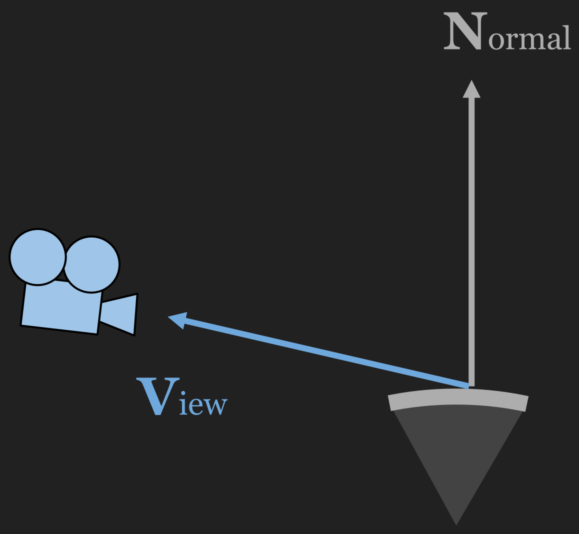
Fundamentals of Computer Graphics and Image Processing

Computer Graphics - Exercise #04

Surface Visibility

How to discard surface (triangles) that is not visible for an observer (camera).

Surface normal vector defines orientation of surface tangential plane.



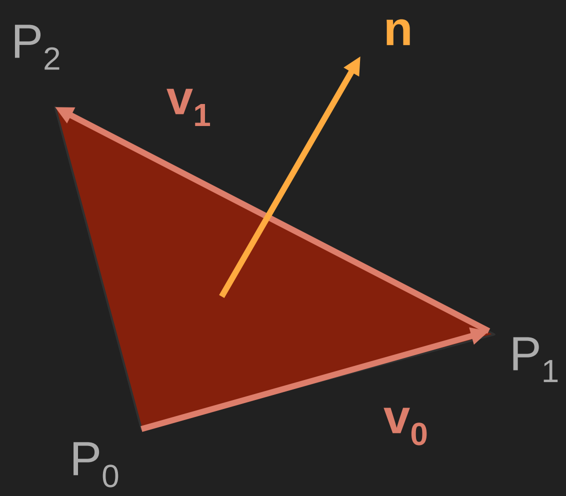
Triangle Normal

Triangle normal specifies an orientation of the triangle in 3D space

$$\overrightarrow{v_0} = P_1 - P_0$$

$$\overrightarrow{v_1} = P_2 - P_1$$

$$\overrightarrow{n} = \overrightarrow{v_0} \times \overrightarrow{v_1}$$



Direction of normal can be ambiguous, therefore vertex order must be explicitly defined (CW or CCW)

Cross product of two vectors

$\vec{a} = \langle a_x, a_y, a_z \rangle$ $= a_x \cdot i + a_y \cdot j + a_z \cdot k$

$\vec{b} = \langle b_x, b_y, b_z \rangle$ $= b_x \cdot i + b_y \cdot j + b_z \cdot k$

$\vec{c} = \vec{a} \times \vec{b} =$

$$\begin{vmatrix} i & j & k \\ a_x & a_y & a_z \\ b_x & b_y & b_z \end{vmatrix} = i \begin{vmatrix} a_y & a_z \\ b_y & b_z \end{vmatrix} - j \begin{vmatrix} a_x & a_z \\ b_x & b_z \end{vmatrix} + k \begin{vmatrix} a_x & a_y \\ b_x & b_y \end{vmatrix}$$

Matrix determinant

$$\begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc$$

Cross product of two vectors

$\vec{a} = \langle a_x, a_y, a_z \rangle$ $= a_x \cdot i + a_y \cdot j + a_z \cdot k$

$\vec{b} = \langle b_x, b_y, b_z \rangle$ $= b_x \cdot i + b_y \cdot j + b_z \cdot k$

$$\vec{c} = \vec{a} \times \vec{b} =$$

$$\begin{vmatrix} i & j & k \\ a_x & a_y & a_z \\ b_x & b_y & b_z \end{vmatrix} = i \begin{vmatrix} a_y & a_z \\ b_y & b_z \end{vmatrix} - j \begin{vmatrix} a_x & a_z \\ b_x & b_z \end{vmatrix} + k \begin{vmatrix} a_x & a_y \\ b_x & b_y \end{vmatrix}$$

$$= i\,(a_y b_z - a_z b_y) - j\,(a_x b_z - a_z b_x) + k\,(a_x b_y - a_y b_x)$$

Cross product of two vectors

$\vec{a} = \langle a_x, a_y, a_z \rangle$ $= a_x \cdot i + a_y \cdot j + a_z \cdot k$

$\vec{b} = \langle b_x, b_y, b_z \rangle$ $= b_x \cdot i + b_y \cdot j + b_z \cdot k$

$\vec{c} = \vec{a} \times \vec{b} =$

$$\begin{vmatrix} i & j & k \\ a_x & a_y & a_z \\ b_x & b_y & b_z \end{vmatrix} = i \begin{vmatrix} a_y & a_z \\ b_y & b_z \end{vmatrix} - j \begin{vmatrix} a_x & a_z \\ b_x & b_z \end{vmatrix} + k \begin{vmatrix} a_x & a_y \\ b_x & b_y \end{vmatrix}$$

$$= i\,(a_y b_z - a_z b_y) - j\,(a_x b_z - a_z b_x) + k\,(a_x b_y - a_y b_x) = \boxed{i \cdot c_x - j \cdot c_y + k \cdot c_z}$$

Cross product of two vectors

$\vec{a} = \langle a_x, a_y, a_z \rangle$

$\vec{b} = \langle b_x, b_y, b_z \rangle$

$\vec{c} = \vec{a} \times \vec{b} =$

$= (a_y b_z - a_z b_y) + (a_z b_x - a_x b_z) + (a_x b_y - a_y b_x) = c_x + c_y + c_z$

$\Rightarrow c_x = a_y b_z - a_z b_y$

$c_y = a_z b_x - a_x b_z$

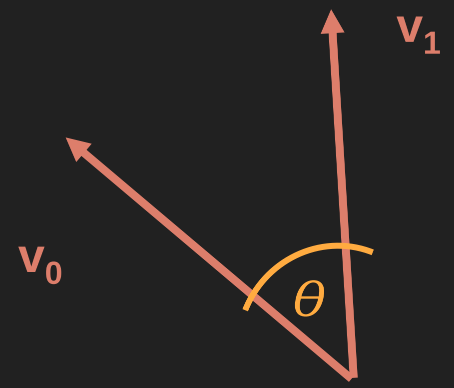
$c_z = a_x b_y - a_y b_x$

Scalar Product and Vector Angle

Having two vectors v_0 and v_1

Angle between them is denoted θ

The dot product can be defined as:



$$\overrightarrow{v_0} \cdot \overrightarrow{v_1} = |\overrightarrow{v_0}| * |\overrightarrow{v_1}| * cos(\theta)$$

When the vectors are normalized (length is equal to 1):

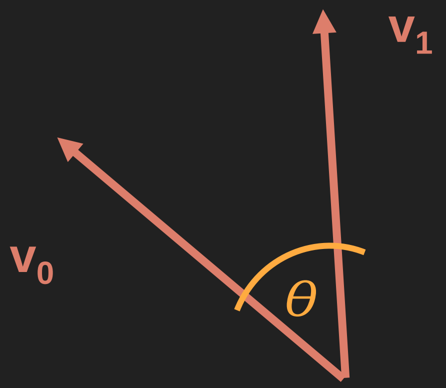
$$\overrightarrow{v_0} \cdot \overrightarrow{v_1} = cos(\theta)$$

Scalar Product and Vector Angle

Having two vectors v_0 and v_1

Angle between them is denoted θ

The dot product can be defined as:



$$\overrightarrow{v_0} \cdot \overrightarrow{v_1} = |\overrightarrow{v_0}| * |\overrightarrow{v_1}| * cos(\theta)$$

When the vectors are normalized (length is equal to 1):

$$\overrightarrow{v_0} \cdot \overrightarrow{v_1} = cos(\theta)$$

$$= v0_x\ v1_x + v0_y\ v1_y + v0_z\ v1_z$$

Back-Face Culling

Triangle is visible to the camera IF

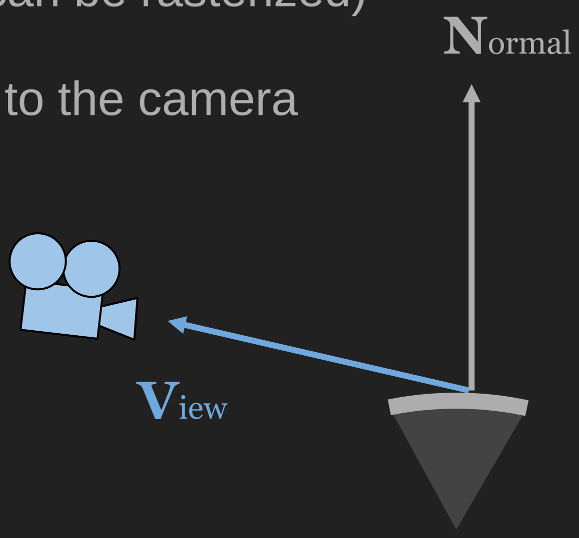
- an angle of view vector and surface normal is less than 90°

OR

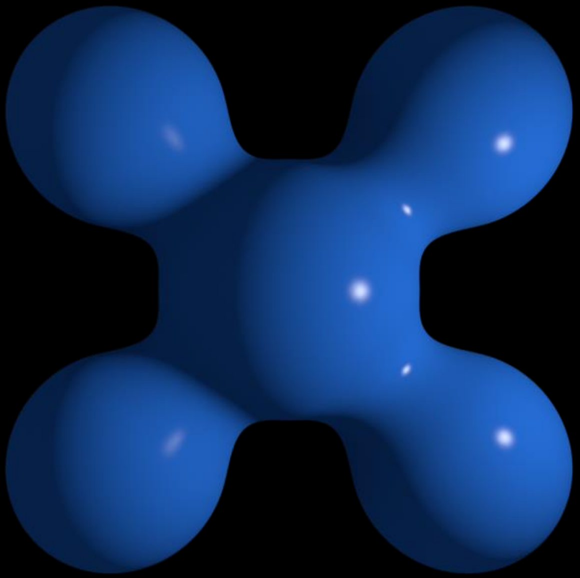
- their dot product is positive:

IF *dot*(V, N) > 0 ~ visible (can be rasterized)

ELSE ~ invisible to the camera

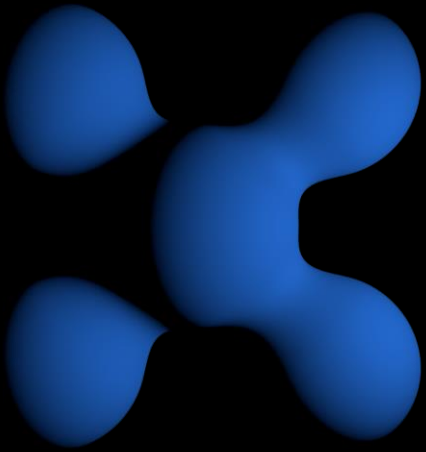


Phong Reflection Model



Final illumination of a surface can be divided into 3 components:
Ambient, Diffuse and Specular

Amount of each component in final mix depends on object’s material

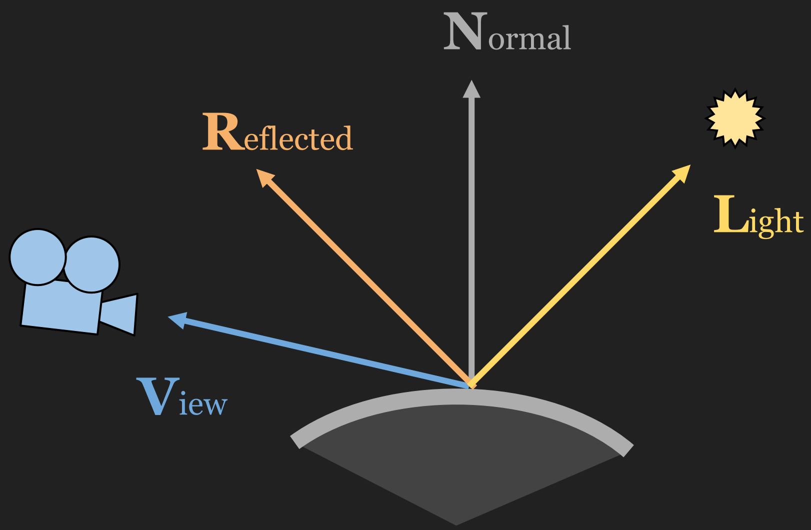


Phong Reflection Model

$$I = k_a I_a + \sum_{L} (k_d I_d + k_s I_s)$$

$$I_d = N \cdot L$$

$$I_s = (R \cdot V)^h$$



Material properties

$\mathbf{k_a}, \mathbf{k_d}, \mathbf{k_s}$ - reflection constants

$\mathbf{h}$ - shininess constant

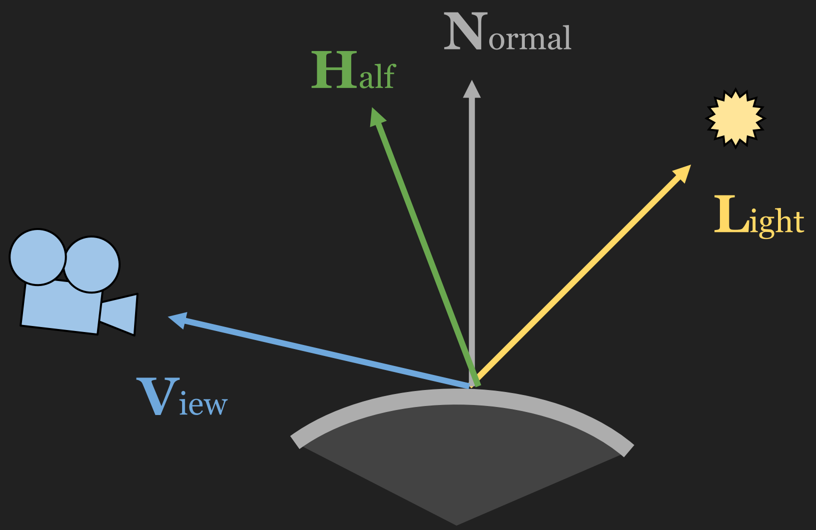
Working with normalized vectors!

Blinn-Phong Reflection Model

$$I = k_a I_a + \sum_L (k_d I_d + k_s I_s)$$

$$I_d = N \cdot L$$

$$I_s = (H \cdot N)^h \qquad H = \frac{V+L}{|V+L|}$$



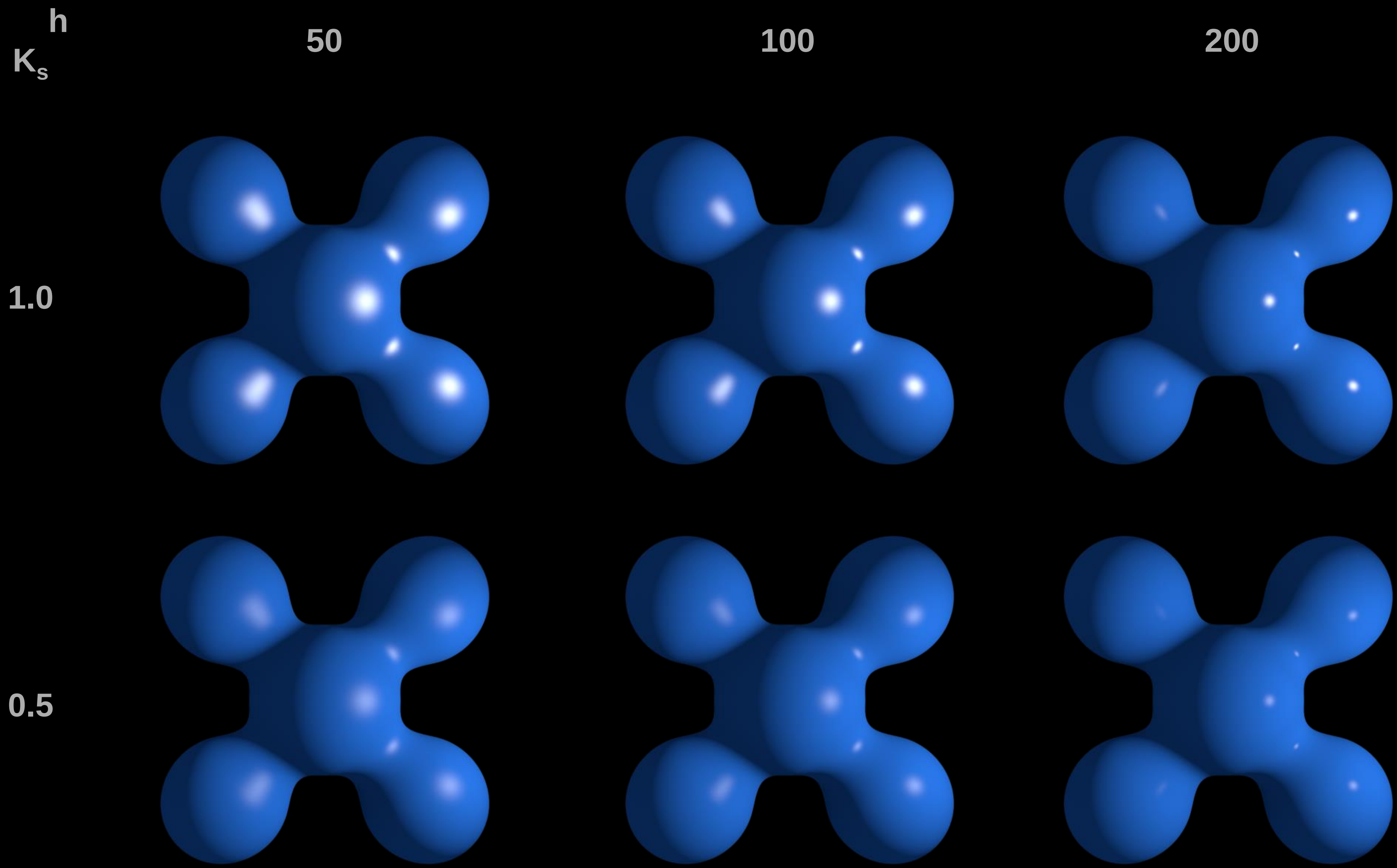
Material properties

$\mathbf{k_a}, \mathbf{k_d}, \mathbf{k_s}$ - reflection constants

$\mathbf{h}$ - shininess constant

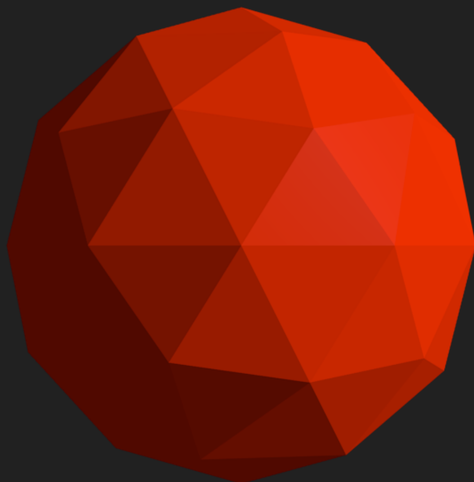
Working with normalized vectors!

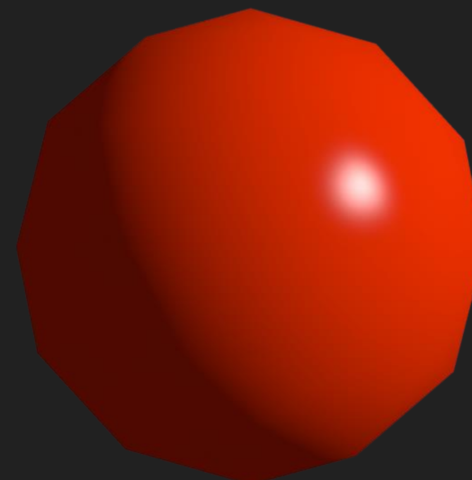
Shininess Constant in Blinn-Phong



Flat Shading

Phong Shading





Note: Phong reflection and Phong shading are not the same! Phong reflection model can be used for calculating lightning of both shading methods