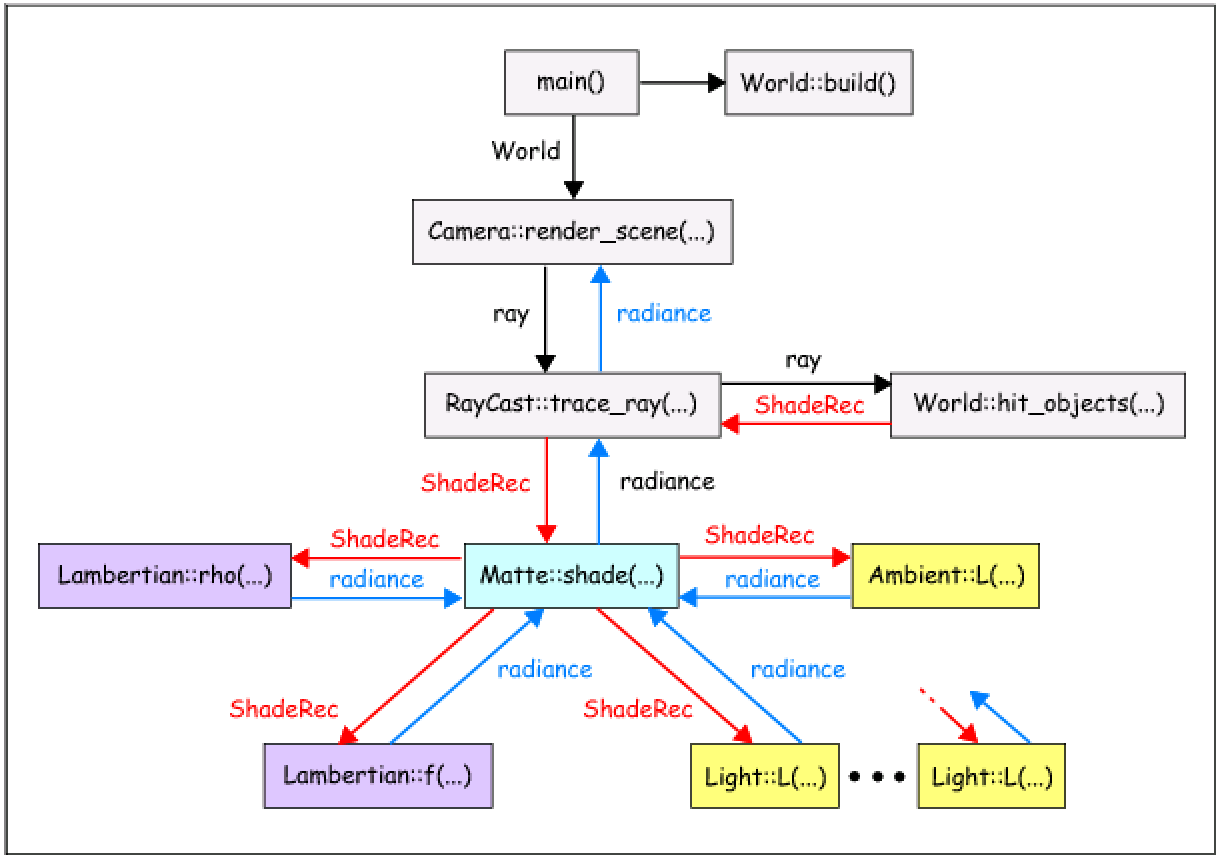
CS 563 Advanced Topics in Computer Graphics
Lights and Materials

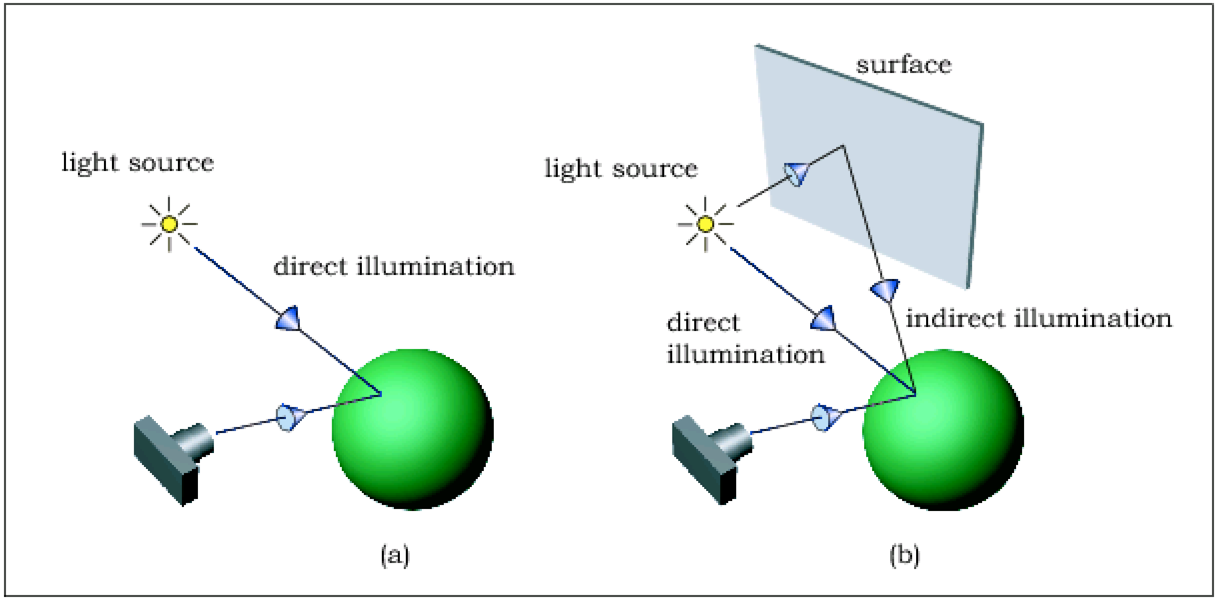
by Steve Olivieri

Shading Architecture



Two Types of Illumination

- Direct Illumination: light that hits a surface by traveling directly from a light source. (a)
- Indirect Illumination: light that hits a surface after being reflected from another surface. (b)



Reflection of Light

- Different materials reflect light in different ways.
  - Perfect Diffuse: light scattered equally in all directions (matte surfaces).
  - Perfect Specular: light reflected in a single mirror ray (mirror surfaces).
  - Glossy Specular: light reflected in a mirror direction, but scattered (shiny surfaces).
- Not all materials fit these three models!

Lights

- Ray tracing is all about modeling the behavior of lights in a scene.
- There are many different types of lights, including ambient, ambient occluder, directional, point, area, and environment.
- Light are typically defined by their power (radiant flux), but we can fudge it with
  - c_l, the color of the light (RGB)
  - l_s, the radiance scaling factor (intensity)

Ambient Lights

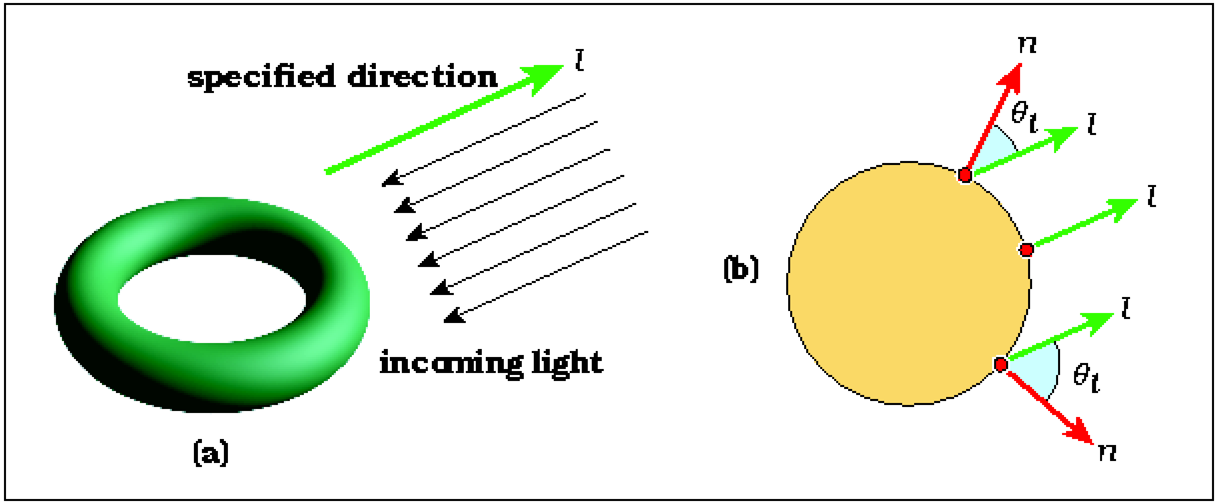
- It is difficult to model indirect, diffuse illumination.
- Instead, we define an ambient light source.
  - Constant color and intensity throughout the scene.
  - Prevents surfaces receiving only indirect illumination from appearing too dark, or black.
  - Not a real, physical light.
- The incident radiance from the ambient light is $L_i = L_s c_l$.
- The reflected ambient radiance is $L_o = \rho_{hh}(p) * l_s c_l$.

Ambient Lights

- Ambient Orbs

Directional Lights

- Directional lights are defined by a single vector that points *toward* the light source.
- The incoming light is a series of parallel rays.
- Directional lights are not physical lights and do not have a real location.
- The Sun is a good approximation of a directional light because the rays that hit the Earth are essentially parallel.



Directional Lights

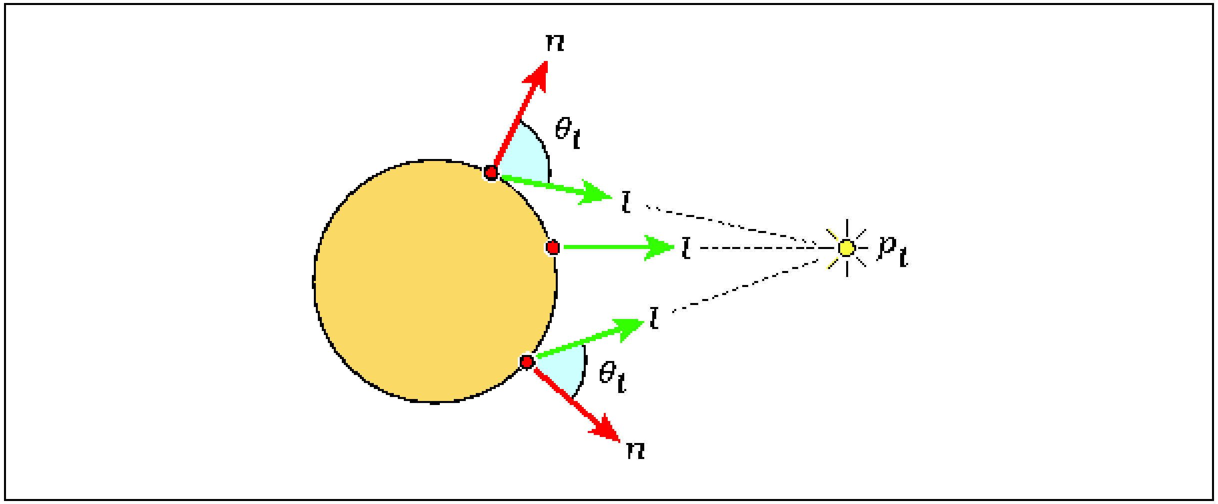
- Directional Light Balls

Directional Lights

- Directional Light Orbs

Point Lights

- Point lights are defined as a location, rather than as a vector.
- Because a point has no surface area, point lights are not physical objects.
- Point lights emit light equally in all directions.
- Light is attenuated according to the inverse square of the distance a surface is from the source.



Point Lights

- Point Balls

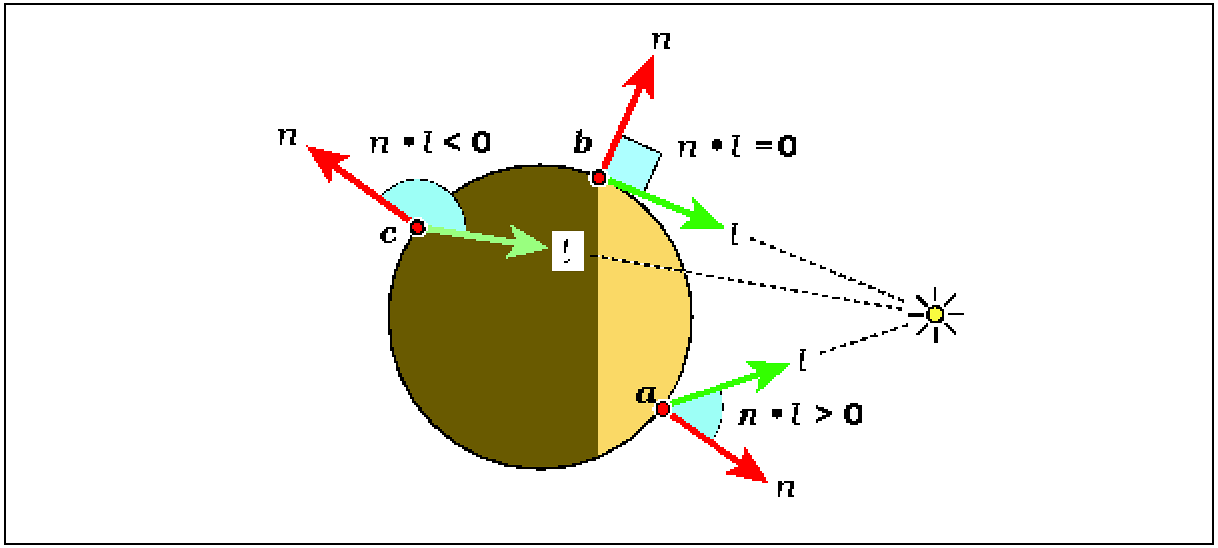
Point Lights

- Point Orbs

Problems

Q: How do we know when to shade a surface?

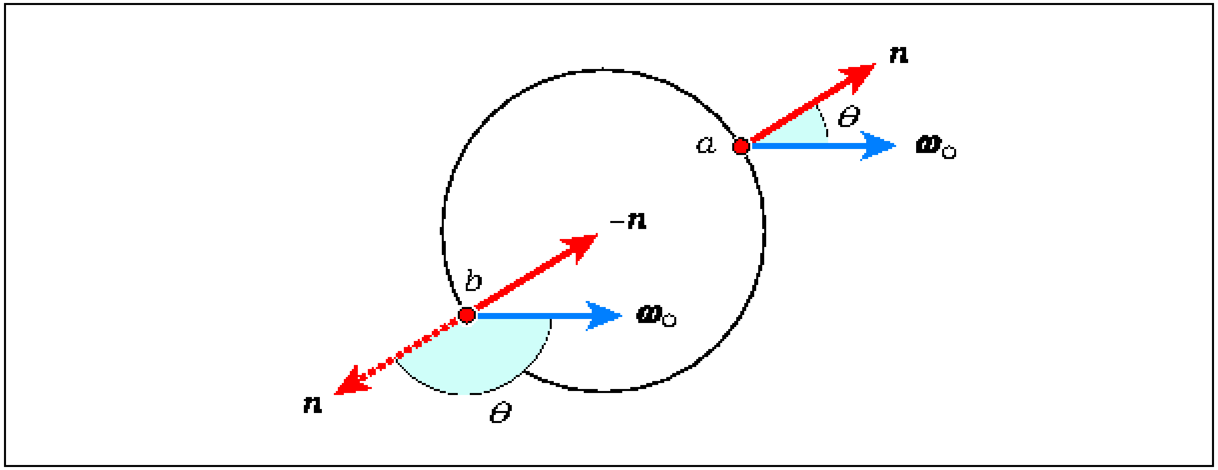
A: When n·l<0, only shade with ambient illumination.



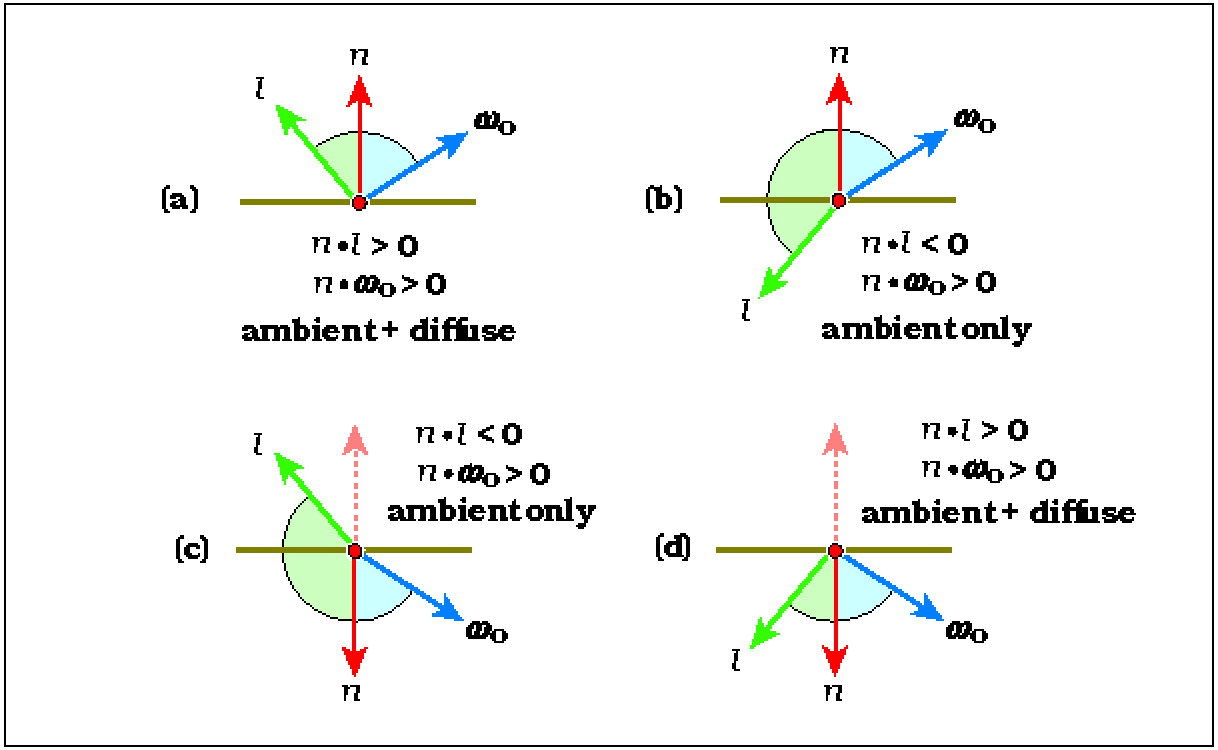
Problems

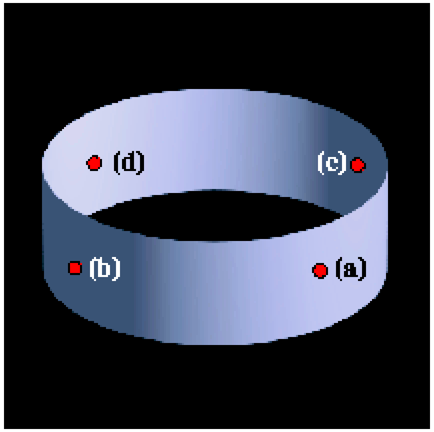
Q: How do we shade the inside surfaces?

A: Reverse the normal!



Problems


(a)
n
l
ωo
n • l > 0
n • ωo > 0
ambient + diffuse
(b)
n
ωo
l
n • l < 0
n • ωo > 0
ambient only
(c)
l
n
ωo
n • l < 0
n • ωo > 0
ambient only
(d)
l
n
ωo
n • l > 0
n • ωo > 0
ambient + diffuse


(d)
(c)
(b)
(a)

Materials

- Different surfaces are made of different materials.
- Different materials reflect light differently. Thus, they require different shading models.
- Two materials that we learned about previously:
  - Matte
  - Phong
- Other surfaces might be translucent, metallic, or reflective.
- Materials are represented with a set of BRDFs and a shade() function.

Matte Material

- The matte material models perfect diffuse reflection.
- Matte contains a Lambertian BRDF for ambient shading and another for diffuse shading.
- The shade() function is simple.

```
RGBColor
Matte::shade(ShadeRec& sr) {
	L = ambient_contribution;
	for(i = 0; i < numLights; i++) {
		calculate ndotwi
		if(ndotwi > 0)
			L += diffuse_contribution;
	}

	return L;
}
```

Matte Material

- Matte Balls, Two Lights

Matte Material

- Matte Orbs

Phong Material

- The Phong material adds specular highlights to the ambient and diffuse shading. So, Phong surfaces appear glossy.
- This material contains the ambient and diffuse Lambertian BRDFs as well as a GlossySpecular BRDF. The GlossySpecular BRDF contains the k_s and e values (remember Phong shading?)
- The shading function remains nearly unchanged.

```
…
        for(i = 0; i < numLights; i++)
            …
                L += specular_contribution
                   + diffuse_contribution;
…
```

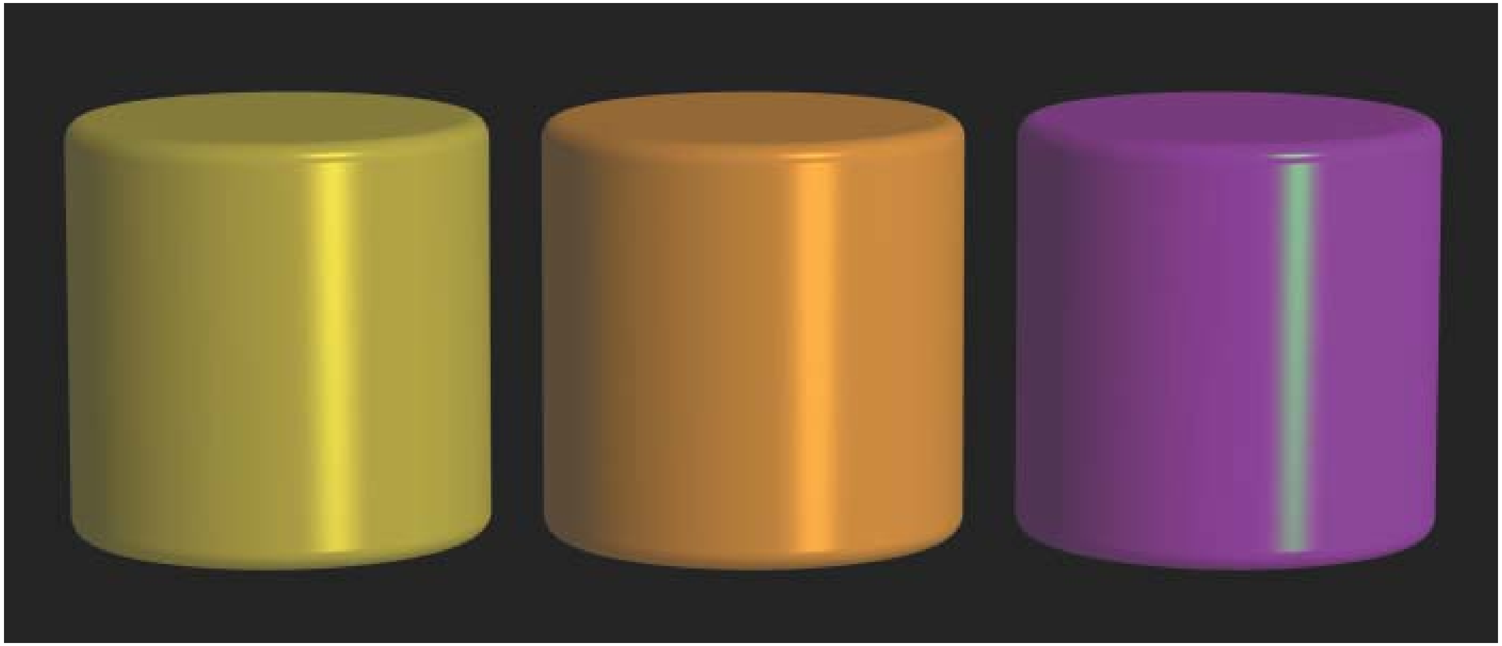
Phong Material

- Specular Balls

Phong Material

- Specular Orbs

Phong Material



QUESTIONS?