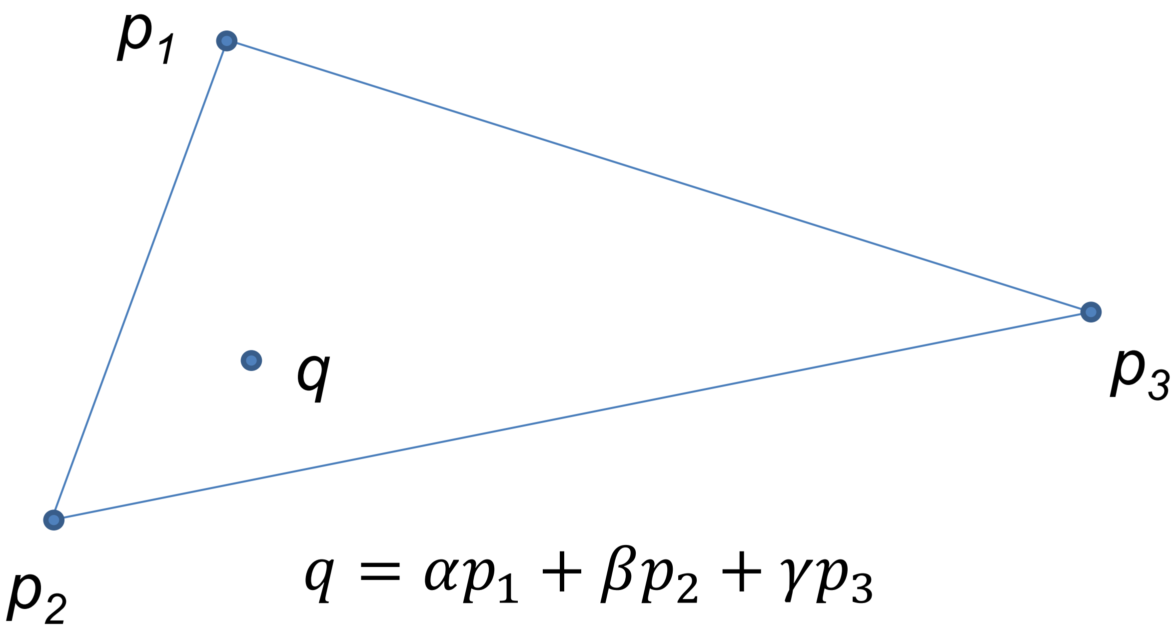
Scan Conversion/Rasterization

- Determine which fragments get generated
- Interpolate parameters (colors, textures, normals, etc.)



- Barycentric coords amongst many other ways..

Barycentric coordinates



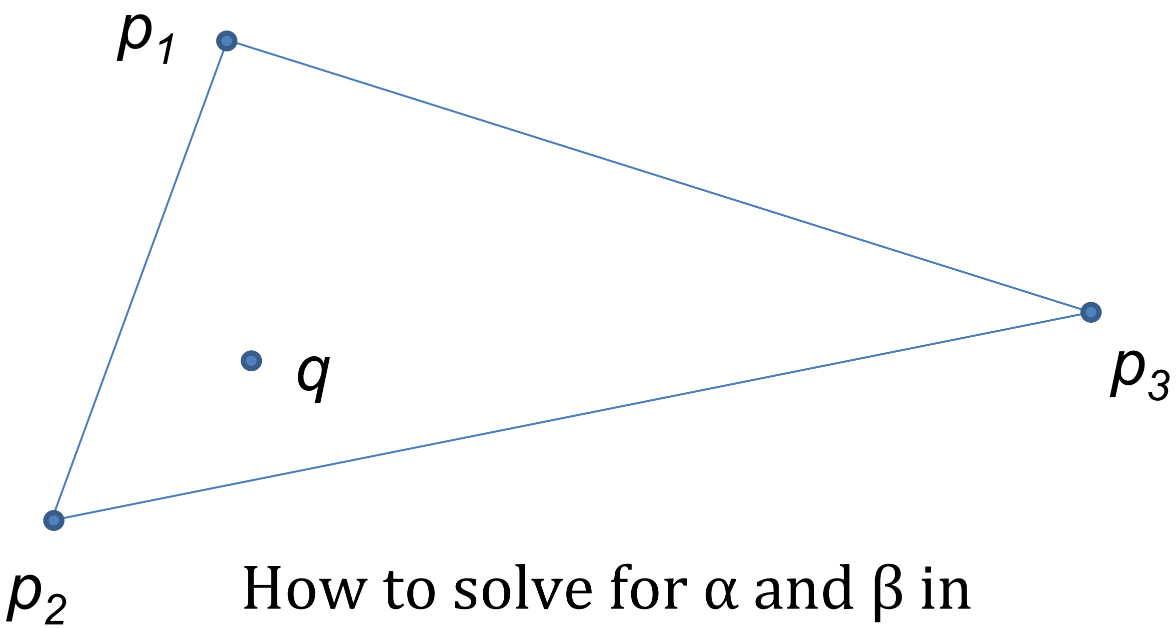
$$q = \alpha p_1 + \beta p_2 + \gamma p_3$$

If $[\alpha + \beta + \gamma = 1 \;\; and \;\; \{\alpha, \beta, \gamma\} \geq 0]$,
then q inside triangle (p_1, p_2, p_3)

Can also write:

$$q = \alpha p_1 + \beta p_2 + (1 - \alpha - \beta) p_3$$

Barycentric coordinates



How to solve for α and β in

$$q = \alpha p_1 + \beta p_2 + (1 - \alpha - \beta) p_3?$$

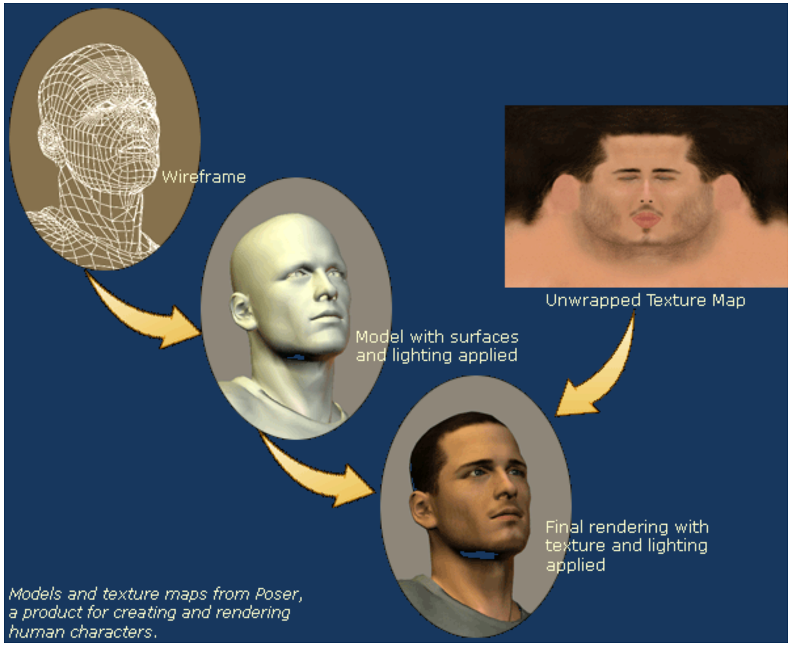
Two equations, two unknowns:
use 2x2 matrix inversion…

Additional concept: Texture mapping

- Model surface-detail with images
  - wrap object with photograph(s)
  - graphics object itself is a simpler model but "looks" more complex

Texture mapping

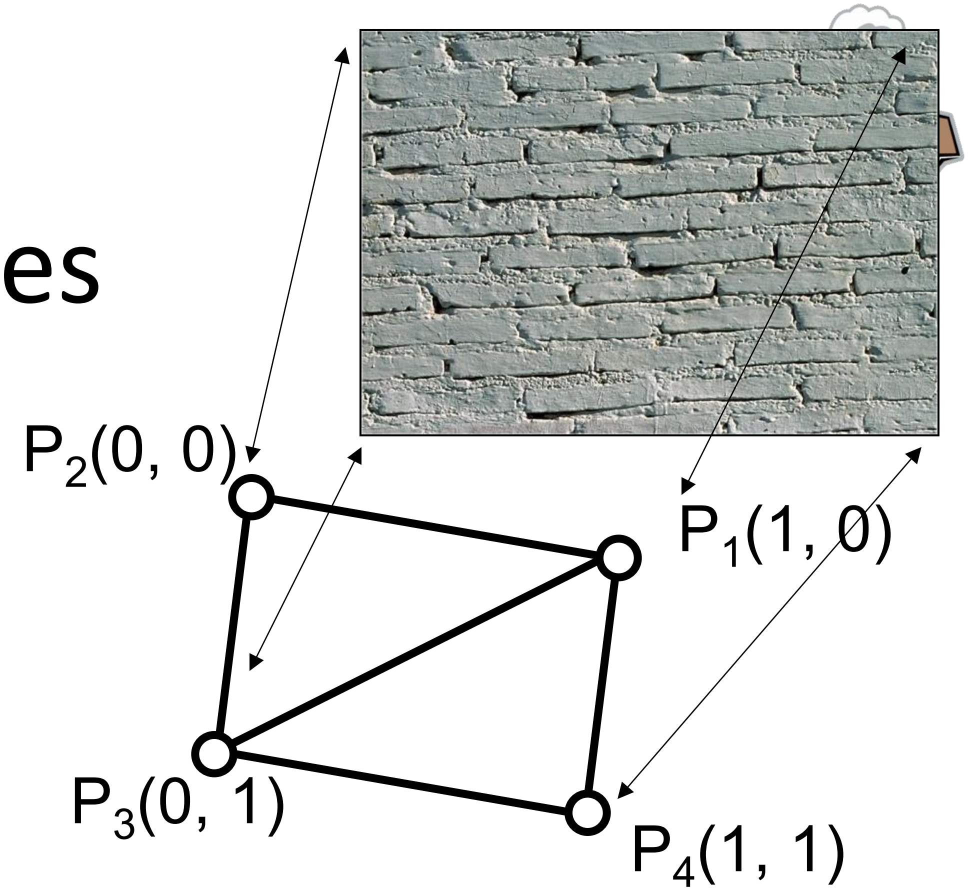
- Model surface-detail with images
  - wrap object with photograph(s)
  - graphics object itself is a simpler model but “looks” more complex

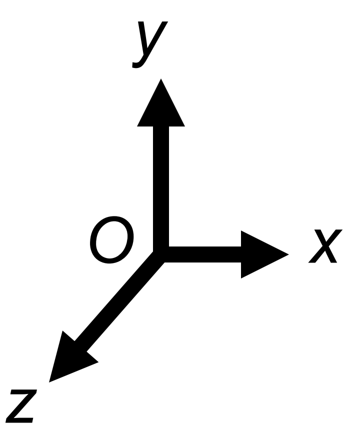


Texture coordinates

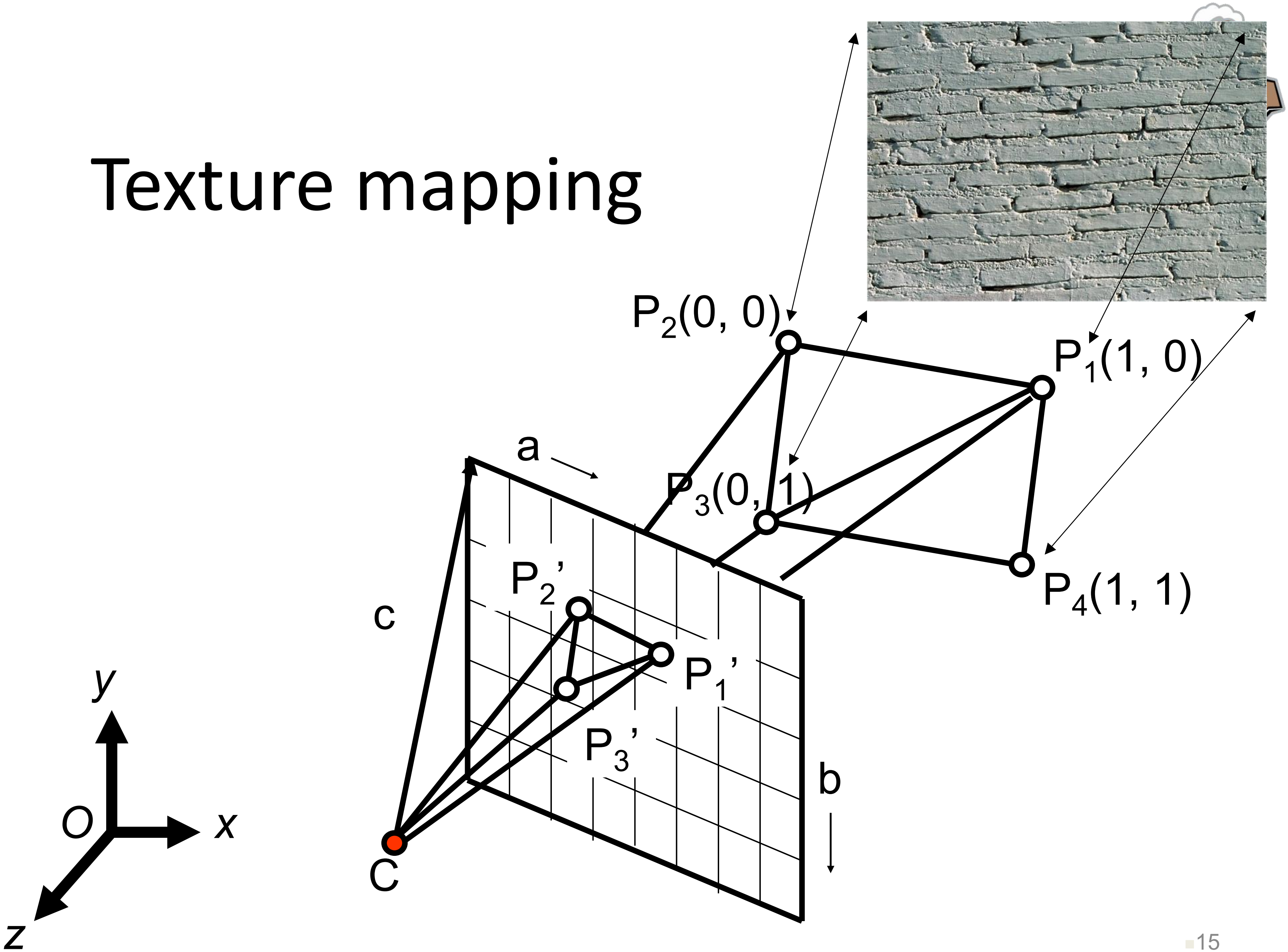
- Mechanism for attaching the texture map to the surface modeled
  - a pair of floats (s, t) for each triangle vertex
  - corners of the image are (0, 0), (0, 1), (1, 1), and (1, 0)
  - tiling indicated with tex. coords. > 1
  - *texels* – color samples in texture maps

Texture coordinates

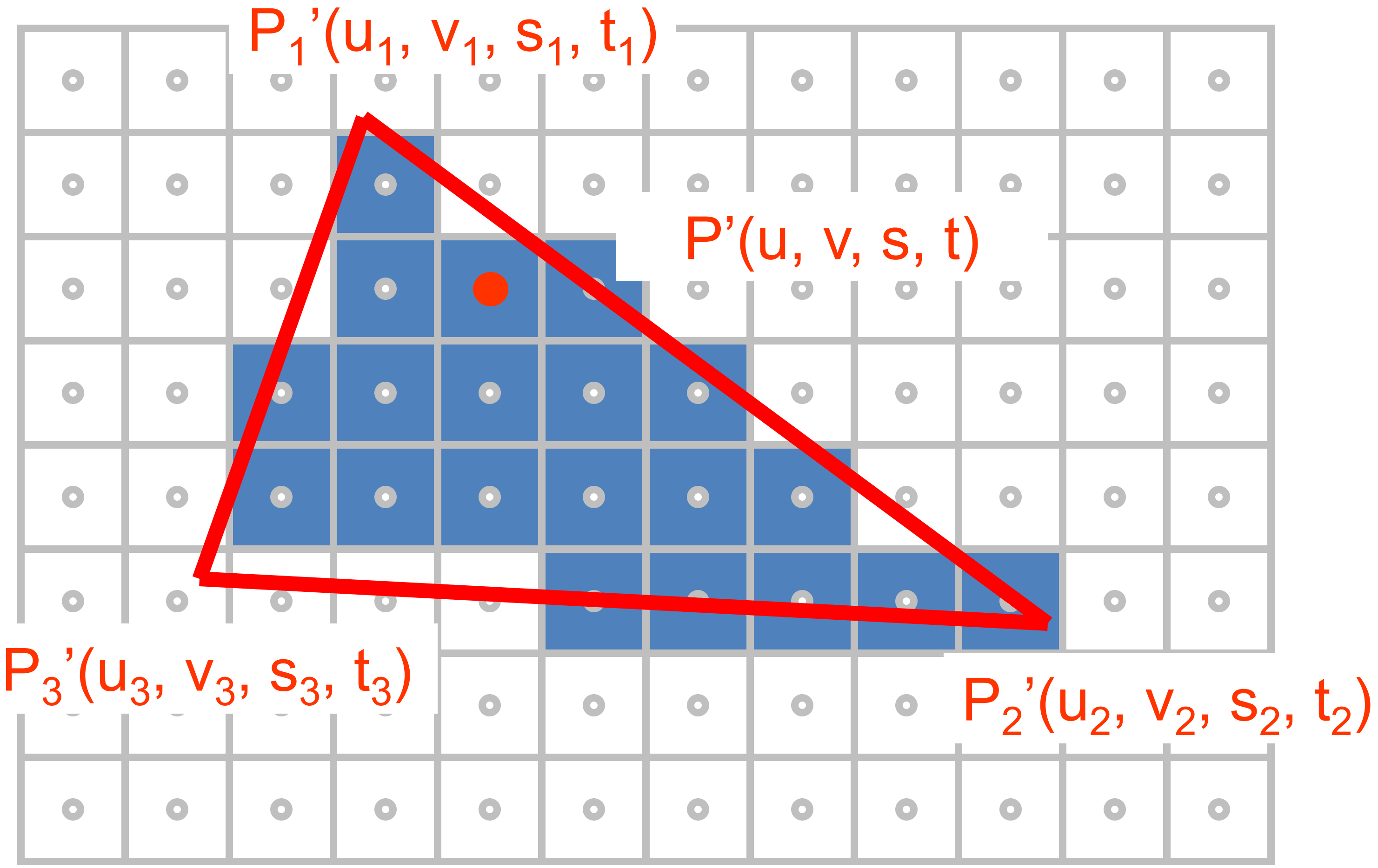




Texture mapping

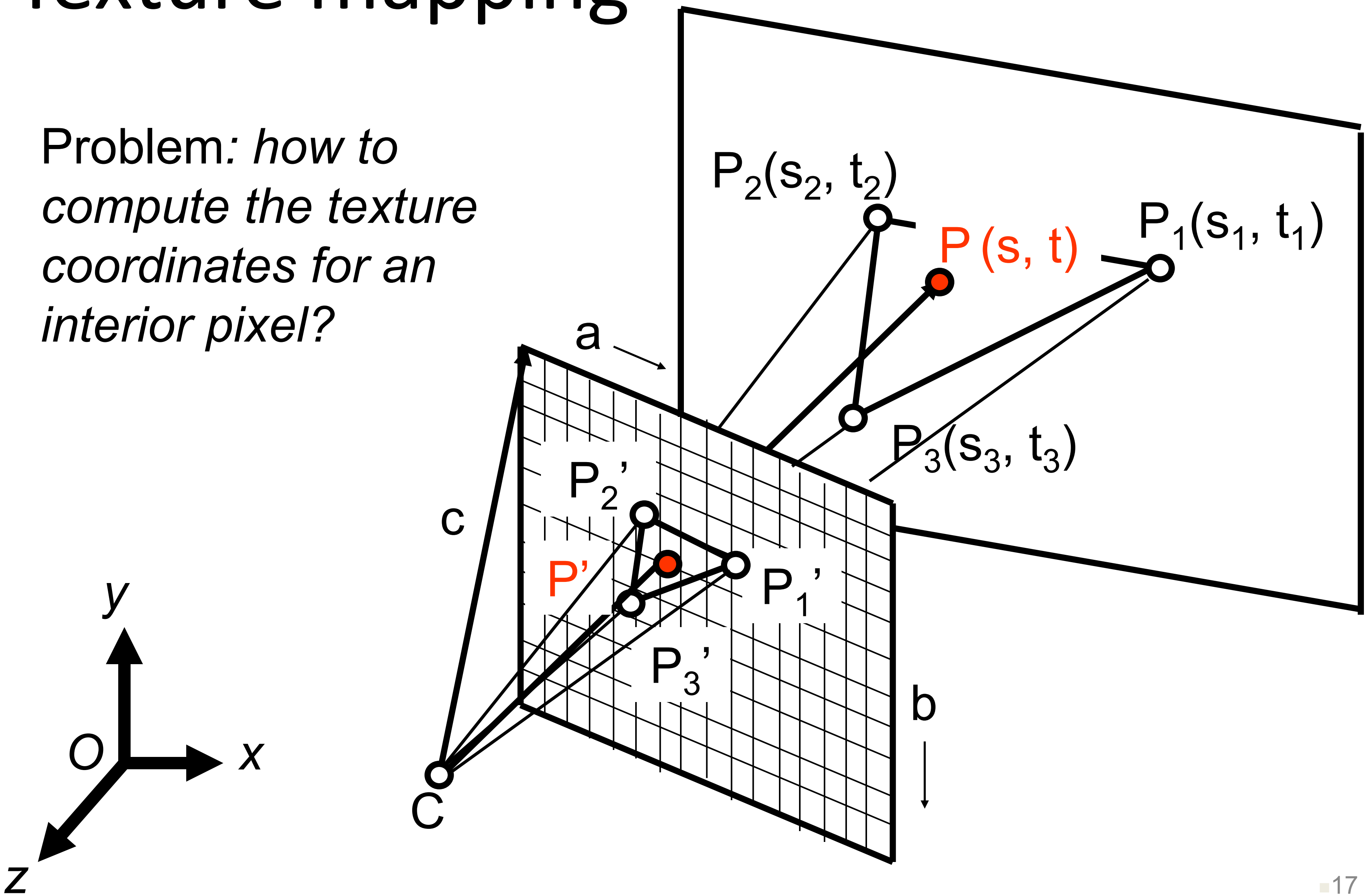


Texels: texture elements



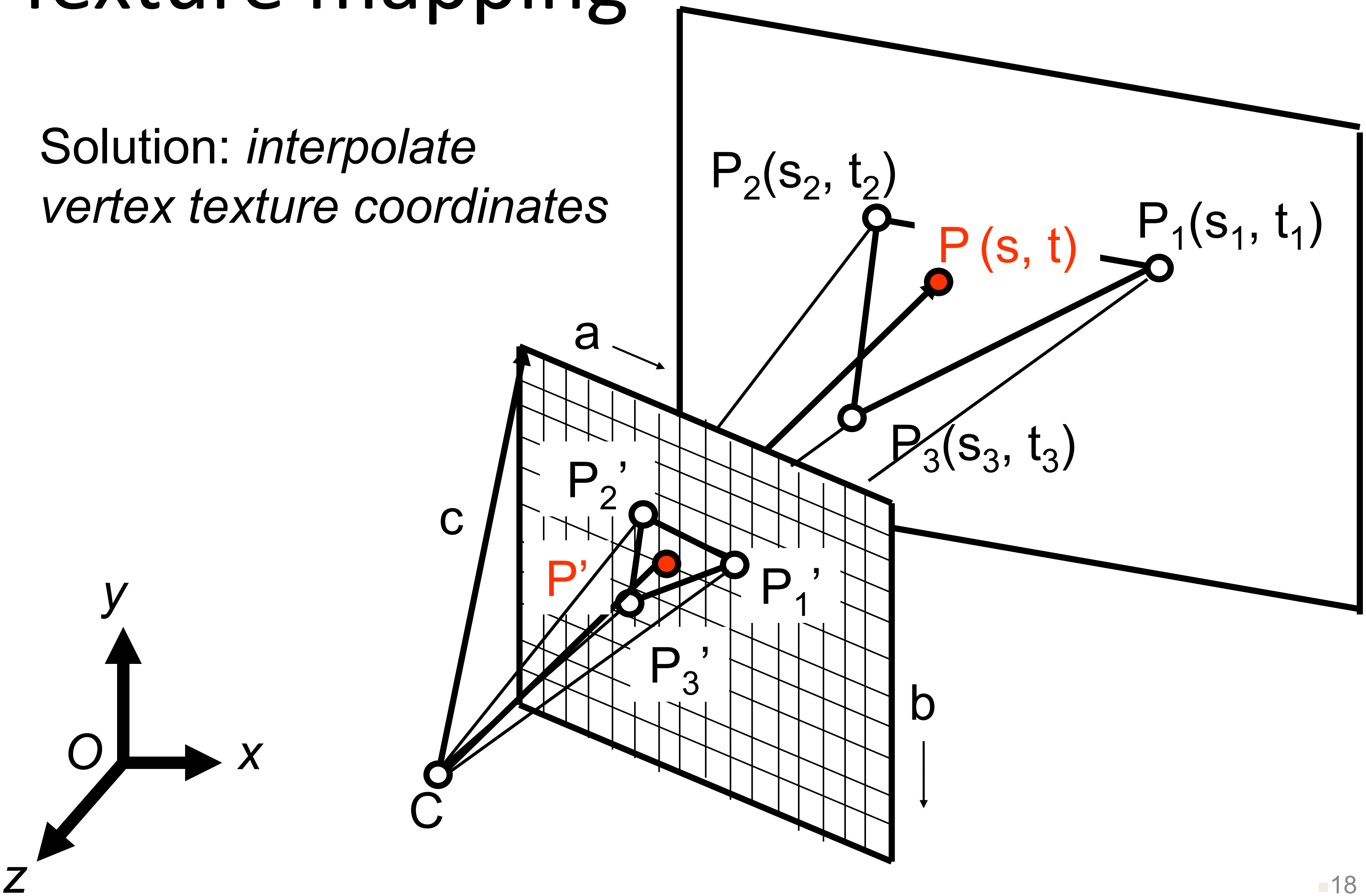
Texture mapping

Problem*: how to compute the texture coordinates for an interior pixel?*



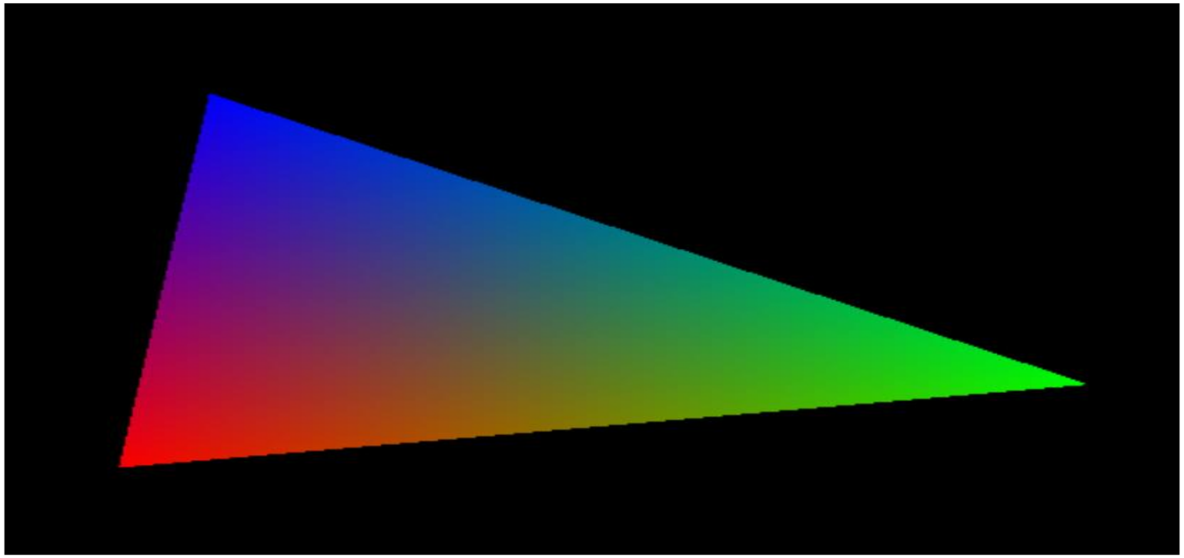
Texture mapping

Solution: *interpolate vertex texture coordinates*



Parameter Interpolation

- Texture coordinates, colors, normals, etc.



- How?
  - Again, use barycentric coordinates…

Computer Graphics Pipeline

Geometry

Modeling	Modeling Transformation	Transform into 3D *world* coordinate system
Illum.	Lighting	Simulate illumination and reflectance
Rendering	Viewing Transformation	Transform into 3D *camera* coordinate system
	Clipping	Clip primitives outside camera's view
	Projection	Transform into 2D camera coordinate system
	Scan Conversion	Draw pixels (incl. texturing, hidden surface…)

Image

Modeling (or Content Generation)

- Manual Modeling (most prevalent approach):
  - Softimage (Microsoft+SoftImage3D, 1989-2002)
  - Maya (SGI+Alias+Wavefront, 1998-today)
  - Blender (open source, 1995-today)
  - Houdini (SideFX, 1996-today)
  - Rhino/Grasshopper (TLM, 1978? – today)
  - And more…
- Problem: TIME CONSUMING!
  - E.g., 3-20 days for a photorealistic character

Modeling (or Content Generation)

- Texture/Image Synthesis
- **Procedural Modeling**
- Generative Modeling
- Neural Modeling (and Rendering)

Illumination

- Local Illumination
  - **Diffuse, Specular**
- Global Illumination
  - **Ray Tracing**
  - Path Tracing (Single and Bidirectional)
  - Photon Mapping
  - **Radiosity**

Rendering

- Point rendering
- **Ray tracing**
- **Polygon/triangle rendering**
- Differentiable Rendering

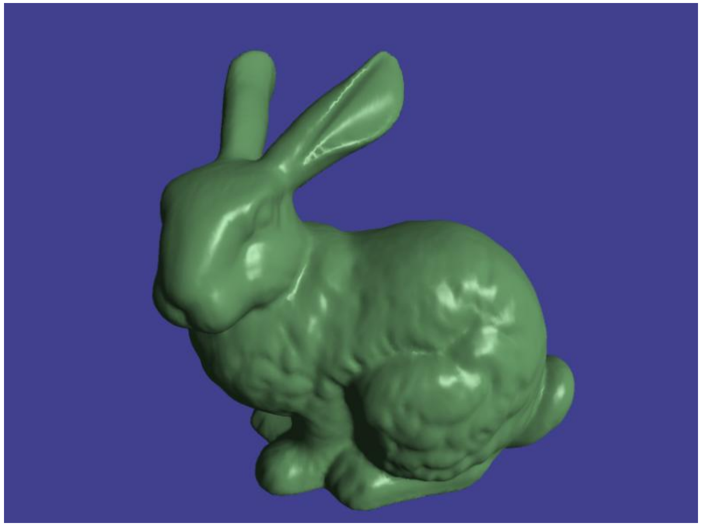
Lighting and Illumination

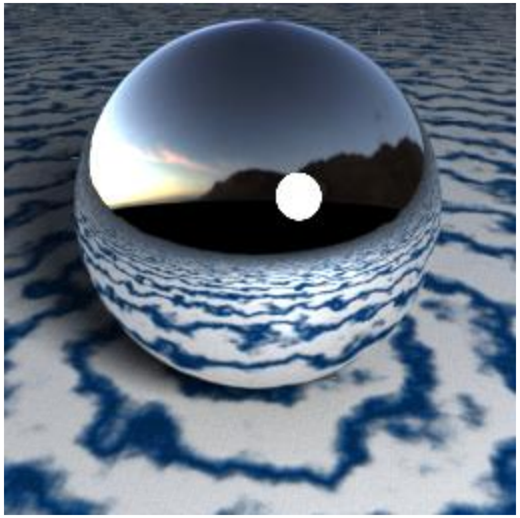
- Local Illumination
  - **Diffuse, Specular**
- Global Illumination
  - **Ray Tracing**
  - Path Tracing (Single and Bidirectional)
  - Photon Mapping
  - **Radiosity**

Diffuse Reflection

(mostly)

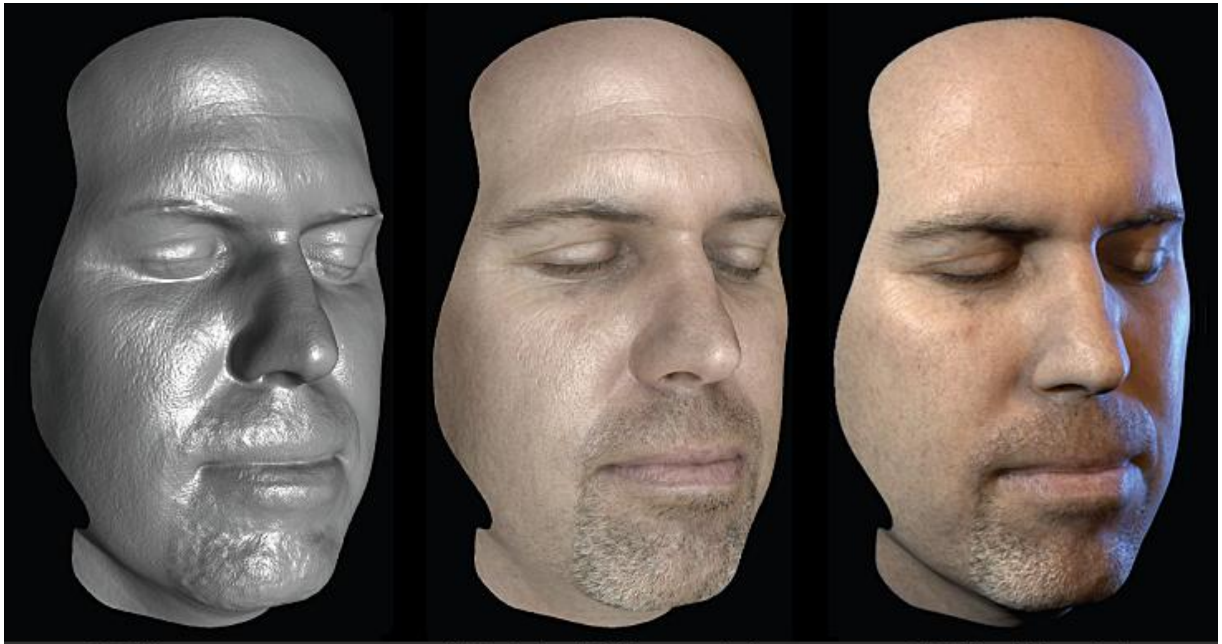
Specular Reflection





also Environment Mapping

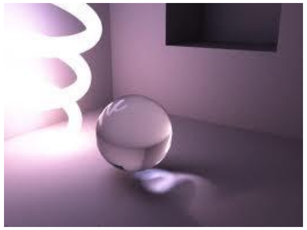
also Subsurface Scatterring

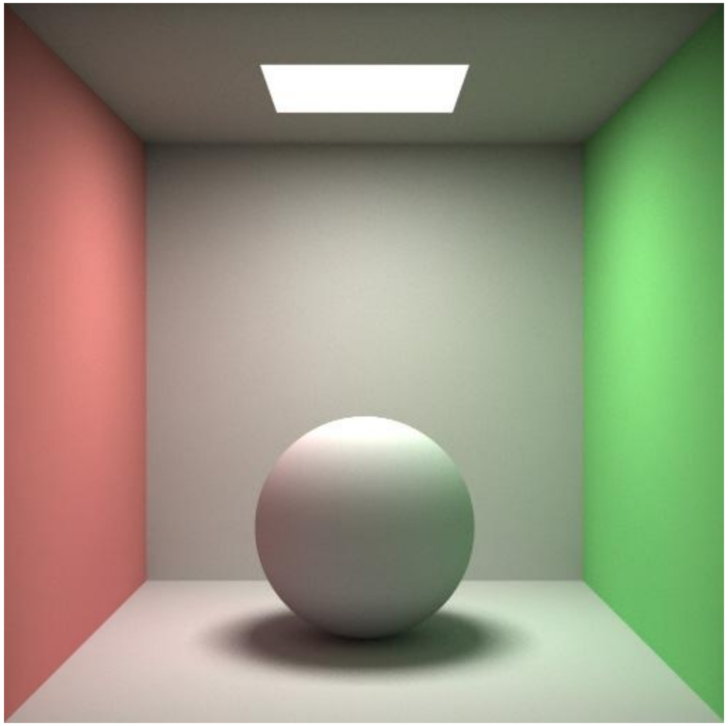


(a) High-res geometry (b) Real-time hybrid map rendering (c) Offline SSS rendering

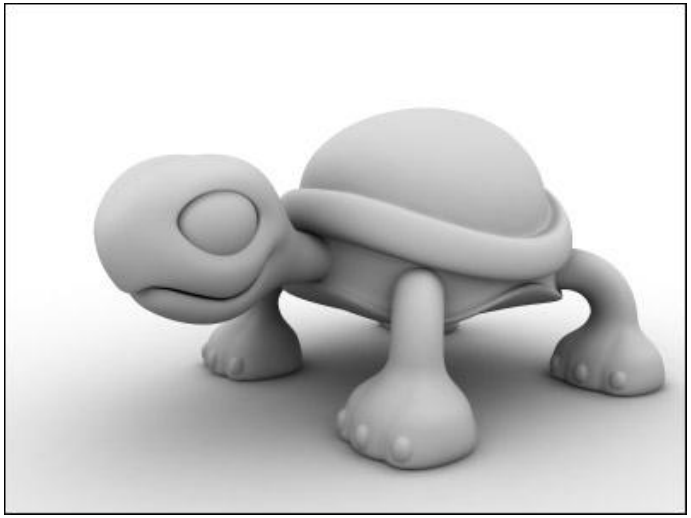
Towards Global Illumination…

Transparency





Radiosity



Ambient occlusion

Lighting and Illumination

- Light sources
  - Point light
    - Models an omnidirectional light source (e.g., a bulb)
  - Directional light
    - Models an omnidirectional light source at infinity
  - Spot light
    - Models a point light with direction
- Basic Direct Lighting Models
  - Ambient light
  - Diffuse reflection
  - Specular reflection

Lighting and Illumination

- Light sources
  - Point light
    - Models an omnidirectional light source (e.g., a bulb)
  - Directional light
    - Models an omnidirectional light source at infinity
  - Spot light
    - Models a point light with direction

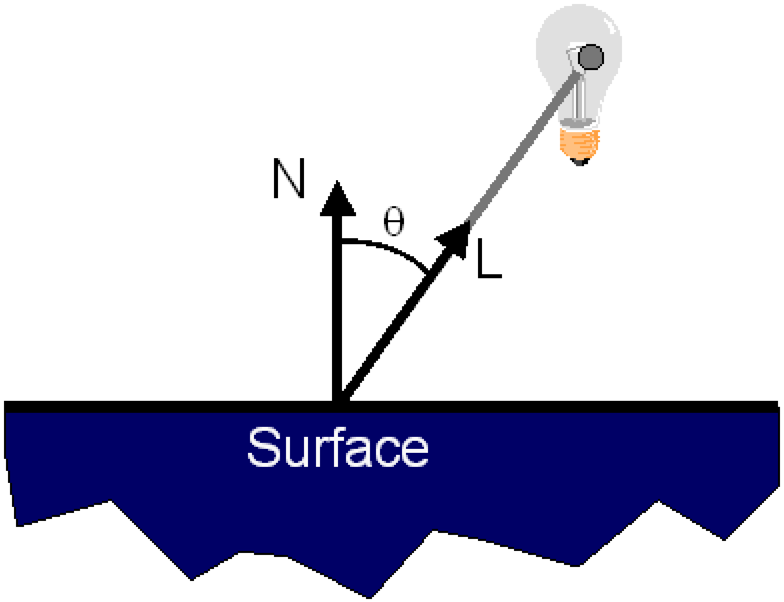
- Basic Direct Lighting Models
  - **Ambient light: just a “constant”, e.g., 0.1 of background color**
  - Diffuse reflection
  - Specular reflection

Lighting and Illumination

- Diffuse reflection
  - Lambertian model

Lighting and Shading

- Diffuse reflection
  - Lambertian model



Lighting and Illumination

- Light sources
  - Point light
    - Models an omnidirectional light source (e.g., a bulb)
  - Directional light
    - Models an omnidirectional light source at infinity
  - Spot light
    - Models a point light with direction

- Basic Direct Lighting Models
  - Ambient light
  - Diffuse reflection
  - Specular reflection

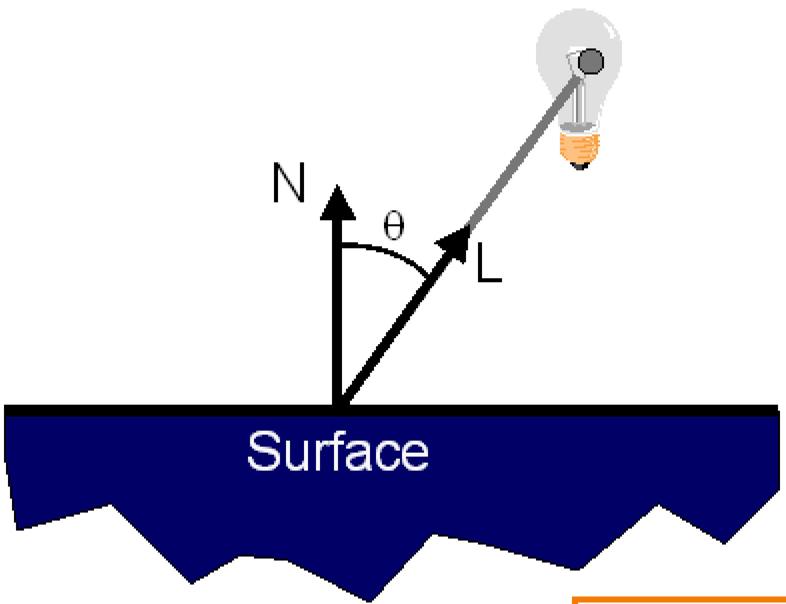
Lighting and Illumination

- Light sources
  - Point light
    - Models an omnidirectional light source (e.g., a bulb)
  - Directional light
    - Models an omnidirectional light source at infinity
  - Spot light
    - Models a point light with direction

- Basic Direct Lighting Models
  - Ambient light: just a "constant", e.g., 0.1 of background color
  - Diffuse reflection
  - Specular reflection

Lighting and Illumination

- Diffuse reflection
  - Lambertian model



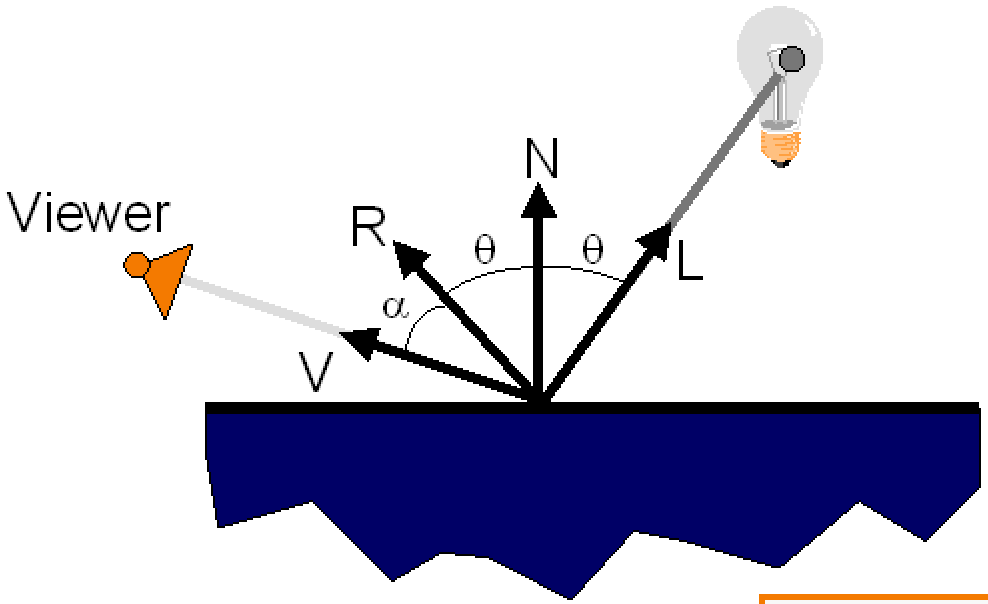
$$I_D = K_D(N \bullet L)I_L$$

Lighting and Illumination

- Specular reflection
  - Phong model

Lighting and Illumination

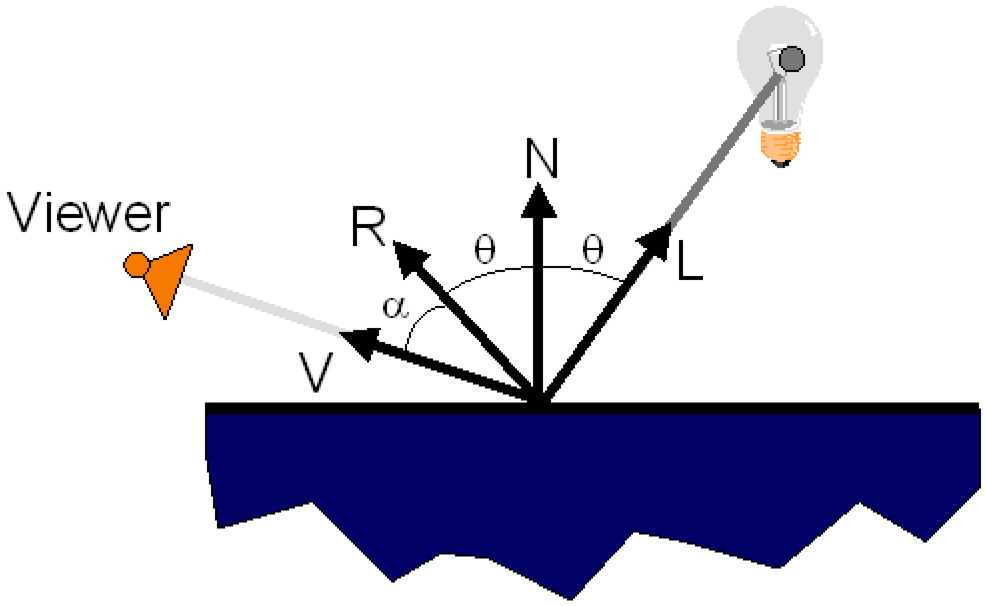
- Specular reflection
  - Phong model



$$I_S = K_S (V \bullet R)^n I_L$$

Lighting and Illumination

- Specular reflection
  - Phong model



But there is more…

Specular Reflection

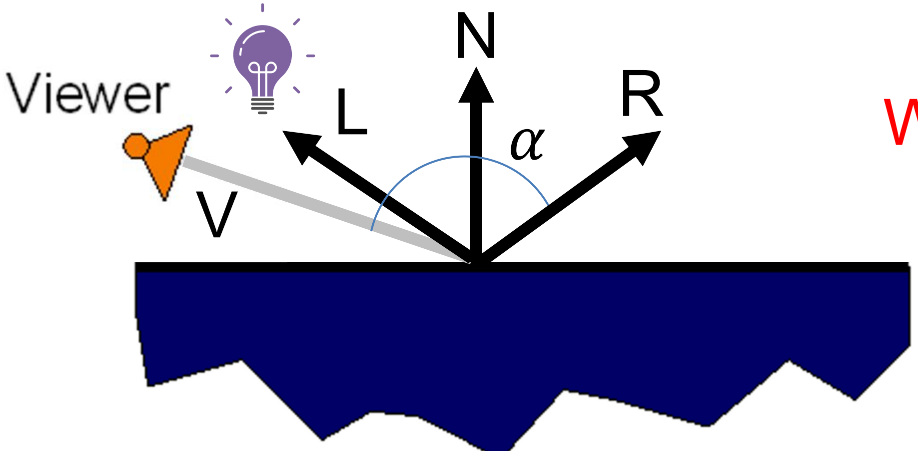
- Problem:

α is ?

$\alpha > 90$

$V \cdot R < 0$

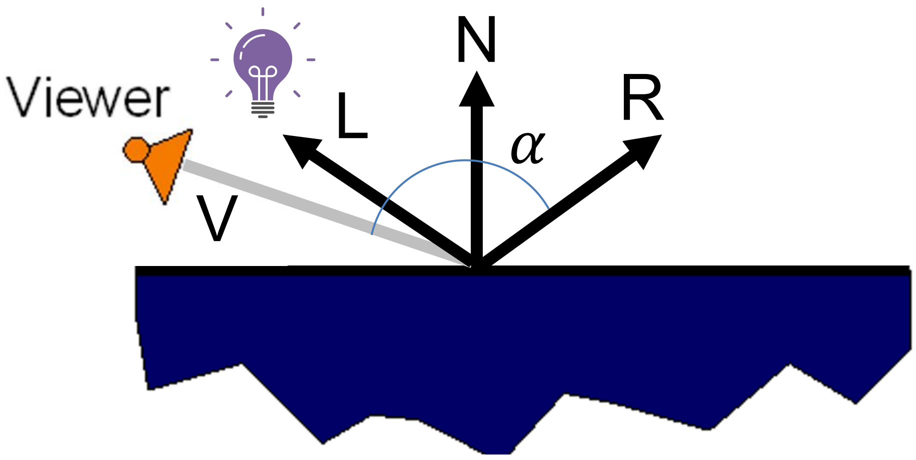
$K_S(V \cdot R)^n I_L < 0$



What can be done?

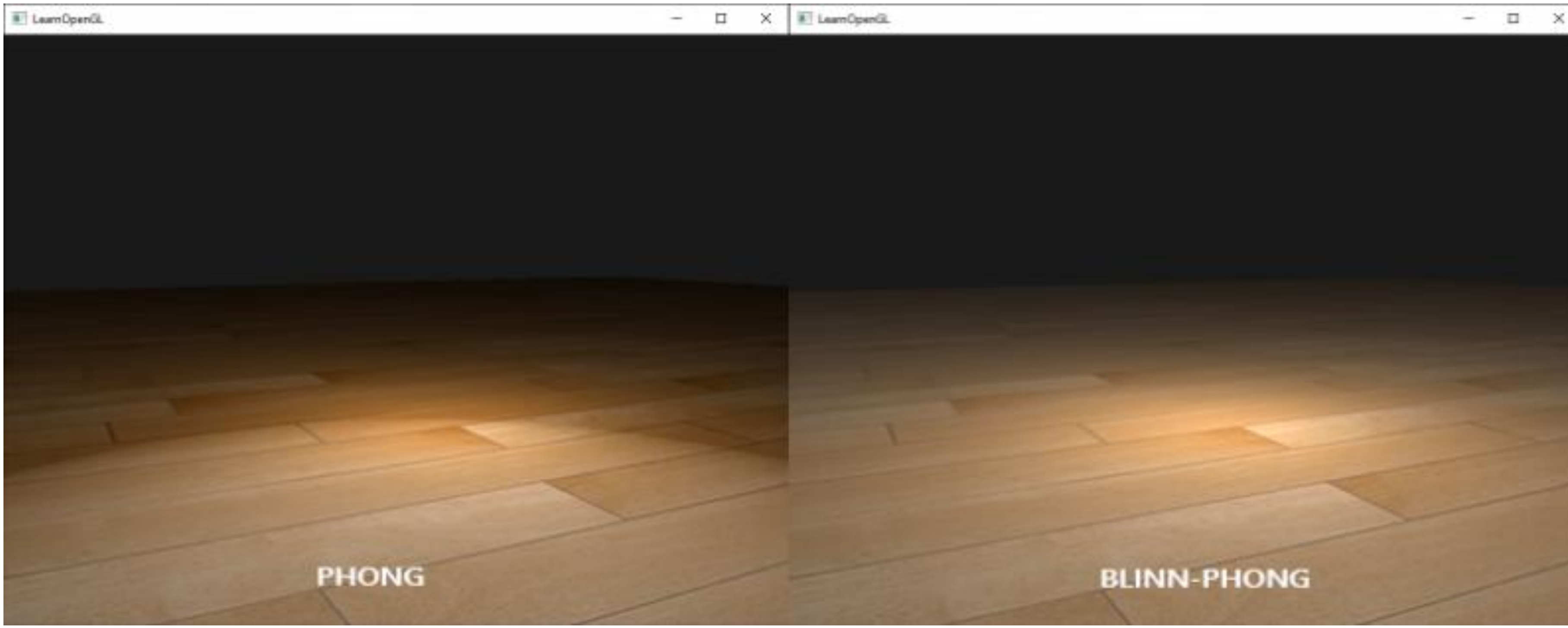
Specular Reflection

- Blinn-Phong Model (or Blinn Model)



$$H = (L + V)/\|L + V\|$$

$$I_{bp} = K_s(H \cdot N)^n I_L$$

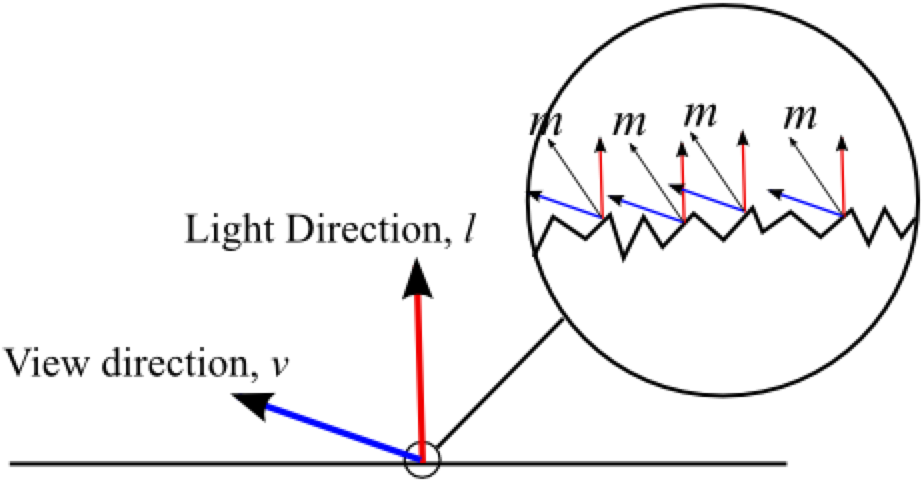

LearnOpenGL
LearnOpenGL
PHONG
BLINN-PHONG

Physically-Based Illumination Models

- Consider
  - Radiance
  - Irradiance
  - Microfacets
  - And more…

Microfacets

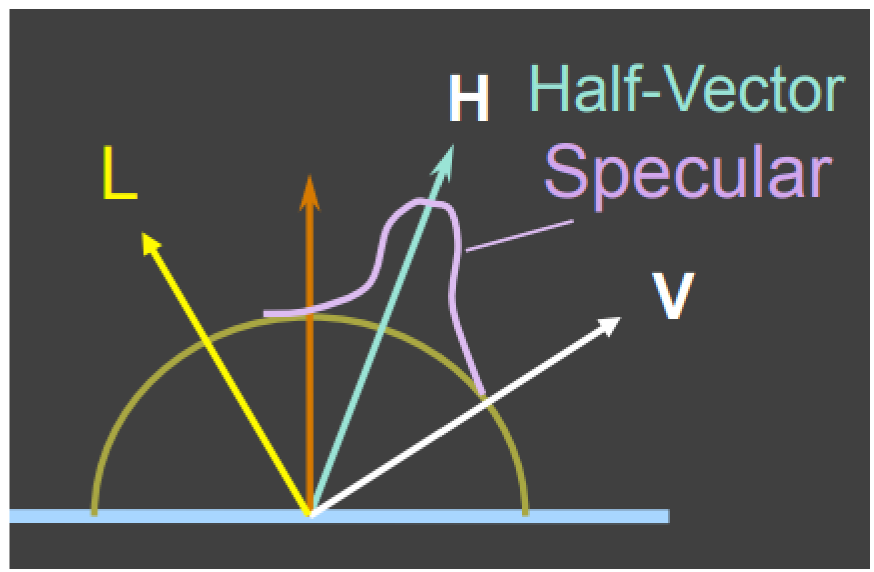
- Key question:
  - What proportion of the microfacets of the surface are oriented in such a way as to specularly reflect light towards the viewer?



Microfacets

- Key question:
  - What proportion of the microfacets of the surface are oriented in such a way as to specularly reflect light towards the viewer?

- If “very specular”, how does shape change?
- If “not very specular”, how does shape change?

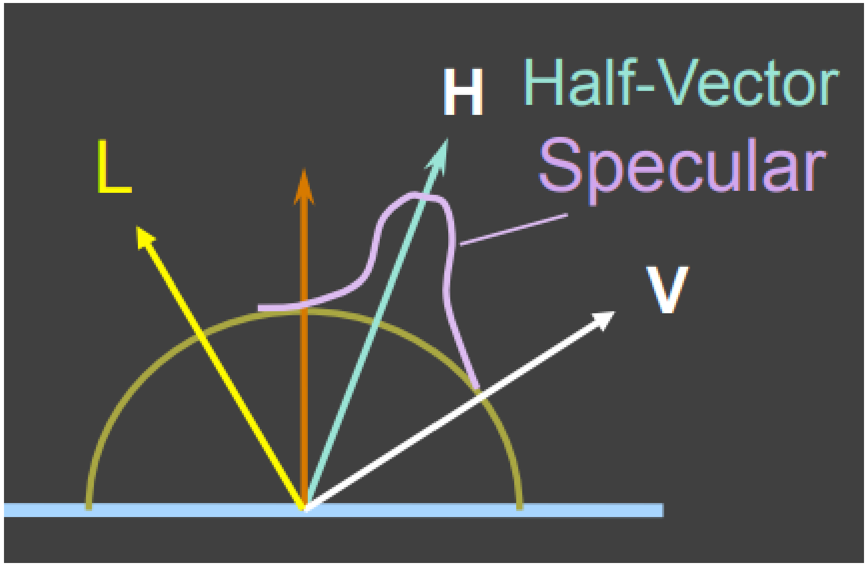


[courtesy Kavita Bala]

Microfacets

- Key question:
  - What proportion of the microfacets of the surface are oriented in such a way as to specularly reflect light towards the viewer?

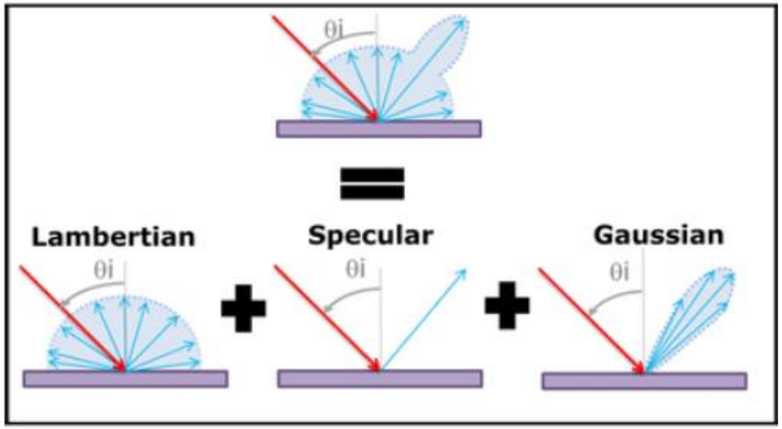
- How else can change shape?
  - Hint: a varying amount of microfacets reflect light in different directions…



[courtesy Kavita Bala]

Microfacets

- Key question:
  - What proportion of the microfacets of the surface are oriented in such a way as to specularly reflect light towards the viewer?
- An approximation:
  - Instead of $H \cdot N$ or $V \cdot R$, use Gaussian…



[courtesy L. Marot]

Microfacets

- Key question:
  – What proportion of the microfacets of the surface are oriented in such a way as to specularly reflect light towards the viewer?
- An approximation:
  – Instead of $H \cdot N$ or $V \cdot R$, use $e^{-\left(\frac{\alpha}{m}\right)^2}$ where $m \in [0,1]$ where larger m means rougher surface (=less specular)

$$I_{bp} = K_s e^{-\left(\frac{\mathrm{acos}(H \cdot N)}{m}\right)^2} I_L$$

Microfacets

- Torrance-Sparrow Model
  - Surface reflection model developed by physicists, assumes isotropic collection of planar microfacets

$$I_s = \frac{F_k}{\pi} \frac{DG}{(N \cdot V)(N \cdot L)}$$

where

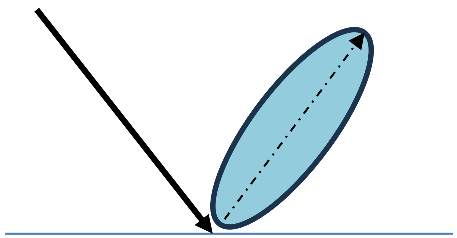
- D is distribution function of microfacet orientations
- G is geometrical attenuation factor (e.g., occlusion, shadowing)
- F_k is the Fresnel term describing the reflection and transmission of light incident on different media

Microfacets

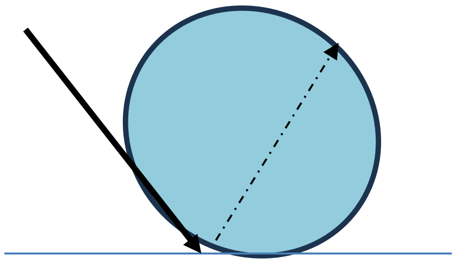
- Torrance-Sparrow Model

$$D = \frac{1}{4m^2 \cos^4 \beta} e^{-\left(\frac{\tan \beta}{m}\right)^2}$$

  - β is angle between N and H
  - m is root-mean-square slope of the microfacets
    - small m = facets similar to normal; highly directional
    - large m = facets slopes are steep; rough surface



m=0.2



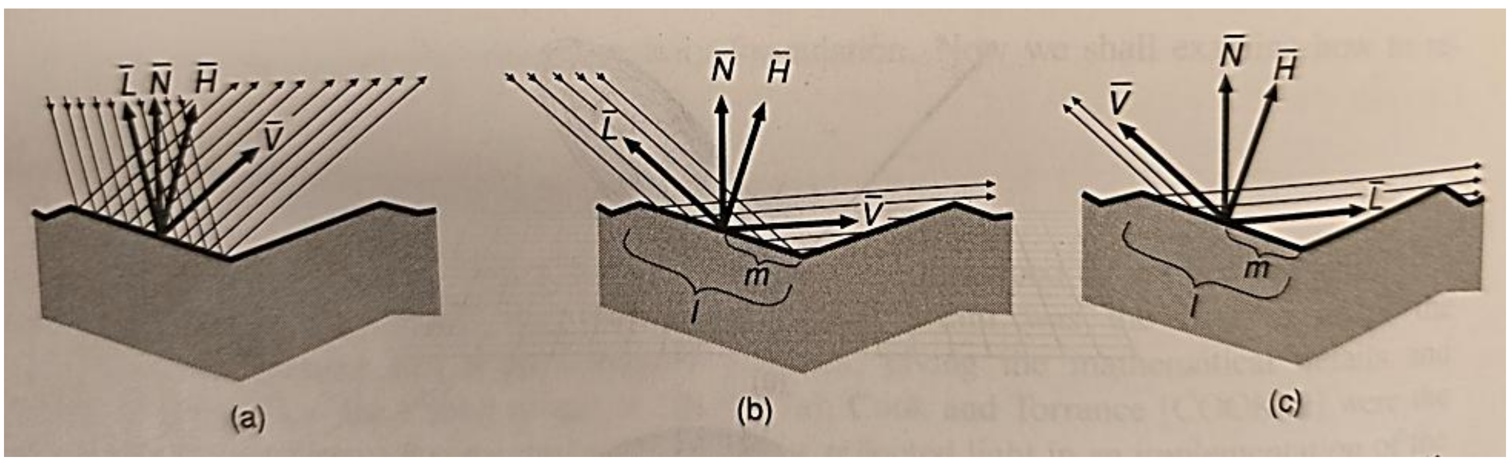
m=0.6

Microfacets

- Torrance-Sparrow Model

$$G = \min\{1, \frac{2(N\cdot H)(N\cdot V)}{V\cdot H}, \frac{2(N\cdot H)(N\cdot L)}{V\cdot H}\}$$

…quantifies how much microfacets shadow each other



Microfacets

- Torrance-Sparrow Model

$$F_k = \frac{1}{2}\frac{(g-c)^2}{(g+c)^2}\left(1+\frac{[c(g+c)-1]^2}{[c(g-c)+1]^2}\right)$$

where $c = \mathrm{L} \cdot H$, $g^2 = \eta_\lambda^2 + c^2 - 1$, $\eta_\lambda = \frac{\eta_{t\lambda}}{\eta_{i\lambda}}$

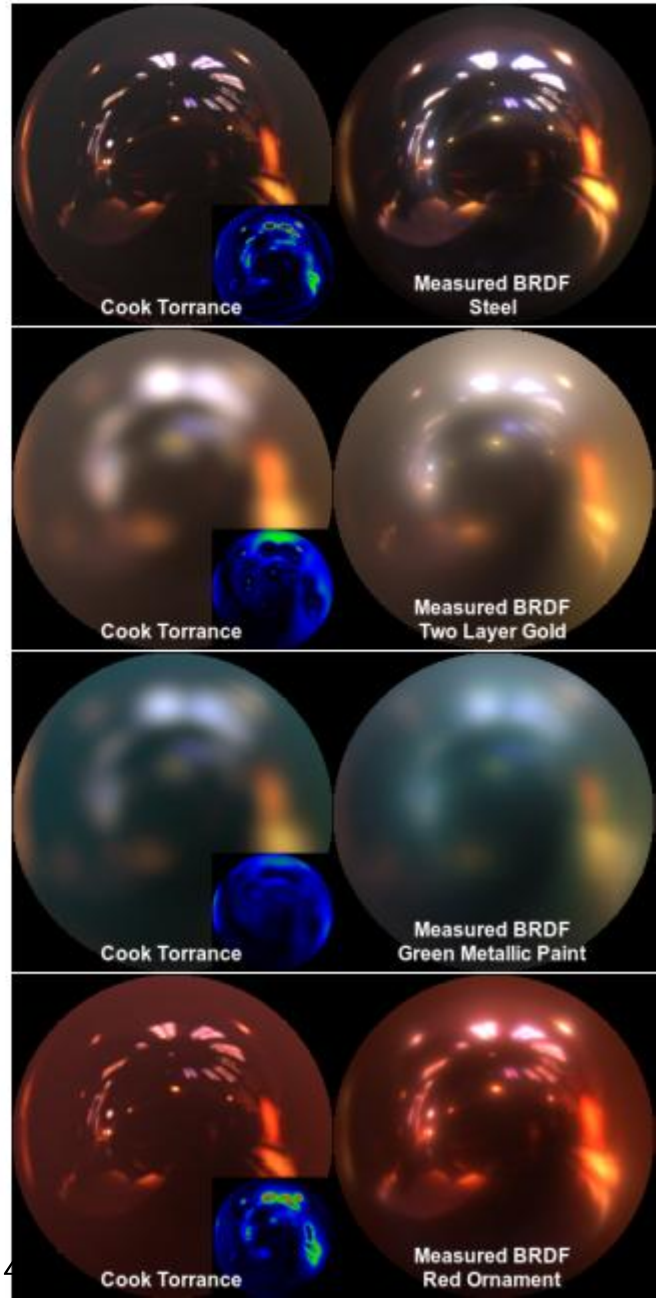
or, simplified to

$F_k \approx F_0 + (1 - F_0)(1 - (V \cdot H))^5$ [Schlick's approximation]

where F_0 is reflectance at normal incidence (e.g., RGB color)

Microfacets

- Cook-Torrance Model
  - Similar, with some improvements
  - D is a weighted sum of distribution functions to model surfaces of multiple scales of roughness



[Brady et al., TOG 2014

What about rough (diffuse) surfaces?

- Must consider occlusion, shadowing, and diffuse interreflection – thoughts?
- $I_d = N \cdot L$
- Is that it?

What about rough (diffuse) surfaces?

- Must consider occlusion, shadowing, and diffuse interreflection – thoughts?
- Derived from microfacets model
  - Oren-Nayar Model:

$$L_r = L_1 + L_2$$

$$L_1 = \frac{\rho}{\pi} E_0 \cos\theta_i \left(C_1 + C_2 \cos(\phi_i - \phi_r) \tan\beta + C_3 (1 - |\cos(\phi_i - \phi_r)|) \tan\frac{\alpha+\beta}{2} \right),$$

$$L_2 = 0.17 \frac{\rho^2}{\pi} E_0 \cos\theta_i \frac{\sigma^2}{\sigma^2 + 0.13} \left[1 - \cos(\phi_i - \phi_r) \left(\frac{2\beta}{\pi} \right)^2 \right],$$

where

$$C_1 = 1 - 0.5 \frac{\sigma^2}{\sigma^2 + 0.33},$$

$$C_2 = \begin{cases} 0.45 \frac{\sigma^2}{\sigma^2 + 0.09} \sin\alpha & \text{if } \cos(\phi_i - \phi_r) \geq 0, \\ 0.45 \frac{\sigma^2}{\sigma^2 + 0.09} \left(\sin\alpha - \left(\frac{2\beta}{\pi} \right)^3 \right) & \text{otherwise,} \end{cases}$$

$$C_3 = 0.125 \frac{\sigma^2}{\sigma^2 + 0.09} \left(\frac{4\alpha\beta}{\pi^2} \right)^2,$$

$$\alpha = \max(\theta_i, \theta_r),$$

$$\beta = \min(\theta_i, \theta_r),$$

What about rough (diffuse) surfaces?

- Oren-Nayar Model (simplified)

$$L_r = \cos\theta_i\,[A + B\max(0, \cos(\theta_r - \theta_i))\sin\alpha\tan\beta]$$

where using numerical experiments:

$$A \approx \rho(\frac{1}{\pi} - 0.09\,\frac{\sigma^2}{\sigma^2 + 0.4})$$

$$B \approx \rho(0.125\,\frac{\sigma^2}{\sigma^2 + 0.18})$$

Real Image Lambertian Model Oren-Nayar Model