Computer Graphics

Lecture 10B
Specular Reflection
Goals

• Know what a **specular surface** looks like.

• Know how to implement the **Phong reflection model**.

• Know how to implement the **Blinn-Phong reflection model**.

• Know what to do with multiple lights.
Not all surfaces are Lambertian
Not all surfaces are Lambertian

Key observation: the specular highlight is not pointing straight towards the light.
Specular Reflection

Specular surfaces appear brighter near "mirror" configuration.

**Phong reflection:** Specular reflection is a function of the angle between mirror direction $r$ and view direction $v$. 
Phong Reflection

Not physically accurate, but perceptually "okay", and intuitive:
A little less physically inaccurate, still perceptually "okay", and less intuitive:
Specular Reflection

• Blinn-Phong: specular reflection is a function of angle between (half-way vector between view and light) and (the normal).

• \( h = \text{bisector}(v, l) \)

• Reflected light proportional to

\[
k_s \max(0, n \cdot h)^p
\]

specular coefficient: determines strength of specularity term

specular exponent: determines sharpness
Computing $\vec{h}$
Computing $\vec{h}$

\[
\text{bisector}(\vec{v}, \vec{\ell}) =
\]
Effect of $\rho$

- $\cos \alpha$
- $\cos^2 \alpha$
- $\cos^8 \alpha$
- $\cos^{64} \alpha$
Putting it all Together: Blinn-Phong Reflection Model

Usually surfaces have both diffuse and specular components, so we'll combine the two:

\[ L = L_d + L_s \]
\[ = k_d I \max(0, \hat{n} \cdot \hat{l}) + k_s I \max(0, \hat{n} \cdot \hat{h})^p \]
Putting it all Together: Blinn-Phong Reflection Model

Usually surfaces have both diffuse and specular components, so we'll combine the two:

\[ L = L_d + L_s = k_d I \max(0, \vec{n} \cdot \vec{l}) + k_s I \max(0, \vec{n} \cdot \vec{h})^p \]
Putting it all Together: Blinn-Phong Reflection Model

Usually surfaces have both diffuse and specular components, so we'll combine the two:

\[ L = L_d + L_s \]

\[ = k_d I \max(0, \vec{n} \cdot \vec{l}) + k_s I \max(0, \vec{n} \cdot \vec{h})^p \]
Putting it all Together: Blinn-Phong Reflection Model

Usually surfaces have both diffuse and specular components, so we'll combine the two:

\[ L = L_d + L_s = k_d I \max(0, \vec{n} \cdot \vec{l}) + k_s I \max(0, \vec{n} \cdot \vec{h})^p \]

- **Light reflected**
- **Diffuse reflection**
- **Specular reflection**
- **Diffuse coefficient** (surface brightness and color)
- **Light intensity**
- **Light direction**
- **Normal**
- **Specular exponent** (sharpness of specularity)
- **Half-vector between \( \vec{l} \) and \( \vec{v} \)
- **Specular coefficient** (strength [and color] of specularity)
Putting it all Together: Blinn-Phong Reflection Model

Usually surfaces have both diffuse and specular components, so we'll combine the two:

\[ L = L_d + L_s = k_d I \max(0, \vec{n} \cdot \vec{l}) + k_s I \max(0, \vec{n} \cdot \vec{h})^p \]

In code: `function shade_light(light, hitrec, ...)`
What if there are multiple lights?

Light is additive - add them together:

\[
L = \sum_{i=1}^{\text{\# lights}} k_d I \max(0, \vec{n} \cdot \vec{l}_i) + k_s I \max(0, \vec{n} \cdot \vec{h}_i)^p
\]

In code:
What if there are multiple lights?

Light is additive - add them together:

\[
L = \sum_{i=1}^{\# \text{ lights}} k_d I \max(0, \vec{n} \cdot \vec{l}_i) + k_s I \max(0, \vec{n} \cdot \vec{h}_i)^p
\]

In code:

```python
function determine_color(hitrec, ray, scene, ...):
```
What if there are multiple lights?

Light is additive - add them together:

\[
L = \sum_{i=1}^{\text{# lights}} k_d I \max(0, \vec{n} \cdot \vec{l}_i) + k_s I \max(0, \vec{n} \cdot \vec{h}_i)^p
\]

In code:

```
function determine_color(hitrec, ray, scene, ...):
    color = black
```
What if there are multiple lights?

Light is additive - add them together:

\[
L = \sum_{i=1}^{\text{\# lights}} k_d I \max(0, \vec{n} \cdot \vec{l}_i) + k_s I \max(0, \vec{n} \cdot \vec{h}_i)^p
\]

In code:

```python
function determine_color(hitrec, ray, scene, ...):
    color = black
    for light in scene.lights:
```
What if there are multiple lights?

Light is additive - add them together:

\[
L = \sum_{i=1}^{\text{# lights}} k_d I \max(0, \vec{n} \cdot \vec{l}_i) + k_s I \max(0, \vec{n} \cdot \vec{h}_i)^p
\]

In code:

```python
function determine_color(hitrec, ray, scene, ...):
    color = black
    for light in scene.lights:
        color += shade_light(light, hitrec, ...)
```
A disclaimer about point lights
A disclaimer about point lights

intensity here: $I$
A disclaimer about point lights

Intensity here: $I/r^2$

Intensity here: $I$
A disclaimer about point lights

In A2, we're ignoring the factor of $1/r^2$, for ease of modeling.
Our images so far:

Point light

smaller $k_s$

larger $k_s$

smaller $p$ larger $p$ red $k_s$

blue $k_d$

gray $k_d$
Partially-Mirrored Surfaces

Notice the floor is gray but also mirror-reflective.

Materials can store a mirror coefficient: fraction of light that is reflected in a mirror-like fashion.
Partially-Mirrored Surfaces

Notice the floor is gray but also mirror-reflective.

Materials can store a mirror coefficient: fraction of light that is reflected in a mirror-like fashion

\[ L = k_m L_r + (1 - k_m)(L_d + L_s) \]
Partially-Mirrored Surfaces

Notice the floor is gray but also mirror-reflective.

Materials can store a mirror coefficient: fraction of light that is reflected in a mirror-like fashion

\[ L = k_m L_r + (1 - k_m)(L_d + L_s) \]

- mirror coefficient
- mirror-reflected light
- "local" color (Blinn-Phong)