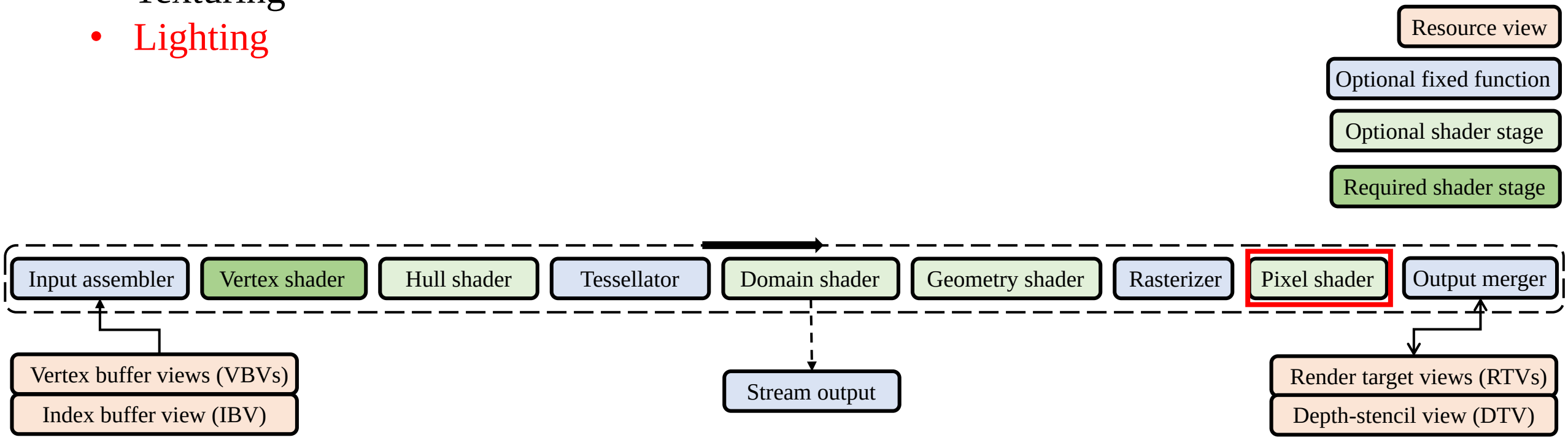
8. Lighting

Prof. HyeongYeop Kang
siamiz@khu.ac.kr
YouTube: HKang IIIXR LAB
IIIXR LAB

Pixel Shader

Rendering pipeline (revisited)

- The per-pixel attributes produced by the rasterizer usually include a normal vector and texture coordinates. Using the attributes, the pixel shader determines the color of each pixel.
  - Texturing
  - Lighting



Phong Lighting Model

Lighting (Illumination)

- Lighting (also known as shading) refers to a technique that handles the interaction between lights and objects.
- The most popular lighting method is based on the *Phong model*. It is widely adopted in commercial games and lays foundations of various advanced lighting techniques.

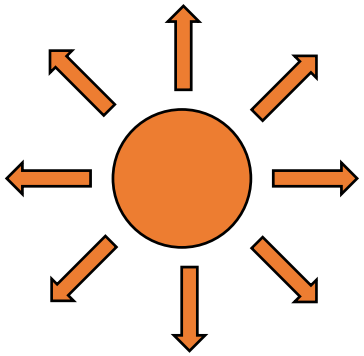
Phone model

- The *phong model* is composed of four terms:
  - Diffuse
  - Specular
  - Ambient
  - Emissive

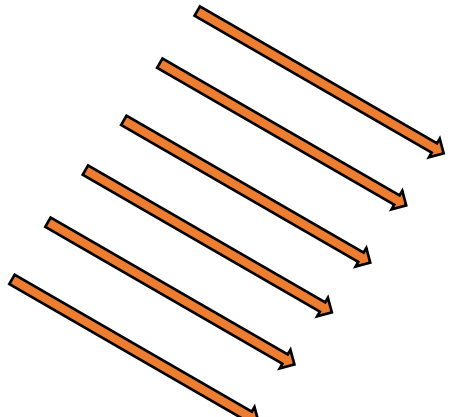
Light Sources

First, let's look at the types of light sources. Different types of light sources illuminate the scene in different ways.

- Point light:
  - This emits light from a single point in all directions.
  - The intensity of the light decreasing with distance.



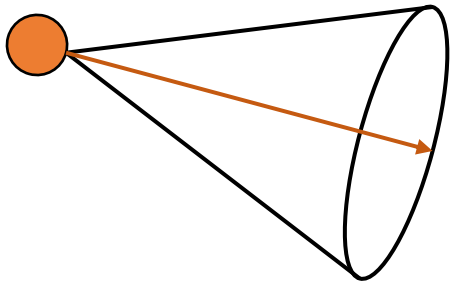
- Directional light (the light which is extremely far away from the scene):
  - This uniformly lights a scene from one direction.
  - The intensity of light does not change with distance.



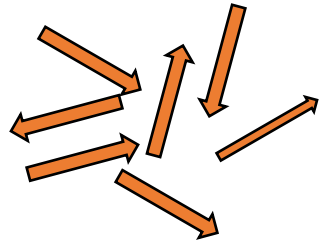
Light Sources

First, let's look at the types of light sources. Different types of light sources illuminate the scene in different ways.

- Spotlight:
  - This makes a cone shaped light.
  - The intensity increases as closer to the center of the light cone and spotlight source.



- Ambient:
  - This illuminates objects even there is no light source. (constant light)
  - The intensity is completely uniform.



constant light C

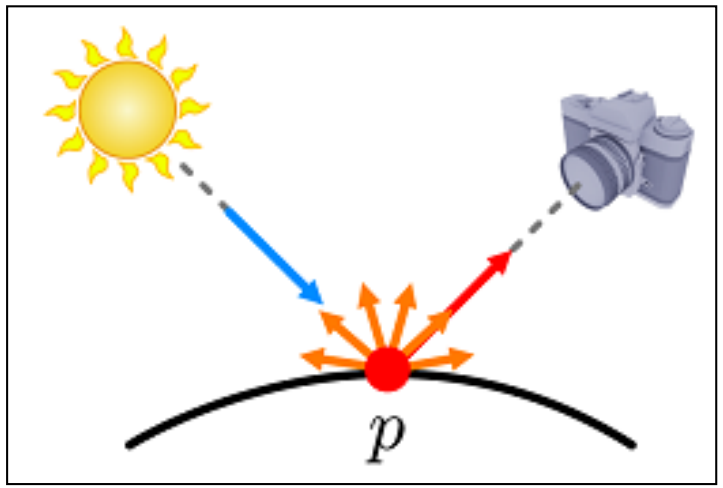
Diffuse Term

What is diffuse light?

- Diffuse light represents the scattered light when direct light hits the surface.
- The light is uniformly distributed.

Diffuse Term in Phong Lighting Model

- The diffuse term is based on Lambert's law, which states that reflections from ideally diffuse surfaces are scattered with equal intensity in all directions.
- Therefore, the amount of perceived reflection is independent of the view direction, and is just proportional to the amount of incoming light.



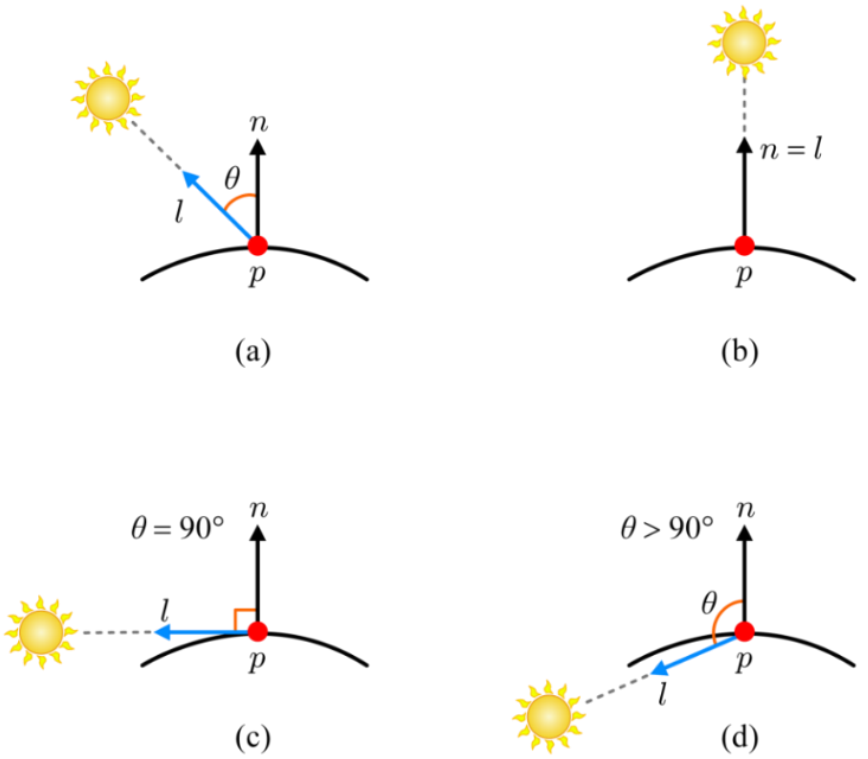
Diffuse Term

Diffuse computation

- Let's consider the light vector (l) and surface normal (n).

- The diffuse term can be determined by $max(n \cdot l, 0)s_d \otimes m_d$, where s_d is the RGB color of the light source, m_d is the *diffuse reflectance* of the object material, and $\otimes$ is component-wise multiplication.

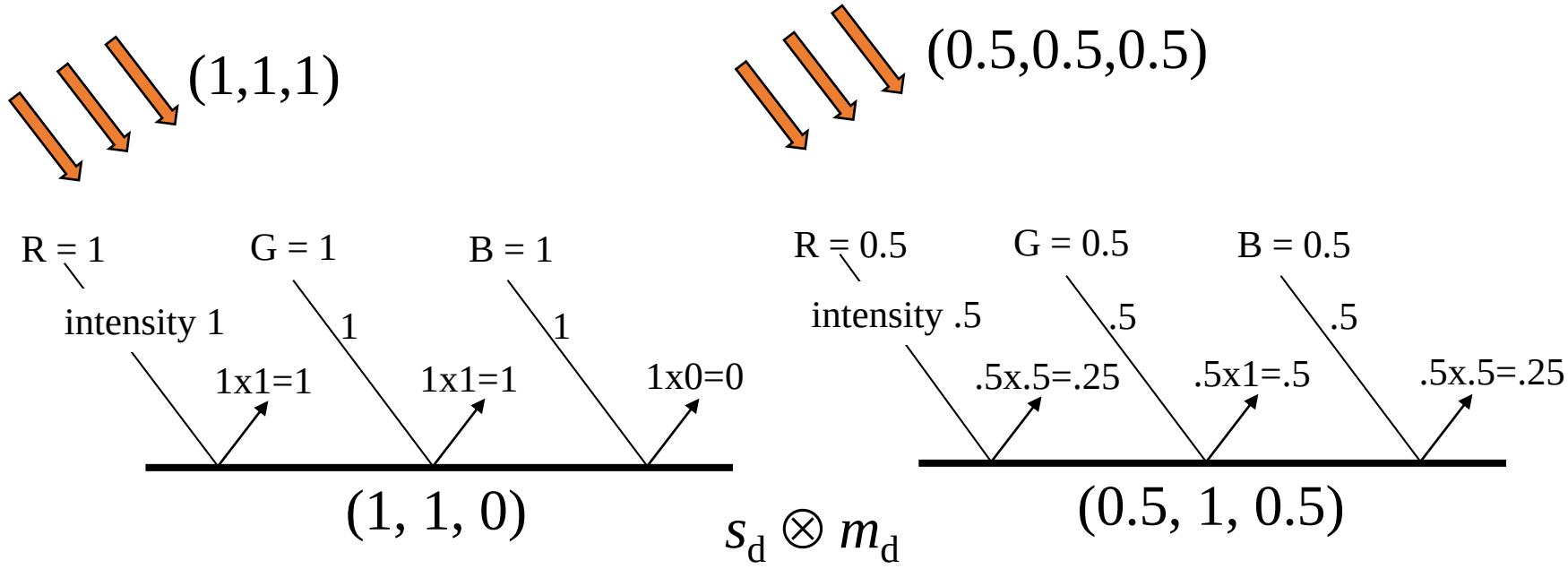
- The term $max(n \cdot l, 0)$ considers the angle of the light to determine the lightness.



Diffuse Term

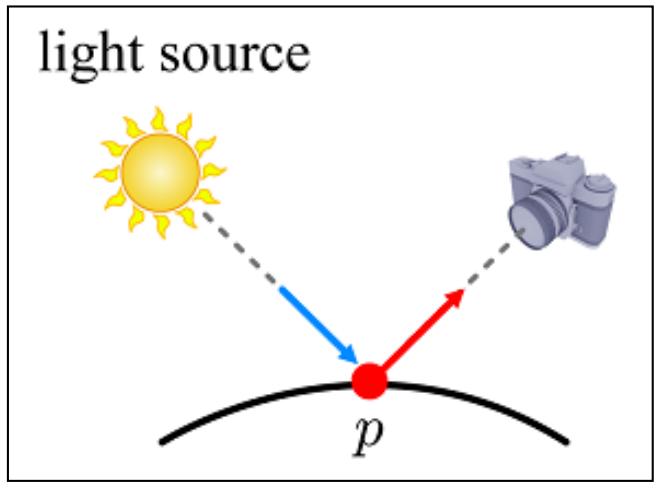
Diffuse computation

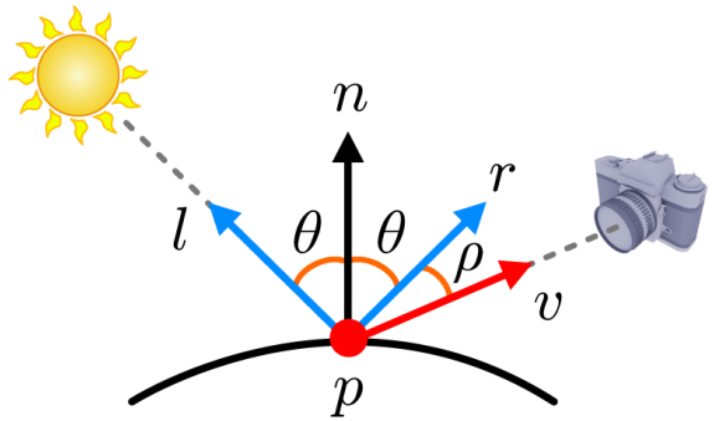
- The material parameter, m_d, changes the color of the light considering the material properties.
- Suppose a white light (1, 1, 1). If an object lit by the light appears yellow, it means that the object reflects R and G and absorbs B. We can easily implement this kind of filtering through material parameter, i.e., if it is (1, 1, 0), then (1, 1, 1)⊗(1, 1, 0)=(1, 1, 0). See right figure.

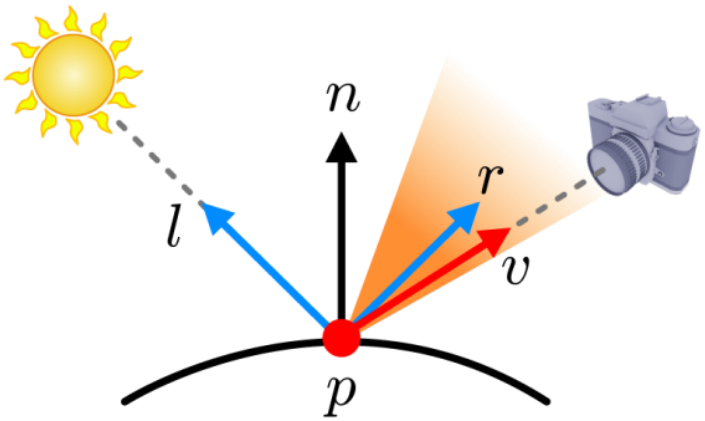


Specular Term

The specular term is used to make a surface look shiny via highlights, and it requires view vector (v) and reflection vector (r) in addition to light vector (l).







Computing the reflection vector

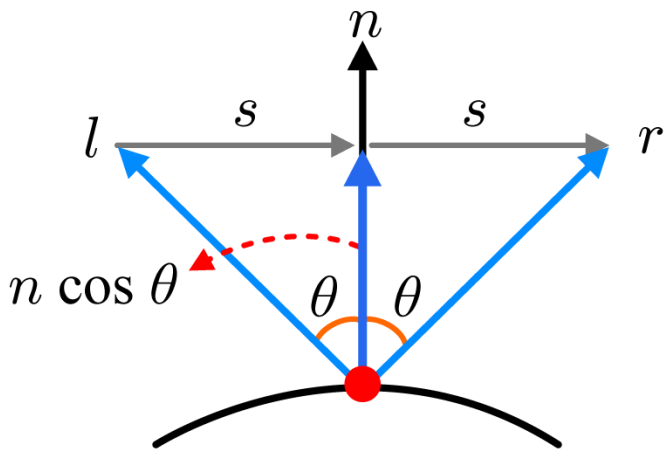
magnitude = $(l \cdot \cos\theta)$

direction = n

$\rightarrow n \cdot (l \cdot \cos\theta)$

$\rightarrow n \cos\theta$

note: n and l are normalized vector.



$s = n\cos\theta - l$

$s = r - n\cos\theta$

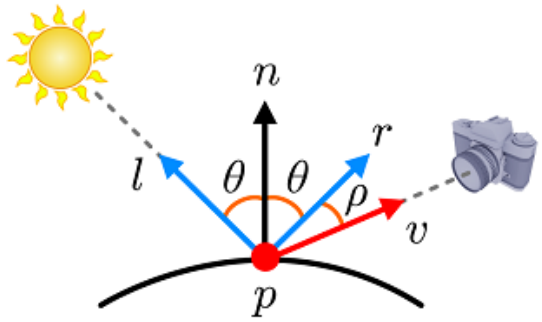
$r = 2n\cos\theta - l$

$= 2n\,(n \cdot l) - l$

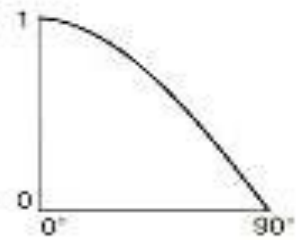
Specular Term

Whereas the diffuse term is view-independent, the specular term is highly view-dependent.

- For a perfectly shiny surface, the highlight at p is visible only when ρ equals 0.
- For a surface that is not perfectly shiny, the maximum highlight occurs when ρ equals 0, but falls off as ρ increases.
- The rapid fall-off of highlights is often approximated by $(r \cdot v)^{sh}$, where sh denotes shininess.



$$(r \cdot v)^{sh} = (r \cdot v \cdot cos\rho)^{sh}$$



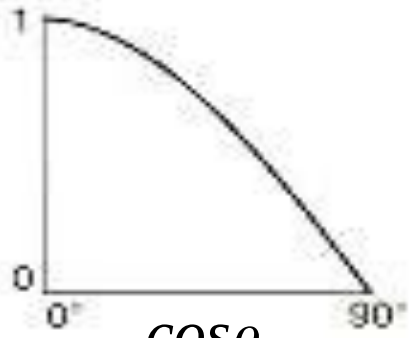
$sh = 1$

Specular Term

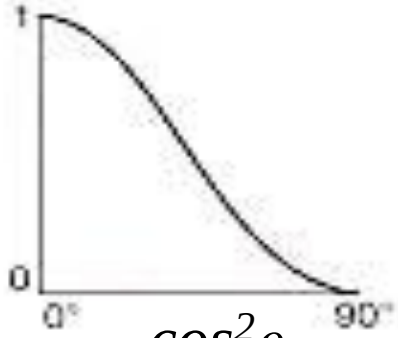
Whereas the diffuse term is view-independent, the specular term is highly view-dependent.

- As *sh* increases, the fall-off becomes steeper.

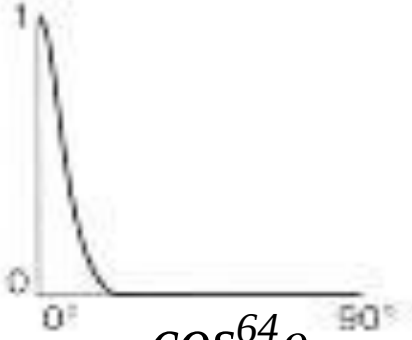
$$(r \cdot v)^{sh} = (r \cdot v \cdot cos\rho)^{sh}$$



$cos\rho$



$cos^2\rho$



$cos^{64}\rho$

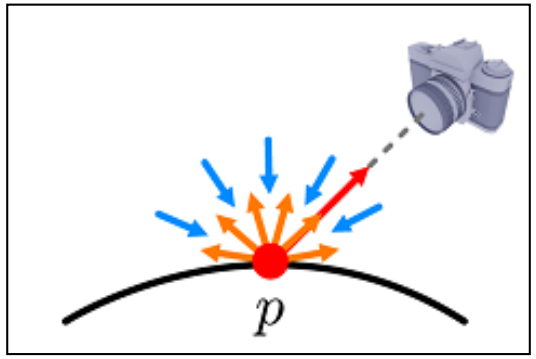
In the actual implementation, the specular term is computed by $(max(r \cdot v,0))^{sh} s_s \otimes m_s$.

- Unlike m_d, m_s is usually a gray-scale value rather than an RGB color.
- m_s enables the highlight on the surface to end up being the color of the light source.

Ambient Term

What is ambient light?

- Ambient light refers to a general level of illumination that accounts for indirect lighting.
- The ambient light describes the light reflected from the various objects in the scene.
- As the ambient light has bounced around so much in the scene, it arrives at a surface point from all directions, and reflections from the surface point are also scattered with equal intensity in all directions.



$s_a \otimes m_a$

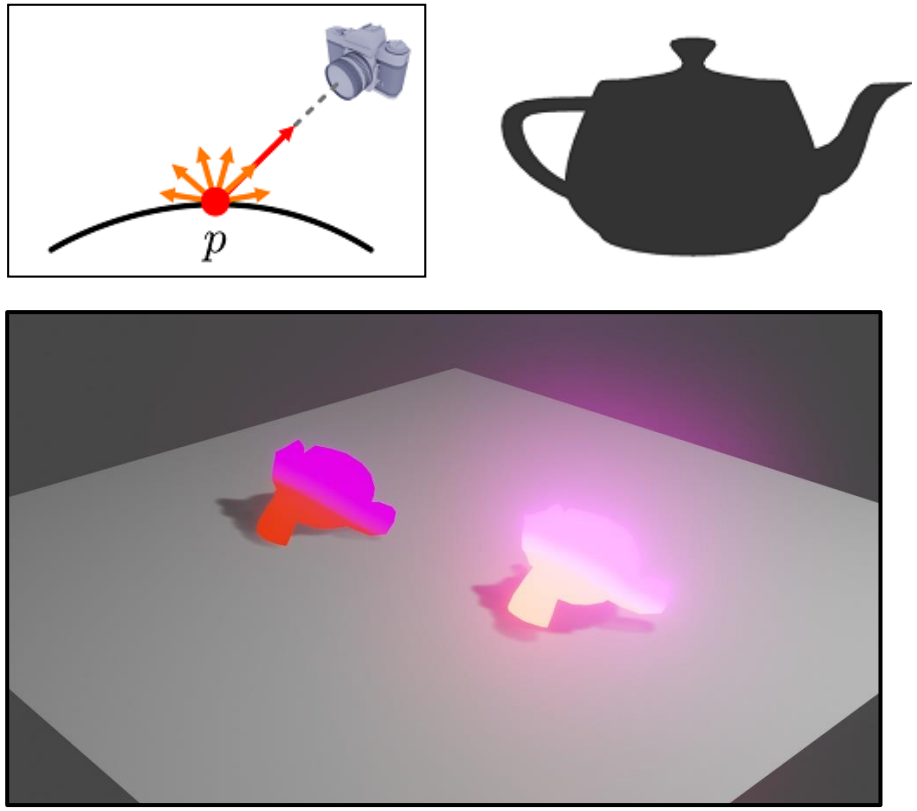
No Ambient

Ambient

Emissive Term

What is emissive term?

- The last term of the Phong model is the emissive term that describes the amount of light emitted by a surface itself.

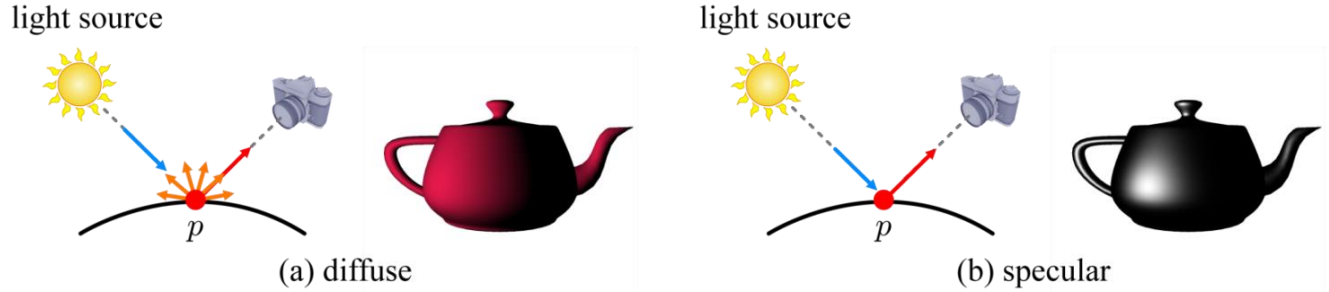


rendering without emissive (left) and with emissive (right)

Phong Lighting Model

The Phong model sums the four terms

$$max(n \cdot l, 0) s_d \otimes m_d + (max(r \cdot v, 0))^{sh} s_s \otimes m_s + s_a \otimes m_a + m_e$$



(a) diffuse (b) specular

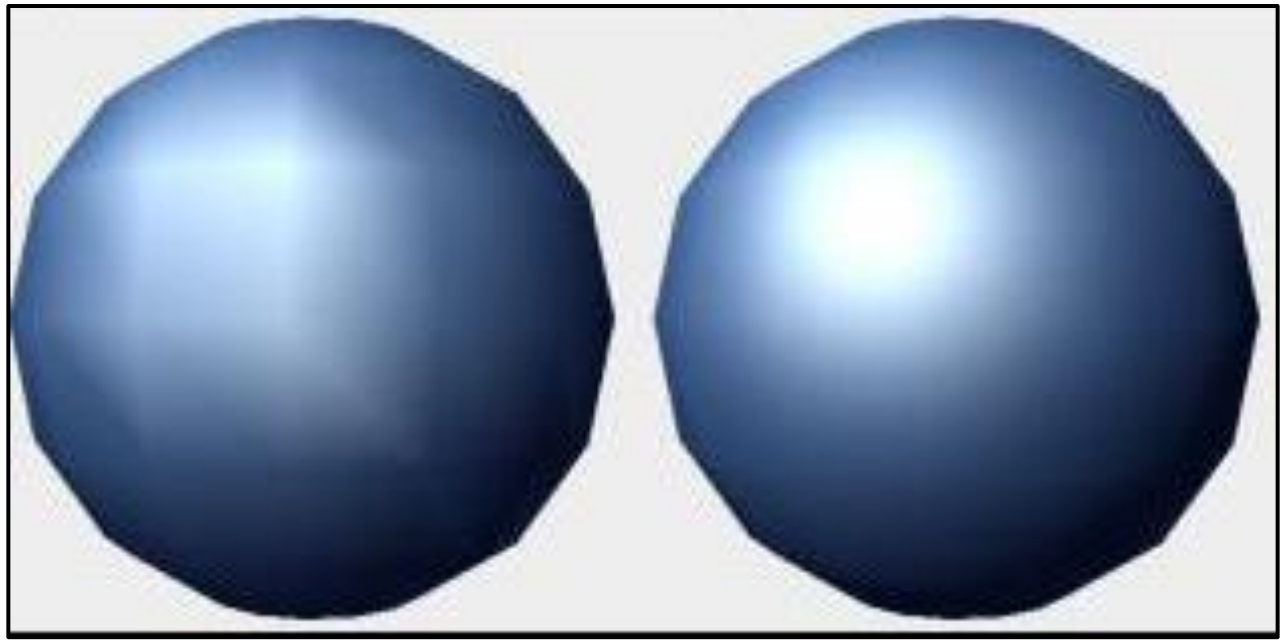
(c) ambient (d) emissive

(e) sum

Per-pixel Lighting

Per-vertex lighting vs. per-pixel lighting

- Per-vertex lighting shows a low quality shading result since it simply interpolates the color of vertices.

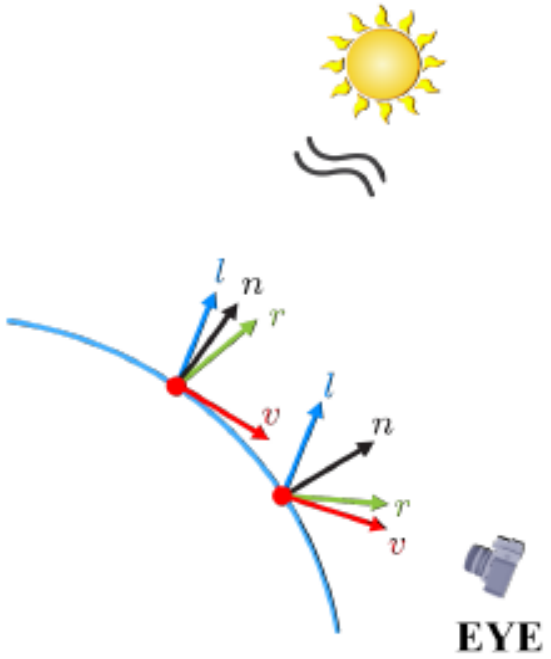


per-vertex lighting vs. per-pixel lighting

Per-pixel Lighting

Consider two points on an object's surface, each of which makes up a distinct pixel. The pixel shader computes lighting using *l*, *n*, *r*, and *v*.

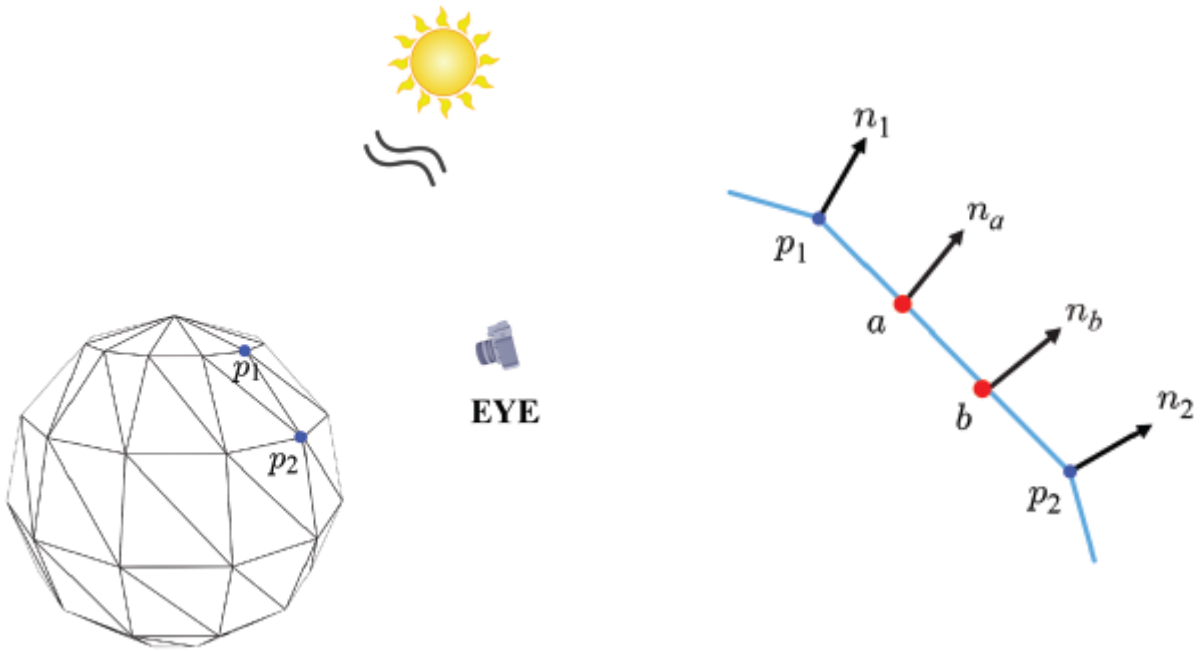
- Since we assume a directional light, *l* is constant for all surface points. It is provided for the pixel shader as a constant vector (through the constant buffer).
- In contrast, *n*, *r*, and *v* vary across the object's surface. A distinct pair of *n* and *v* is given to each execution of the pixel shader, which then computes *r*: $r = 2n(n \cdot l)-l$.



Per-pixel Lighting

Normal for each pixel

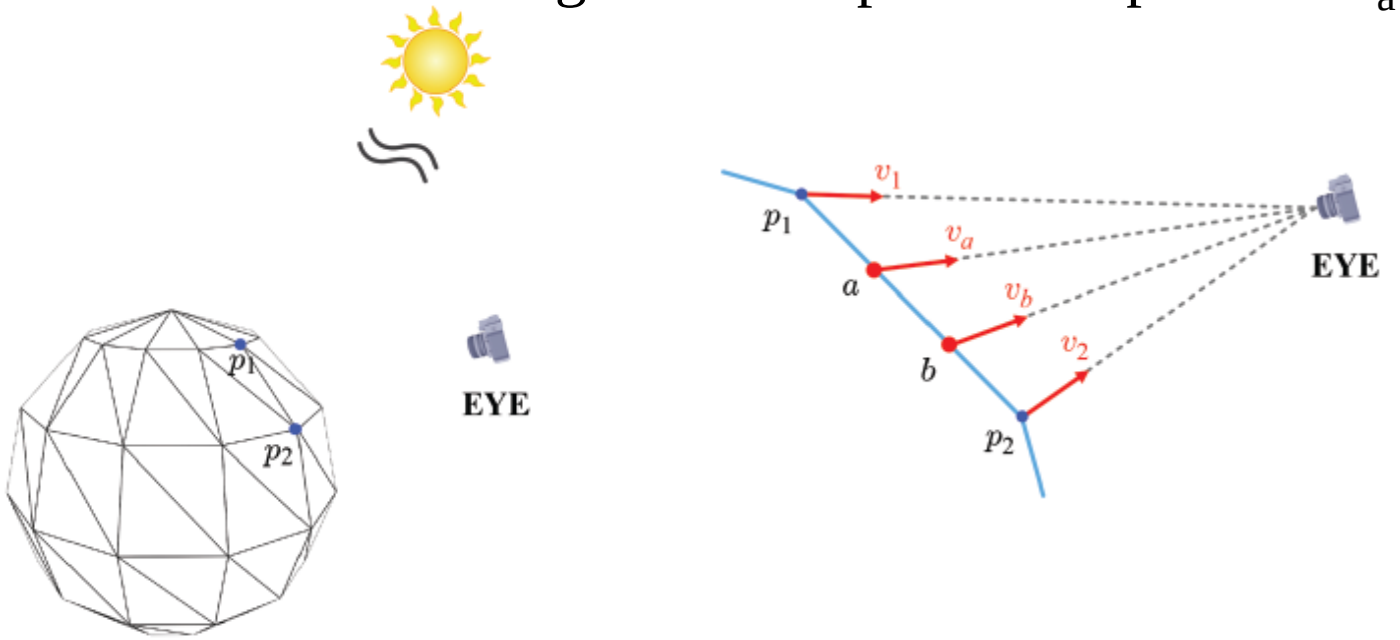
- Because l is a world-space vector, n should also be defined in the world space.
- For this, the vertex shader world-transforms the object-space normal of each vertex and passes the world-space normal to the rasterization stage.
- In the rasterization stage, vertex normals are interpolated to provide n for each pixel.
- To a and b, each of which is assumed to make up a distinct pixel, the rasterizer assigns normals, n_a and n_b, by interpolating n_1 and n_2.



Per-pixel Lighting

View vector for each pixel

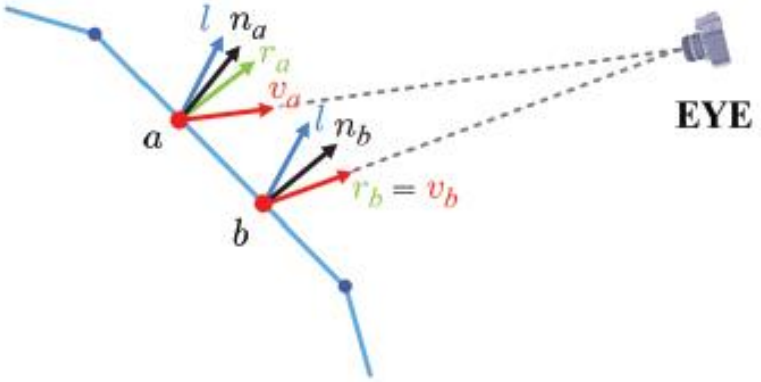
- For the pixels, *a* and *b*, we also need world-space view vectors denoted by v_a and v_b, respectively.
- For this, the vertex shader transforms the object-space position of each vertex to the world space and connects it to the camera position, EYE, defined in the world space.
- The per-vertex world-space view vectors are v_1 and v_2.
- They are passed to the rasterization stage and interpolated to produce v_a and v_b.



Per-pixel Lighting

Pixel shader determines the pixel color

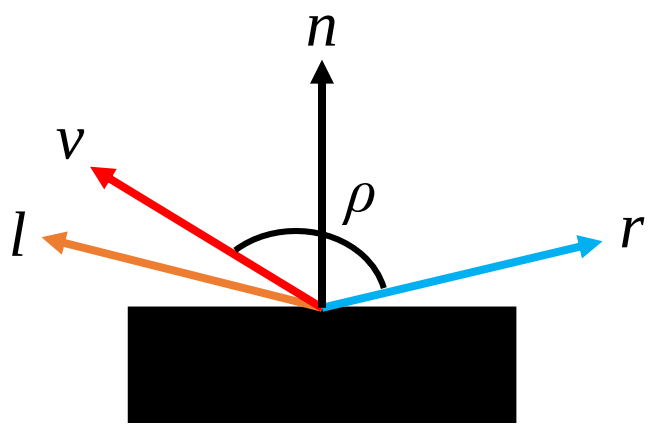
- Given *n* and *v* provided as an input to the pixel shader and *l* provided as a constant buffer by the Direct3D program, the pixel shader first computes the reflection vector, i.e., $r=2n(n\cdot l)-l$, and finally implements the Phong model.
- Note that *l*, *n*, *r*, and *v* are all defined in the world space.

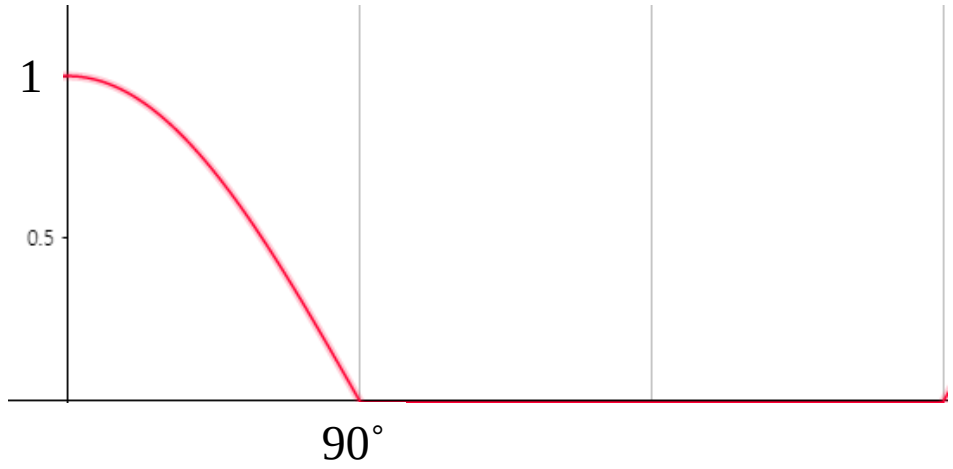


Blinn-Phong Reflection Model

The Glinn-phong reflection model is a modified version of the Phong reflection model.

- Although Phong lighting is a very efficient approximation of lighting, its specular term shows unrealistic shading results in certain conditions.
- Recall that the specular term is calculated by $(max(r \cdot v, 0))^{sh} s_s \otimes m_s$.
  - Here, dot product equals a cosine calculation.
- If the angle is greater than 90˚, the result of the dot product becomes negative.
  - Then, *max*() makes the result 0.





Blinn-Phong Reflection Model

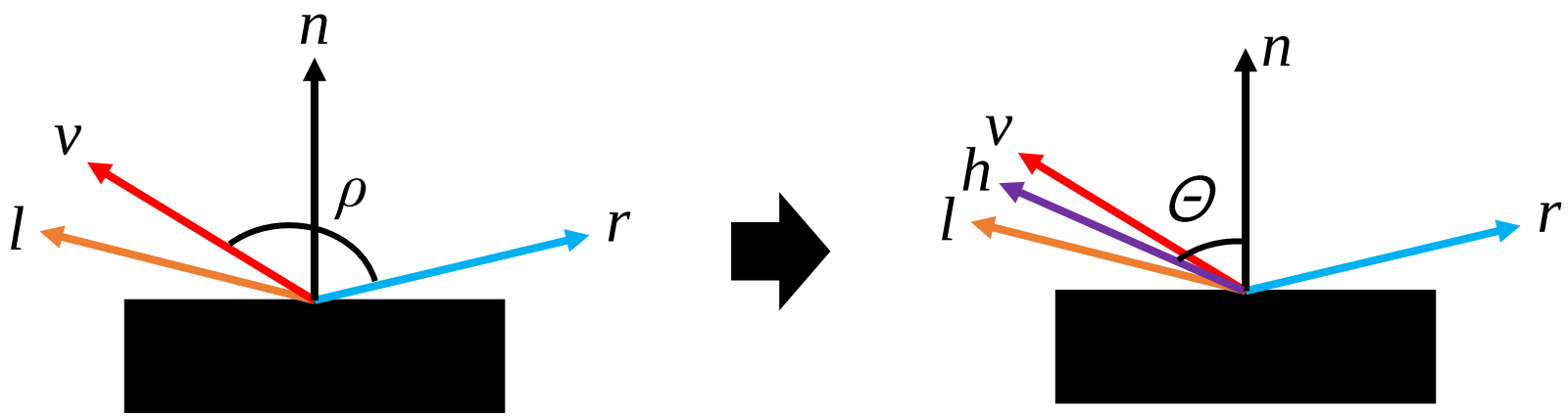
The Glinn-phong reflection model is a modified version of the Phong reflection model.

- This instantaneous cut-off creates a boundary on the surface.
- To overcome this problem, Blinn-phong model approaches the specular model differently.

Blinn-Phong Reflection Model

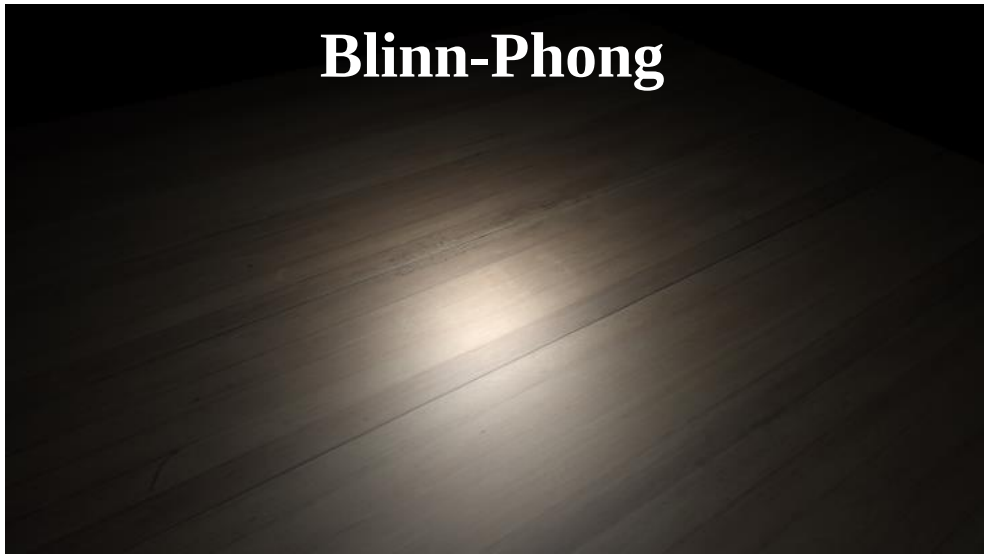
The Glinn-phong reflection model is a modified version of the Phong reflection model.

- In the Blinn-Phong reflection model, reflection term relies on a *halfway* vector (h) instead of reflection vector.
- The h is a unit vector that exactly bisects the view vector (v) and the light vector (l).
  - $h = \frac{l+v}{\|l+v\|}$
- Then the specular term is calculated as: $(max(n \cdot \mathrm{h}, 0))^{sh} s_s \otimes m_{s.}$



Blinn-Phong Reflection Model

$sh = 1$



$sh = 2$