



COMP220: Graphics & Simulation
6: Materials and Lighting

Learning outcomes

- ▶ **Explain** the Blinn-Phong illumination model
- ▶ **Implement** Blinn-Phong illumination in your own programs
- ▶ **Describe** how effects such as normal mapping can be used to render realistic materials

Vector products



Dot and cross product

Dot and cross product

$$a \cdot b = |a||b| \cos \theta$$

where θ is the **angle** between a and b

Dot and cross product

$$\mathbf{a} \cdot \mathbf{b} = |\mathbf{a}| |\mathbf{b}| \cos \theta$$

where θ is the **angle** between a and b

$$\mathbf{a} \times \mathbf{b} = (|\mathbf{a}| |\mathbf{b}| \sin \theta) \mathbf{n}$$

where n is a unit vector **perpendicular** to both a and b
with direction given by the **right-hand rule**

Uses

Uses

- Both dot and cross product are **quick to calculate**

Uses

- ▶ Both dot and cross product are **quick to calculate**
- ▶ Dot product can be used to find the **angle** between vectors

Uses

- ▶ Both dot and cross product are **quick to calculate**
- ▶ Dot product can be used to find the **angle** between vectors
 - ▶ Actually the **cosine** of the angle

Uses

- ▶ Both dot and cross product are **quick to calculate**
- ▶ Dot product can be used to find the **angle** between vectors
 - ▶ Actually the **cosine** of the angle
 - ▶ If $a \cdot b = 0$ (and a, b are non-zero) then $\cos \theta = 0$, i.e. $\theta = 90^\circ$ — a and b are **perpendicular**

Uses

- ▶ Both dot and cross product are **quick to calculate**
- ▶ Dot product can be used to find the **angle** between vectors
 - ▶ Actually the **cosine** of the angle
 - ▶ If $a \cdot b = 0$ (and a, b are non-zero) then $\cos \theta = 0$, i.e. $\theta = 90^\circ$ — a and b are **perpendicular**
 - ▶ If $a \cdot b = 1$ and a, b are unit vectors then $\cos \theta = 1$, i.e. $\theta = 0^\circ$ — a and b are **parallel**

Uses

- ▶ Both dot and cross product are **quick to calculate**
- ▶ Dot product can be used to find the **angle** between vectors
 - ▶ Actually the **cosine** of the angle
 - ▶ If $a \cdot b = 0$ (and a, b are non-zero) then $\cos \theta = 0$, i.e. $\theta = 90^\circ$ — a and b are **perpendicular**
 - ▶ If $a \cdot b = 1$ and a, b are unit vectors then $\cos \theta = 1$, i.e. $\theta = 0^\circ$ — a and b are **parallel**
- ▶ Cross product can be used to find a vector **perpendicular** to two others

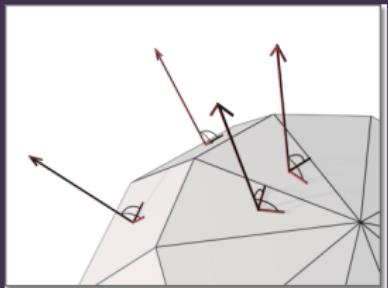
Uses

- ▶ Both dot and cross product are **quick to calculate**
- ▶ Dot product can be used to find the **angle** between vectors
 - ▶ Actually the **cosine** of the angle
 - ▶ If $a \cdot b = 0$ (and a, b are non-zero) then $\cos \theta = 0$, i.e. $\theta = 90^\circ$ — a and b are **perpendicular**
 - ▶ If $a \cdot b = 1$ and a, b are unit vectors then $\cos \theta = 1$, i.e. $\theta = 0^\circ$ — a and b are **parallel**
- ▶ Cross product can be used to find a vector **perpendicular** to two others
- ▶ $\text{vector} \cdot \text{vector} = \text{number}$; $\text{vector} \times \text{vector} = \text{vector}$

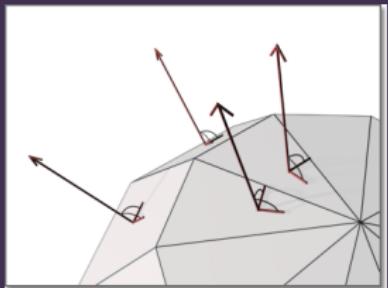
Surface normals

Surface normals

- ▶ The **normal** to a surface is a **unit vector** that is **perpendicular** to the surface

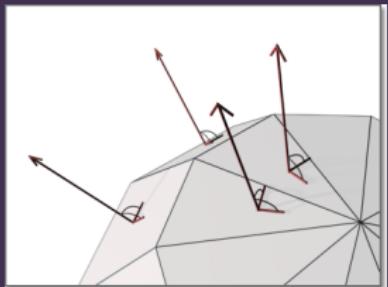


Surface normals

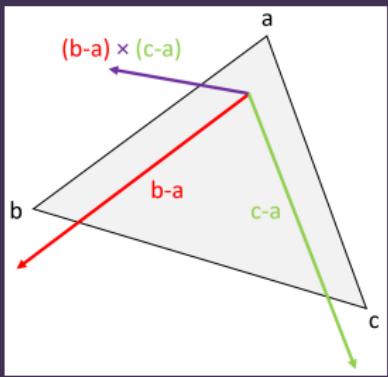


- ▶ The **normal** to a surface is a **unit vector** that is **perpendicular** to the surface
- ▶ If we have two non-parallel vectors that are **tangent** to the surface, we can use the **cross product** to find the normal

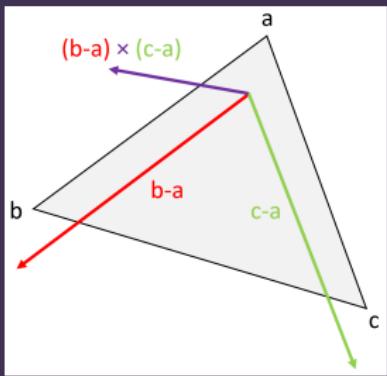
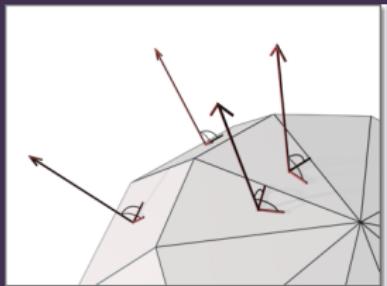
Surface normals



- ▶ The **normal** to a surface is a **unit vector** that is **perpendicular** to the surface
- ▶ If we have two non-parallel vectors that are **tangent** to the surface, we can use the **cross product** to find the normal
- ▶ For a triangle with vertices a, b, c , two such vectors are $b - a$ and $c - a$



Surface normals



- ▶ The **normal** to a surface is a **unit vector** that is **perpendicular** to the surface
- ▶ If we have two non-parallel vectors that are **tangent** to the surface, we can use the **cross product** to find the normal
- ▶ For a triangle with vertices a, b, c , two such vectors are $b - a$ and $c - a$
- ▶ So the normal is
$$\frac{n}{|n|} \quad \text{where} \quad n = (b - a) \times (c - a)$$

The Blinn-Phong illumination model



The Blinn-Phong illumination model

The Blinn-Phong illumination model

Jim Blinn, "Models of light reflection for computer synthesized pictures". *ACM SIGGRAPH Computer Graphics*, 11(2):192–198, 1977.

The Blinn-Phong illumination model

Jim Blinn, "Models of light reflection for computer synthesized pictures". *ACM SIGGRAPH Computer Graphics*, 11(2):192–198, 1977.

The Blinn-Phong model builds on the Phong shading model

The Blinn-Phong illumination model

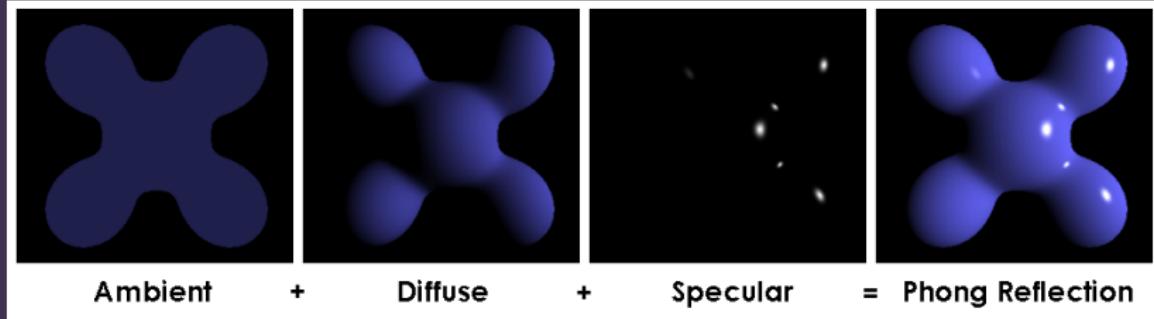
Jim Blinn, "Models of light reflection for computer synthesized pictures". *ACM SIGGRAPH Computer Graphics*, 11(2):192–198, 1977.

The Blinn-Phong model builds on the Phong shading model. It breaks lighting down into three parts: **ambient**, **diffuse**, and **specular**.

The Blinn-Phong illumination model

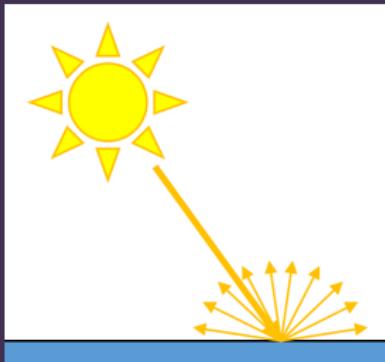
Jim Blinn, "Models of light reflection for computer synthesized pictures". *ACM SIGGRAPH Computer Graphics*, 11(2):192–198, 1977.

The Blinn-Phong model builds on the Phong shading model. It breaks lighting down into three parts: **ambient**, **diffuse**, and **specular**.



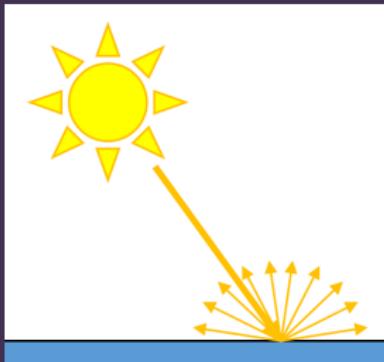
Diffuse lighting

Diffuse lighting



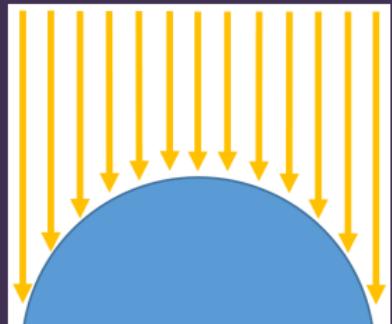
When light hits a “rough surface, it is
scattered equally in all directions

Diffuse lighting

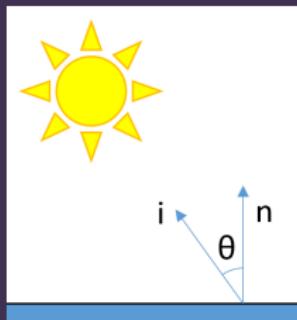


When light hits a “rough surface, it is
scattered equally in all directions

The amount of light hitting the surface
depends on the **angle** between the
surface and the light source

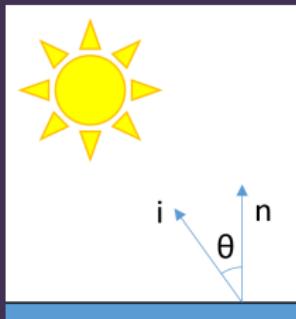


Diffuse lighting formula



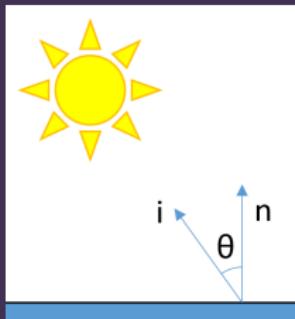
Diffuse lighting formula

- ▶ Light intensity is proportional to the **cosine** of the angle between the **light direction** and the **surface normal**



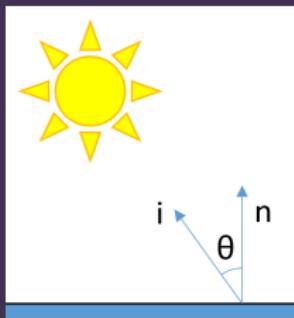
Diffuse lighting formula

- ▶ Light intensity is proportional to the **cosine** of the angle between the **light direction** and the **surface normal**
- ▶ Let n be the normal, and i be a unit vector pointing towards the light source

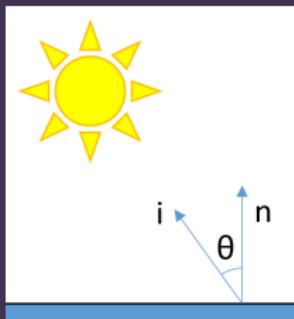


Diffuse lighting formula

- ▶ Light intensity is proportional to the **cosine** of the angle between the **light direction** and the **surface normal**
- ▶ Let n be the normal, and i be a unit vector pointing towards the light source
- ▶ Light intensity is proportional to $\cos \theta = n \cdot i$



Diffuse lighting formula



- ▶ Light intensity is proportional to the **cosine** of the angle between the **light direction** and the **surface normal**
- ▶ Let n be the normal, and i be a unit vector pointing towards the light source
- ▶ Light intensity is proportional to $\cos \theta = n \cdot i$
- ▶ If the surface is **pointing away** from the light source, we get $\theta > \frac{\pi}{2}$ so $\cos \theta < 0$ — in this case we **clamp** the answer to 0

Light direction and intensity

Light direction and intensity

- ▶ For a **distant** light source (e.g. the sun), direction and intensity are **constant**

Light direction and intensity

- ▶ For a **distant** light source (e.g. the sun), direction and intensity are **constant**
- ▶ For a **point** light source (e.g. a lightbulb):

Light direction and intensity

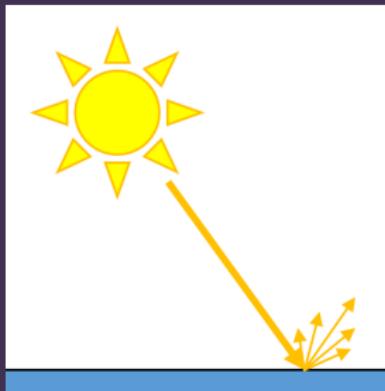
- ▶ For a **distant** light source (e.g. the sun), direction and intensity are **constant**
- ▶ For a **point** light source (e.g. a lightbulb):
 - ▶ Direction is calculated by subtracting the light position from the fragment position

Light direction and intensity

- ▶ For a **distant** light source (e.g. the sun), direction and intensity are **constant**
- ▶ For a **point** light source (e.g. a lightbulb):
 - ▶ Direction is calculated by subtracting the light position from the fragment position
 - ▶ Intensity obeys an **inverse square law**: if the distance between the fragment and the light source is d , then the light intensity is $\frac{1}{d^2}$

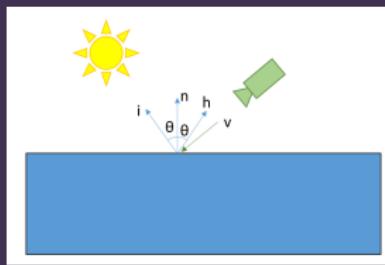
Specular lighting

Specular lighting

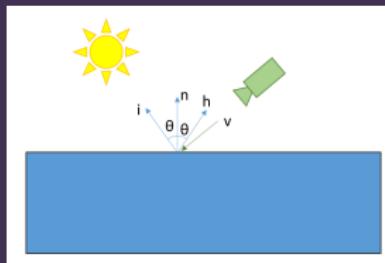


When light hits a “smooth” surface, it is **reflected** across a narrow range of angles

Specular lighting formula

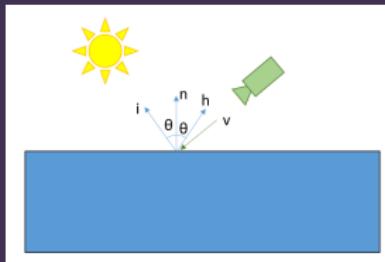


Specular lighting formula



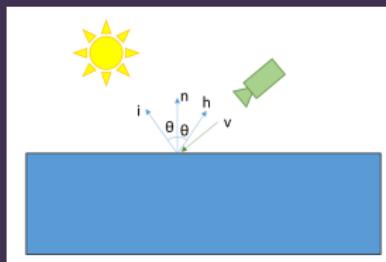
- ▶ Let h be the halfway vector, calculated by $i + v$

Specular lighting formula



- ▶ Let h be the halfway vector, calculated by $i + v$
- ▶ h can be thought of as the surface normal which leads to the maximum reflection in v

Specular lighting formula



- ▶ Let h be the halfway vector, calculated by $i + v$
- ▶ h can be thought of as the surface normal which leads to the maximum reflection in v
- ▶ Specular light intensity is proportional to

$$\text{clamp}(n \cdot h)^s$$

where s is a “shininess” parameter, and $\text{clamp}(x)$ clamps its argument between 0 and 1

Ambient lighting

Ambient lighting

- ▶ Currently, surfaces pointing away from the light are completely **black** (light intensity = 0)

Ambient lighting

- ▶ Currently, surfaces pointing away from the light are completely **black** (light intensity = 0)
- ▶ In the real world, light scattered from one surface illuminates others

Ambient lighting

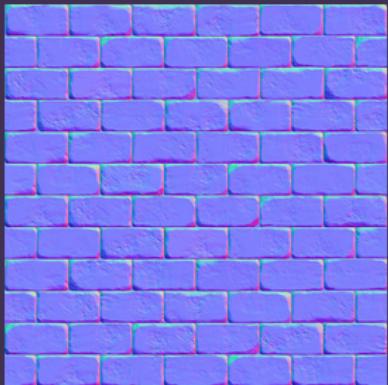
- ▶ Currently, surfaces pointing away from the light are completely **black** (light intensity = 0)
- ▶ In the real world, light scattered from one surface illuminates others
- ▶ In the Phong model, we cheat and add a little **ambient** intensity to the lighting

Ambient lighting

- ▶ Currently, surfaces pointing away from the light are completely **black** (light intensity = 0)
- ▶ In the real world, light scattered from one surface illuminates others
- ▶ In the Phong model, we cheat and add a little **ambient** intensity to the lighting
- ▶ Another option would be to add more light sources...

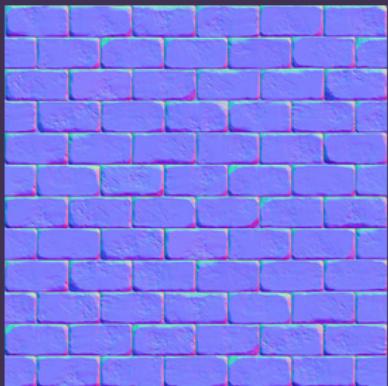
Normal mapping

Normal mapping



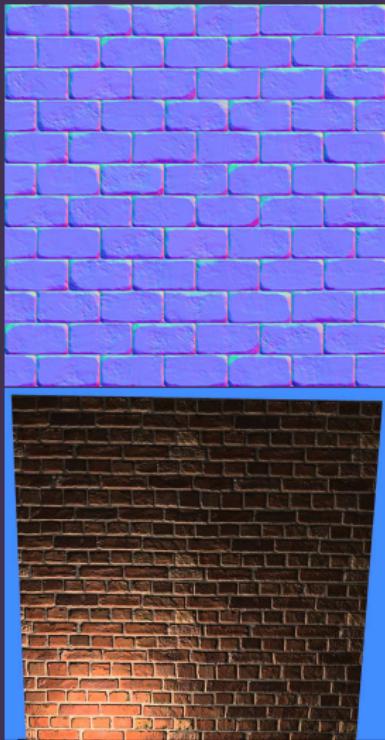
- ▶ A **normal map** is a texture which is used to slightly alter the normal across a surface

Normal mapping



- ▶ A **normal map** is a texture which is used to slightly alter the normal across a surface
 - ▶ Each pixel in the normal map represents a 3D vector, with xyz mapped to RGB

Normal mapping



- ▶ A **normal map** is a texture which is used to slightly alter the normal across a surface
 - ▶ Each pixel in the normal map represents a 3D vector, with xyz mapped to RGB
- ▶ Can be used to add detail to flat, low-poly surfaces

Normal mapping



- ▶ A **normal map** is a texture which is used to slightly alter the normal across a surface
 - ▶ Each pixel in the normal map represents a 3D vector, with xyz mapped to RGB
- ▶ Can be used to add detail to flat, low-poly surfaces
- ▶ Can use textures to change other lighting parameters across a surface, e.g. **specular mapping**