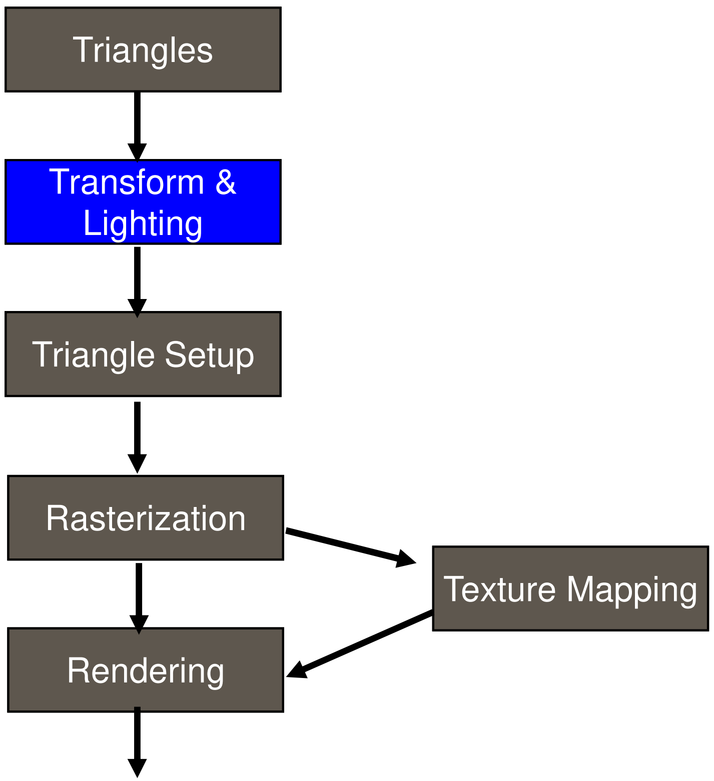
Computer Graphics

by Ruen-Rone Lee
ICL/ITRI

Wrap up from last course

- Geometrical Transformation
- Viewing Transformation
- Projection Transformation
- Viewport Transformation

Part I:
Conventional 3D Graphics Pipeline

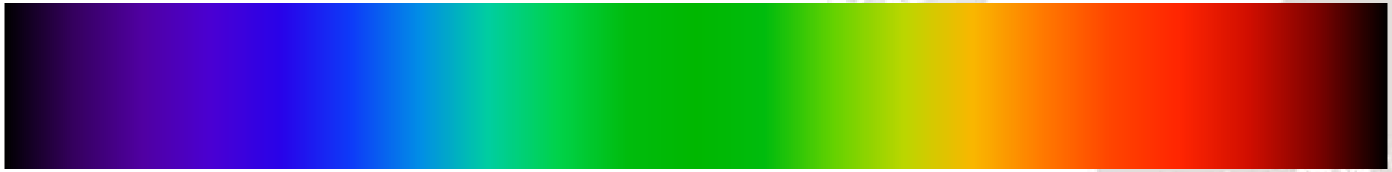


Conventional 3D Graphics Pipeline

Lighting

Color Model
Illumination Model
Polygon Shading

Color

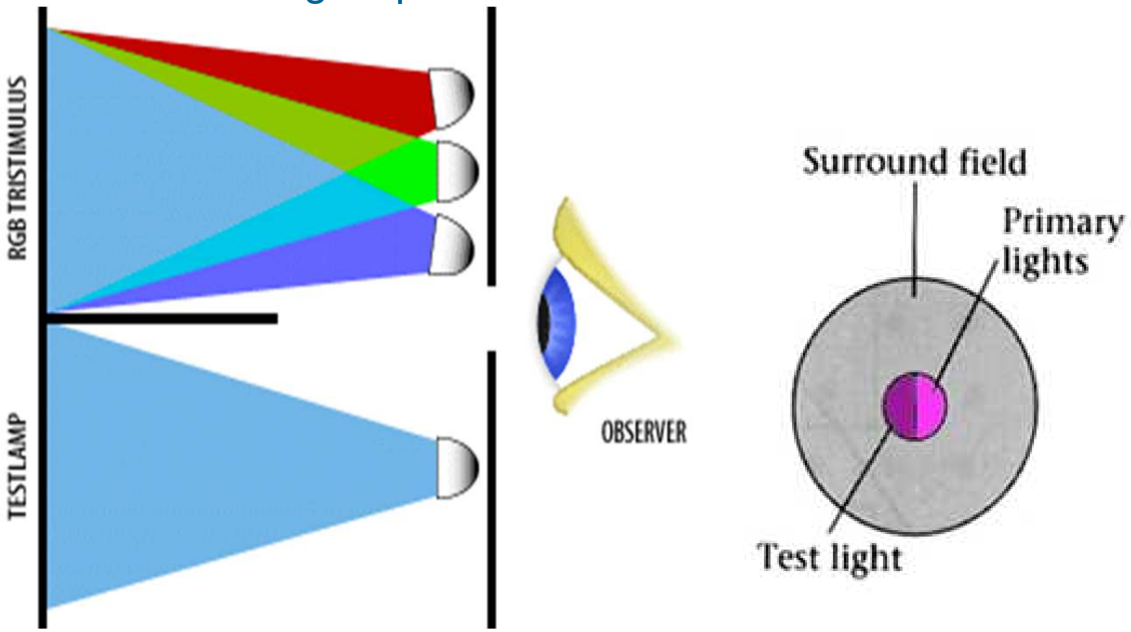


- **Stimulation of cone cells in the human eye by electromagnetic radiation in the visible spectrum**

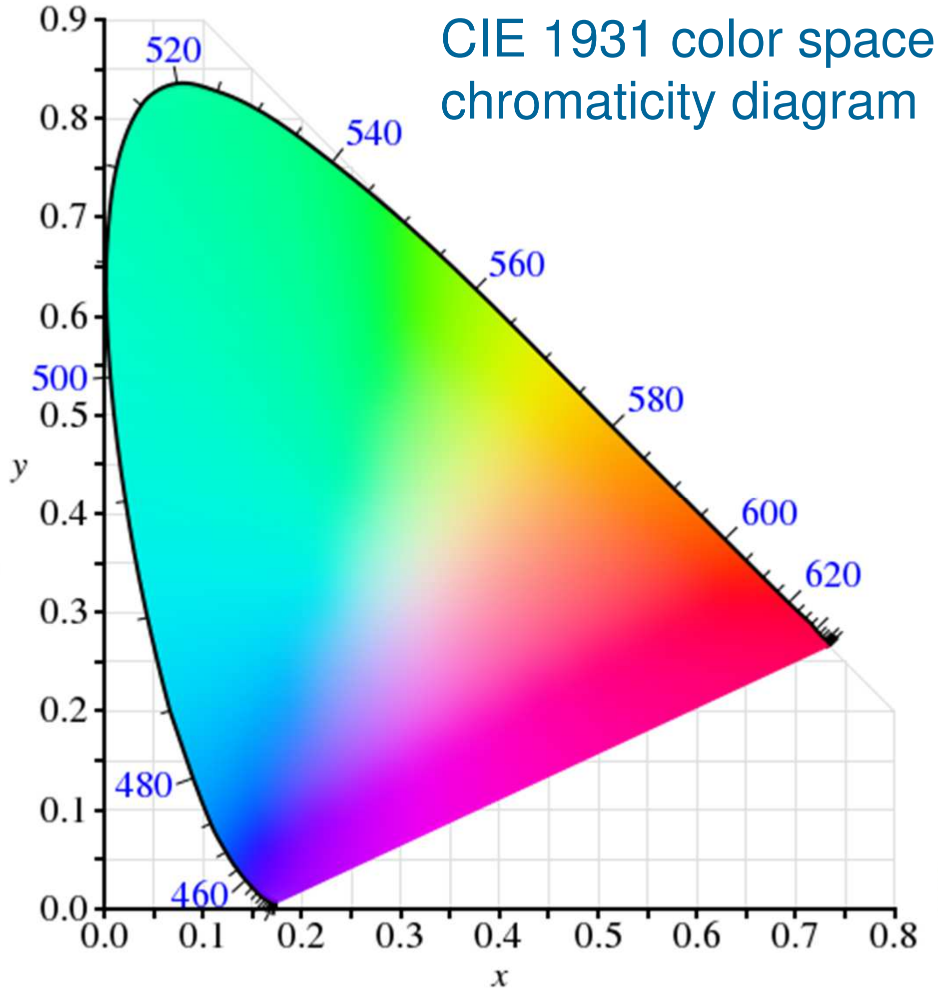
Color = (brightness, chromaticity)

$$\text{Color} = (\underbrace{\text{brightness}}_{\text{amplitude}}, \underbrace{\text{chromaticity}}_{\text{frequency}})$$

Color Matching Experiment

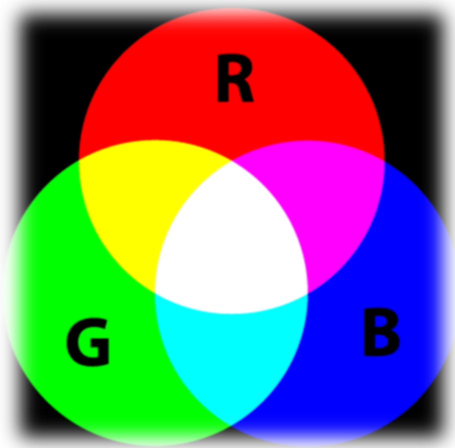


CIE 1931 color space chromaticity diagram



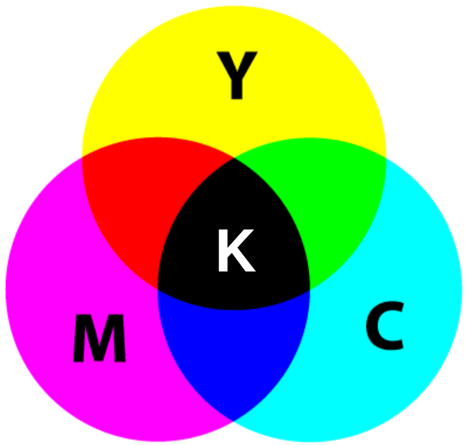
Color Model

- An abstract model to describe colors
- Representation: A tuple (three or four values/components) is used to represent a specific color, such as (*r*, *g*, *b*) in RGB color model or (*c*, *m*, *y*, *k*) in CMYK color model



- Additive color model
- Used in sensing, representation, and display of images in electronic systems, such as TV and computer monitor

RGB Color Model

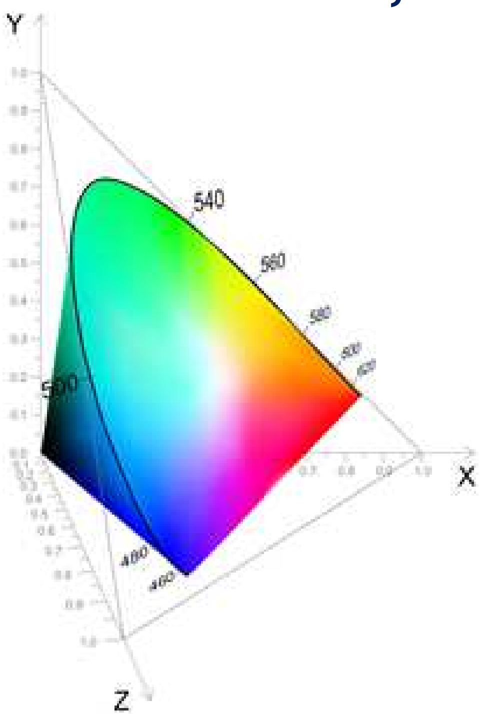


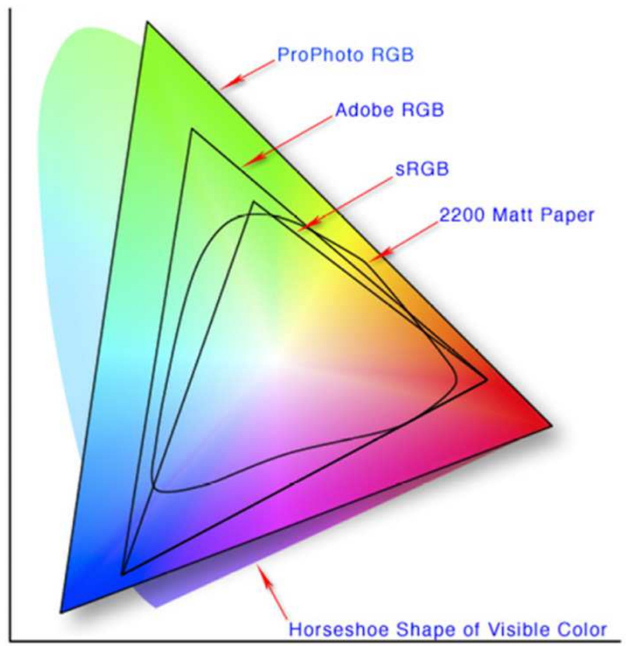
- Subtractive color model
- Used in color printing

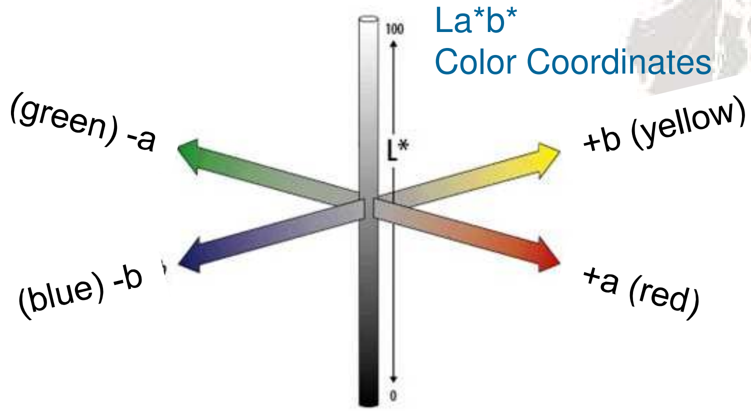
CMYK Color Model

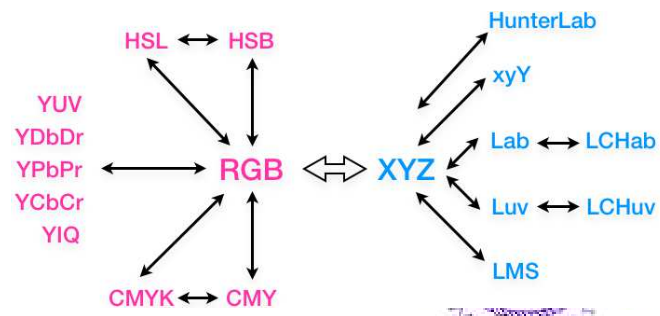
Color Space

- Define the range and tones of colors
  - Device-invariant: human visible colors
    - CIE-RGB, CIE-XYZ, …
  - Device-variant: device producible colors
    - sRGB, Adobe RGB, CMYK, …



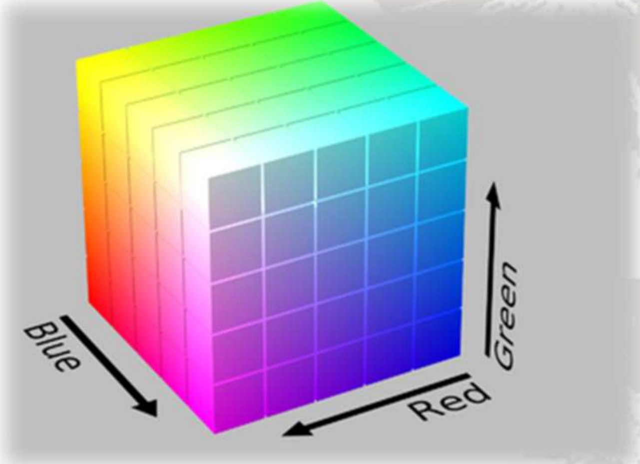


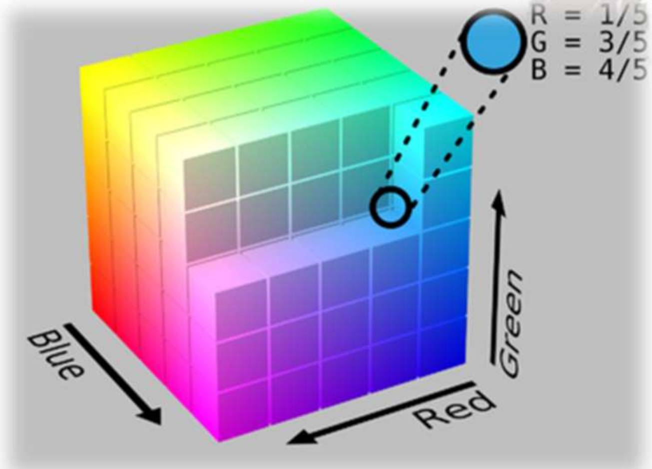




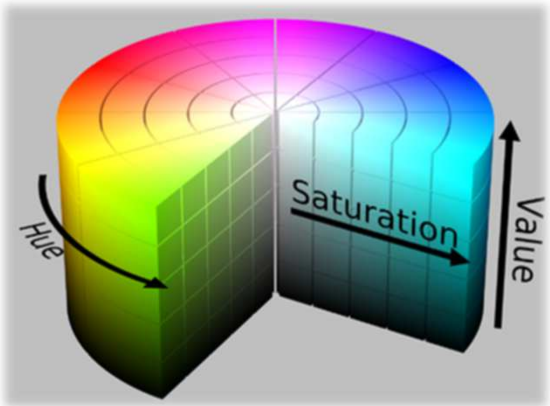
RGB Color Model and Color Space

- **Three additive primary colors, red, green and blue light are added together in various ways to synthesize the colors in the associated color space**
  - **HSV and HSL are transformation of RGB color space**

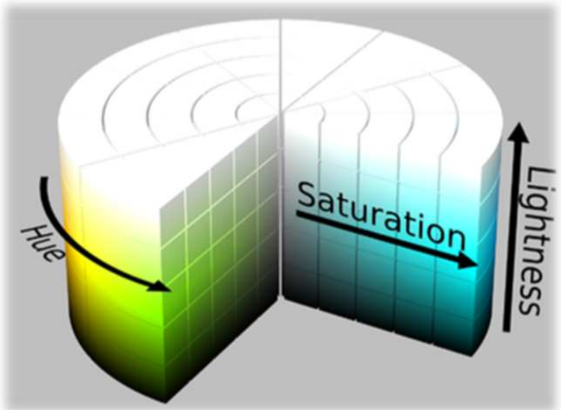




HSV

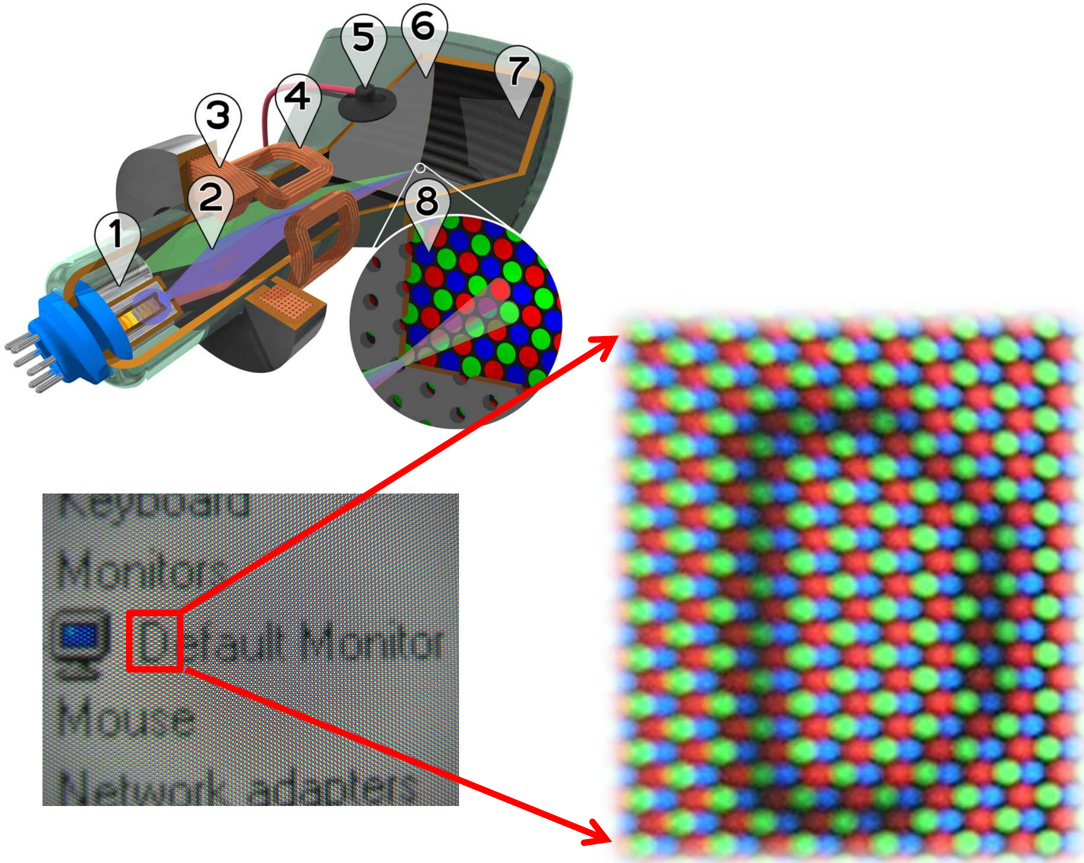


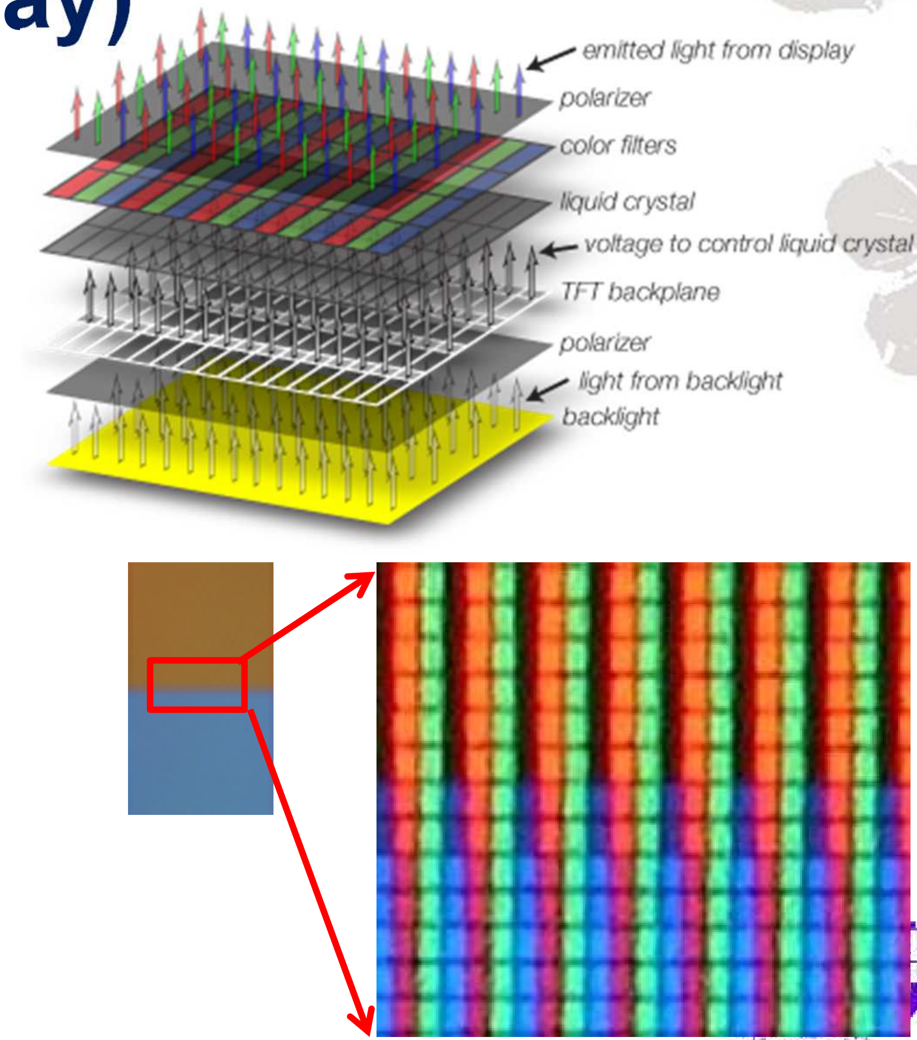
HSL



Displays using RGB Color Model

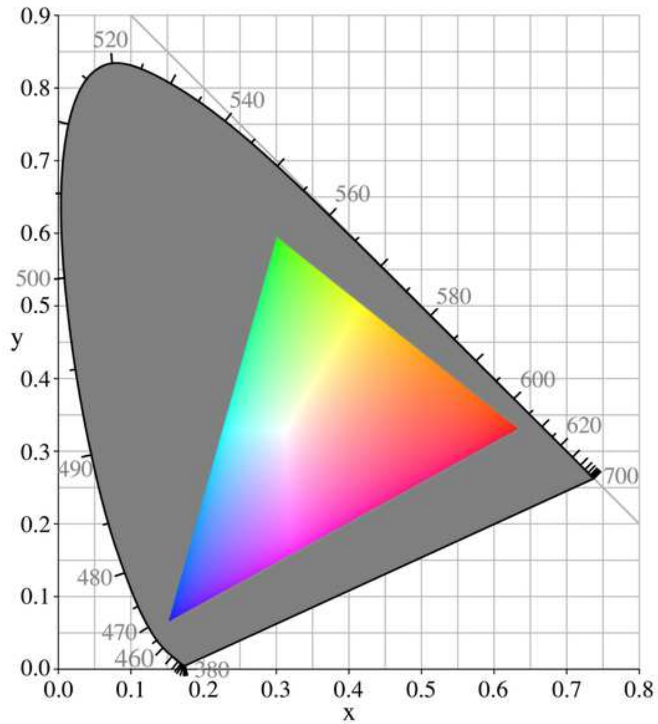
- CRT (cathode ray tube)
- LCD (liquid crystal display)





Color Depth

- Gamut (色域): a subset of colors which can be represented in a given color space
- Color depth: the number of bits to represent the color of a pixel (bpp: bits per pixel)
  - 15/16 bits, 24/32 bits, …
  - The available colors of a given color depth d is 2^d
    - e.g., 24-bit (R8G8B8) → 2^{24} colors



sRGB Gamut

Shading

- Shading is a light-material interaction in determining the colors of pixels of rendered objects or scenes

Flat Shading

Smooth Shading

Advanced Shading

- Global illumination
  - Direct illumination plus indirect illumination
    - Reflection
    - Refraction
    - Shadow

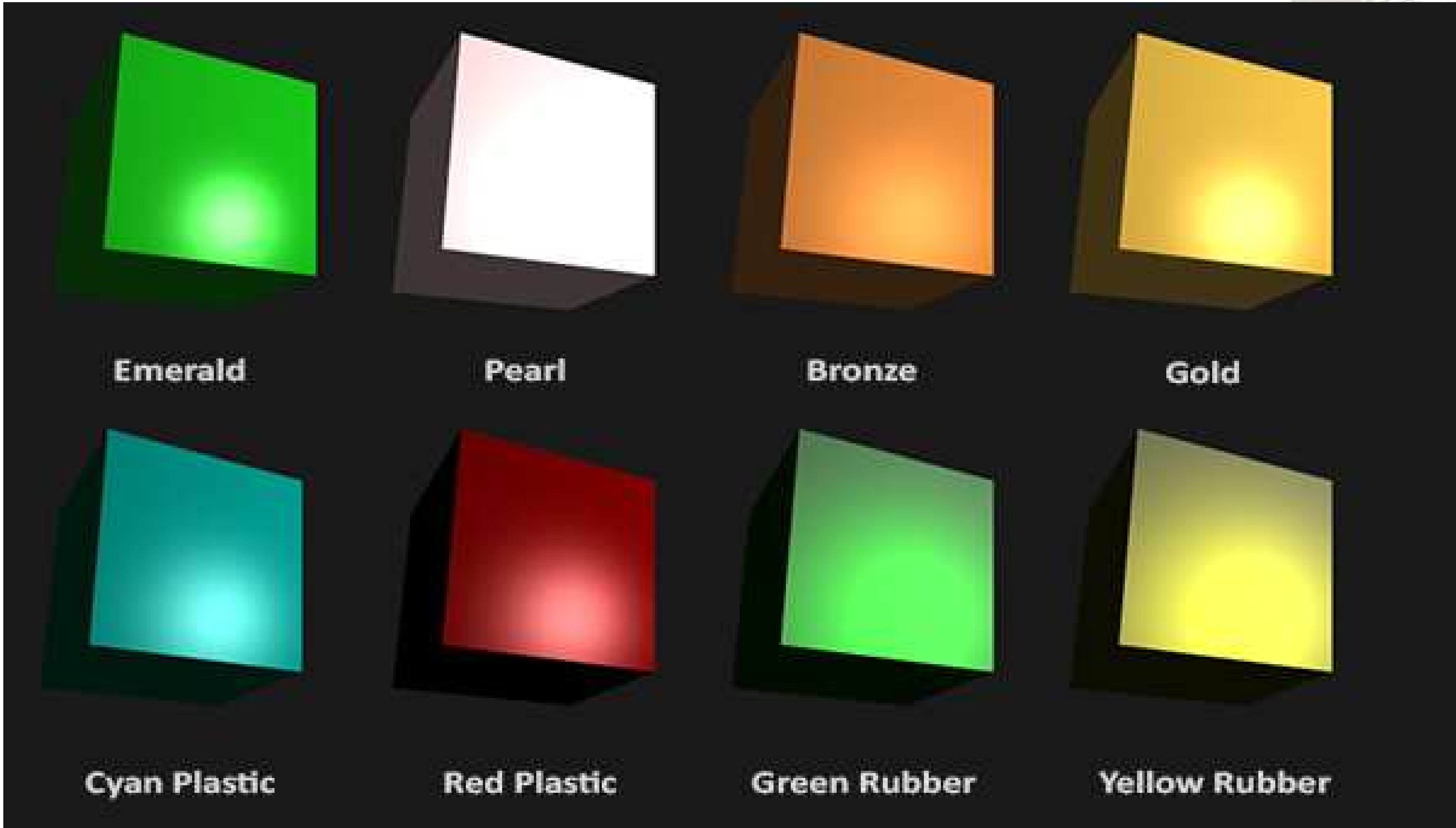
Factors that affect Shading

- Light sources
  - Ambient light, directional light, positional light, spot light…
- Material properties
  - Ambient reflection, diffuse reflection, specular reflection…
- Location of the viewer
  - Position of perceiving specular highlight
- Surface orientation
  - Surface normal, vertex normal

Example of Materials

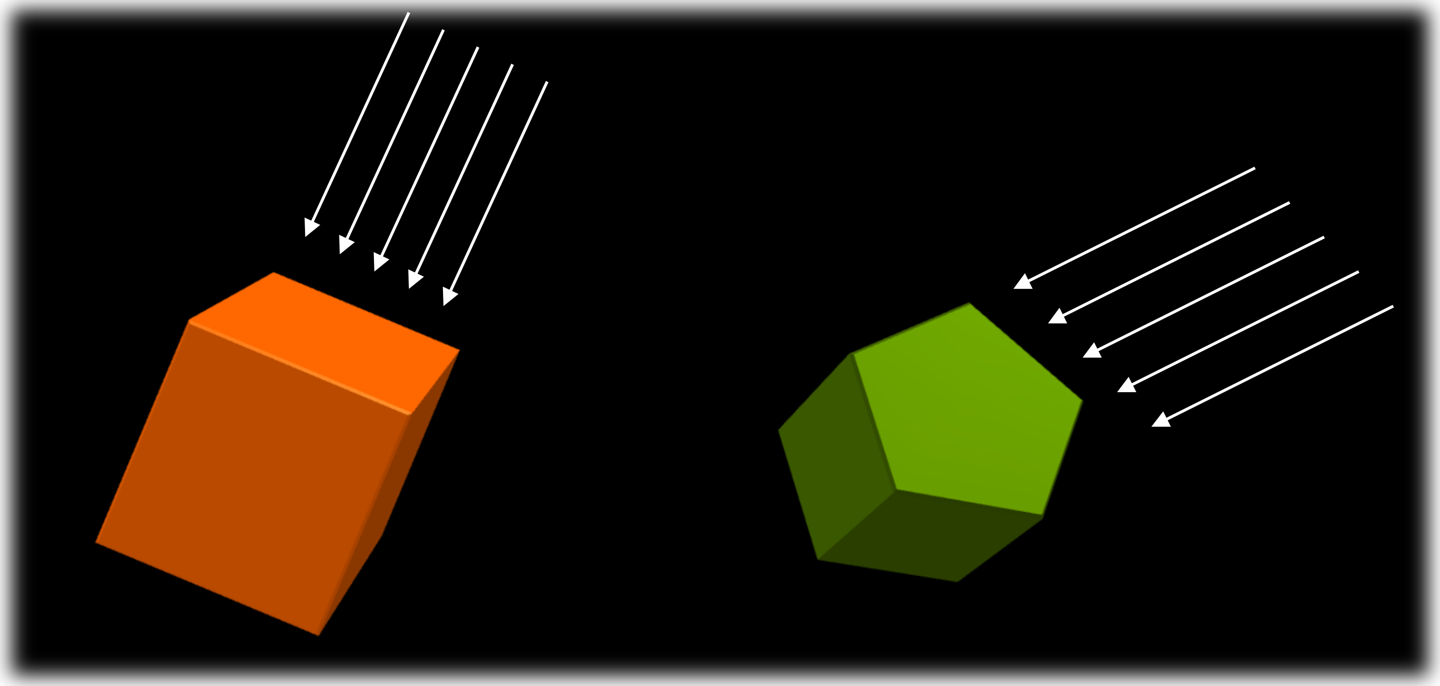
Name	Ambient			Diffuse			Specular			Shininess
emerald	0.0215	0.1745	0.0215	0.07568	0.61424	0.07568	0.633	0.727811	0.633	0.6
jade	0.135	0.2225	0.1575	0.54	0.89	0.63	0.316228	0.316228	0.316228	0.1
obsidian	0.05375	0.05	0.06625	0.18275	0.17	0.22525	0.332741	0.328634	0.346435	0.3
pearl	0.25	0.20725	0.20725	1	0.829	0.829	0.296648	0.296648	0.296648	0.088
ruby	0.1745	0.01175	0.01175	0.61424	0.04136	0.04136	0.727811	0.626959	0.626959	0.6
turquoise	0.1	0.18725	0.1745	0.396	0.74151	0.69102	0.297254	0.30829	0.306678	0.1
brass	0.329412	0.223529	0.027451	0.780392	0.568627	0.113725	0.992157	0.941176	0.807843	0.21794872
bronze	0.2125	0.1275	0.054	0.714	0.4284	0.18144	0.393548	0.271906	0.166721	0.2
chrome	0.25	0.25	0.25	0.4	0.4	0.4	0.774597	0.774597	0.774597	0.6
copper	0.19125	0.0735	0.0225	0.7038	0.27048	0.0828	0.256777	0.137622	0.086014	0.1
gold	0.24725	0.1995	0.0745	0.75164	0.60648	0.22648	0.628281	0.555802	0.366065	0.4
silver	0.19225	0.19225	0.19225	0.50754	0.50754	0.50754	0.508273	0.508273	0.508273	0.4
black plastic	0.0	0.0	0.0	0.01	0.01	0.01	0.50	0.50	0.50	.25
cyan plastic	0.0	0.1	0.06	0.0	0.50980392	0.50980392	0.50196078	0.50196078	0.50196078	.25
green plastic	0.0	0.0	0.0	0.1	0.35	0.1	0.45	0.55	0.45	.25
red plastic	0.0	0.0	0.0	0.5	0.0	0.0	0.7	0.6	0.6	.25
white plastic	0.0	0.0	0.0	0.55	0.55	0.55	0.70	0.70	0.70	.25
yellow plastic	0.0	0.0	0.0	0.5	0.5	0.0	0.60	0.60	0.50	.25
black rubber	0.02	0.02	0.02	0.01	0.01	0.01	0.4	0.4	0.4	.078125
cyan rubber	0.0	0.05	0.05	0.4	0.5	0.5	0.04	0.7	0.7	.078125
green rubber	0.0	0.05	0.0	0.4	0.5	0.4	0.04	0.7	0.04	.078125
red rubber	0.05	0.0	0.0	0.5	0.4	0.4	0.7	0.04	0.04	.078125
white rubber	0.05	0.05	0.05	0.5	0.5	0.5	0.7	0.7	0.7	.078125
yellow rubber	0.05	0.05	0.0	0.5	0.5	0.4	0.7	0.7	0.04	.078125

Example of Materials



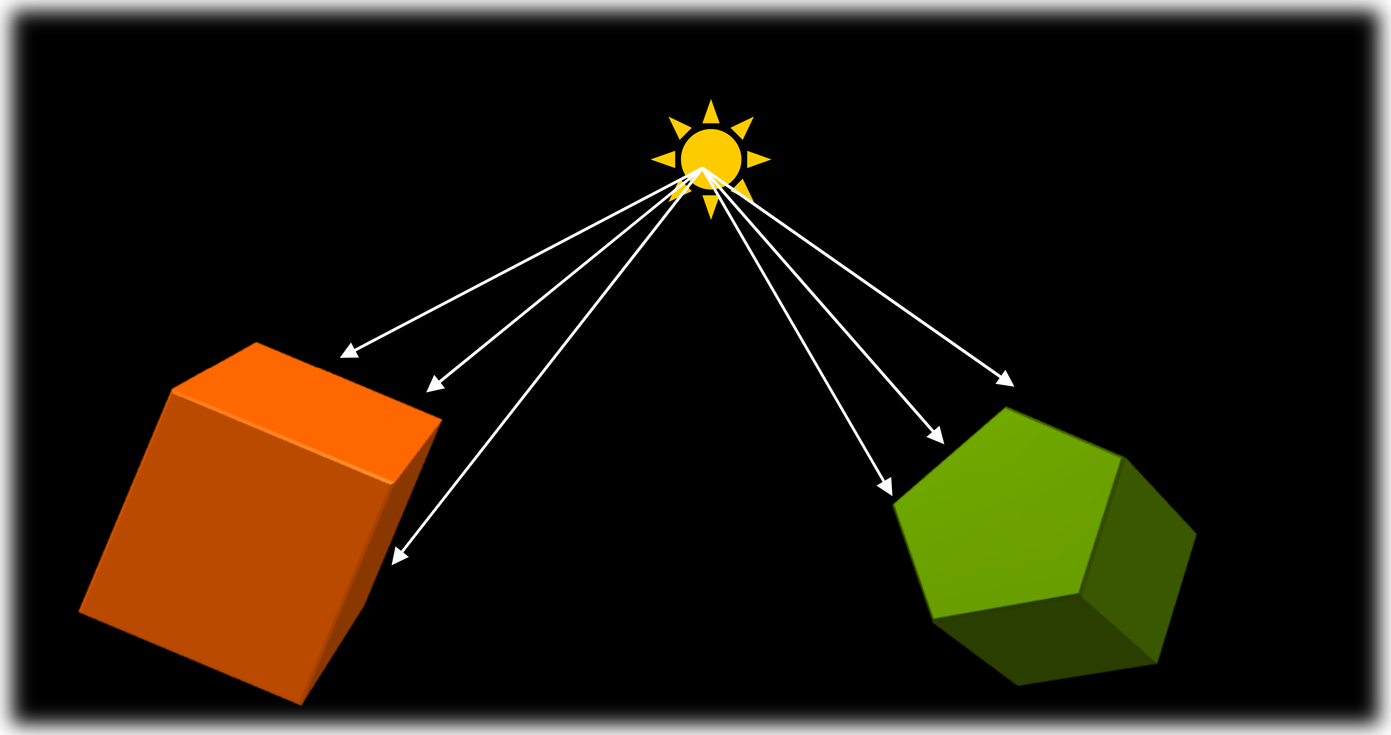
Light Sources

- Directional Light
  - Light source located at infinite far away such as the sun



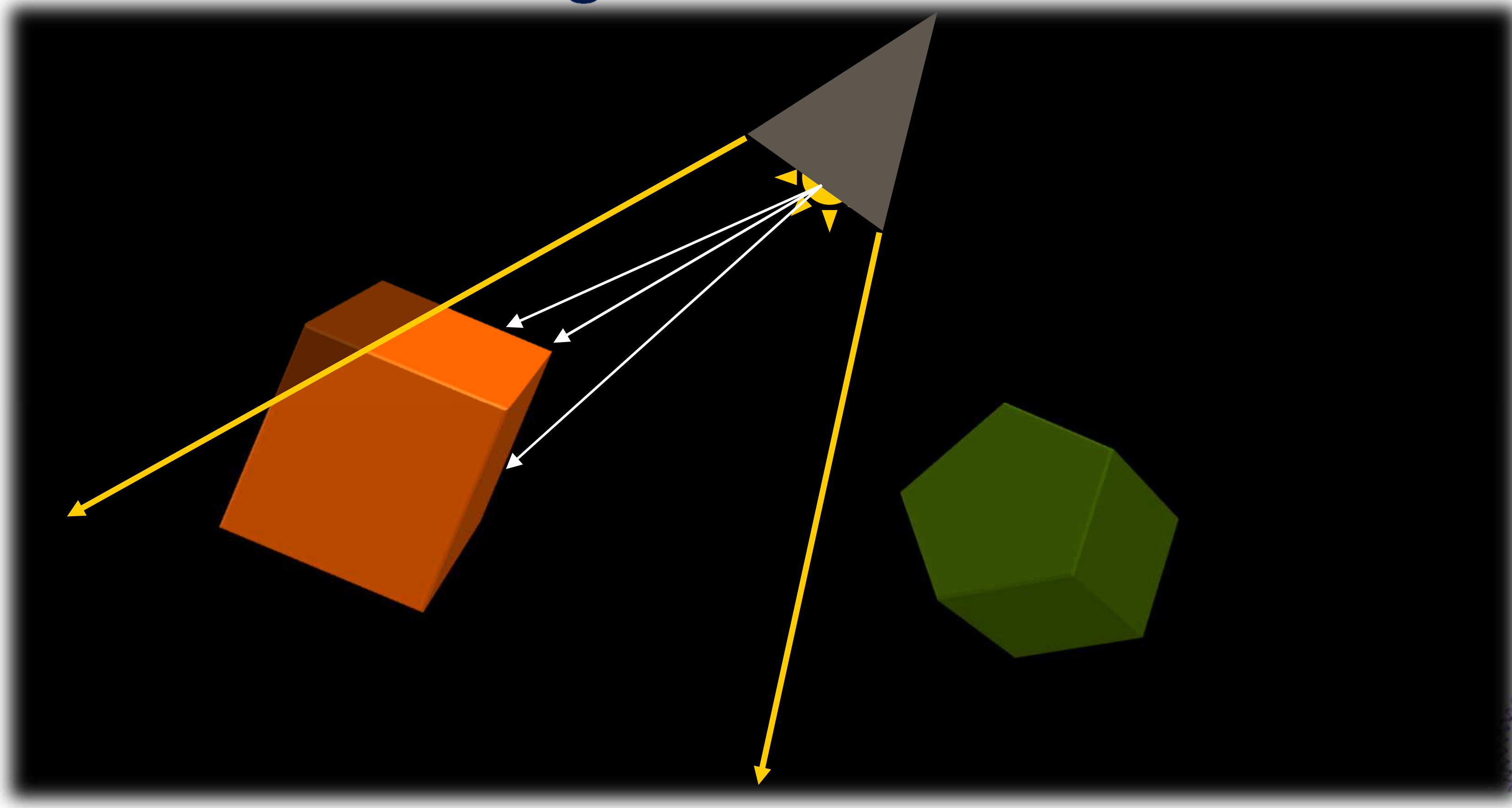
Light Sources

- Positional Light (Point Light)
  - Light source located at a specific position



Light Sources

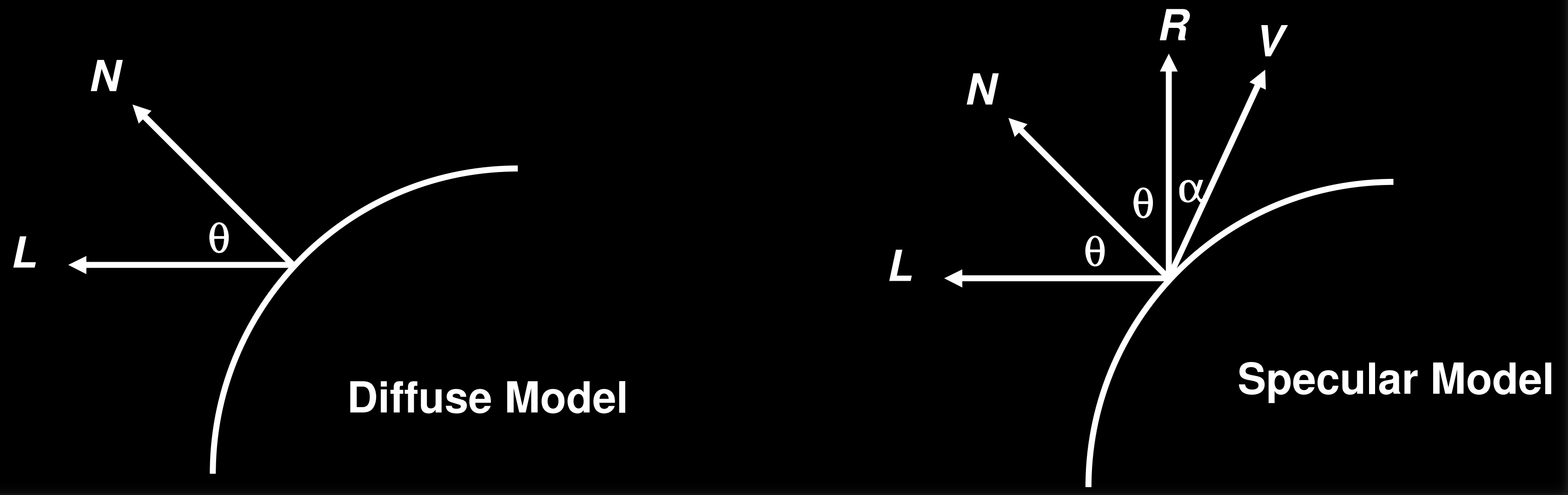
- **Spot Light**
  - **Light source located at a specific position with certain cutoff range**



Lighting Equation

- Intensity = Ambient + Diffuse + Specular

$$I = \boxed{I_a k_a} + \sum_{p=1}^{m} f_p I_p \left(\boxed{k_d (N \cdot L_p)} + \boxed{k_s (R_p \cdot V)^n} \right)$$



Ambient Light

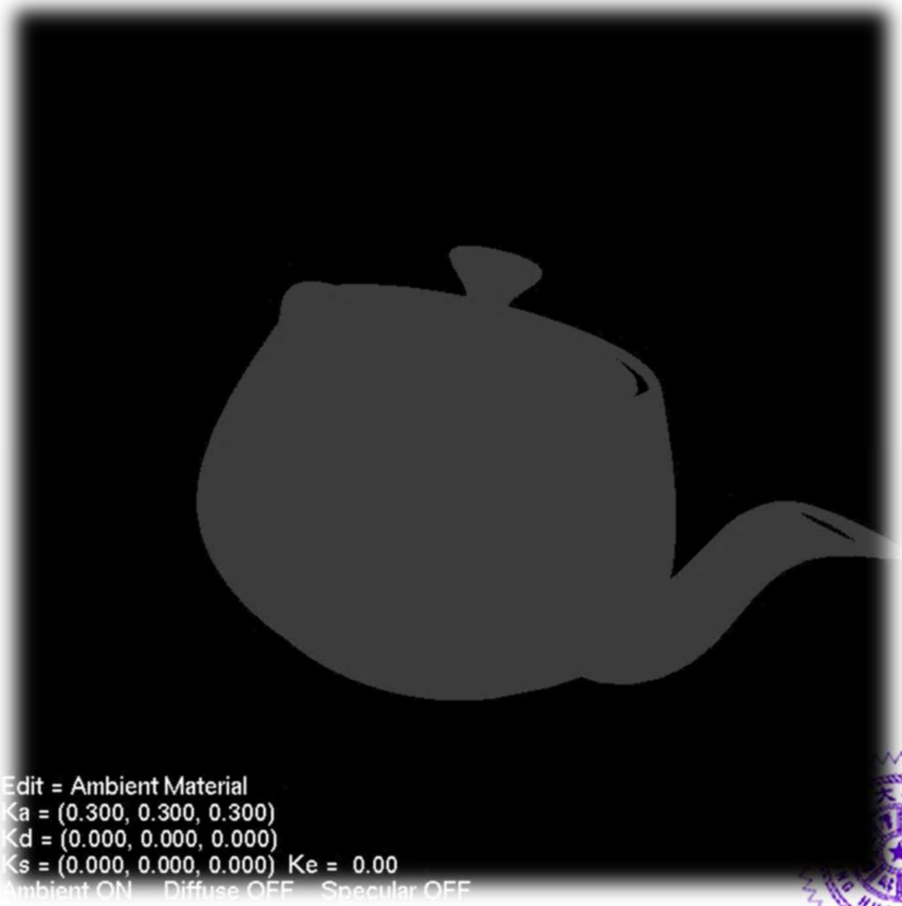
- **Illumination surrounding a scene without providing any specific light source**

$$I = I_a k_a$$

I: resulting intensity

I_a: ambient light intensity

k_a : ambient reflection coefficient

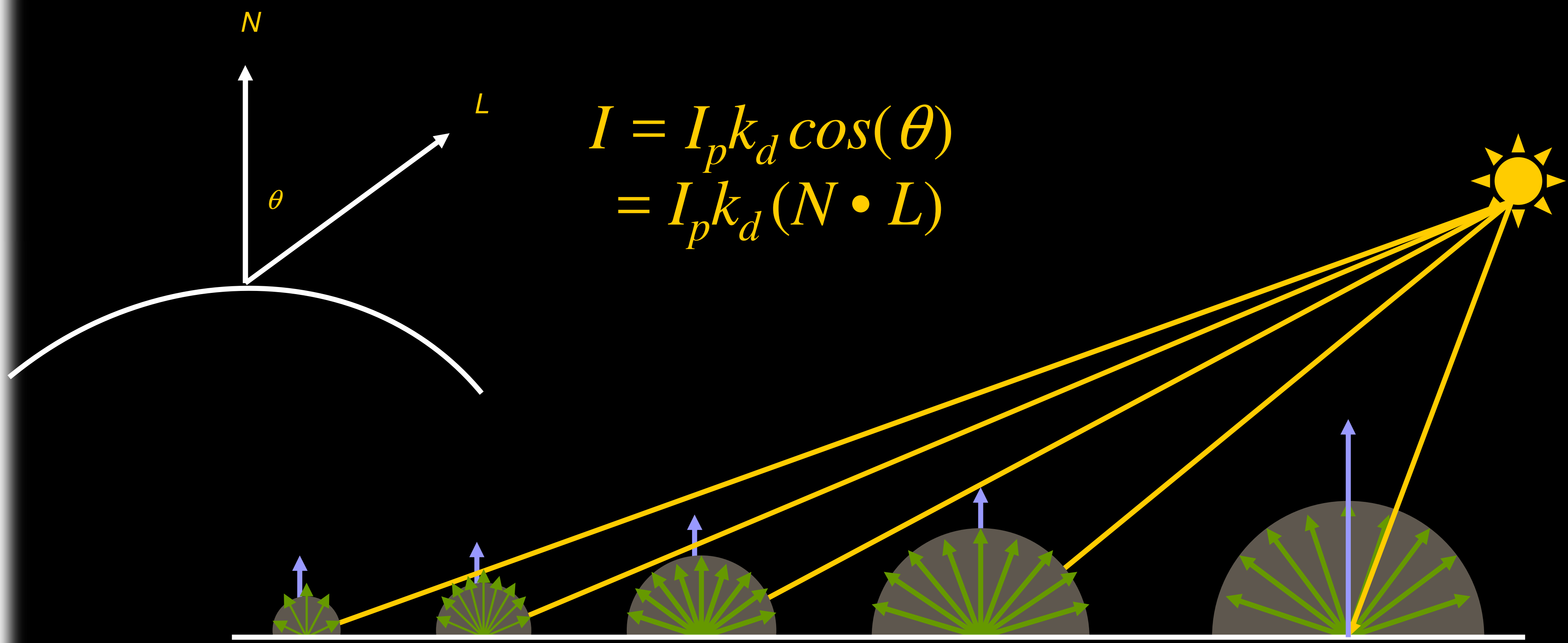


Diffuse Reflection

- **Lambert's Cosine Law**

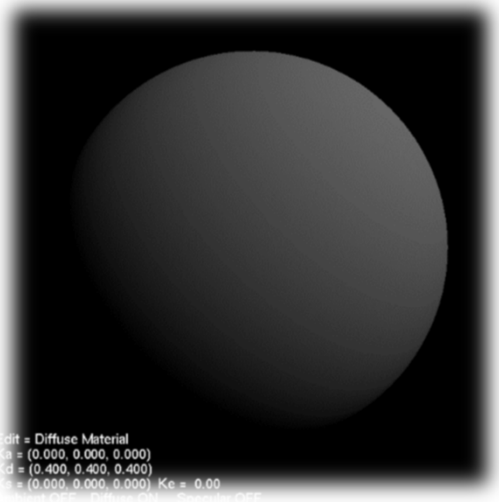
I_p: point light source intensity
k_d: diffuse reflection coefficient
N: normalized normal vector
L: normalized light direction vector

$$I = I_p k_d \cos(\theta) = I_p k_d (N \bullet L)$$

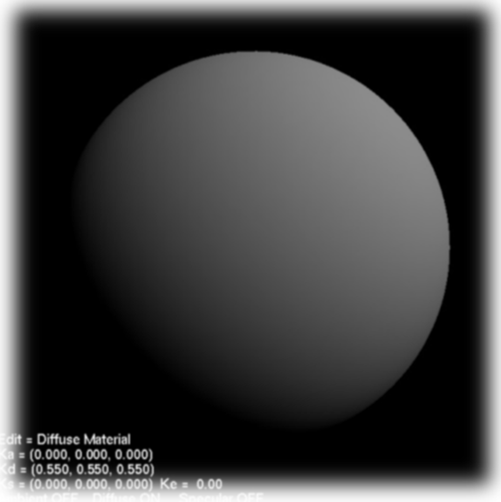


Diffuse Reflection

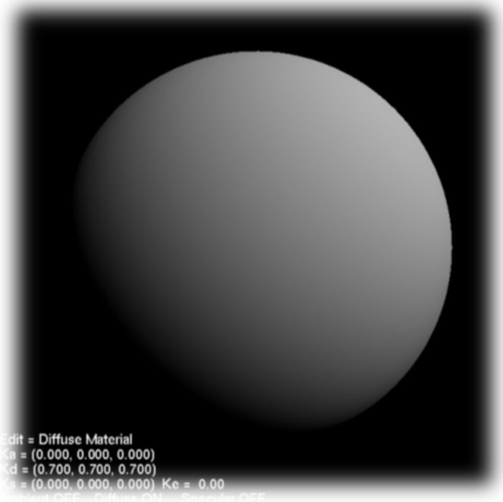
- **Example**
  - **Fixed point light source at (1.0, 1.0, 1.0)**



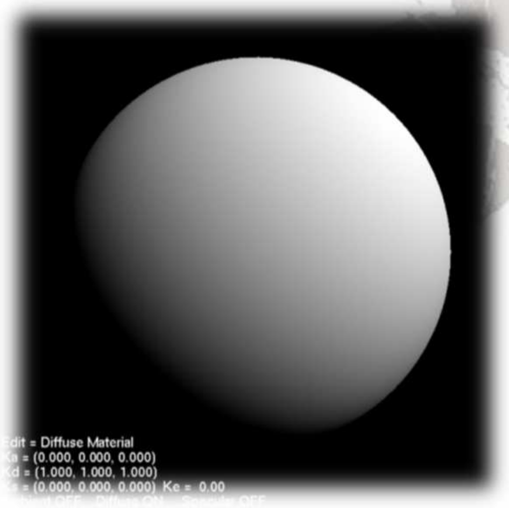
$k_d = 0.4$



$k_d = 0.55$



$k_d = 0.7$

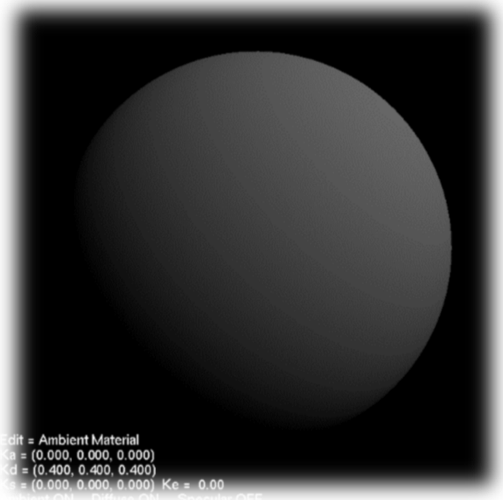


$k_d = 1.0$

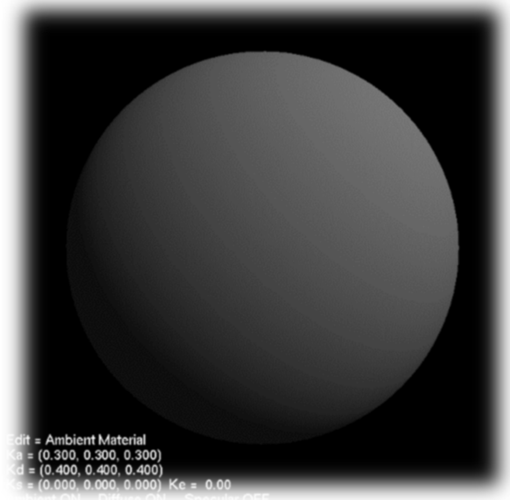
Ambient + Diffuse Reflection

- **Example**
  - $k_d = 0.4$

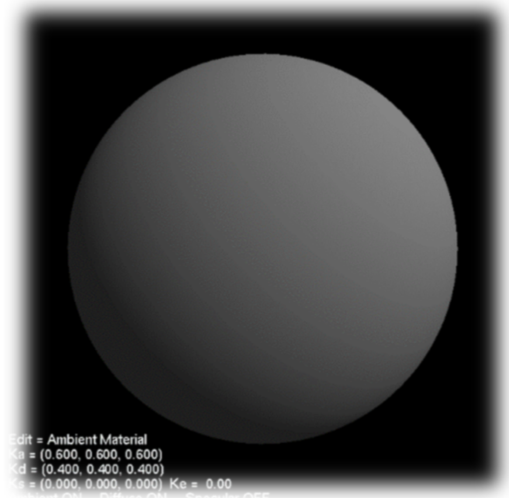
$$I = I_a k_a + I_p k_d (N \bullet L)$$



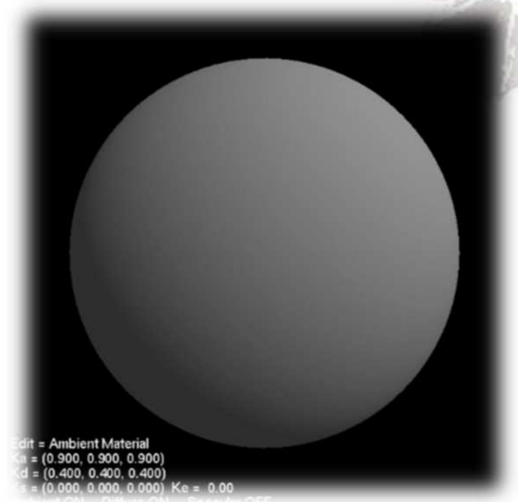
$k_a = 0.0$



$k_a = 0.3$



$k_a = 0.6$

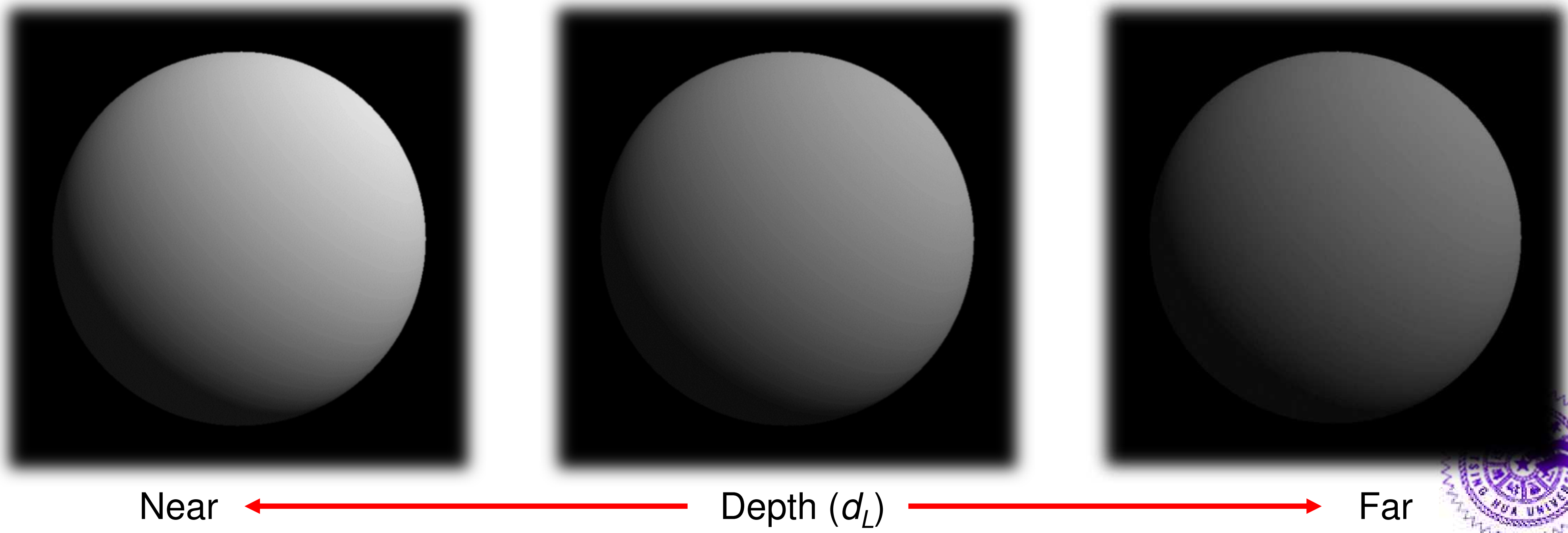


$k_a = 0.9$

Light Source Attenuation

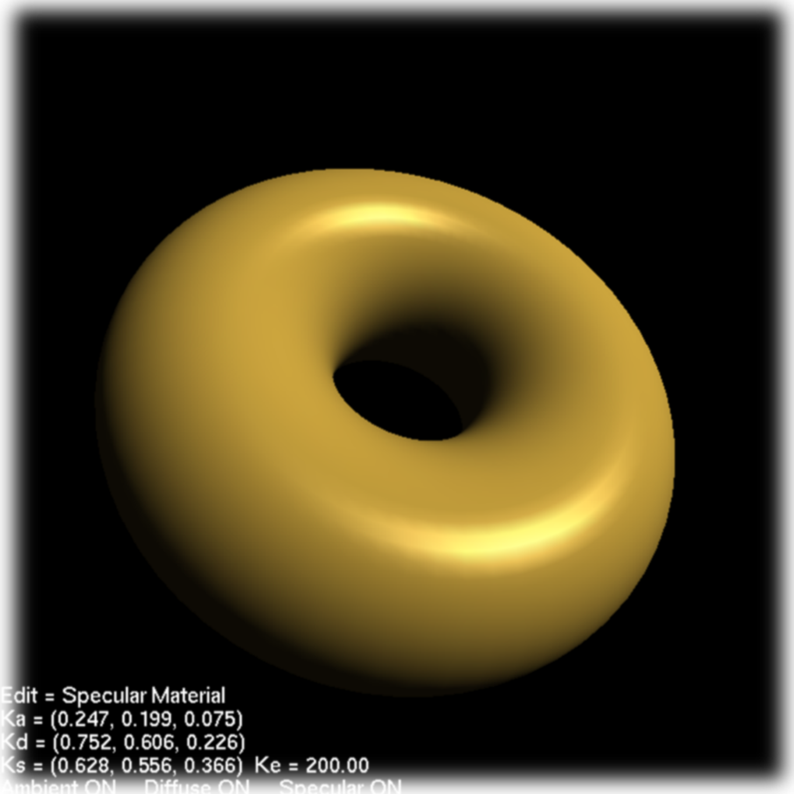
- Light source intensity will attenuate with respect to the distance between the light source and the object

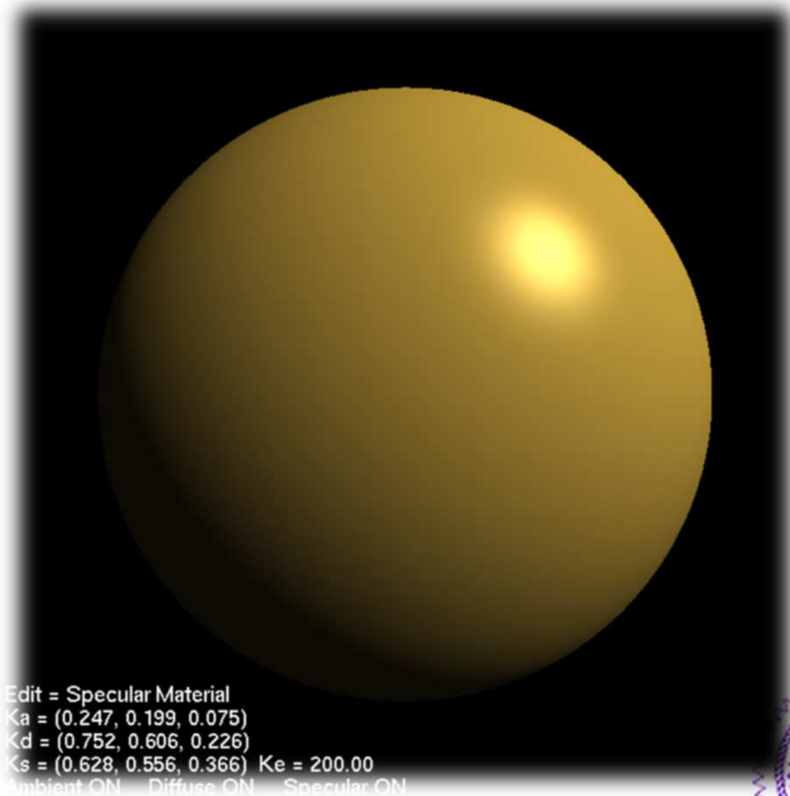
$$I = I_a k_a + f_{att} I_p k_d (N \bullet L) \qquad f_{att} = \min\left(\frac{1}{c_1 + c_2 d_L + c_3 d_L^2}, 1\right)$$



Specular Highlight

- Specular highlight is the bright spot regions **observed** on the object being illuminated by light sources





Specular Highlight

$$I_s = I_p k_s \cos^n \alpha = I_p k_s (R_p \cdot V)^n$$

$$I = I_a k_a + f_p I_p (k_d (N \cdot L_p) + k_s (R_p \cdot V)^n)$$

I: Intensity of final illumination

I_a: Intensity of ambient light

k_a: Ambient reflection coefficient

f_p: Attenuation function of point light source *p*

k_d: Diffuse reflection coefficient

N: Normalized normal vector

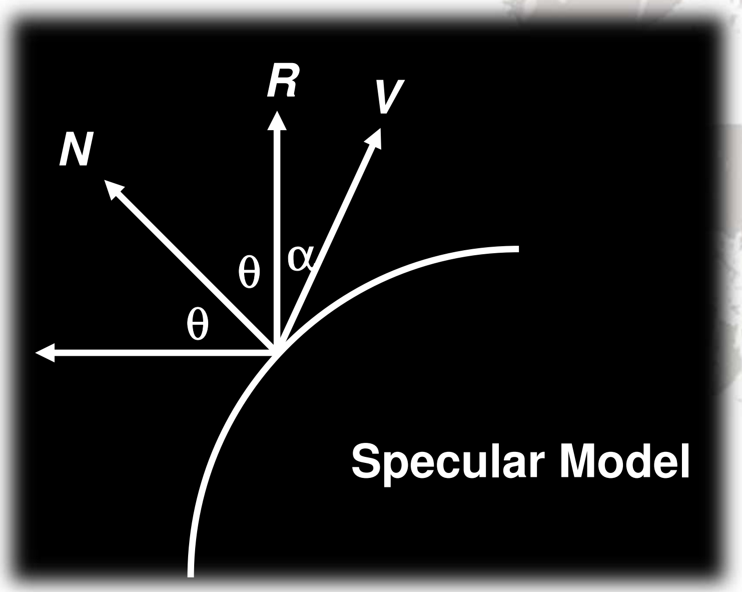
L_p: Normalized light source direction of point light source *p*

k_s: Specular reflection coefficient

R_p: Normalized light source reflection vector of point light source *p*

V: Normalized viewpoint direction vector

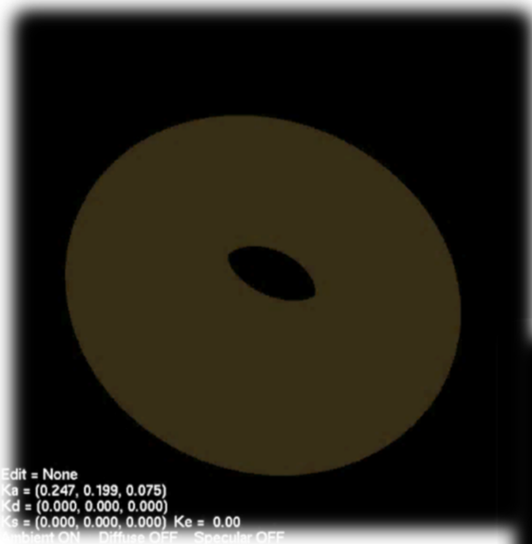
n: Material's specular reflection exponent



Specular Highlight

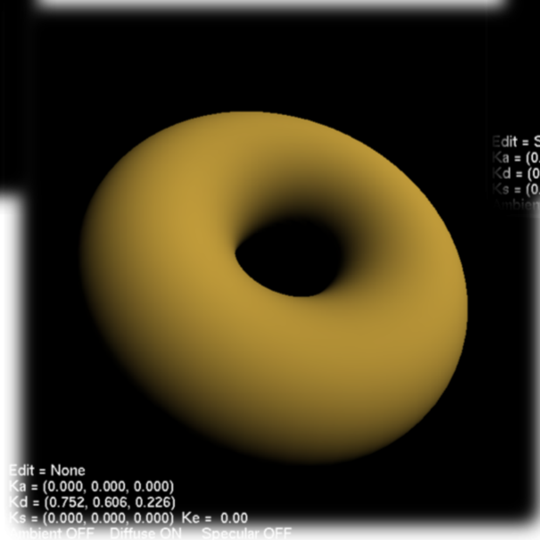
◆ Phong Reflection Model

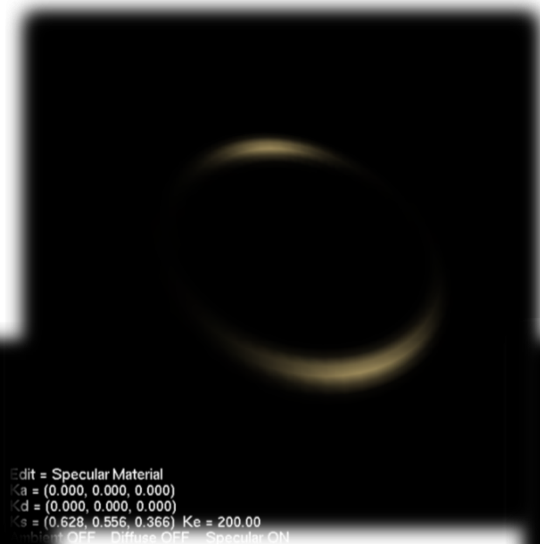
$$I = \boxed{I_a k_a} + f_p I_p \left(\boxed{k_d (N \cdot L_p)} + \boxed{k_s (R_p \cdot V)^n}\right)$$



Ambient

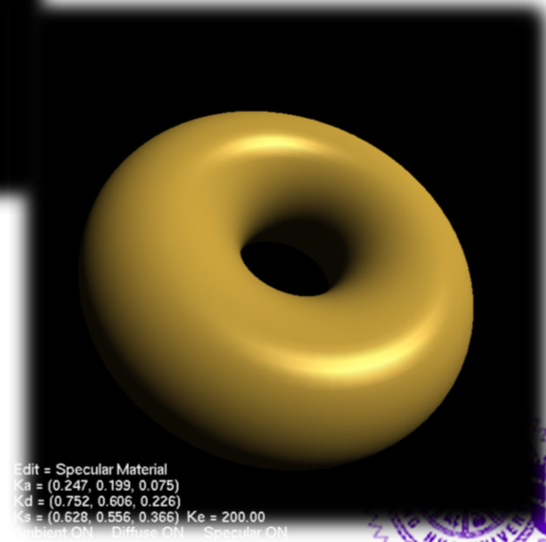
Diffuse





Specular

Final illumination



Modified Phong Reflection Model

- Also called Blinn-Phong Reflection Model
- Original Phong model requires to calculate the reflection vector and view vector for each point
- Blinn suggested to use an approximated way to calculated the specular reflection term by introducing the halfway vector

$$I_s = I_p k_s \cos^n \alpha = I_p k_s (\underbrace{H_p}_{\text{Normalized halfway vector}} \cdot \underbrace{N}_{\text{Normalized normal vector}})^{\boxed{n'}}$$

The Halfway Vector

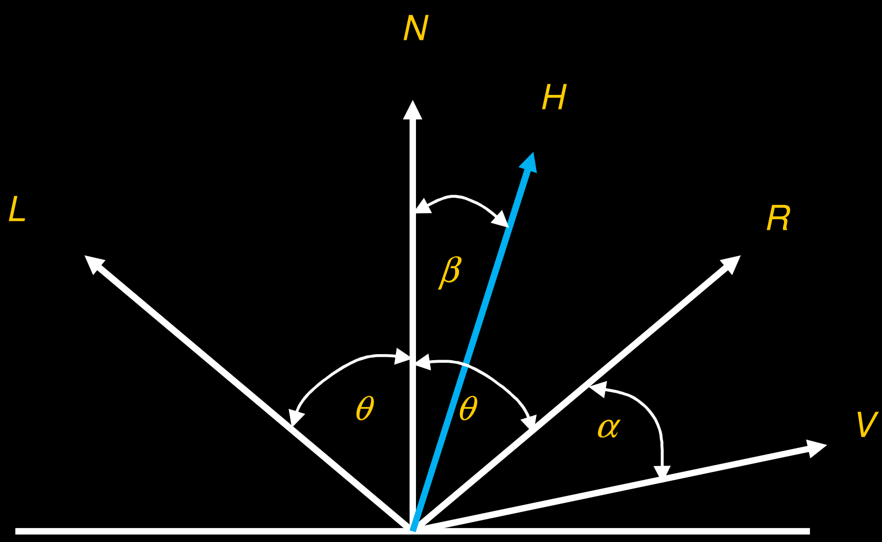
$$I_s = I_p k_s \cos^n \alpha = I_p k_s (R_p \cdot V)^n$$
$$\approx I_p k_s (H_p \cdot N)^{n'}$$

- The halfway vector is the normalized vector halfway between the viewpoint and the light vector

$$H = \frac{L+V}{|L+V|}$$

$$\theta + \beta = \theta - \beta + \alpha$$

$$\beta = \frac{1}{2}\alpha$$



Multiple Light Sources

$$I = I_a k_a + \sum_{p=1}^{m} f_p I_p (k_d (N \cdot L_p) + \underline{k_s (N \cdot H_p)^{n'}})$$

I: Intensity of final illumination

I_a: Intensity of ambient light

k_a: Ambient reflection coefficient

f_p: Attenuation function of point light source *p*

k_d: Diffuse reflection coefficient

N: Normalized normal vector

L_p: Normalized light source direction of point light source *p*

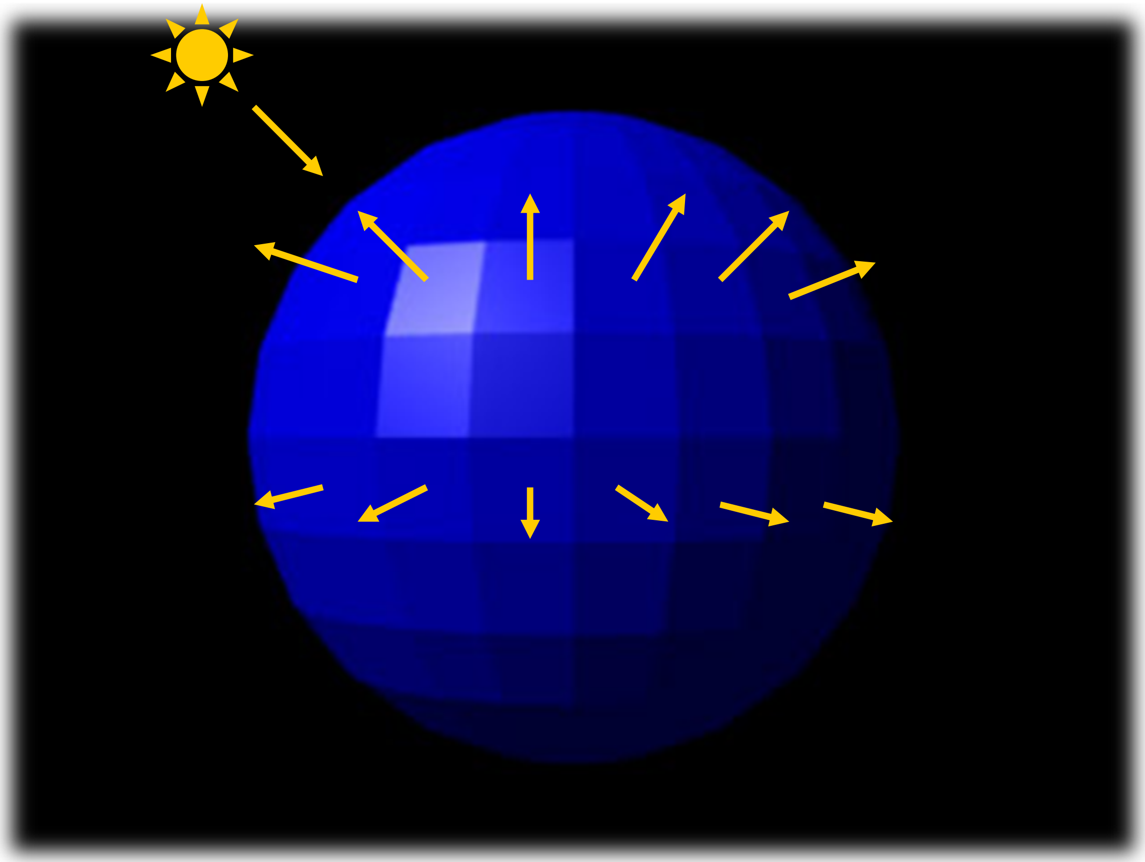
k_s: Specular reflection coefficient

H_p: Normalized half vector between viewpoint and the point light source *p*

n': Material's specular reflection exponent (Blinn-Phong Reflection Model)

Flat Shading

- Using face normal to derive polygon color

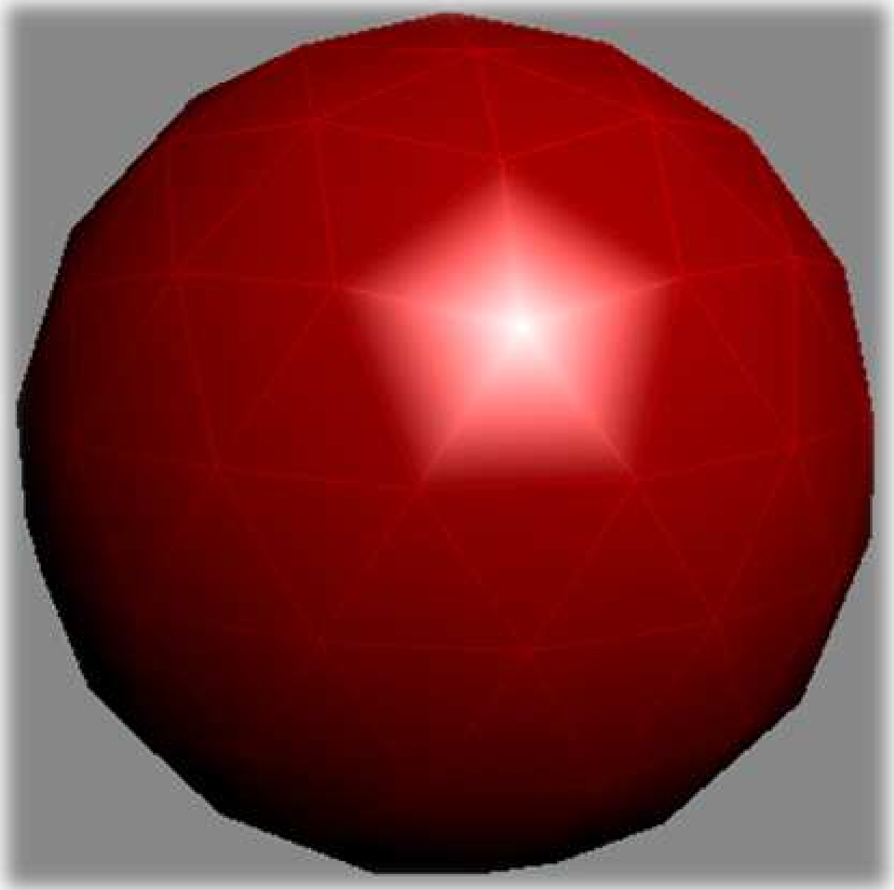


Smooth Shading

- The colors for the interior of the polygon are interpolated between vertex colors

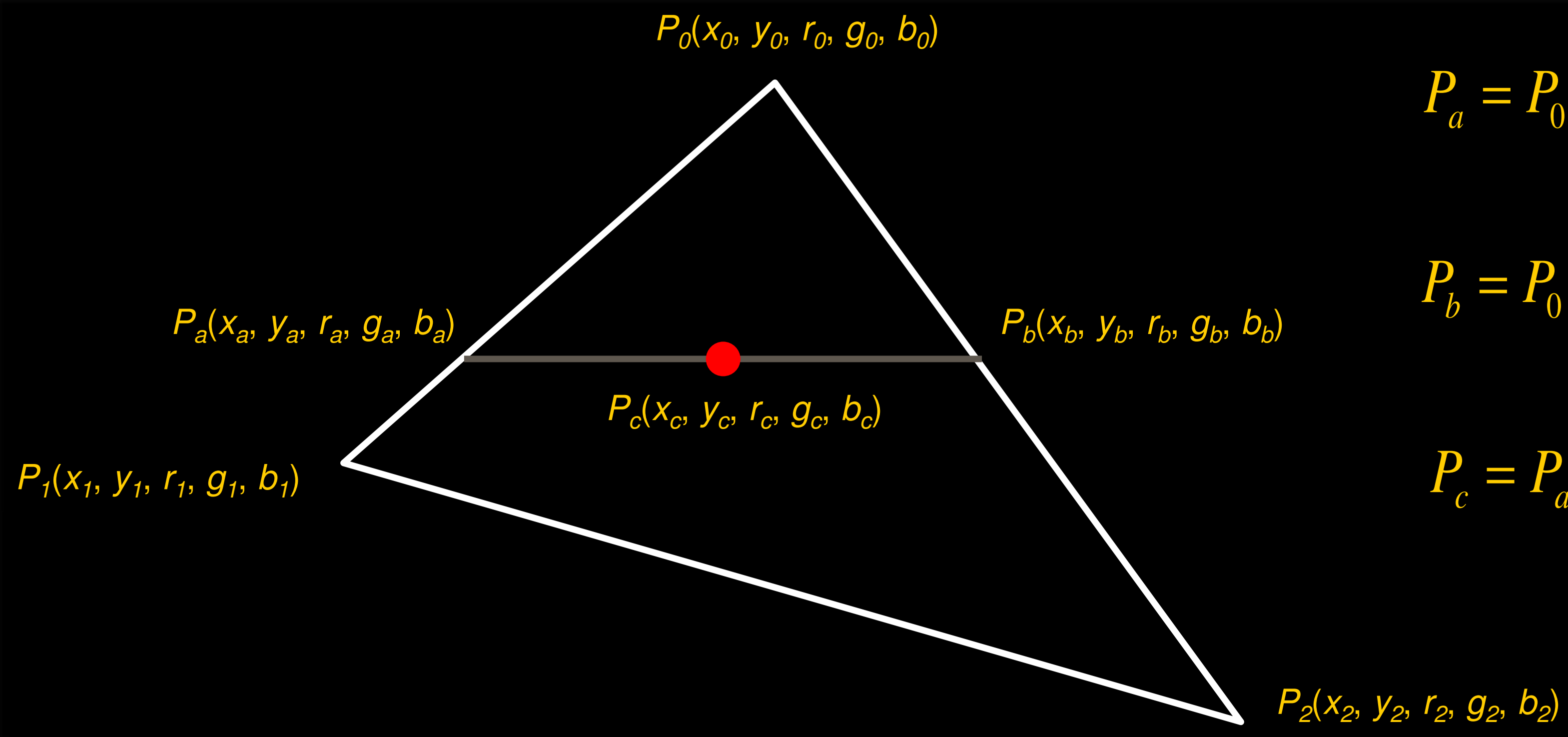
Smooth Shading

- Gouraud Shading



Smooth Shading

- Gouraud Shading
  - Compute colors for each vertices respectively
  - Interpolate pixel colors through vertex colors



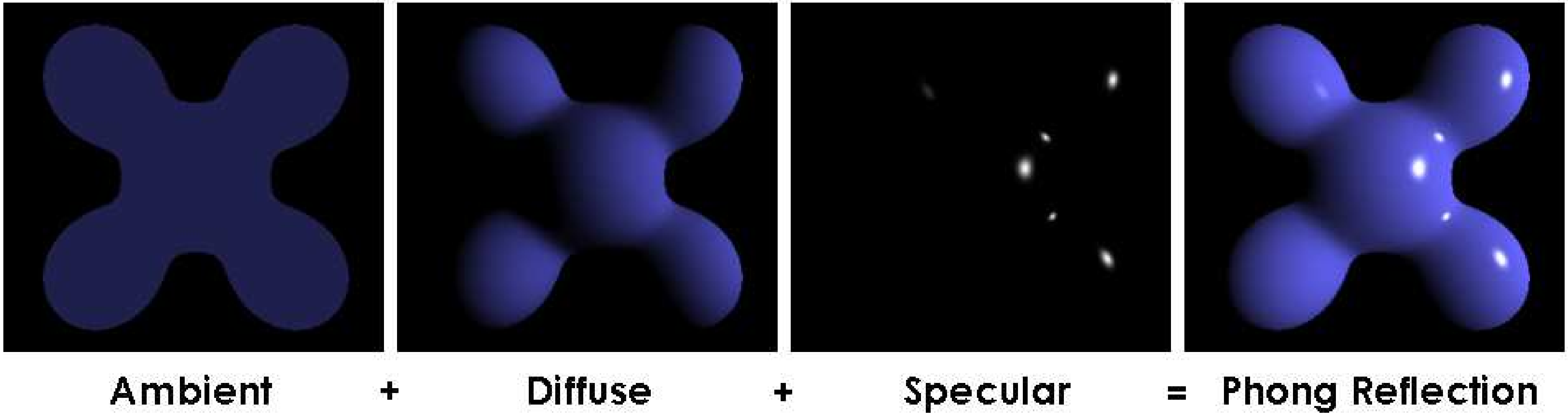
$$P_a = P_0\left(1 - \frac{y_a - y_0}{y_1 - y_0}\right) + P_1 \frac{y_a - y_0}{y_1 - y_0}$$

$$P_b = P_0\left(1 - \frac{y_b - y_0}{y_2 - y_0}\right) + P_2 \frac{y_b - y_0}{y_2 - y_0}$$

$$P_c = P_a\left(1 - \frac{x_c - x_a}{x_b - x_a}\right) + P_b \frac{x_c - x_a}{x_b - x_a}$$

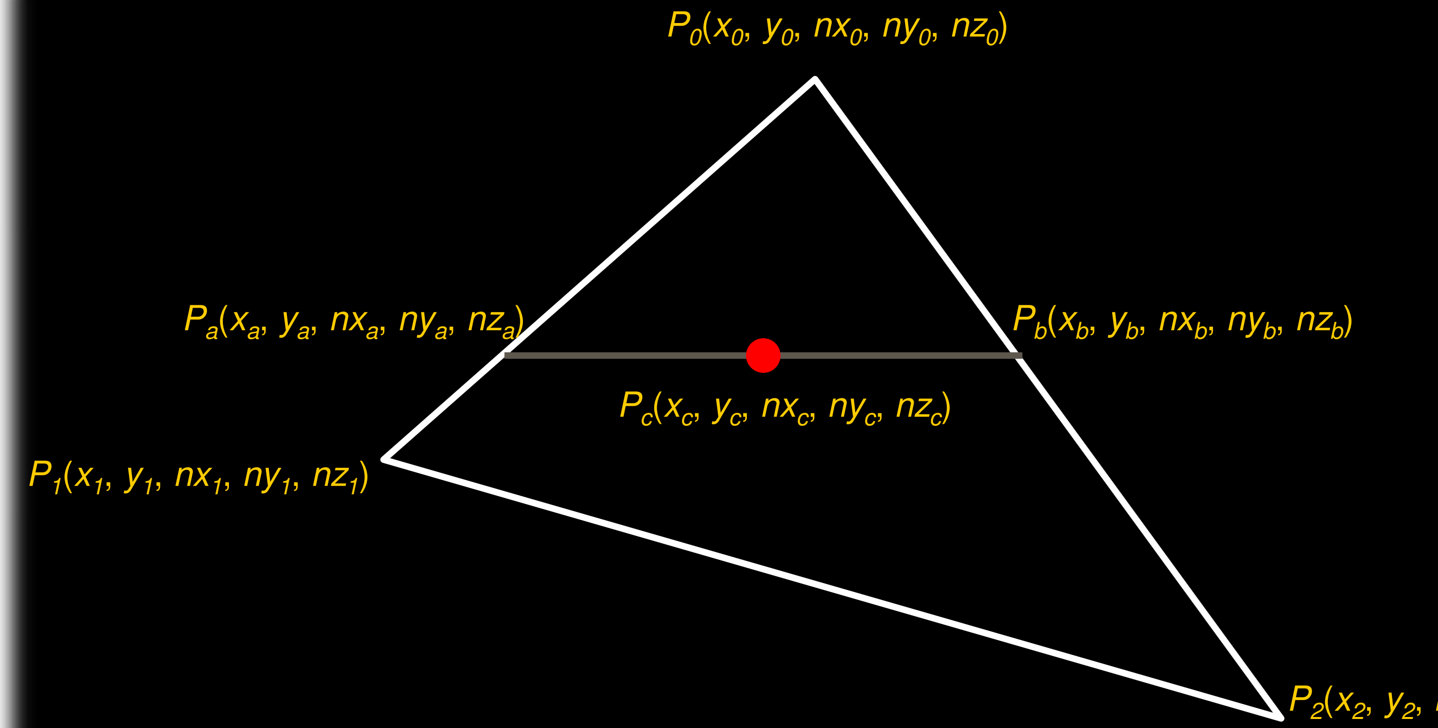
Smooth Shading

- Phong Shading



Smooth Shading

- **Phong Shading**
  - **Interpolate pixel normal through vertex normals**
  - **Compute pixel color with derived pixel normal**



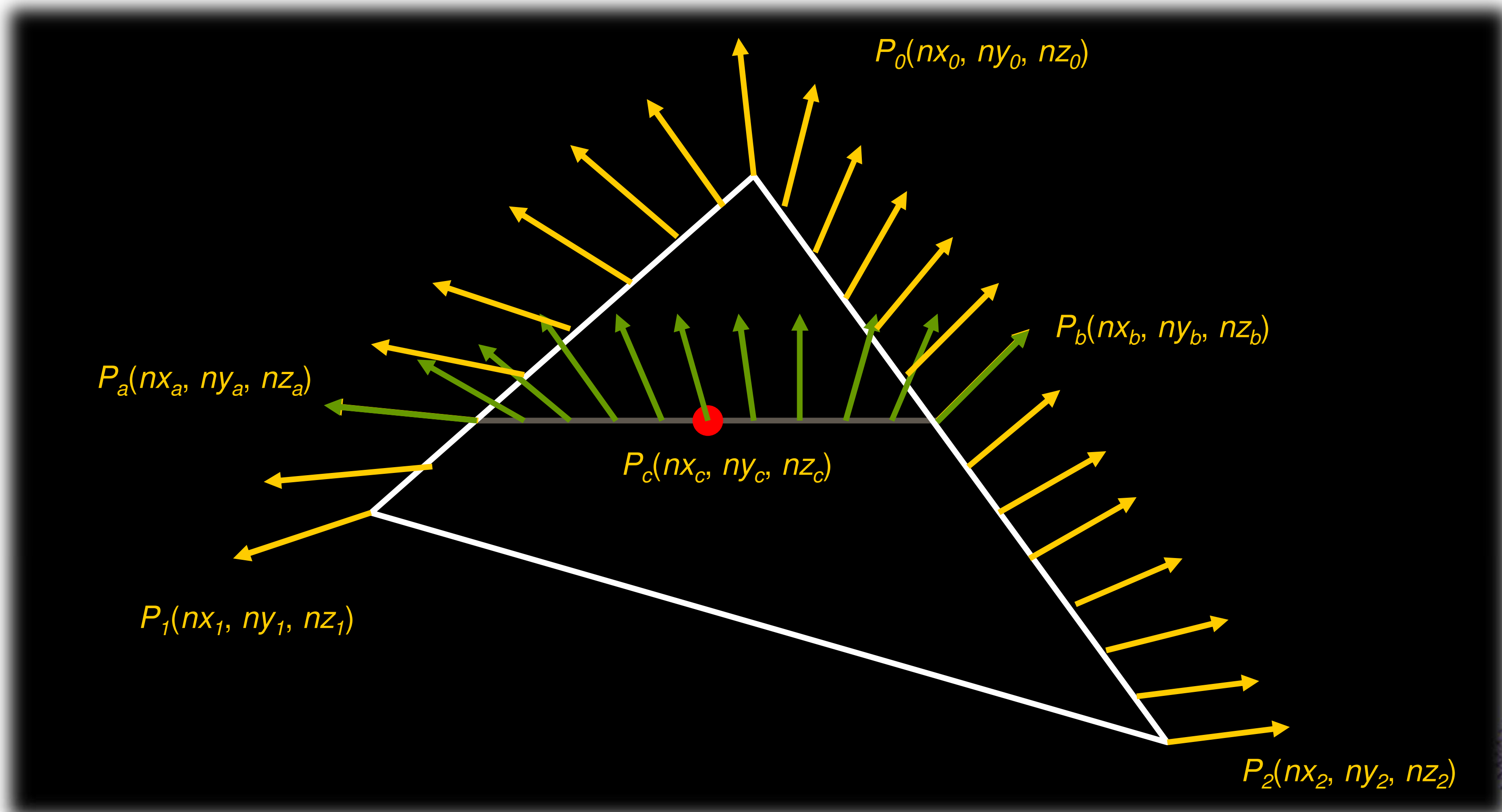
$$P_a = P_0\left(1-\frac{y_a-y_0}{y_1-y_0}\right)+P_1\frac{y_a-y_0}{y_1-y_0}$$

$$P_b = P_0\left(1-\frac{y_b-y_0}{y_2-y_0}\right)+P_2\frac{y_b-y_0}{y_2-y_0}$$

$$P_c = P_a\left(1-\frac{x_c-x_a}{x_b-x_a}\right)+P_b\frac{x_c-x_a}{x_b-x_a}$$

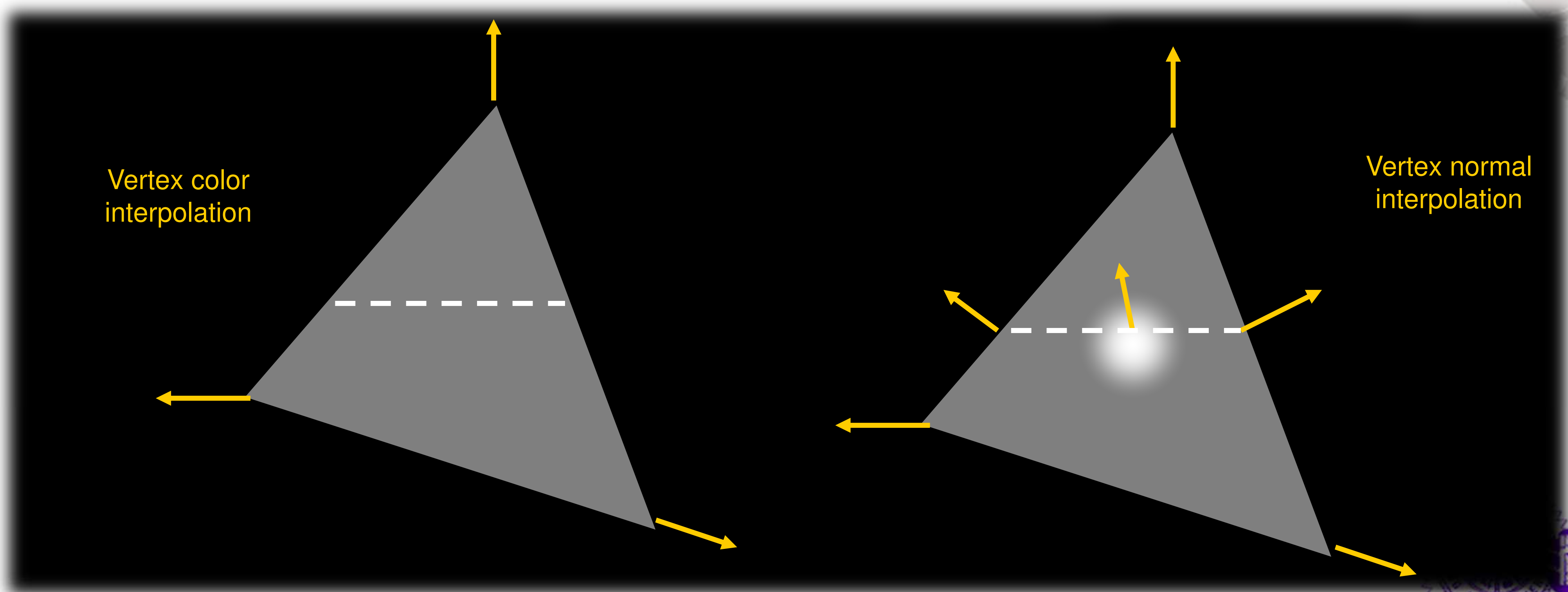
Smooth Shading

- **Phong Shading**
  - **Interpolate pixel normal through vertex normals**



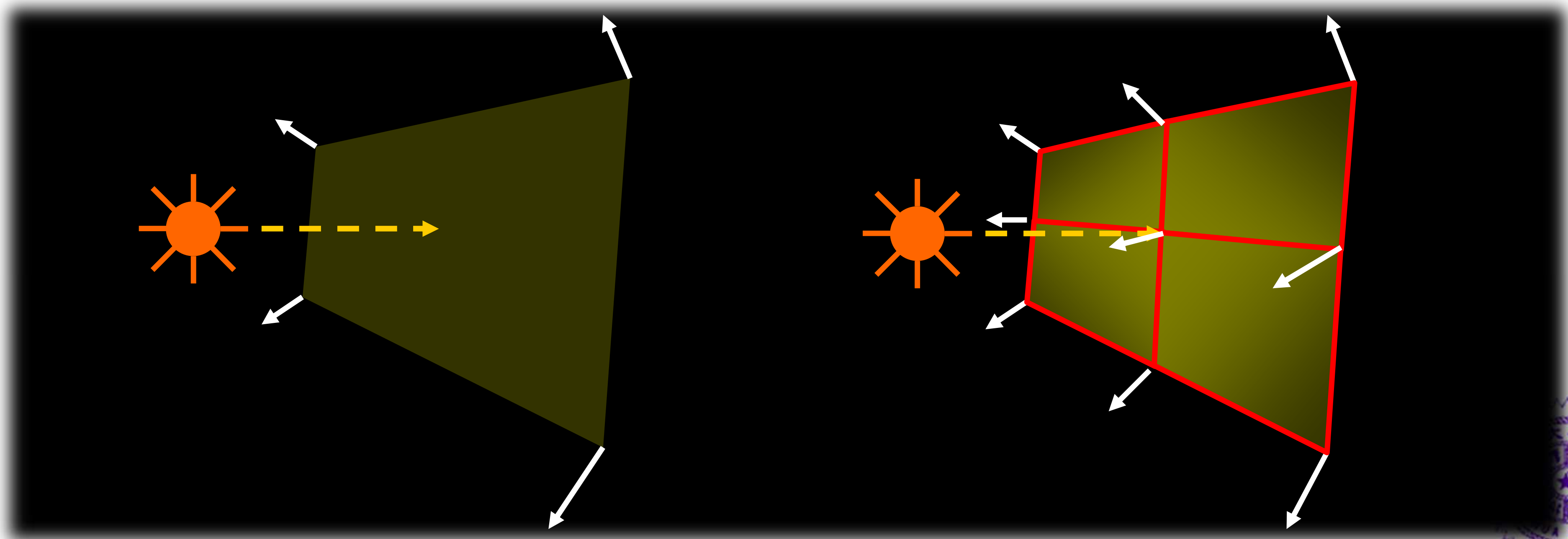
Smooth Shading

- Specular highlight issue in Gouraud shading
  - If specular highlight is not on the vertices but occurs inside the triangle, then the Gouraud shading cannot reveal such kind of highlight



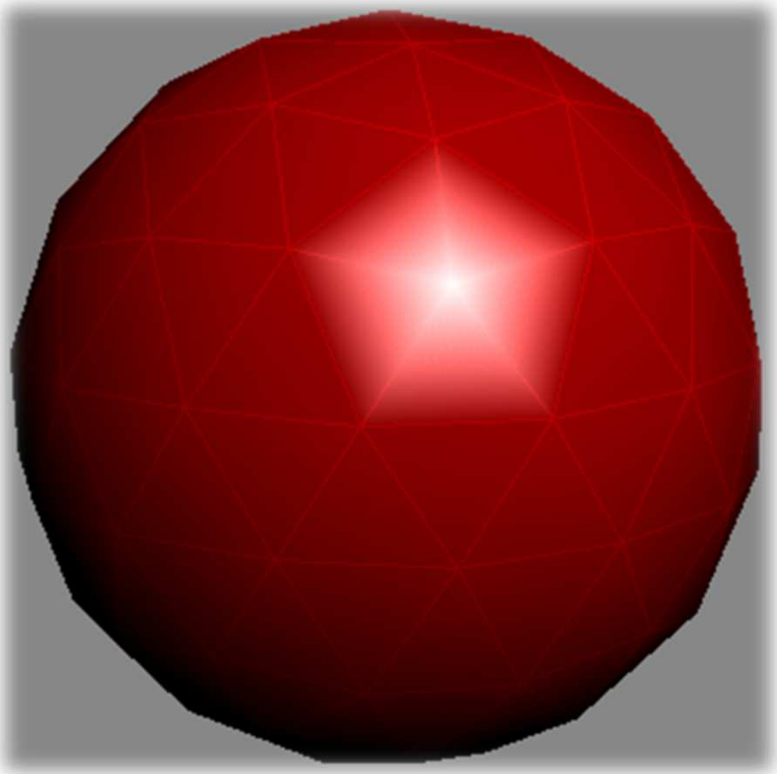
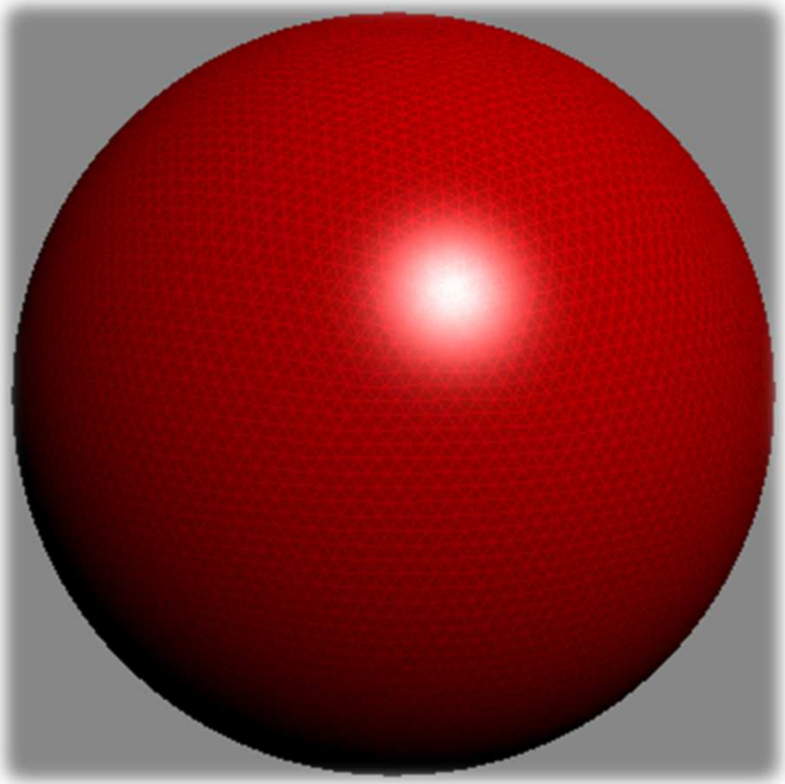
Polygon Size Mattered

- Large polygon size might degrade the lighting quality
  - Darker if the light source is closer to the polygon
  - No specular highlight can be perceived in the middle of a polygon



Polygon Size Mattered

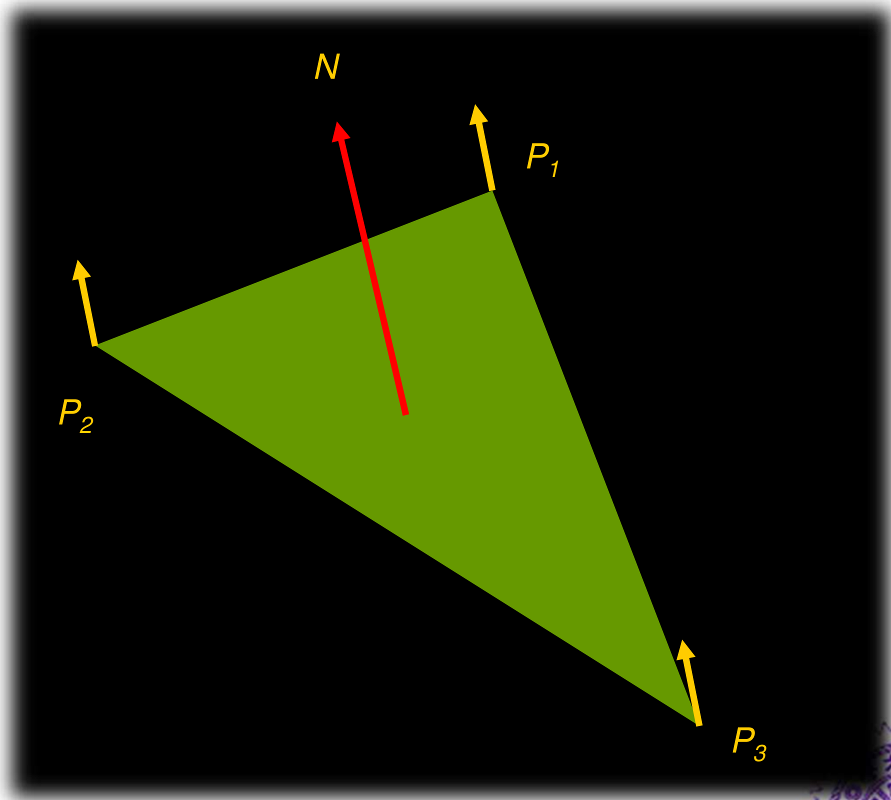
- Subdivide large polygon into smaller polygons to gain better shading result

Vertex Normal Derivation

- Flat shading
  - Vertex normal is equal to polygon face normal

$$N = (P_2 - P_1) \times (P_3 - P_1)$$

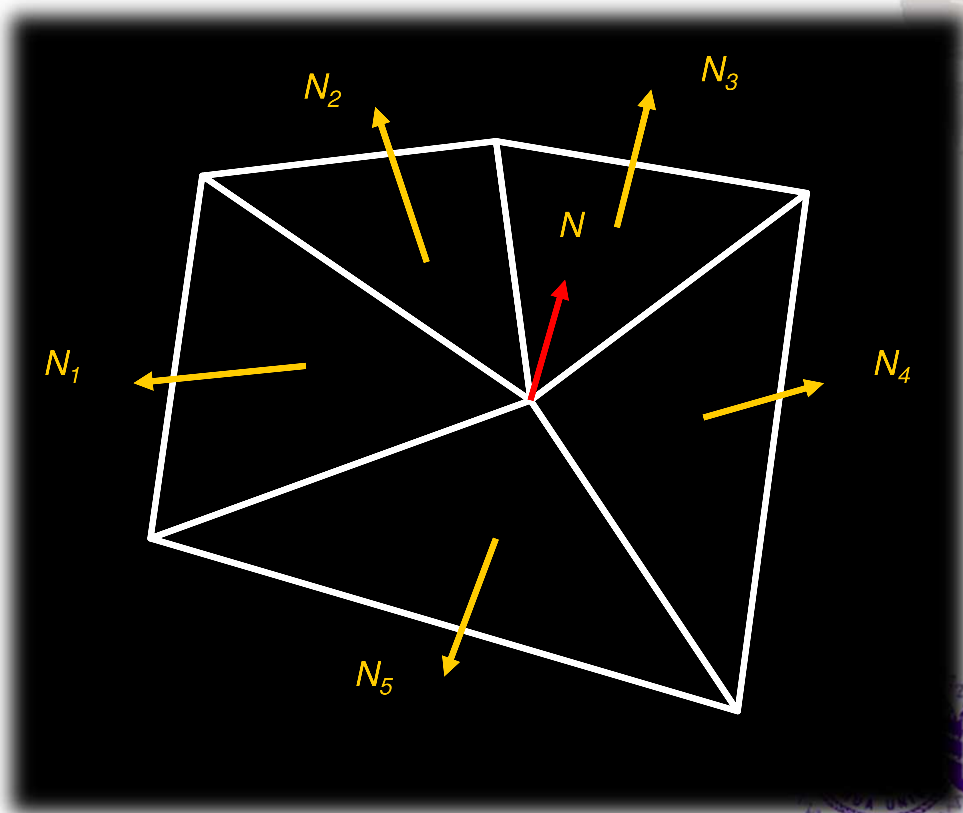


Vertex Normal Derivation

- Smooth shading
  - Vertex normal is equal to the sum of polygon face normals of adjacent polygons

$$N_{sum} = (N_1 + N_2 + \ldots + N_m)$$

$$N = N_{sum} / |N_{sum}|$$

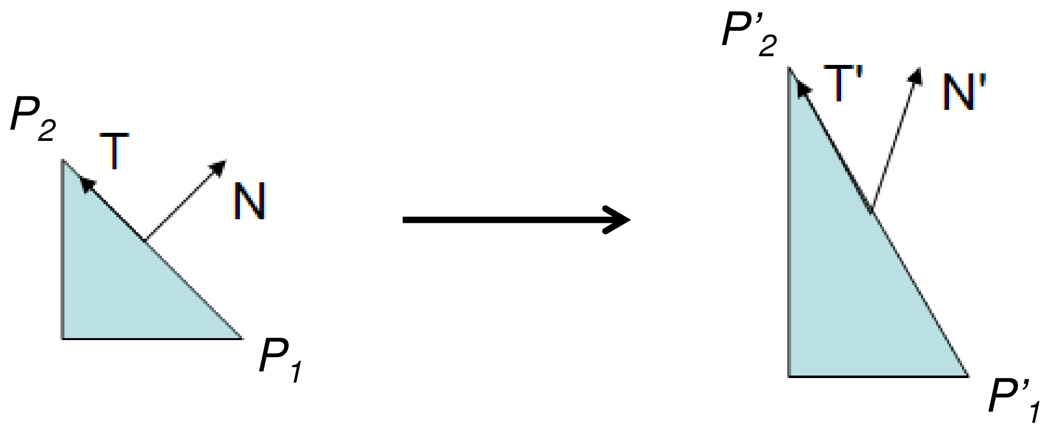


Normal Transformation

- Model transform and viewing transform together can transform a vertex from object space to eye space
- Normal is related to lighting process and the lighting calculation is performed in eye space
- So, normal has to transform to eye space as well
- But, what will happen if we transform normal using the same model-view matrix?

Normal Transformation

- If we transform normal using model-view matrix M, then…



$T = P_2 - P_1$
$M \cdot T = M \cdot (P_2 - P_1)$
$T' = M \cdot P_2 - M \cdot P_1 = P'_2 - P'_1$

$N = Q_2 - Q_1$
$M \cdot N = M \cdot (Q_2 - Q_1)$
$N' = M \cdot Q_2 - M \cdot Q_1 = Q'_2 - Q'_1$

$N \cdot T = 0$
But, after normal transformed by model-view matrix M
$N' \cdot T' \neq 0$

Normal Transformation

- **Normal transformation should be taking care if you have done any model and viewing transformations to the geometric data**

A vector in homogeneous coordinate is represented as
$T = (x, y, z, 0) = (x_2, y_2, z_2, 1) – (x_1, y_1, z_1, 1)$

A normal in homogeneous coordinate is represented as $N = (n_x, n_y, n_z, 0)$

Since T and N are orthogonal, thus $N \cdot T = 0$
We also know that after model-view transformation, T' and N' should remain orthogonal.
That is, $N' \cdot T' = 0$.

Normal Transformation

- Represent the dot product by matrix multiplication and let *M* be the model-view matrix, we have

$$N \cdot T = \begin{pmatrix} n_x & n_y & n_z & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 0 \end{pmatrix} = 0$$

Eye space normal

Object space normal

Normal Transformation

$$N' \cdot T' = \underbrace{\boxed{\begin{pmatrix} n_x & n_y & n_z & 0 \end{pmatrix} M^{-1}}}_{\text{Normal Transformation}} \underbrace{\boxed{M \begin{pmatrix} x \\ y \\ z \\ 0 \end{pmatrix}}}_{\text{Model-view Transformation}} = 0 \Rightarrow \begin{pmatrix} n'_x \\ n'_y \\ n'_z \\ 0 \end{pmatrix} = (M^{-1})^T \begin{pmatrix} n_x \\ n_y \\ n_z \\ 0 \end{pmatrix}$$

Model-view Transformation

Lighting Procedure

- Define the vertex normals
  - Lighting is the interaction between vertex normals and the light source
- Define light sources
  - Light source properties
- Select lighting model
  - Determine which lighting equation is used
- Define material properties
  - Define the percentage of reflectance to the light source

Complete OpenGL Lighting Formula

Object can emit light itself

Global ambient light

Light source contribution

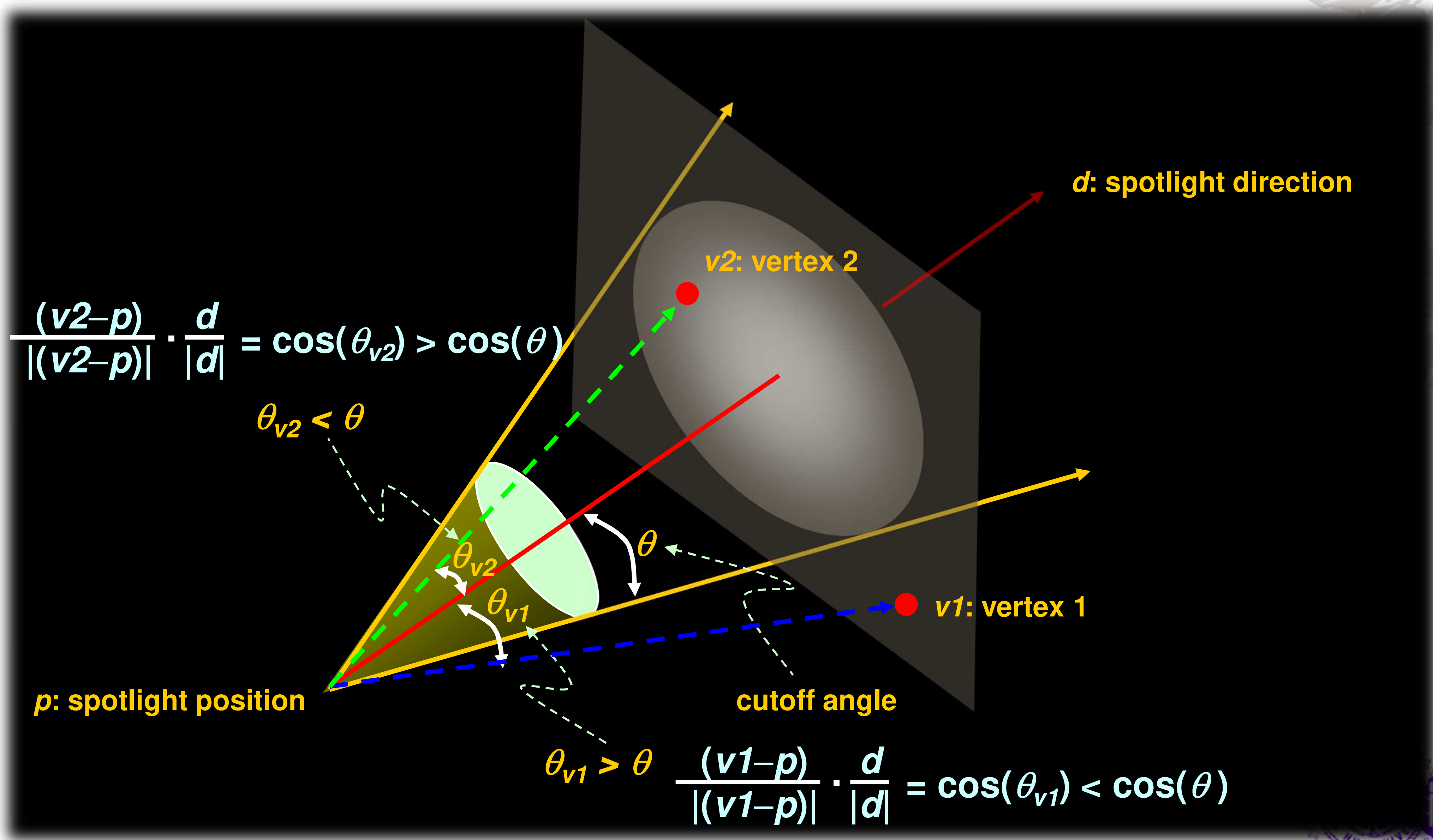
$$\text{vertex color} = \boxed{\text{emission}_{\text{material}}} +$$

$$\boxed{\text{ambient}_{\text{light model}} * \text{ambient}_{\text{material}}} +$$

$$\boxed{\begin{aligned} &\sum_{i=0}^{n-1} \left(\frac{1}{k_c + k_l d + k_q d^2} \right)_i * (\text{spotlight effect})_i * \\ &[\text{ambient}_{\text{light}} * \text{ambient}_{\text{material}} + \\ &(\max\{\mathbf{L} \cdot \mathbf{n}, 0\}) * \text{diffuse}_{\text{light}} * \text{diffuse}_{\text{material}} + \\ &(\max\{\mathbf{s} \cdot \mathbf{n}, 0\})^{\text{shininess}} * \text{specular}_{\text{light}} * \text{specular}_{\text{material}}]_i \end{aligned}}$$

Reference: http://what-when-how.com/opengl-programming-guide/the-mathematics-of-lighting-opengl-programming/

Spotlight Effect



Spotlight Effect

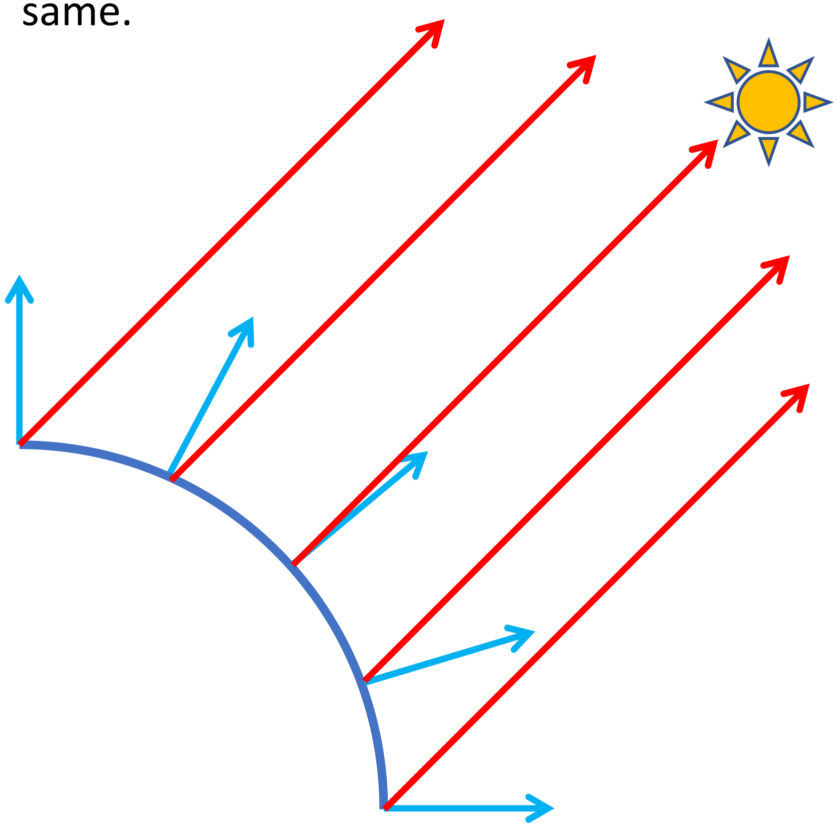
- Spotlight Effect =
  - 1, if the light source is not a spotlight
  - 0, if the light source is a spotlight but the vertex lies outside the cone of illumination produced by the spotlight
  - Otherwise, spotlight effect = $(\max\{v \cdot d, 0\})^{\text{spot\_exp}}$
    - v is the unit vector from the spotlight to the vertex
    - d is the spotlight direction

Q&A

Difference between Directional Light and Positional (Point) Light

Directional Light:

- Indicated by a vector (x, y, z) to the light source
- No need to calculate the light vector between each vertex and light source. They are all the same.



Positional Light (point light):

- Indicated by a position (x, y, z) of the light source
- Need to calculate the light vector between each vertex and light source. They are not all the same.