Advanced Computer Graphics

Lecture-07 Shading and Lighting

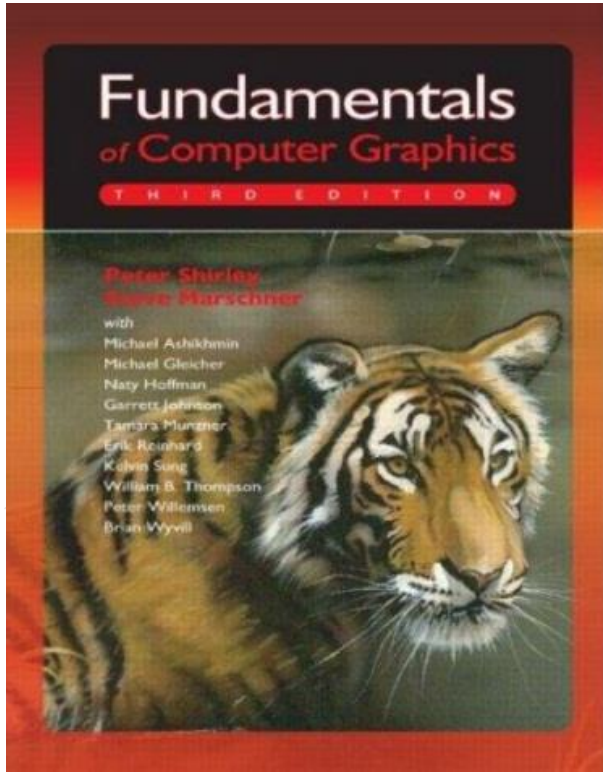
Tzung-Han Lin
National Taiwan University of Science and Technology
Graduate Institute of Color and Illumination Technology
e-mail: thl@mail.ntust.edu.tw

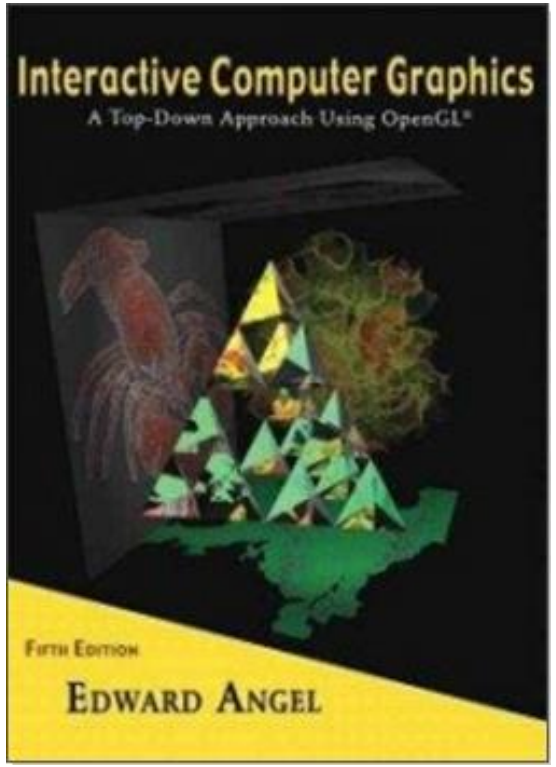
TAIWAN TECH National Taiwan University of Science and Technology

CIT 色彩與照明科技研究所 Graduate Institute of Color and Illumination Technology

Content in textbook

- Fundamentals of Computer Graphics, Chapter 9
- Interactive Computer Graphics, 5th edition, Chapter 6

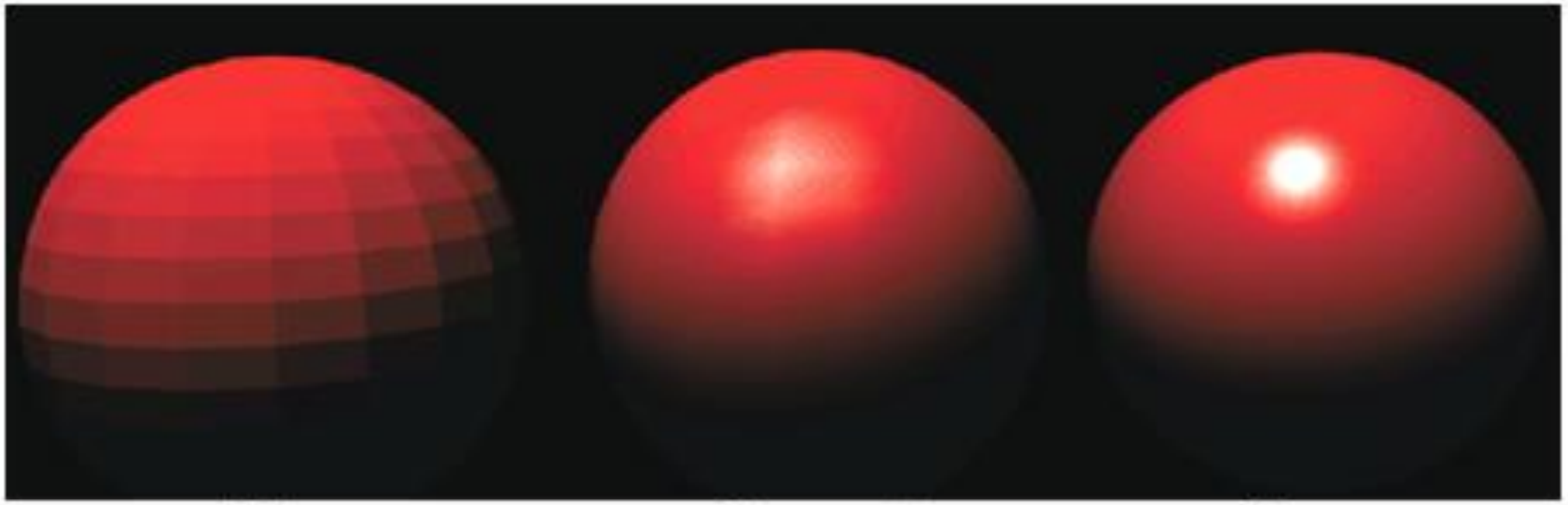




NTUST. CIIC.

Shading algorithm

- Flat shading
- Smooth shading
  - Gouraud Shading (by vertex)
  - Phong Shading (by fragment)

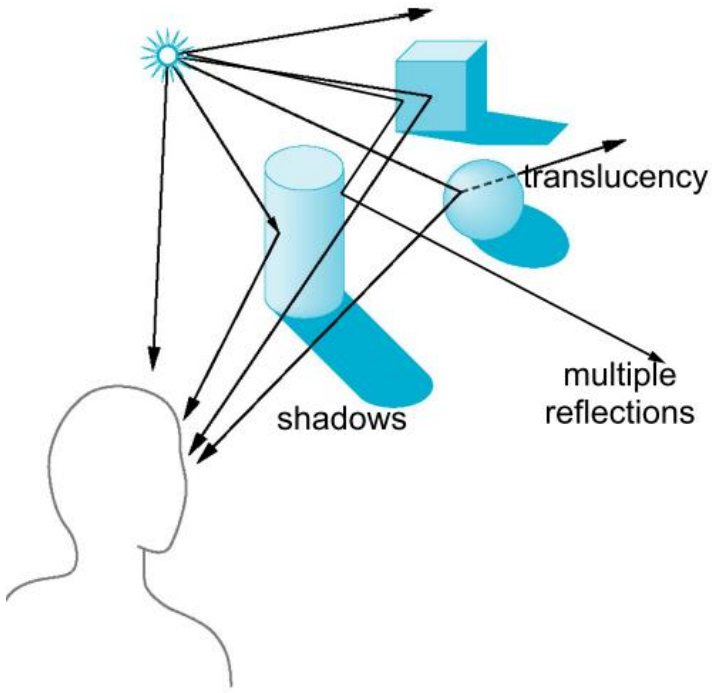


NTUST. CIIC.

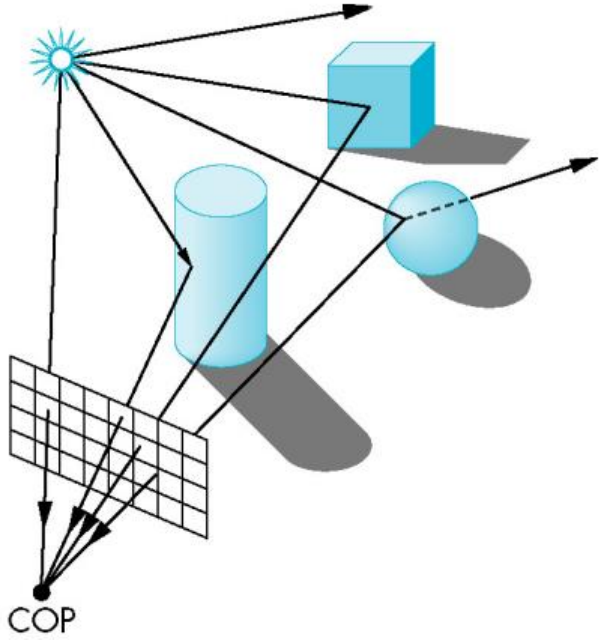
Real scene and graphics shading

NTUST. CIIC.

Real scene



Computer Graphics



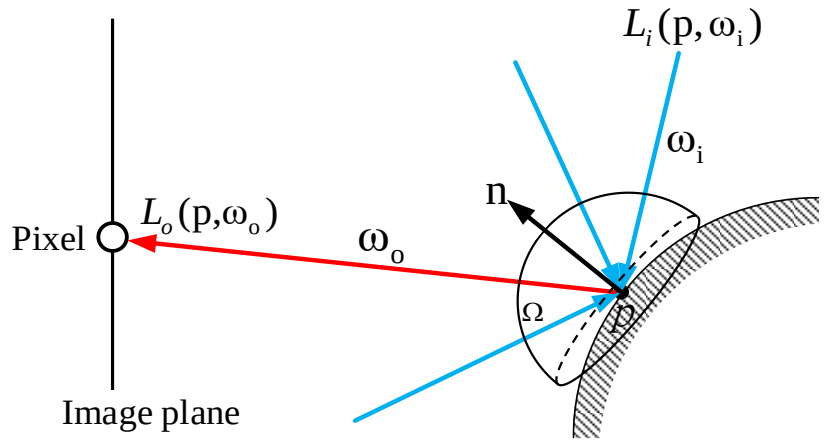
From: Interactive Computer Graphics

Physical based or not

- Physical based render the conservation of energy

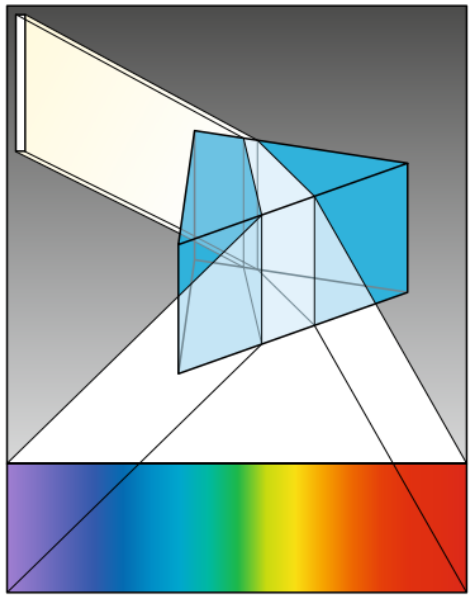
$$L_o(p,\omega_o) = \int_\Omega f_r(p,\omega_i,\omega_o) L_i(p,\omega_i) n \cdot \omega_i d\omega_i$$

$$L = \frac{d^2\Phi}{dA^\perp d\omega} = \frac{d^2\Phi}{dA d\omega \cos\theta}$$

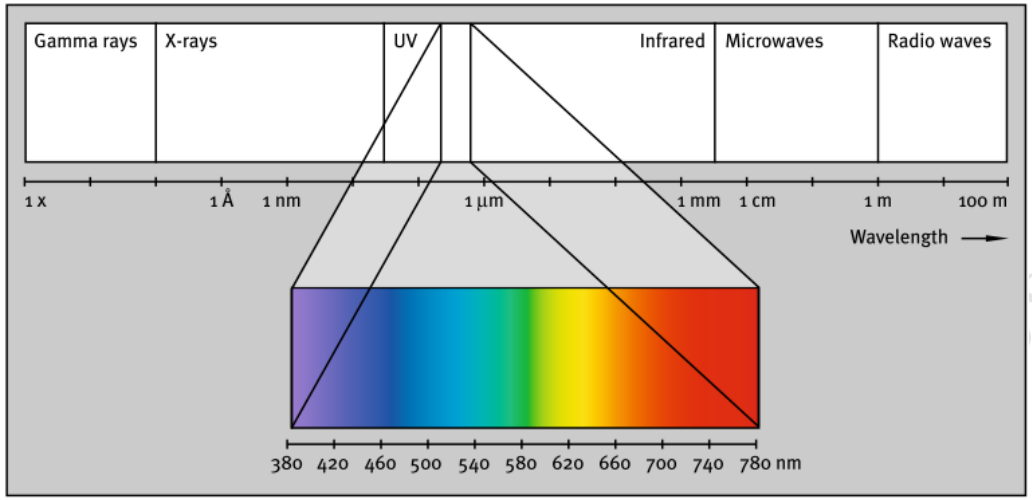


How “color” formed

- Without light we see nothing. This simple truth becomes more complex upon closer inspection, because light is not just light. Colloquially one talks of cold and warm light. The photographer differentiates between daylight and artificial light. In reprography there is standardized lighting for color matching artwork, print proofs and final runs. As light sets the basic conditions for the perception of color, this book begins with light. (statement from textbook)



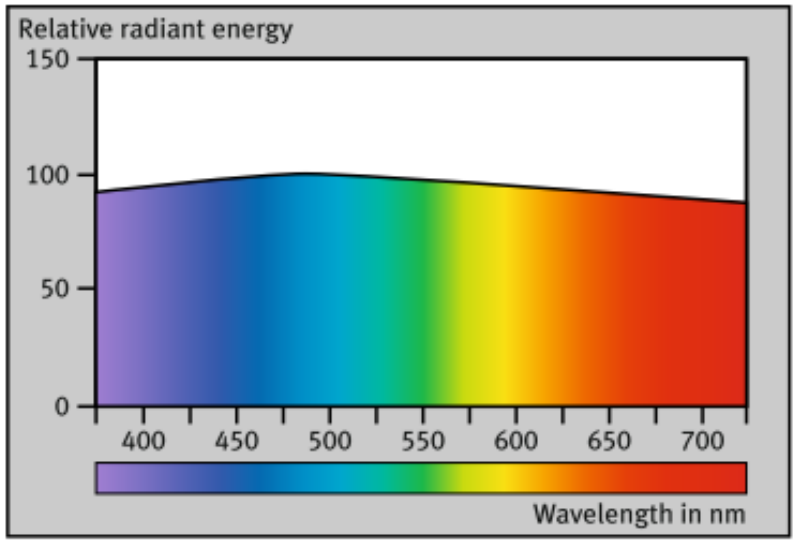
Refraction through a prism



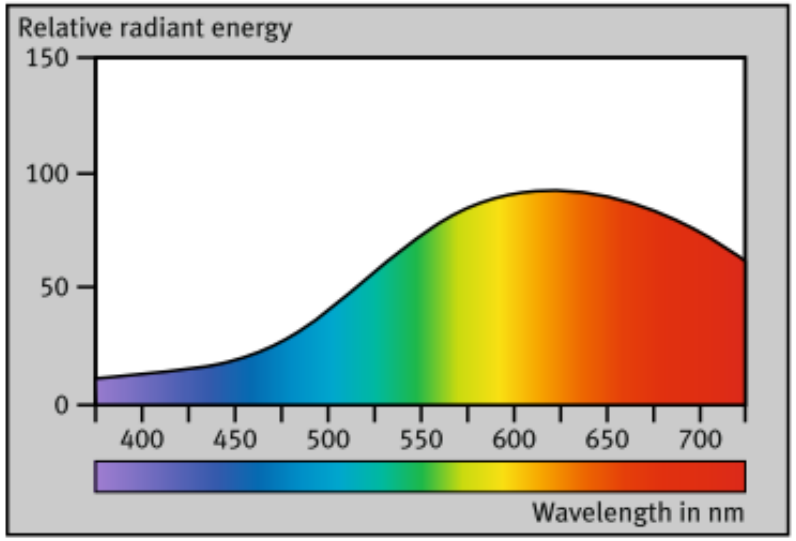
How “color” formed

- Radiant energy vs spectrum →color

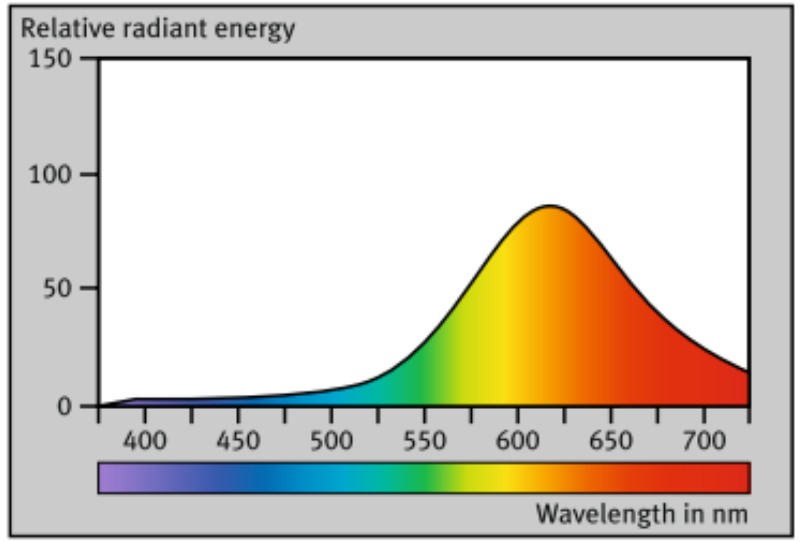
NTUST. CIIC.



Daylight’s spectrum

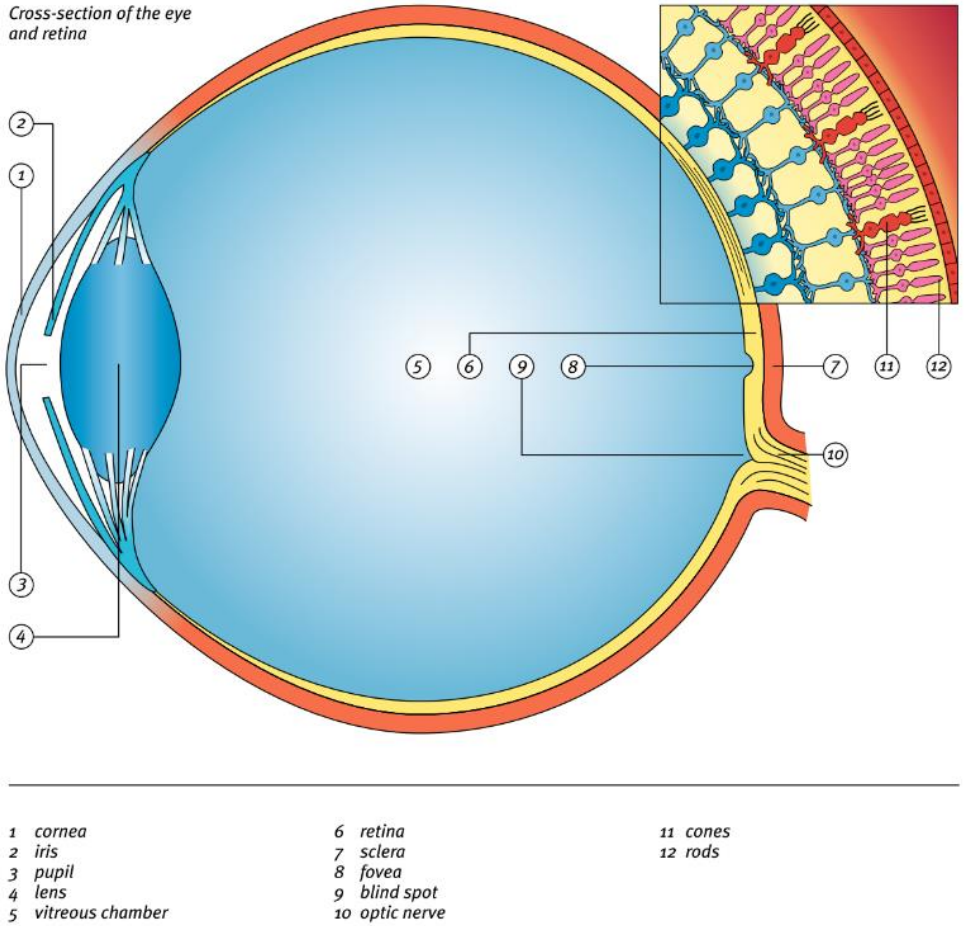


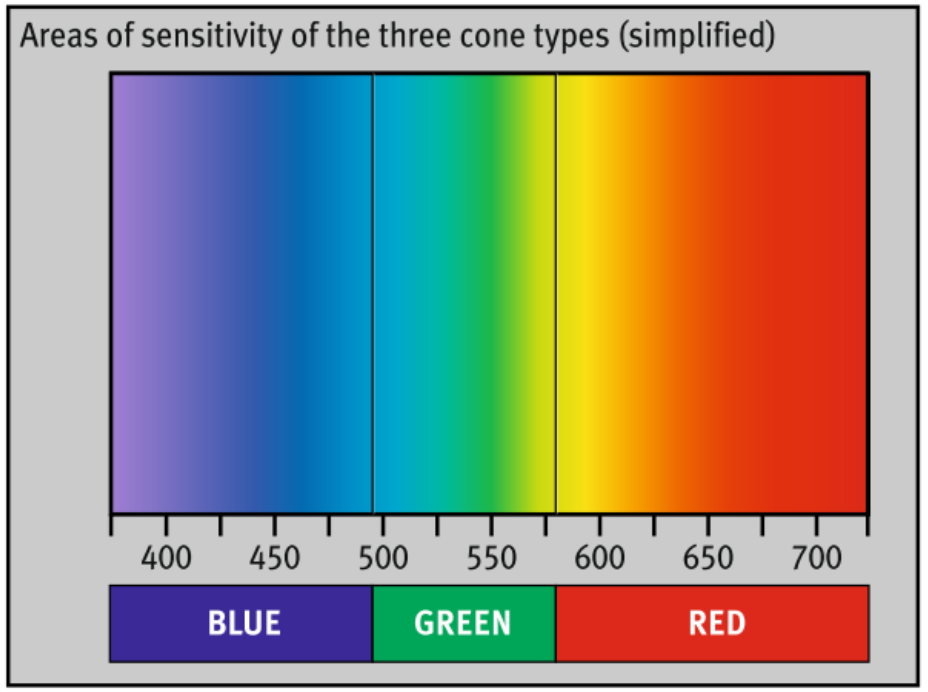
A light bulb’s spectrum



The spectrum of a red traffic light

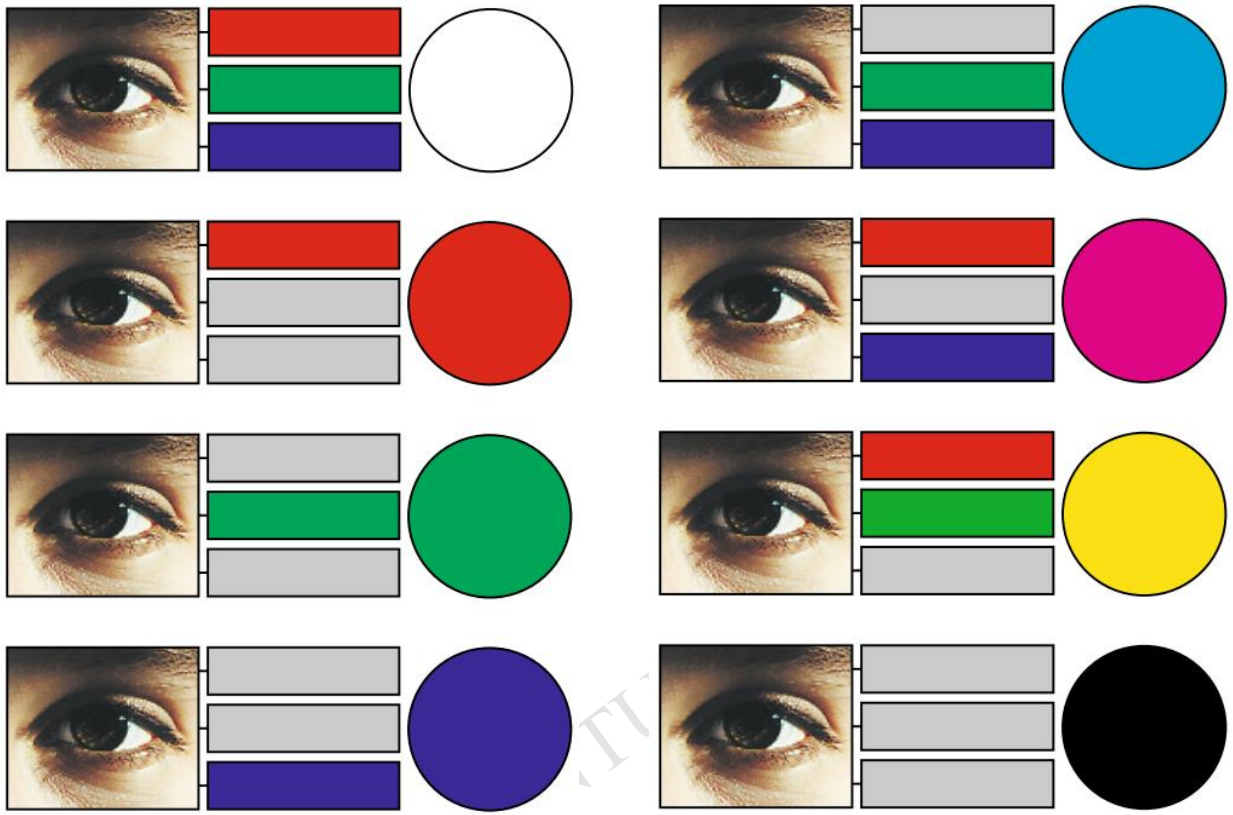
How “color” formed

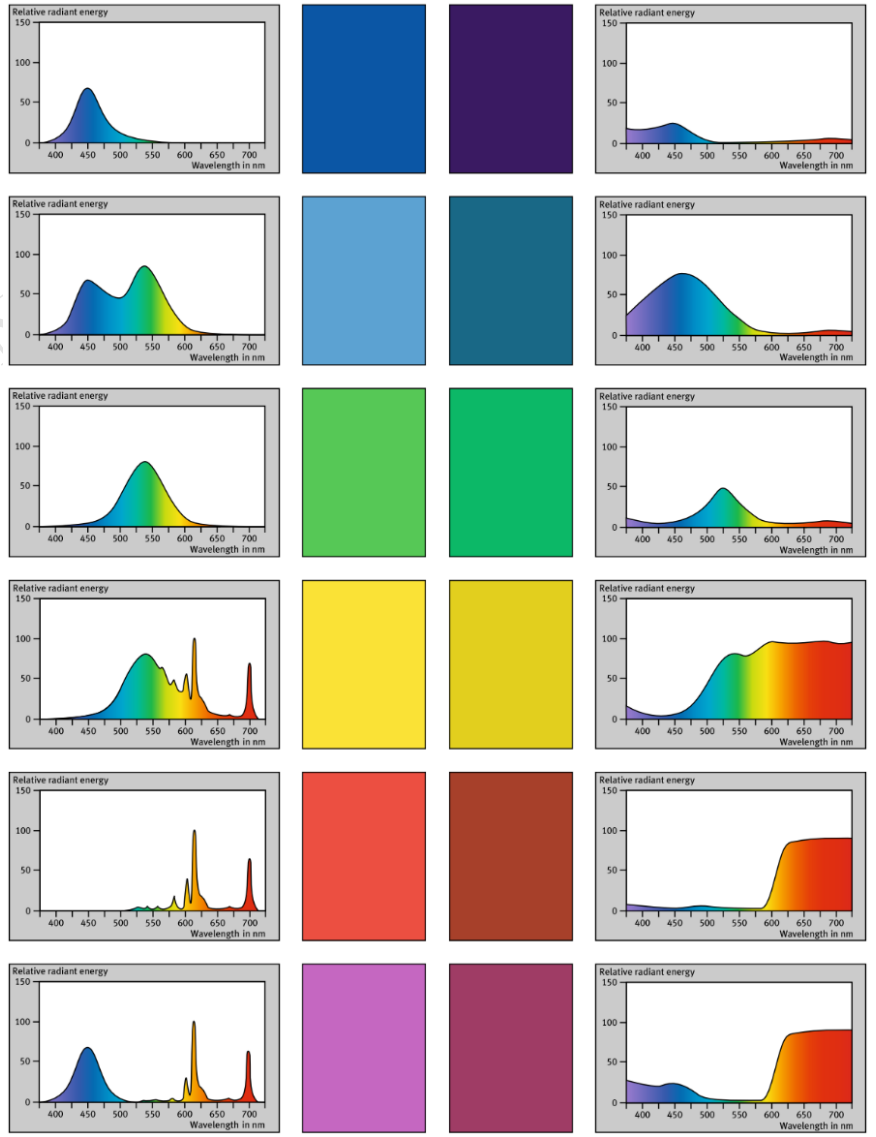




Three types of cone are sensitive to different areas of the spectrum

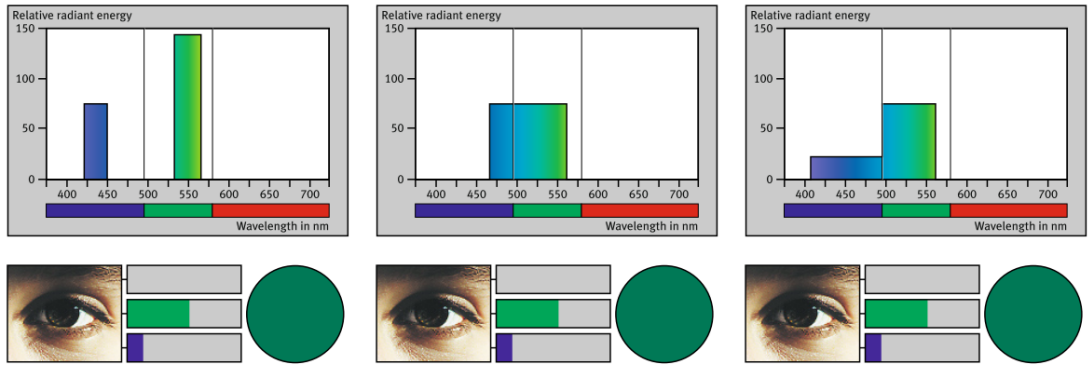
How “color” formed



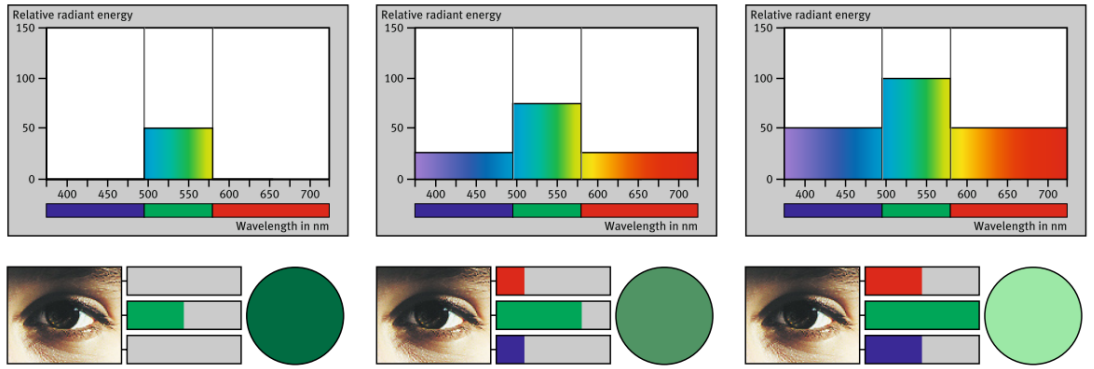


How “color” formed

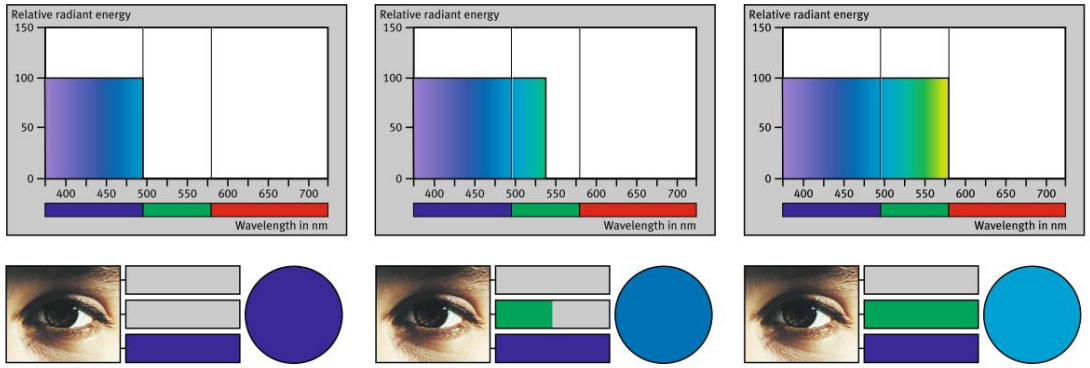
Different spectra can create the same impression of color in the eye



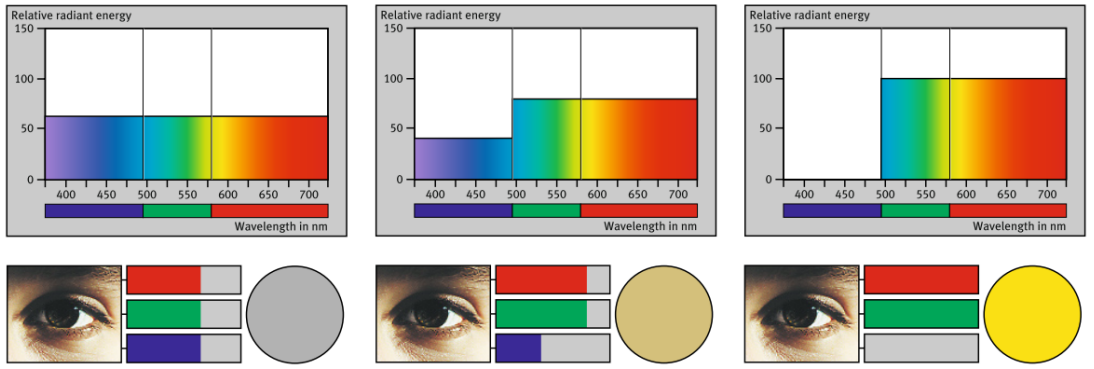
Different lightness



Different hue



Different saturation



Human vs Bee

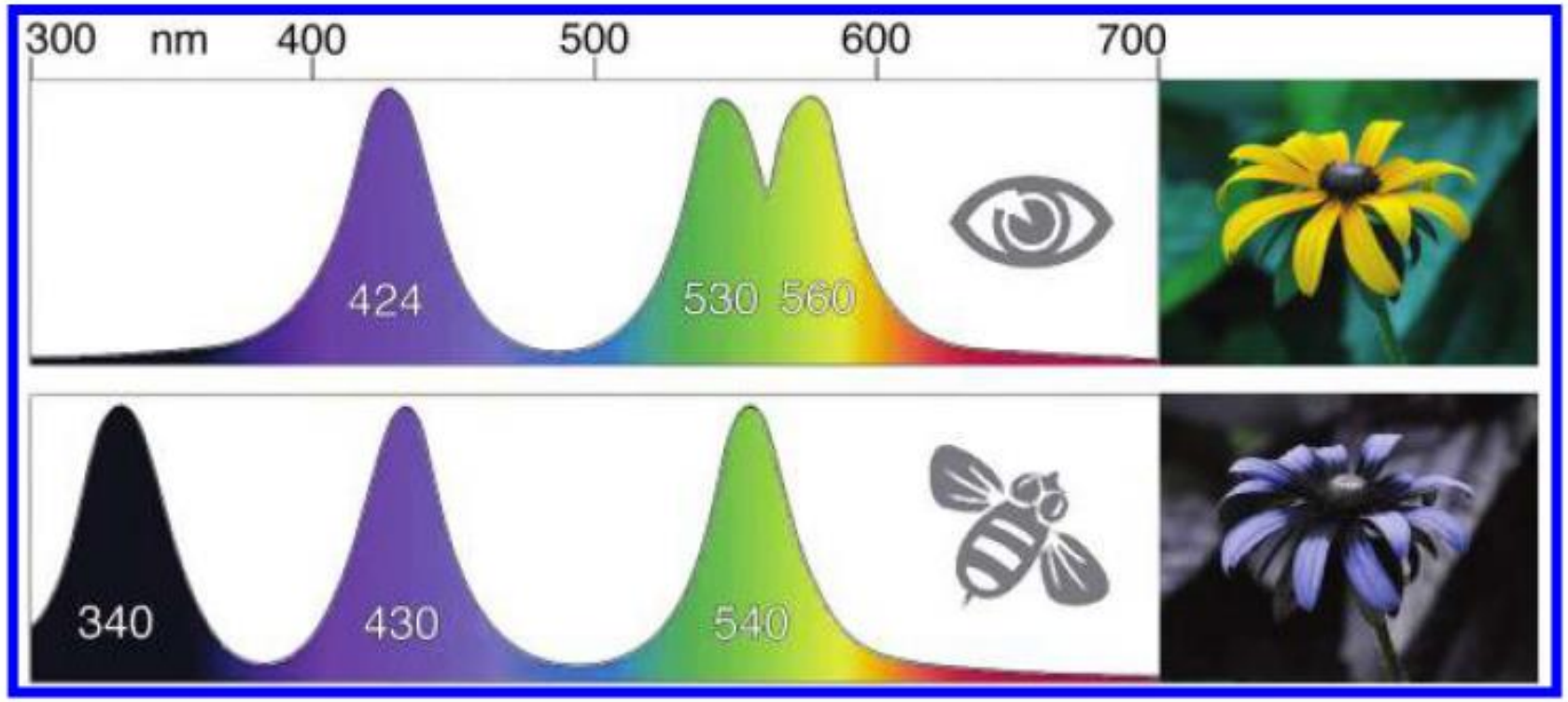


Figure 3.9 *Wavelength sensitivity of the color receptors in the eye of human and bee.*

How “color” formed

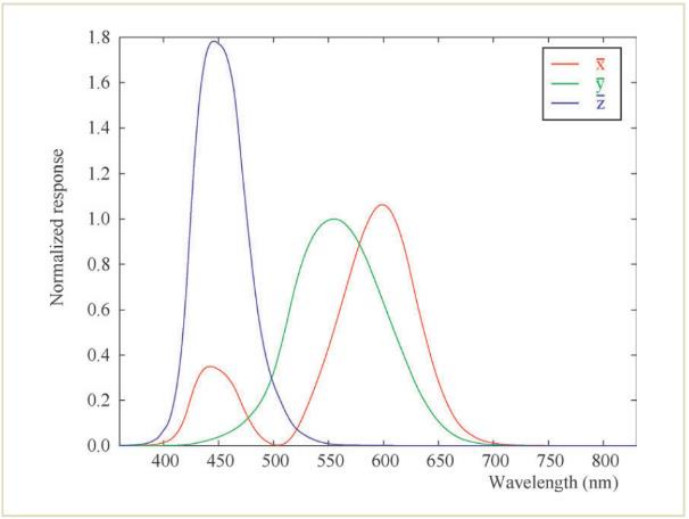


FIGURE 2.9 CIE 1931 2-degree XYZ color-matching functions.

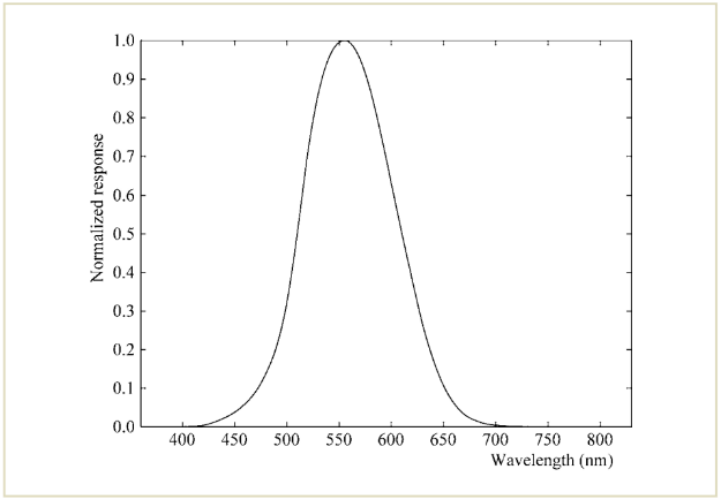
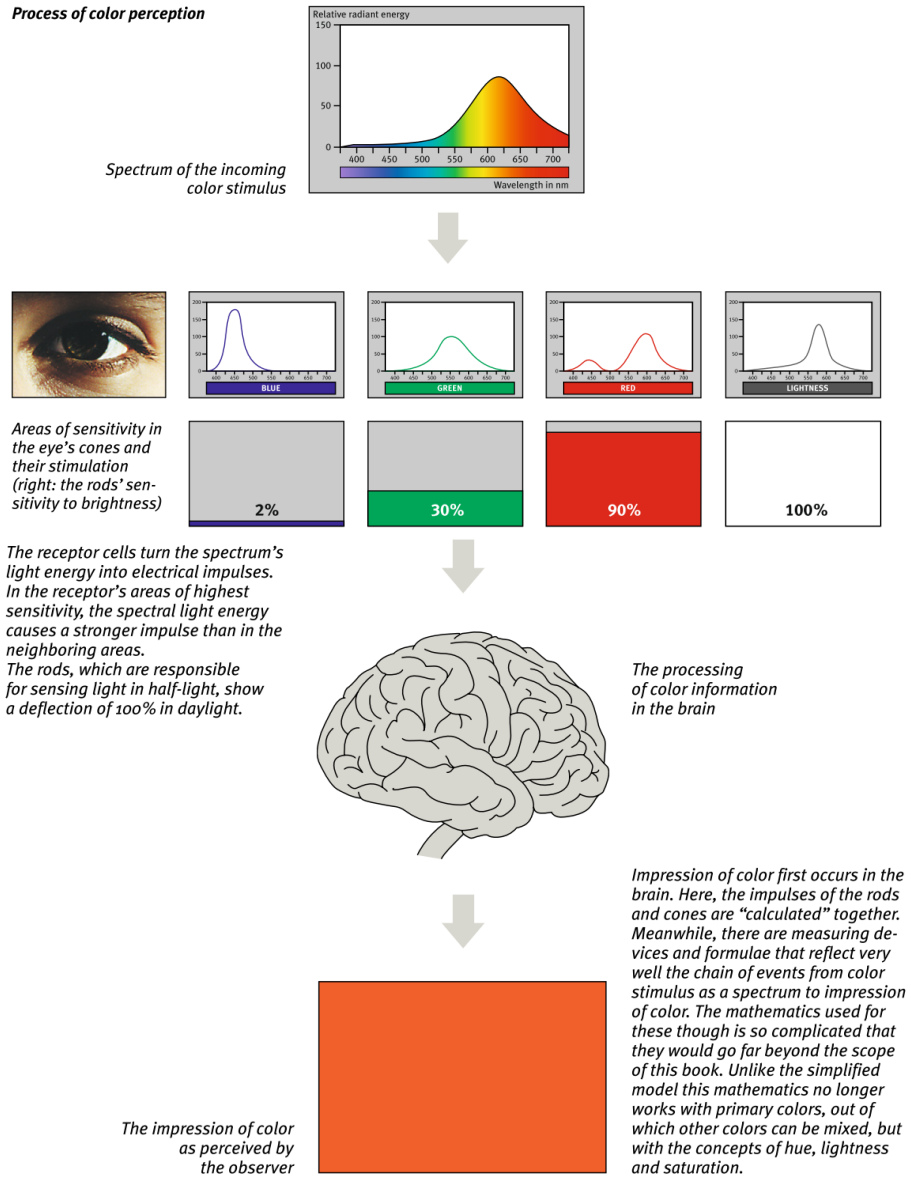


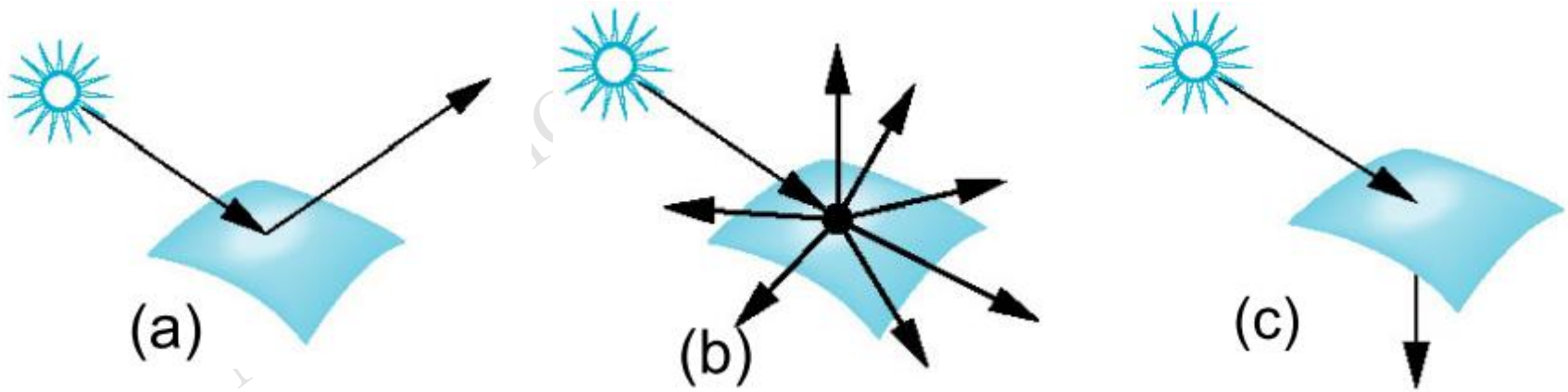
FIGURE 2.7 CIE standard observer photopic luminous efficiency curve.



2009 - Digital Color Management Principles and Strategies for the Standardized Print Production - Homann

General material properties for light

- (a) Specular surfaces
- (b) Diffuse surfaces
- (c) Translucent surfaces



Specular / reflection　　Diffusion　　Refraction

NTUST. CIIC.

From: Interactive Computer Graphics

Reflection model

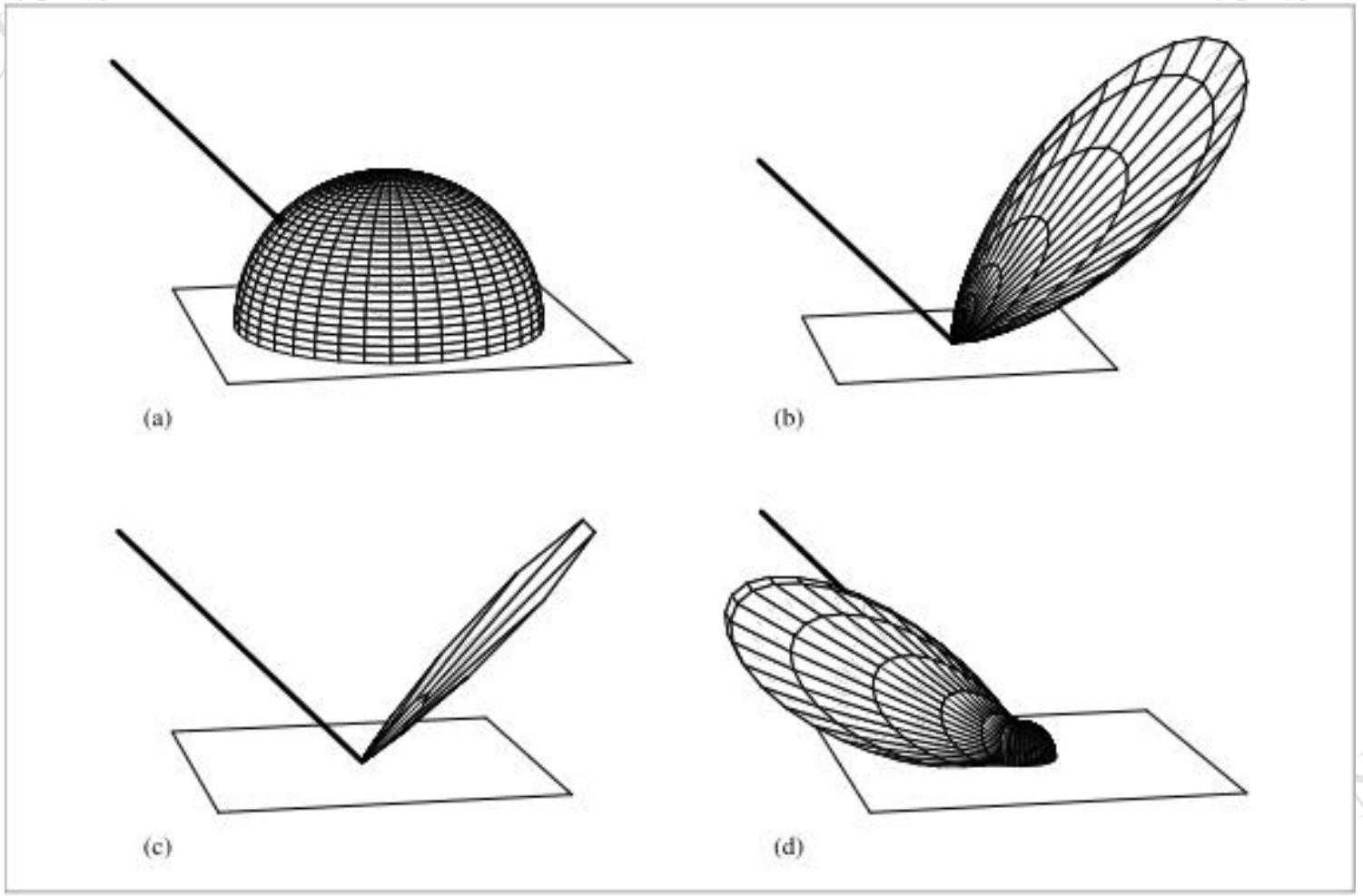


Figure 8.1: Reflection from a surface can be generally categorized by the distribution of reflected light from an incident direction (heavy lines): (a) diffuse, (b) glossy specular, (c) perfect specular, and (d) retro-reflective distributions.

NTUST. CIIC.

Shading Level

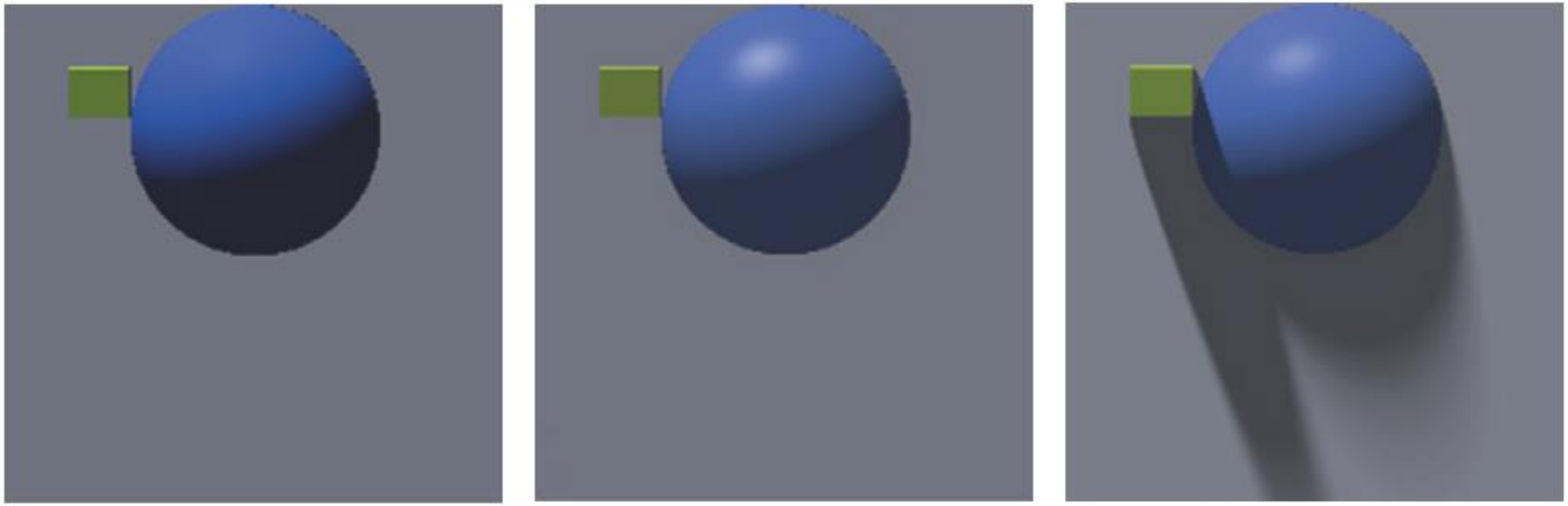


Figure 2.6 (a) Lambertian shading only. (b) Lambertian shading with specular and ambient shading. (c) Lambertian shading with specular, ambient, and cast shadows.

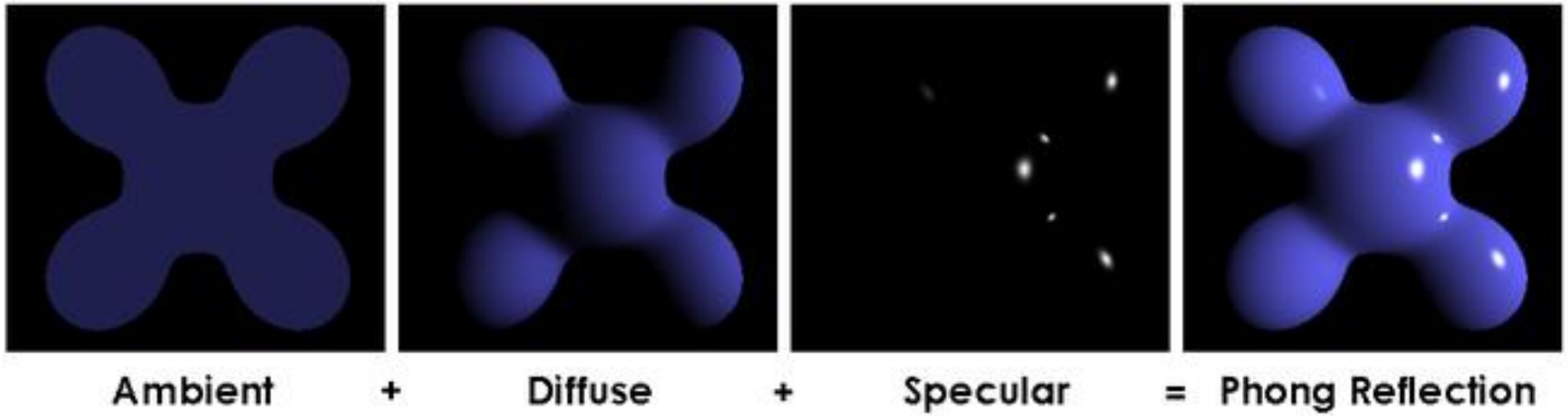
What property do we need in color?

- In computer graphics
- Light source
  - Specular light source
  - Diffuse light source
  - Ambient light source
- Material property
  - Reflection to specular light
  - Reflection to diffuse light
  - Refection to ambient light

NTUST. CIIC.

How CG deal with “color”

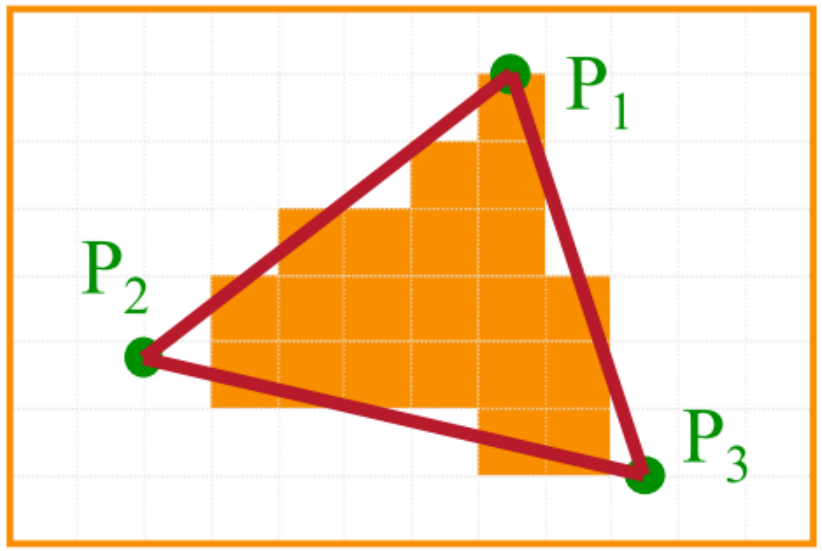
- Normally, in computer graphics, the color consists of at least three layers:
  - Ambient
  - Diffuse
  - Specular

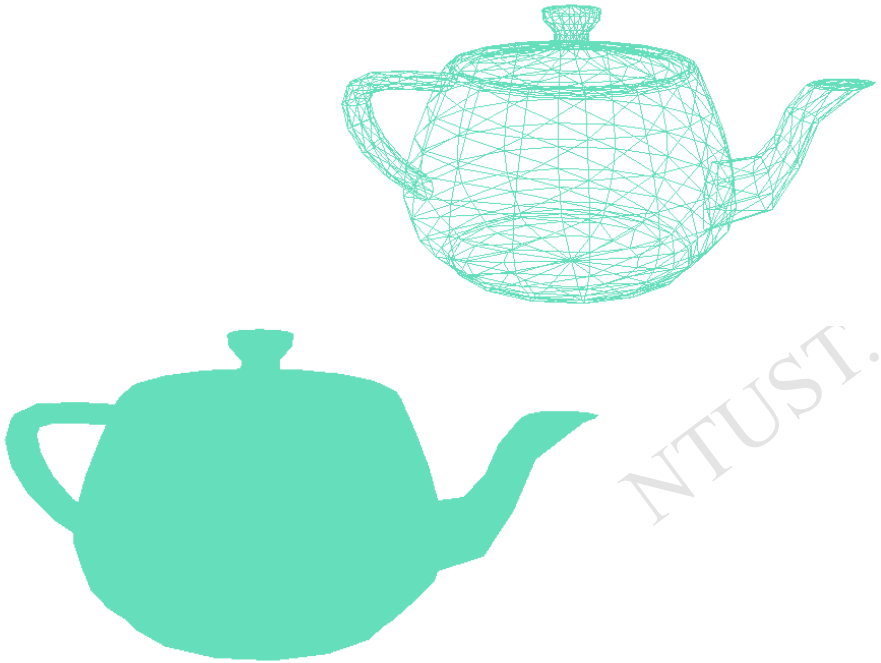


http://en.wikipedia.org/wiki/Phong_shading

Filling color in polygon

- In this situation, there is NO light source. Each polygon has a constant "color".

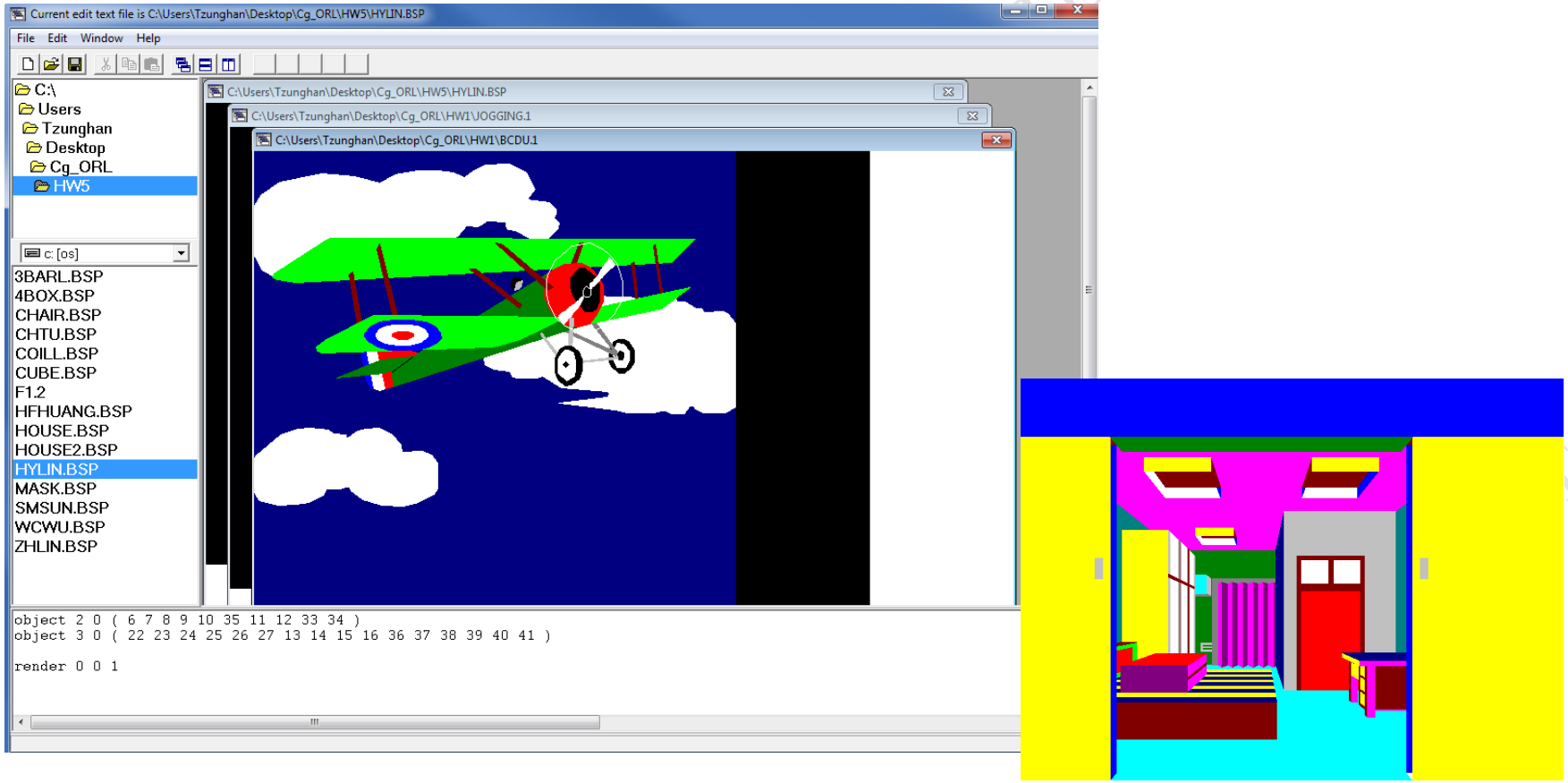




NTUST. CIIC.

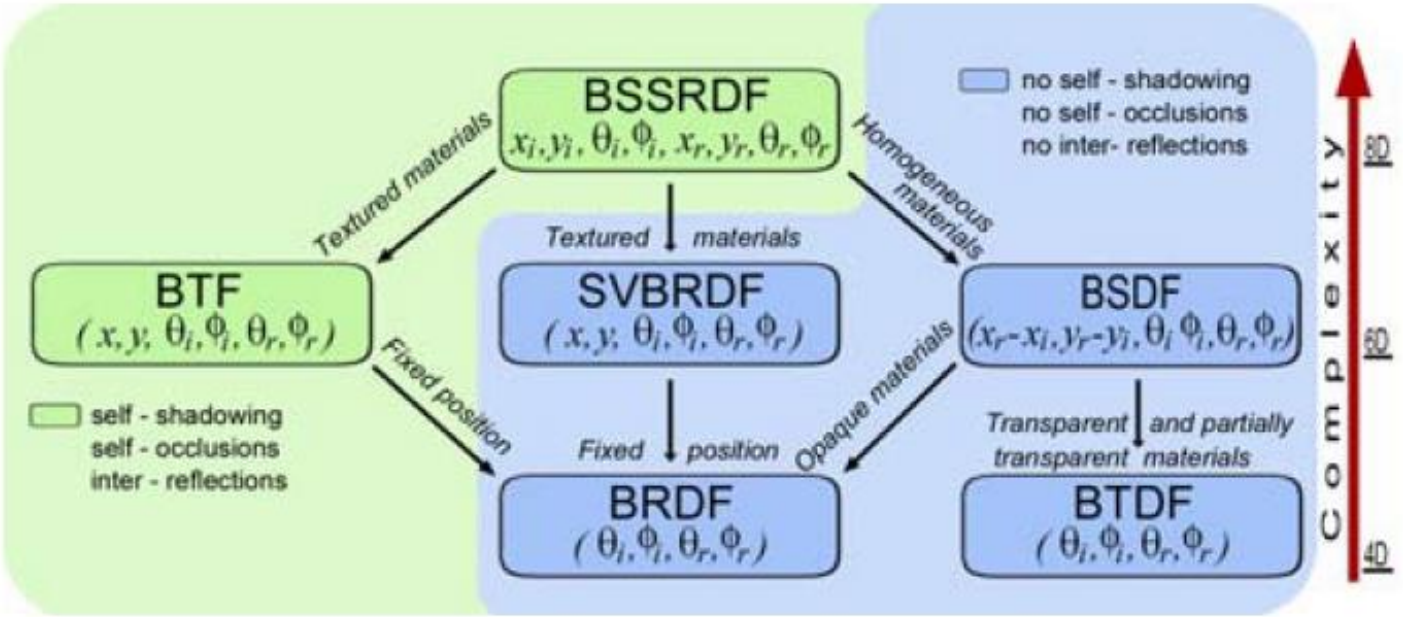
Filling color in polygon

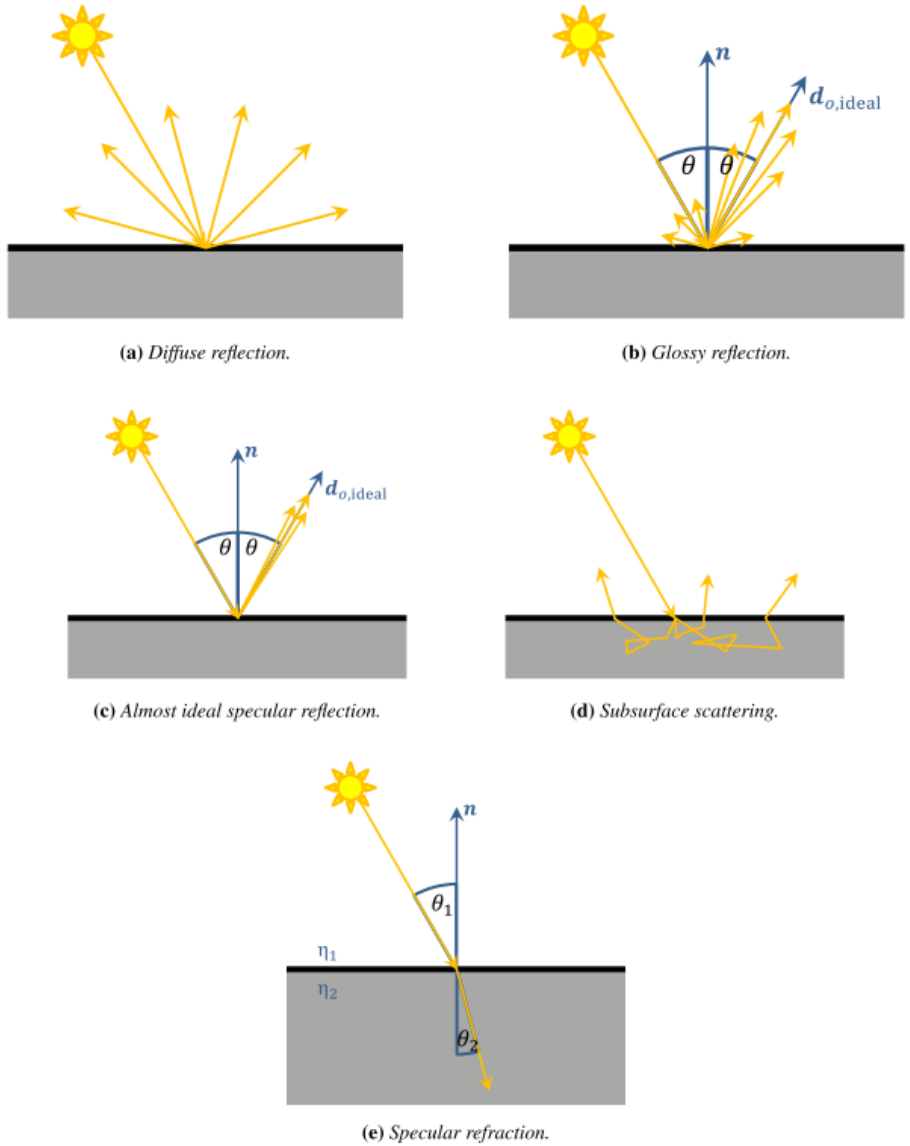
- For example



Physically Based Rendering

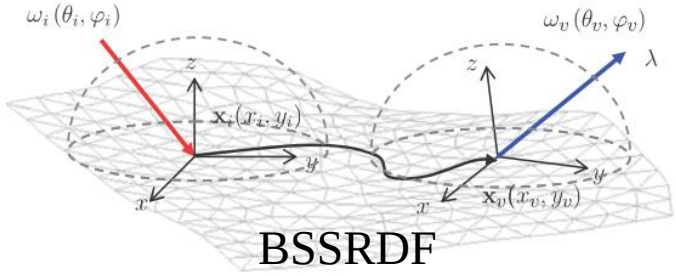
NTUST. CIIC.



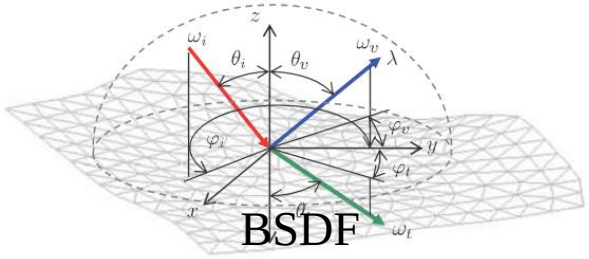


(a) *Diffuse reflection.*
(b) *Glossy reflection.*
(c) *Almost ideal specular reflection.*
(d) *Subsurface scattering.*
(e) *Specular refraction.*

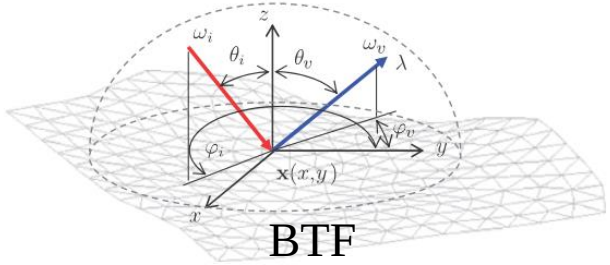
Physically Based Rendering



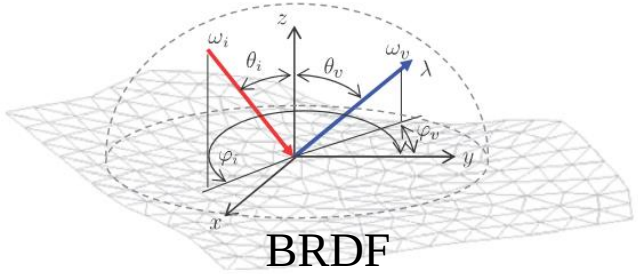
BSSRDF



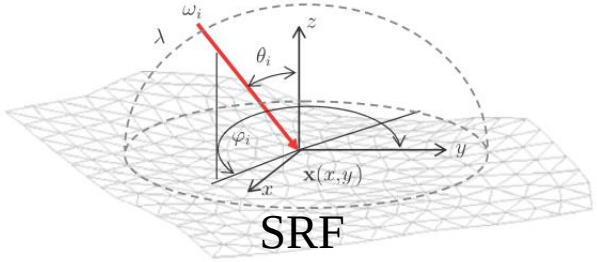
BSDF



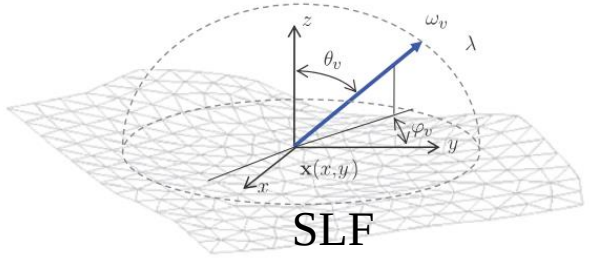
BTF



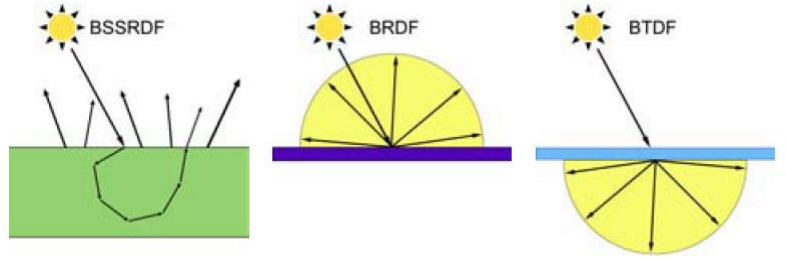
BRDF



SRF



SLF



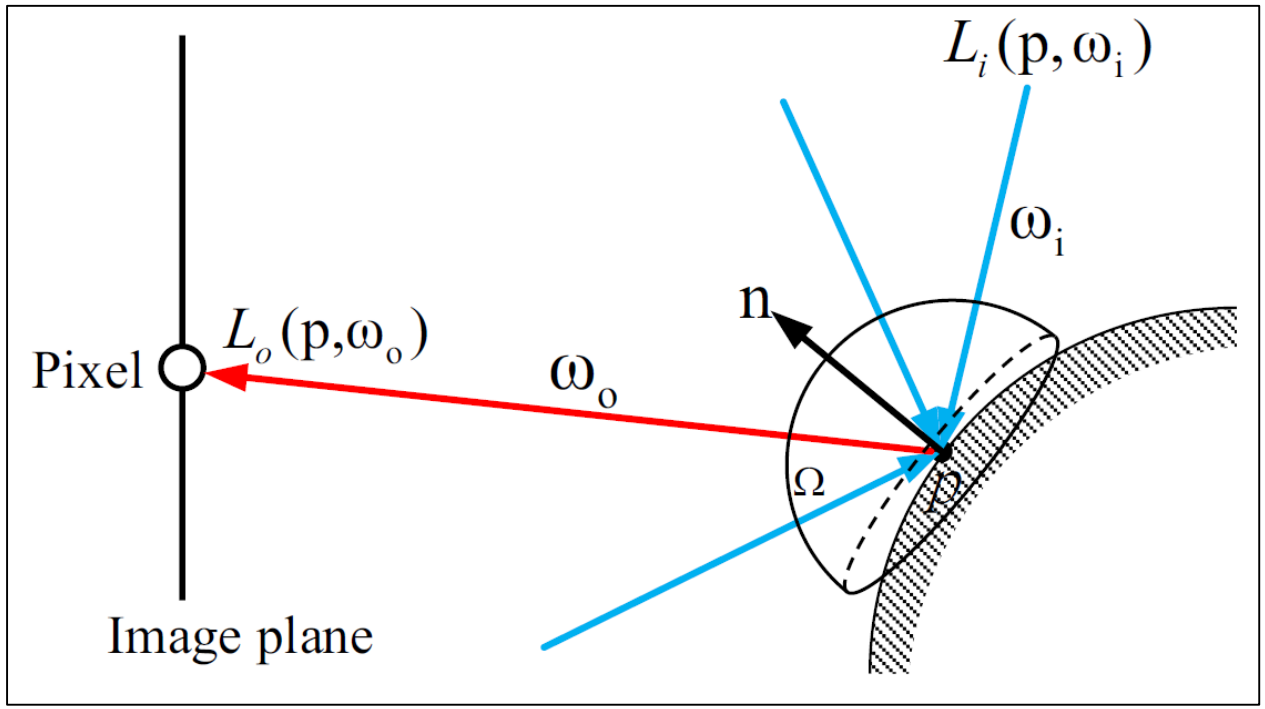
(a) BSSRDF (b) BRDF (c) BTDF


(a) Basic reflection models (b) Diffuse (c) Glossy (d) Specular

D. Guarnera, G. C. Guarnera, a. Ghosh, C. Denk, and M. Glencross, "BRDF representation and acquisition," *Comput. Graph. Forum*, vol. 35, no. 2, pp. 625–650, 2016.

Rendering Equation

$$L_o(p,\overrightarrow{\omega_o}) = \int_{\Omega} f_r(p,\overrightarrow{\omega_i},\overrightarrow{\omega_o}) L_i(p,\overrightarrow{\omega_i}) \vec{n} \cdot \overrightarrow{\omega_i} d\overrightarrow{\omega_i}$$



Radiant Energy: Q

Radiant flux: Φ $\Phi = \frac{dQ}{dt}$

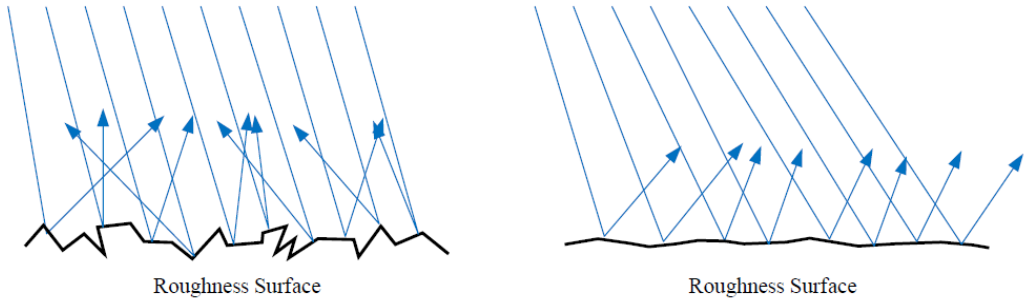
Radiance: L $L = \frac{d^2\Phi}{dA d\omega \cos\theta}$

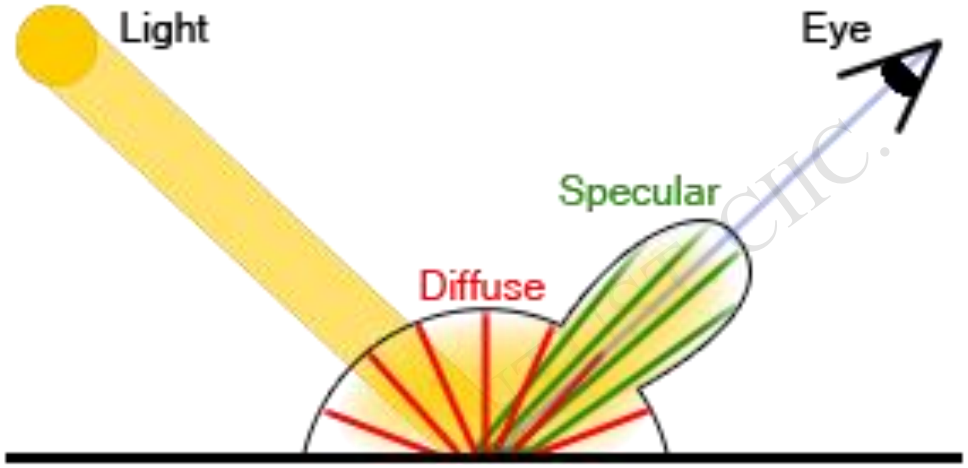
Unit area: A

Unit solid angle: ω

NTUST. CIIC.

Physically based rendering model: Cook–Torrance Model





"The approximation model of the surface that Cook and Torrance have used falls into the category of the Microfacet models which revolves around the idea that rough surfaces can be modelled as a collection of small microfacets."

k_d is the diffuse reflection constant for the material.
k_s is the specular reflection constant for the material.

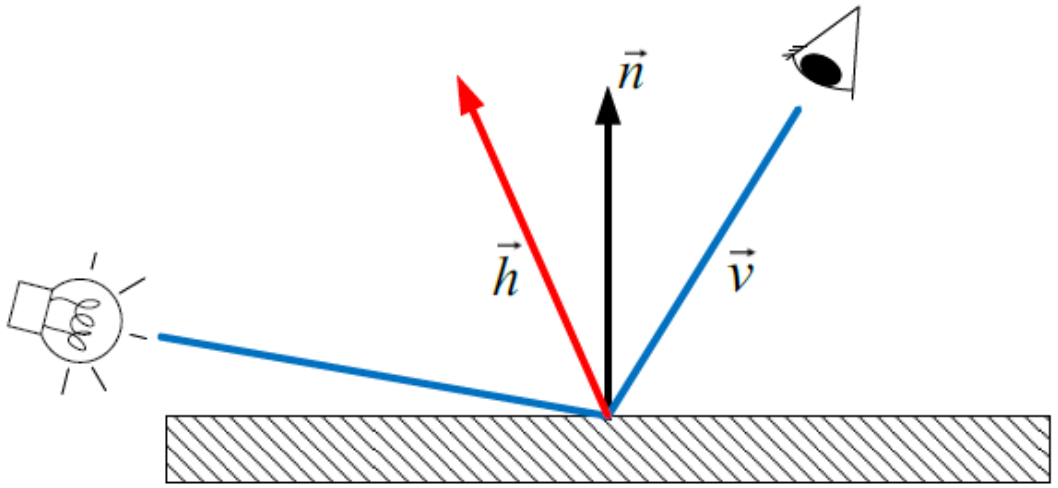
$$f_r = k_d f_{lambert} + k_s f_{cook-torrance}$$

$$f_{lambert} = \frac{c}{\pi}$$

$$f_{cook-torrance} = \frac{D \cdot F \cdot G}{4(\overrightarrow{\omega_o}, \vec{n})(\overrightarrow{\omega_i}, \vec{n})}$$

http://www.codinglabs.net/article_physically_based_rendering_cook_torrance.aspx

Physically based rendering model: Cook–Torrance Model-Normal Distribution Function



$$f_{cook-torrance} = \frac{D \cdot F \cdot G}{4(\overrightarrow{\omega_o}, \vec{n})(\overrightarrow{\omega_i}, \vec{n})}$$

Normal Distribution Function

$$DNF_{GGX_{TR}}(\vec{n}, \vec{h}, \alpha) = \frac{\alpha^2}{\pi\left[(\vec{n} \cdot \vec{h})^2(\alpha^2 - 1) + 1\right]}$$

α : Roughness of the surface

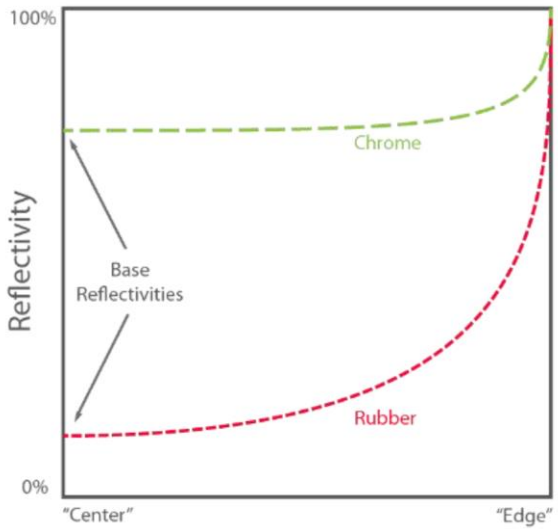
h: Halfway vector

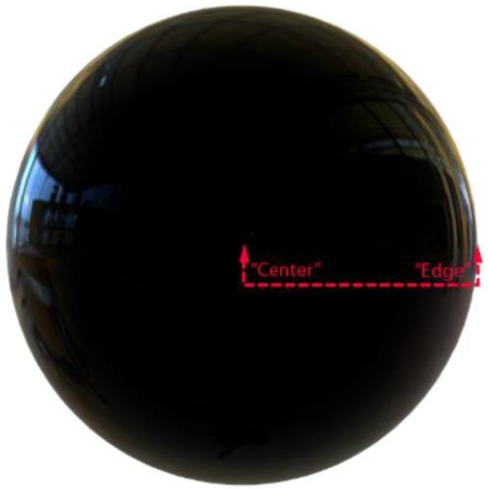
n: Normal vector

http://www.codinglabs.net/article_physically_based_rendering_cook_torrance.aspx

Physically based rendering model: Cook–Torrance Model- Fresnel Equation

$$f_{cook-torrance} = \frac{D \cdot F \cdot G}{4(\overrightarrow{\omega_o}, \vec{n})(\overrightarrow{\omega_i}, \vec{n})}$$





Fresnel Equation：

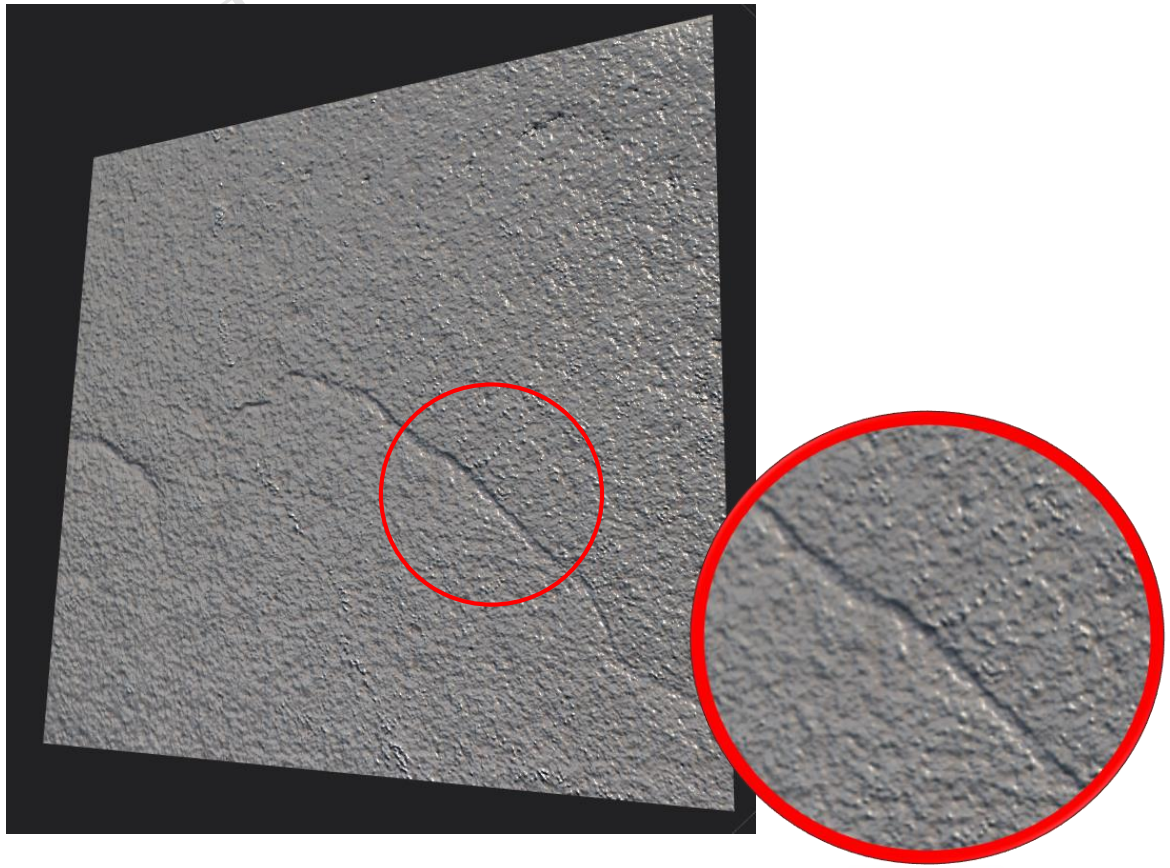
$$F_{Schlick}(\vec{n}, \vec{v}, F_0) = F_0 + (1 - \mathrm{F_0})\left(1 - (\vec{n} \cdot \vec{v})\right)^5$$

F_0: Basic reflectance, got by Indices of Refraction, IOR

Insulator	F_0 (Linear)	F_0 (sRGB)	Color
Water	0.02, 0.02, 0.02	0.15, 0.15, 0.15	
Plastic / Glass (Low)	0.03, 0.03, 0.03	0.21, 0.21, 0.21	
Plastic High	0.05, 0.05, 0.05	0.24, 0.24, 0.24	
Glass (High) / Ruby	0.08, 0.08, 0.08	0.31, 0.31, 0.31	
Diamond	0.17, 0.17, 0.17	0.45, 0.45, 0.45	

http://www.codinglabs.net/article_physically_based_rendering_cook_torrance.aspx

Physically based rendering model: Cook–Torrance Model- Geometry Function



$$f_{cook-torrance} = \frac{D \cdot F \cdot G}{4(\overrightarrow{\omega_o}, \vec{n})(\overrightarrow{\omega_i}, \vec{n})}$$

Geometry Function ：

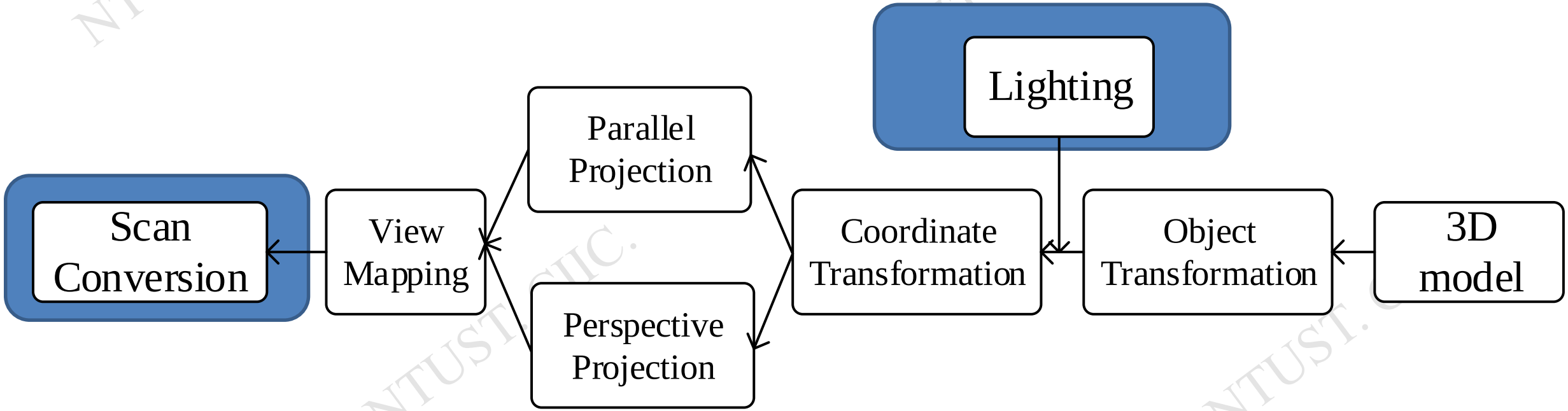
$$G_{SchlickGXX}(\vec{n}, \vec{v}, k) = \frac{\vec{n} \cdot \vec{v}}{(\vec{n} \cdot \vec{v})(1 - \alpha/2) + \alpha/2}$$

α : Roughness of the surface

NTUST. CIIC.

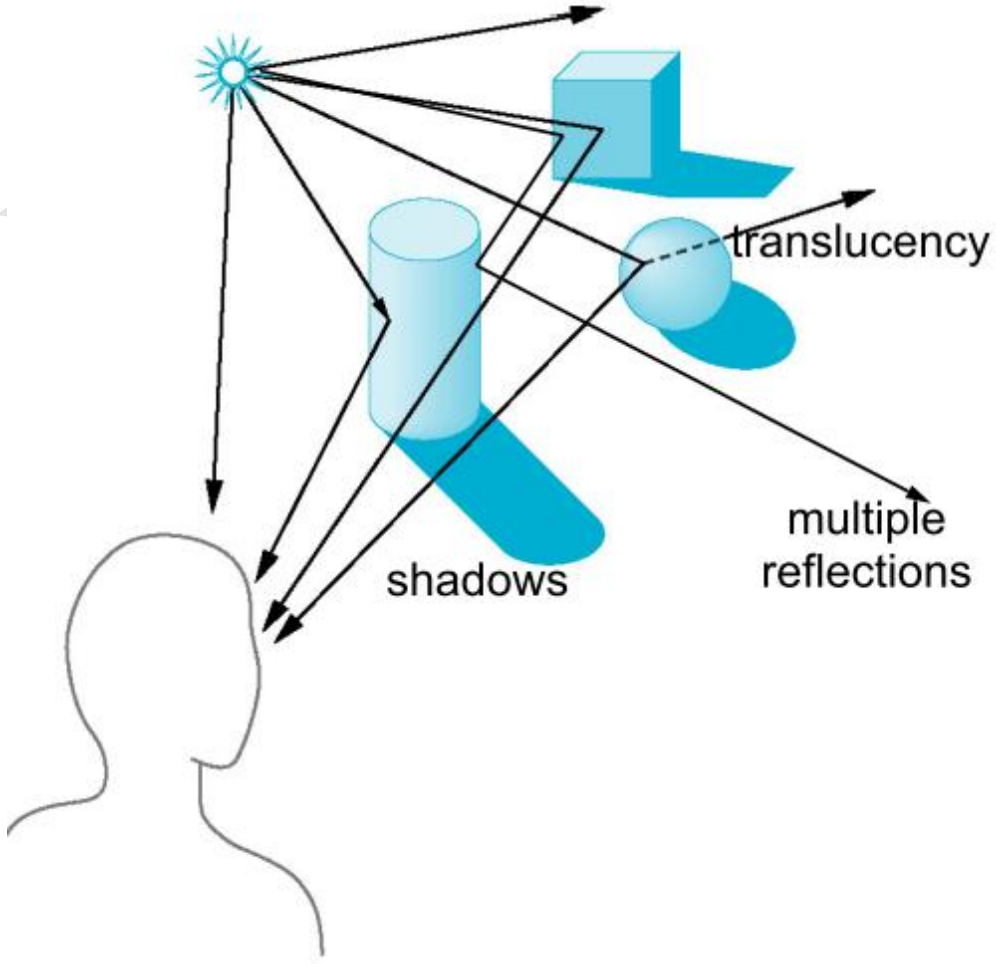
http://www.codinglabs.net/article_physically_based_rendering_cook_torrance.aspx

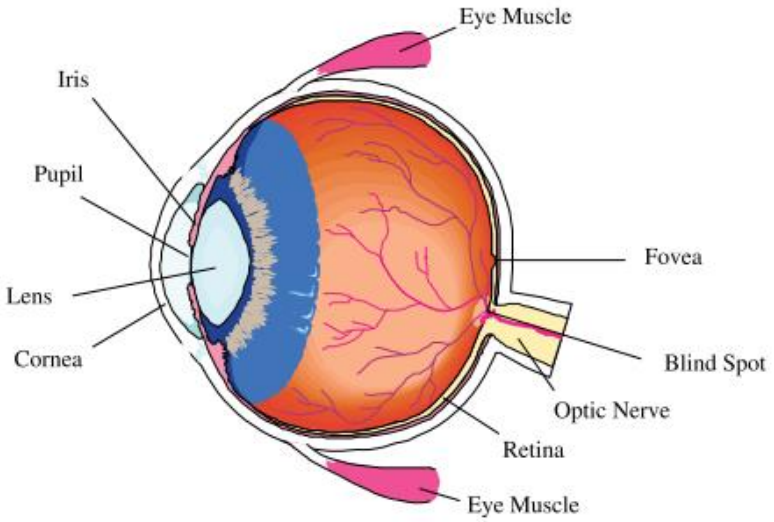
Lighting in CG pipeline

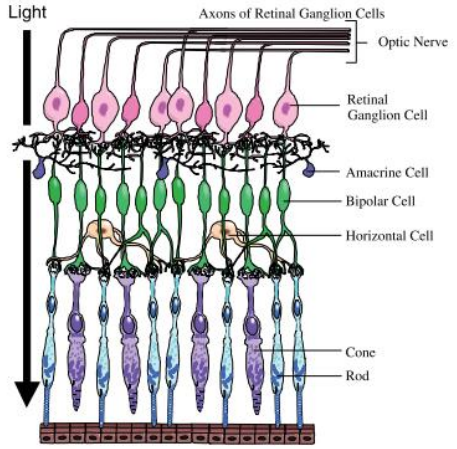


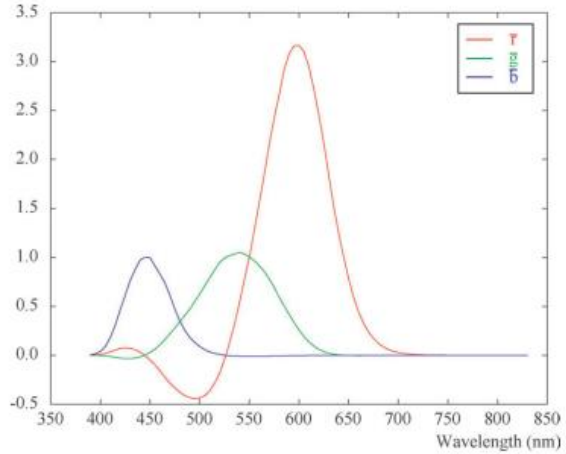
NTUST. CIIC.

Color and tristimulus



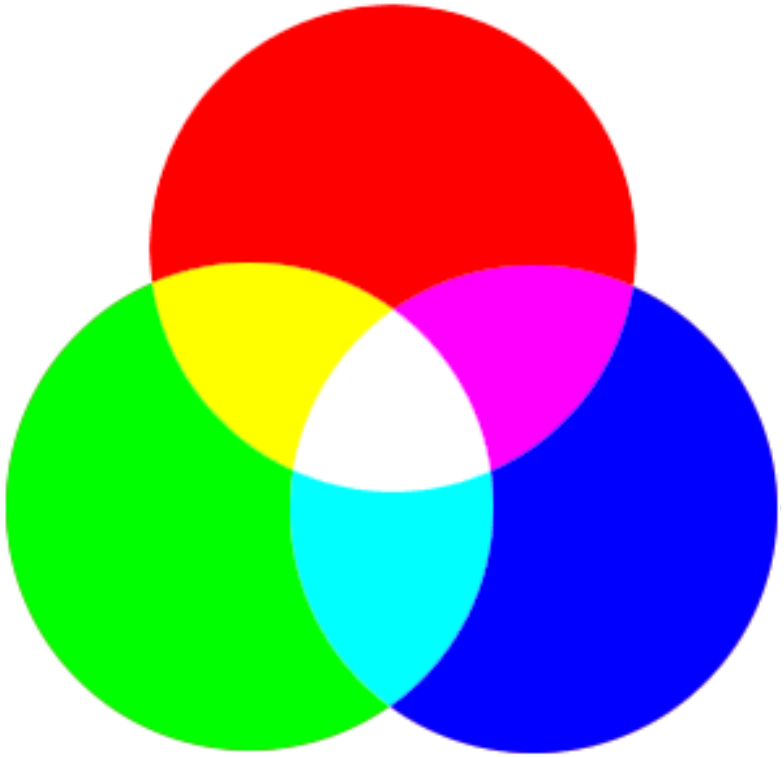




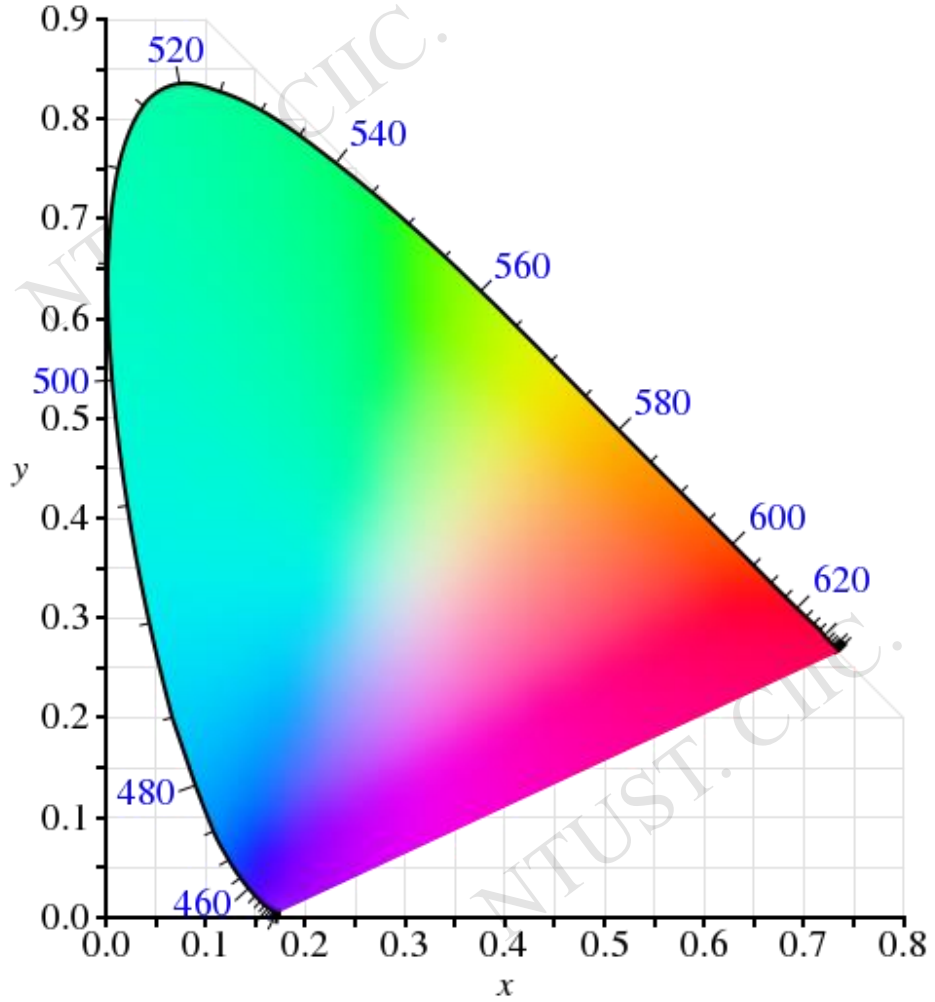


From: Interactive Computer Graphics
From Book: Color Constancy

Color and tristimulus



Color addition



CIE1931xy

Color and tristimulus

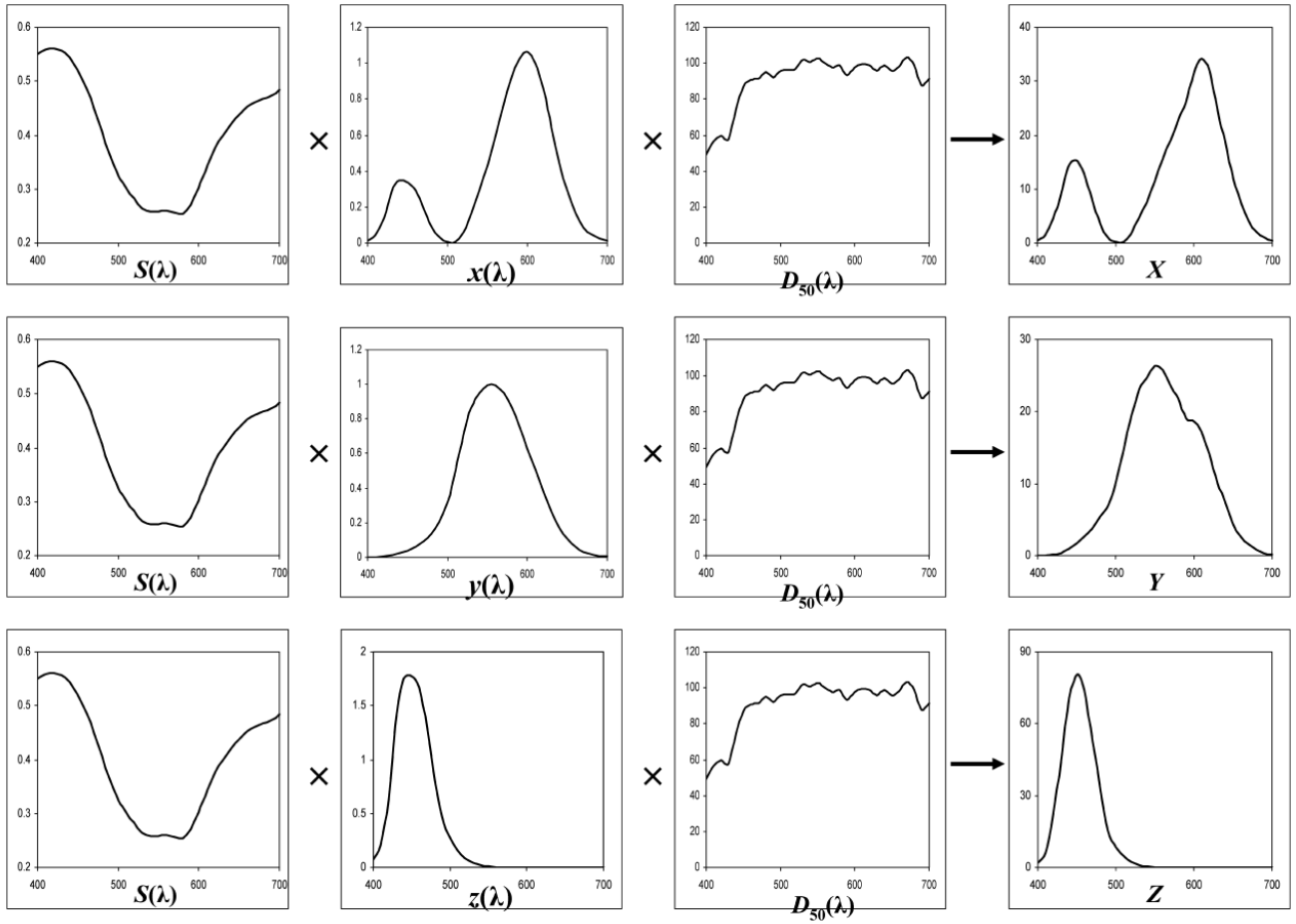
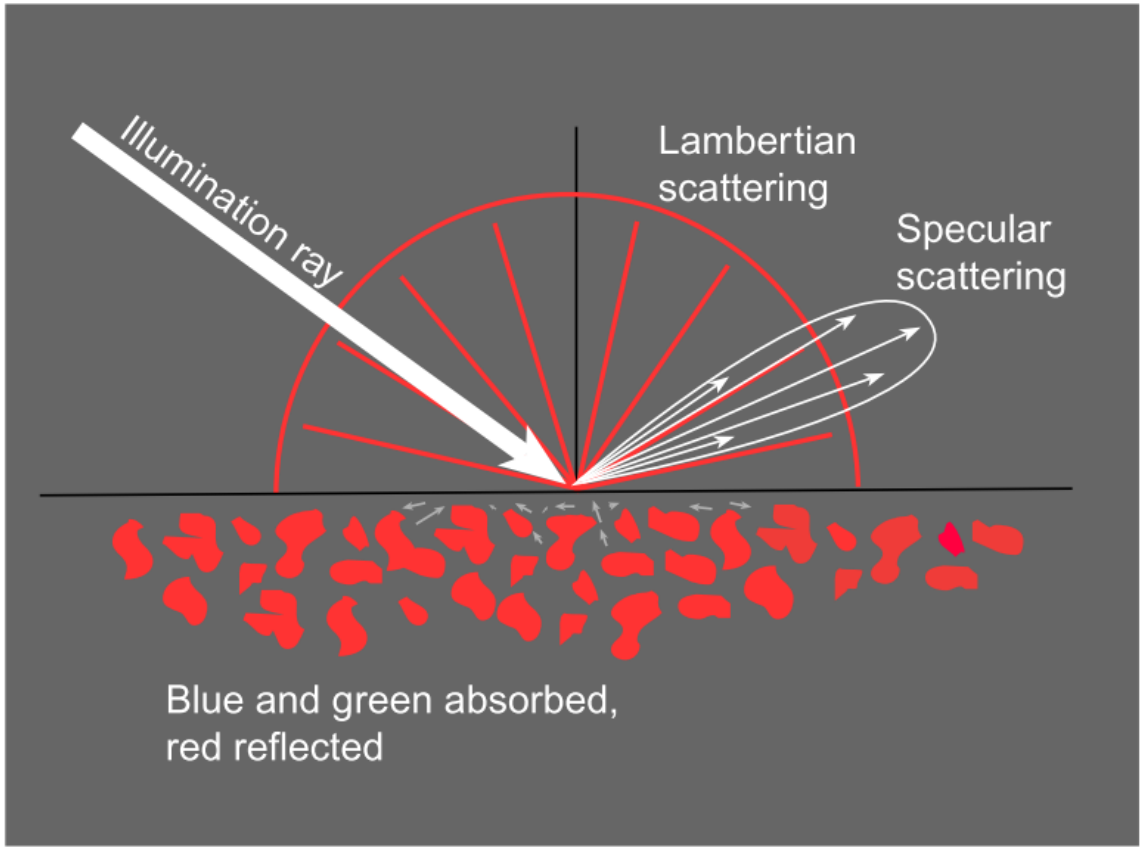
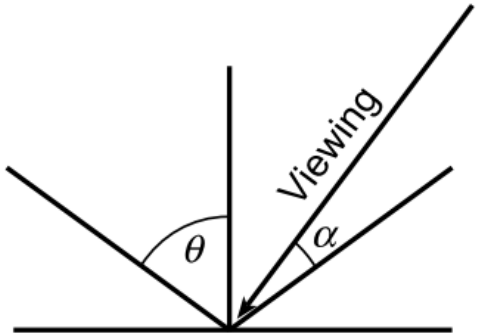


Figure 1.1 Graphic illustration of the CIE tristimulus computation.

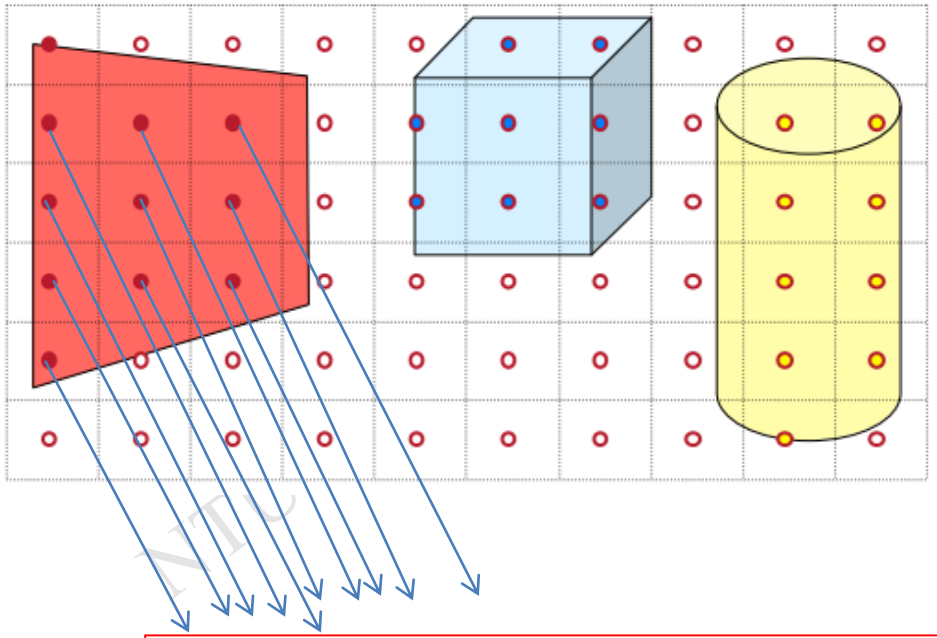
Photometry and light

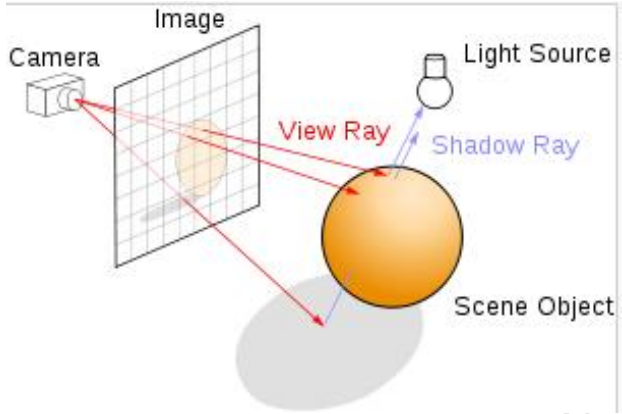




How color formed in CG?

- Determine the intensity (RGB/color) of every pixel on the image





$$I = I_E + K_A I_{AL} + \sum_i [K_D (N \cdot L_i) I_{Di} + K_S (V \cdot R_i)^n I_{Si}]$$

http://en.wikipedia.org/wiki/Ray_tracing_(graphics)

How color formed in CG?

- For “light”, tree types of lights are often used
  - I : Intensity (color) you see on screen (outcome)
  - I_E : Intensity (color) of emission
  - I_{AL} : Intensity (color) of ambient light
  - I_{Di} : Intensity (color) of the i-th light (diffuse)
  - I_{Si} : Intensity (color) of the i-th light (specular)
- In most conditions, people usually use “white” light

$$I = I_E + K_A I_{AL} + \sum_i [K_D (N \cdot L_i) I_{Di} + K_S (V \cdot R_i)^n I_{Si}]$$

NTUST. CIIC.

How color formed in CG?

- For “material property”, several coefficients are defined:
  - K_A : Reaction to ambient light (color)
  - K_D : Reaction to diffuse light (color)
  - K_S : Reaction to specular light (color)
  - n : The gloss coefficient

$$I = I_E + K_A I_{AL} + \sum_i [K_D (N \cdot L_i) I_{Di} + K_S (V \cdot R_i)^n I_{Si}]$$

NTUST. CIIC.

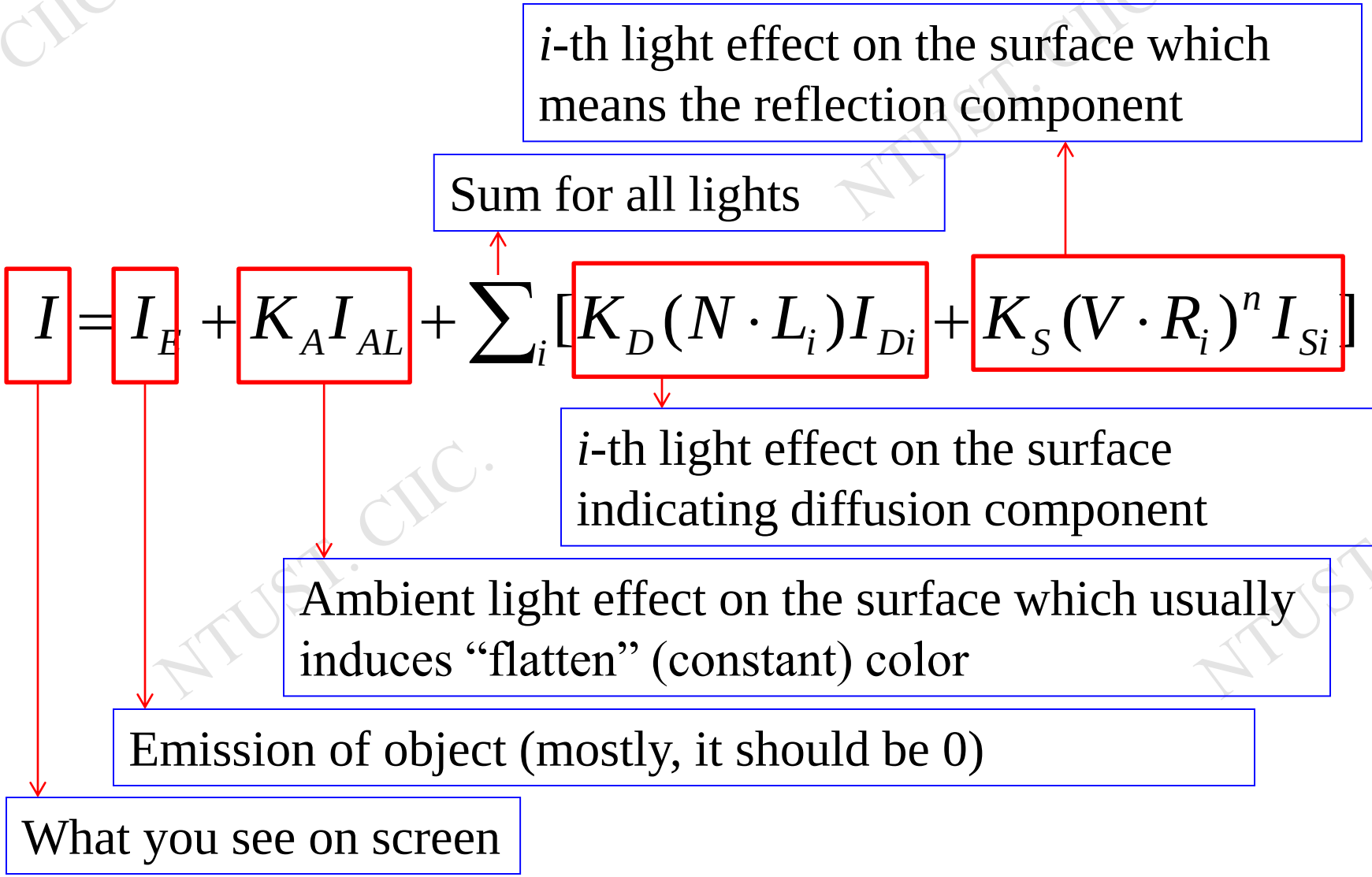
How color formed in CG?

- For “geometry”, several directions are defined:
  - N : Normal vector of vertex (or face)
  - L_i : The vector to i-th light
  - R_i : Reflection direction of i-th light

$$I = I_E + K_A I_{AL} + \sum_i [K_D (N \cdot L_i) I_{Di} + K_S (V \cdot R_i)^n I_{Si}]$$

NTUST. CIIC.

How color formed in CG?



NTUST. CIIC.

How color formed in CG? (ambient part)

- What does ambient light contribute?

$$I = I_E + K_A I_{AL} + \sum_i [K_D (N \cdot L_i) I_{Di} + K_S (V \cdot R_i)^n I_{Si}]$$

Ambient light (sun light, moon light, etc.)
Could be [r, g, b] format

Material property: reaction for ambient light
Could be [r, g, b] format

NTUST. CIIC.

How color formed in CG? (ambient part)

- What the general conditions should be?

$$I = K_A I_{AL}$$

$$\begin{bmatrix} r \\ g \\ b \end{bmatrix} = \begin{bmatrix} K_A[r] \\ K_A[g] \\ K_A[b] \end{bmatrix} \begin{bmatrix} I_{AL}[r] \\ I_{AL}[g] \\ I_{AL}[b] \end{bmatrix}$$

Ambient outcome | Material property | Light property

Recall the *.mtl in *.obj file

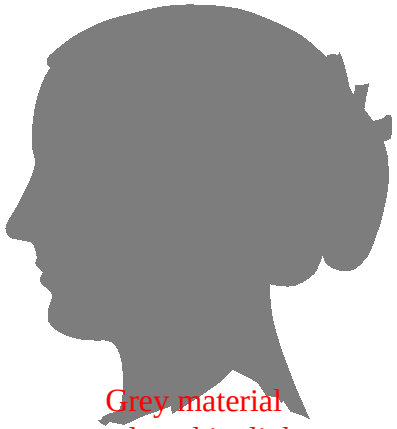
```
newmtl footMaterial
        Ka 0.1 0.1 0.1
        Kd 0.7 0.7 0.8
        Ks 0.9 0.9 0.9
        Ns 50
        d 1
        Tr 0.5
        illum 1
```

NTUST. CIIC.

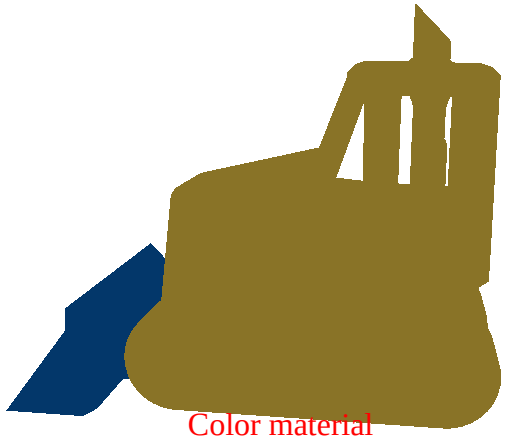
How color formed in CG? (ambient part)

- What does shading result look like?
  - Constant color Only.
  - Light does NOT affect the intensity distribution.
  - Color changes monotonously.

$$I = K_A I_{AL}$$



Grey material
under white light

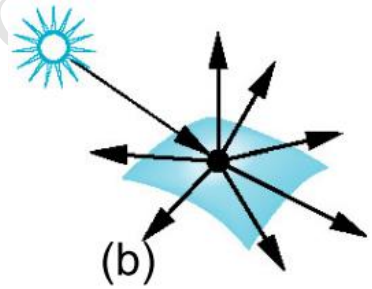


Color material
under white light

NTUST. CIIC.

How color formed in CG? (diffuse part)

- Diffusion: accumulate all “lights”
  - Note: the normal vector N.



Normal vector of a pixel (or face)

The vector to *i*-th light source

$$I = I_E + K_A I_{AL} + \sum_i [K_D (N \cdot L_i) I_{Di} + K_S (V \cdot R_i)^n I_{Si}]$$

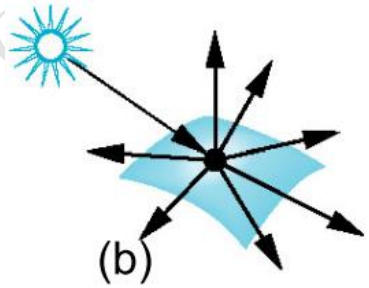
Material property for diffusion

i-th Light diffuse intensity

NTUST. CIIC.

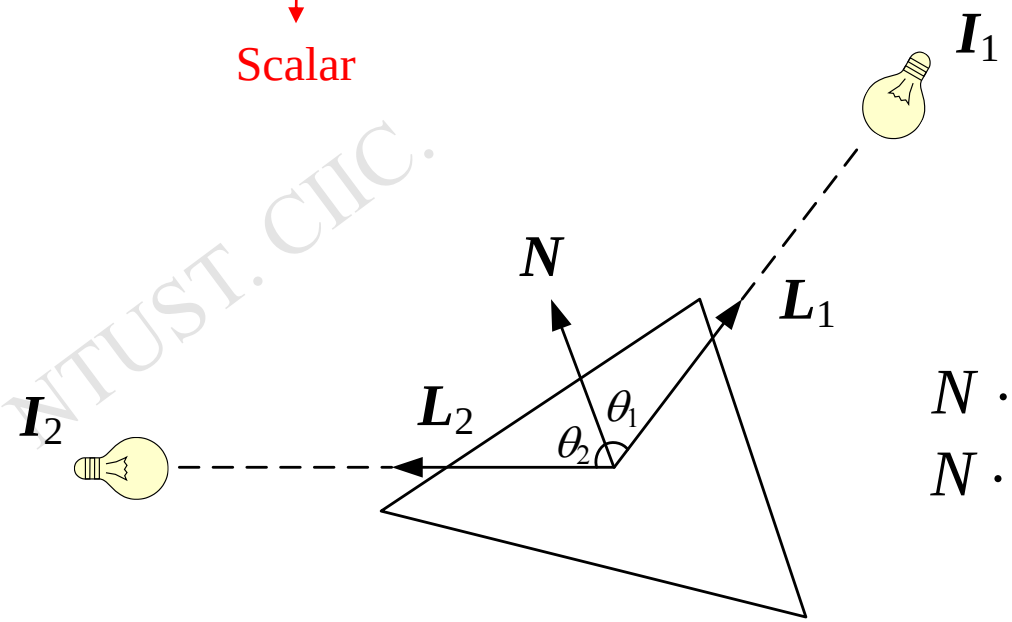
How color formed in CG? (diffuse part)

- Diffusion: accumulate all "lights"



$$I = \sum_i K_D (N \cdot L_i) I_{Di}$$

Scalar



$$N \cdot L_1 = \cos\theta_1$$
$$N \cdot L_2 = \cos\theta_2$$
.....

Note: all vectors are unit vectors.

How color formed in CG? (diffuse part)

- What the general conditions should be?

$$I = \sum_i K_D (N \cdot L_i) I_{Di}$$

Geometrical decay (due to 1st light)

Geometrical decay (due to 2nd light)

$$\begin{bmatrix} r \\ g \\ b \end{bmatrix} = \cos\theta_1 \begin{bmatrix} K_D[r] \\ K_D[g] \\ K_D[b] \end{bmatrix} \begin{bmatrix} I_{D1}[r] \\ I_{D1}[g] \\ I_{D1}[b] \end{bmatrix} + \cos\theta_2 \begin{bmatrix} K_D[r] \\ K_D[g] \\ K_D[b] \end{bmatrix} \begin{bmatrix} I_{D2}[r] \\ I_{D2}[g] \\ I_{D2}[b] \end{bmatrix} + \ldots$$

Diffuse outcome

Material property

1st light diffuse property

Material property

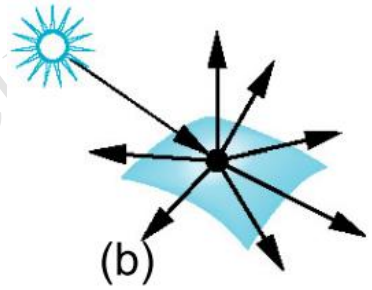
2nd light diffuse property

Recall the *.mtl in *.obj file

```
newmtl footMaterial
      Ka 0.1 0.1 0.1
      Kd 0.7 0.7 0.8
      Ks 0.9 0.9 0.9
      Ns 50
      d 1
      Tr 0.5
      illum 1
```

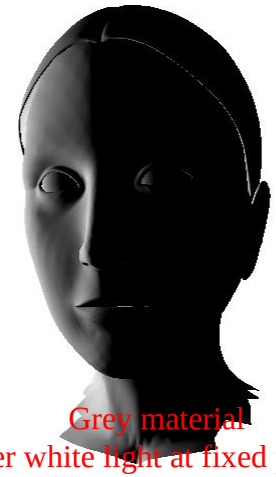
NTUST. CIIC.

How color formed in CG? (diffuse part)

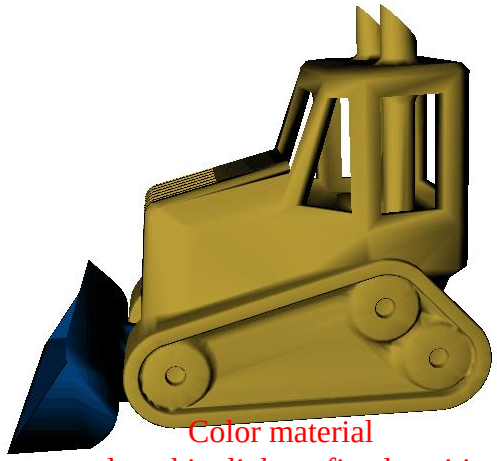


- What does the shading result look like?
  - Intensity continuously changes.
  - For fixed light positions, intensity is always the same no matter what the observation direction.

$$I = \sum_i K_D (N \cdot L_i) I_{Di}$$



Grey material
under white light at fixed position

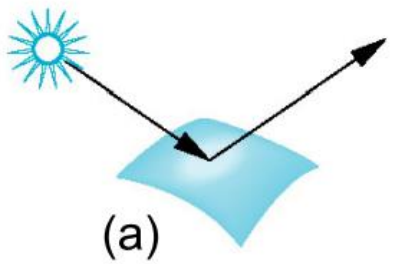


Color material
under white light at fixed position

NTUST. CIIC.

How color formed in CG? (specular part)

- Specular: accumulate all “reflection lights”
  - Note: the vector N and Ri.



The vector from surface to camera (eye)

Reflection vector of the i-th Light

Material property (glossiness)

$$I = I_E + K_A I_{AL} + \sum_i [K_D (N \cdot L_i) I_{Di} + K_S (V \cdot R_i)^n I_{Si}]$$

Material property for reflection (specular)

i-th Light specular intensity

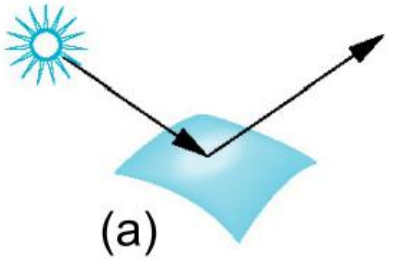
NTUST.CIIC.

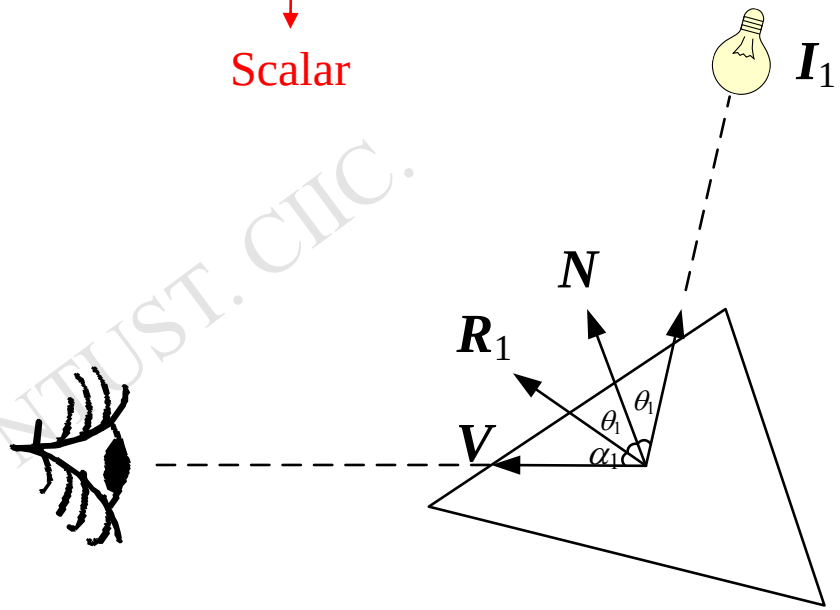
How color formed in CG? (specular part)

- Reflection (specular): accumulate all "lights"

$$I = \sum_i K_S (V \cdot R_i)^n I_{Si}$$

Scalar





$$V \cdot R_1 = \cos \alpha_1$$

.....

NTUST. CIIC.

How color formed in CG? (specular part)

- What the general conditions should be?

$$I = \sum_i K_S (V \cdot R_i)^n I_{Si}$$

Geometrical & viewing direction decay (due to 1st light)

Geometrical & viewing direction decay (due to 2nd light)

$$\begin{bmatrix} r \\ g \\ b \end{bmatrix} = \cos^n \alpha_1 \begin{bmatrix} K_S[r] \\ K_S[g] \\ K_S[b] \end{bmatrix} \begin{bmatrix} I_{S1}[r] \\ I_{S1}[g] \\ I_{S1}[b] \end{bmatrix} + \cos^n \alpha_2 \begin{bmatrix} K_S[r] \\ K_S[g] \\ K_S[b] \end{bmatrix} \begin{bmatrix} I_{S2}[r] \\ I_{S2}[g] \\ I_{S2}[b] \end{bmatrix} + \ldots$$

Specular outcome

Material property

1st light specular property

Material property

2nd light specular property

Recall the *.mtl in *.obj file

```
newmtl footMaterial
      Ka 0.1 0.1 0.1
      Kd 0.7 0.7 0.8
      Ks 0.9 0.9 0.9
      Ns 50
      d 1
      Tr 0.5
      illum 1
```

NTUST. CIIC.

How color formed in CG? (specular part)

- What does the shading result look like?
  - Intensity continuously changes according viewing direction and light position.

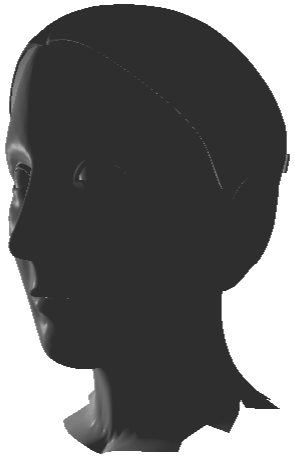
$$I = \sum_i K_S (V \cdot R_i)^n I_{Si}$$

NTUST. CIIC.

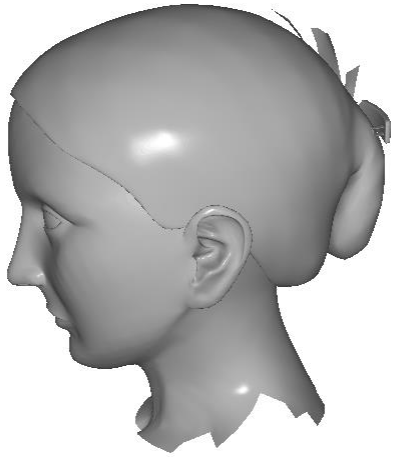
How color formed in CG?

- What the linear combination will look like?

$$\begin{bmatrix} r \\ g \\ b \end{bmatrix}_{\text{Result}} = \begin{bmatrix} r \\ g \\ b \end{bmatrix}_{\text{Emission}} + \begin{bmatrix} r \\ g \\ b \end{bmatrix}_{\text{Ambient}} + \begin{bmatrix} r \\ g \\ b \end{bmatrix}_{\text{Diffuse}} + \begin{bmatrix} r \\ g \\ b \end{bmatrix}_{\text{Specular}}$$



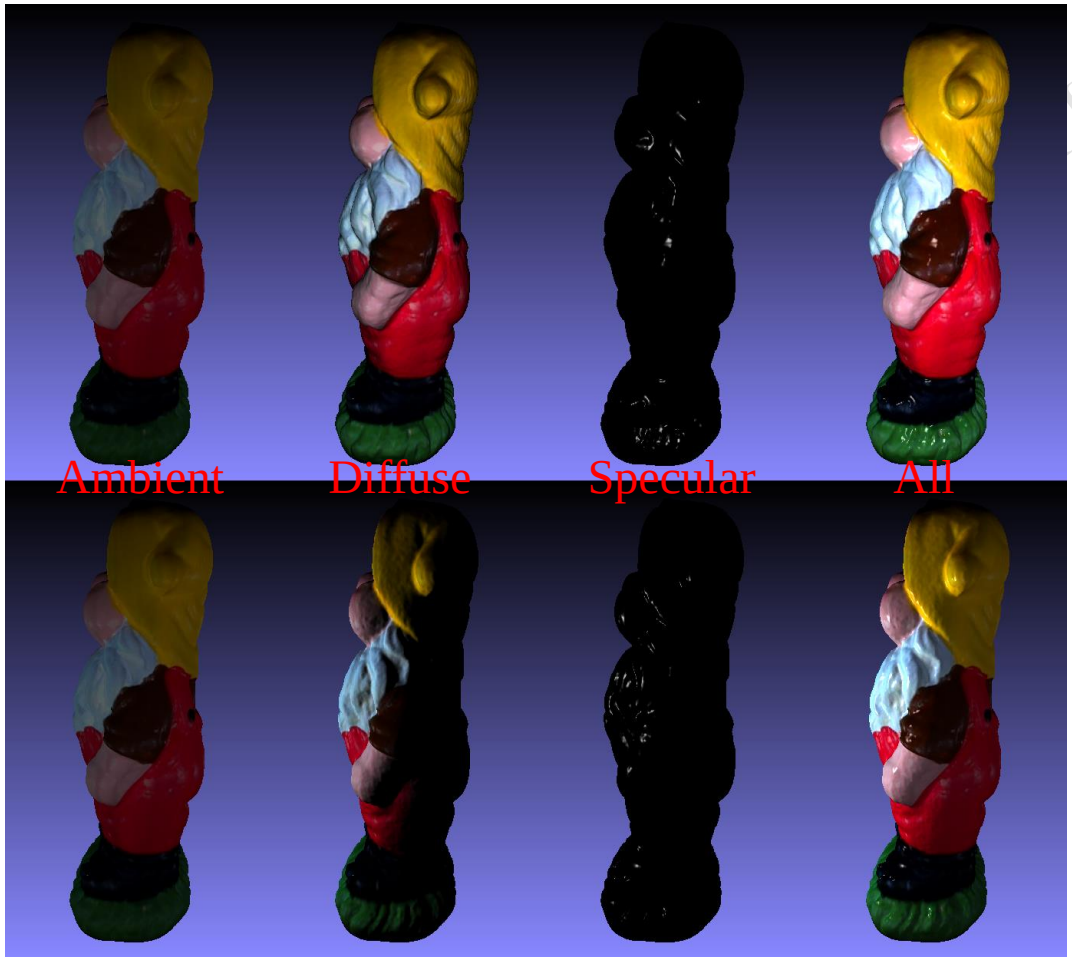
White light at fixed position



White light following the camera position

NTUST. CIIC.

Shading vs Lighting

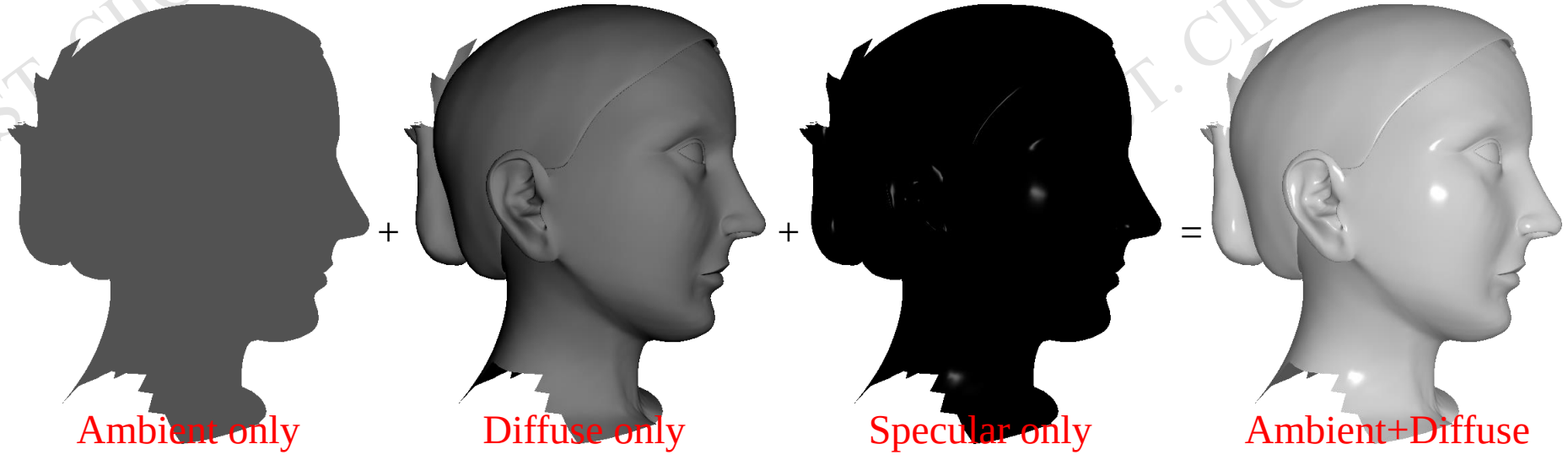


One white light following the camera

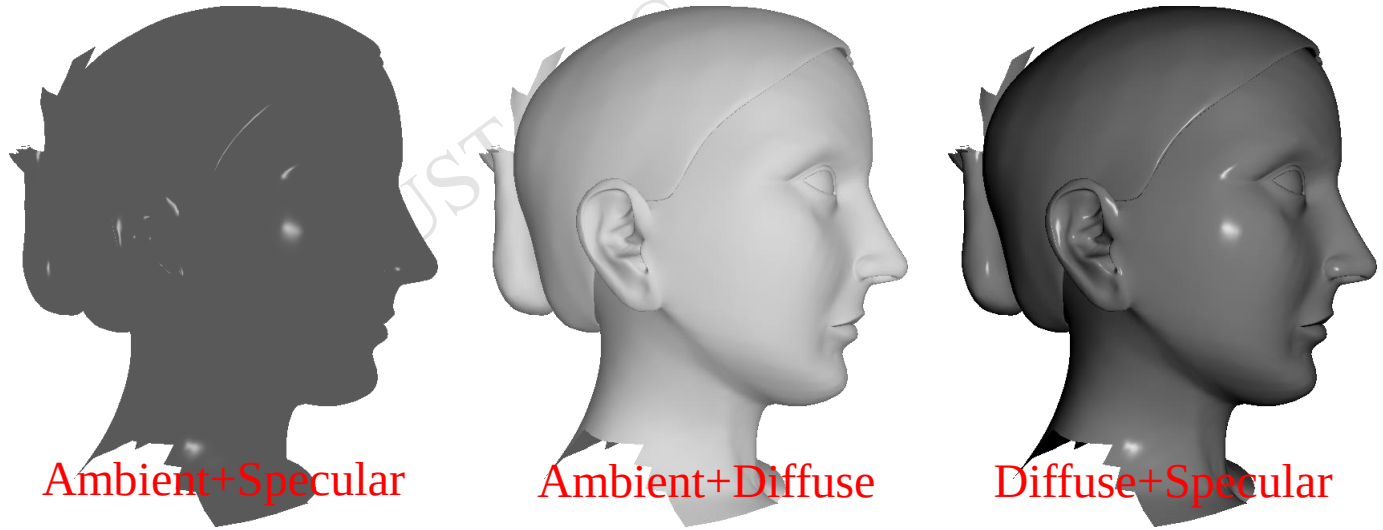
One white light in front of the object (fixed position)

NTUST. CIIC.

Phong shading—monocular example.



Ambient only | Diffuse only | Specular only | Ambient+Diffuse +Specular



Ambient+Specular | Ambient+Diffuse | Diffuse+Specular

NTUST. CIIC.

Phong shading—color example.

Ambient component (color material + white light)

Diffuse component (color material + white light)

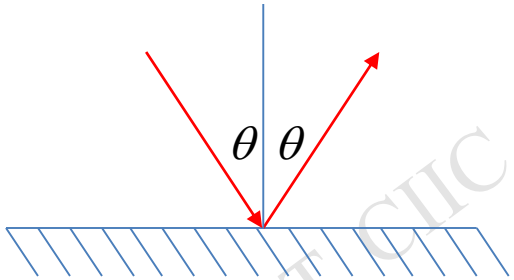
Specular component (white material reflection+ white light)

Final shading result (ambient + diffuse + specular)

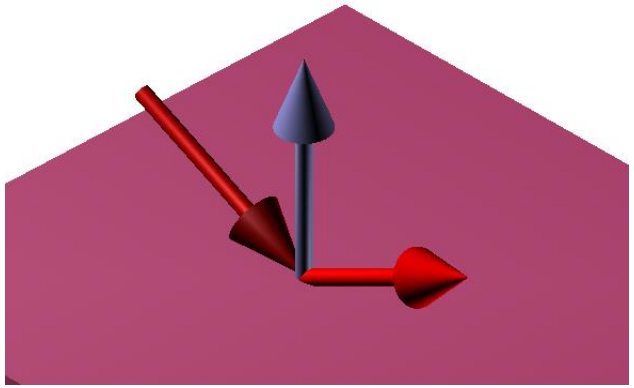
NTUST. CIIC.

Mathematic method—remind

- Reflection rule



2D



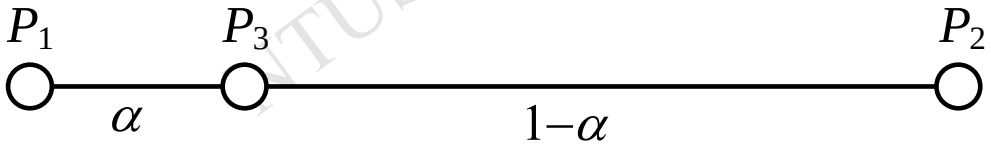
3D

NTUST. CIIC.

Mathematic method—remind

- Linear interpolation
  - To interpolate the value of P3 from P1 and P2.
  - Summation of coefficients must be “1”.

$$P_3 = \frac{A}{A+B}P_2 + \frac{B}{A+B}P_1$$

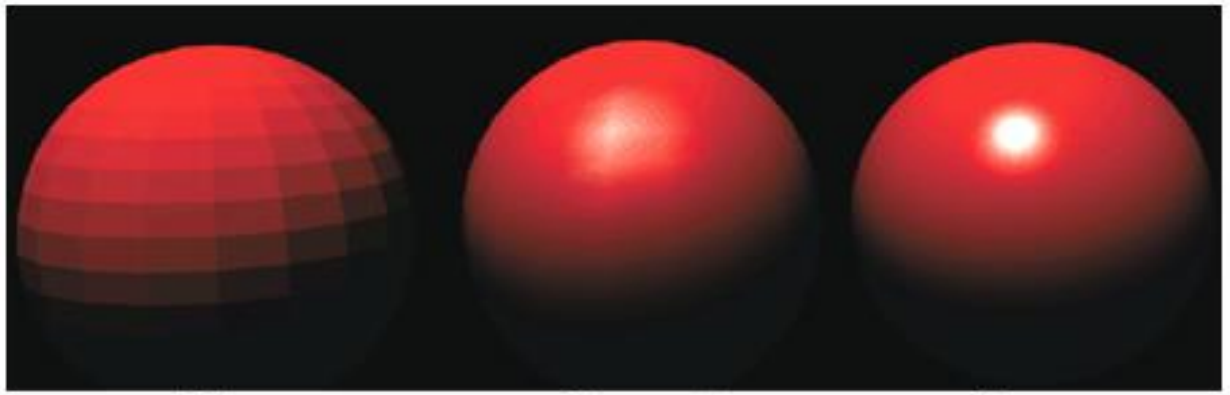


$$P_3 = \alpha P_2 + (1-\alpha)P_1$$

NTUST. CIIC.

Shading algorithm

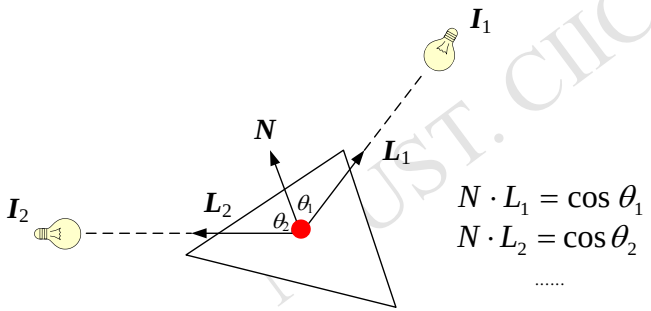
- Flat shading
- Smooth shading
  - Gouraud Shading (by vertex)
  - Phong Shading (by fragment)

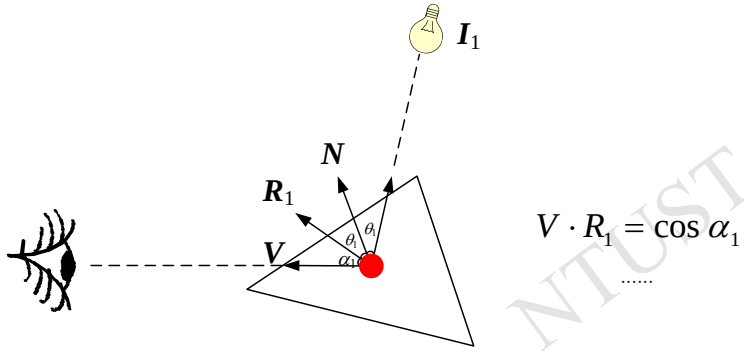


NTUST. CIIC.

How to render a scene

- Decompose into several parts
  - Ambient → fill in a constant color for polygons
  - Diffuse → what is the target? depending on algorithm
  - Specular → what is the target? depending on algorithm



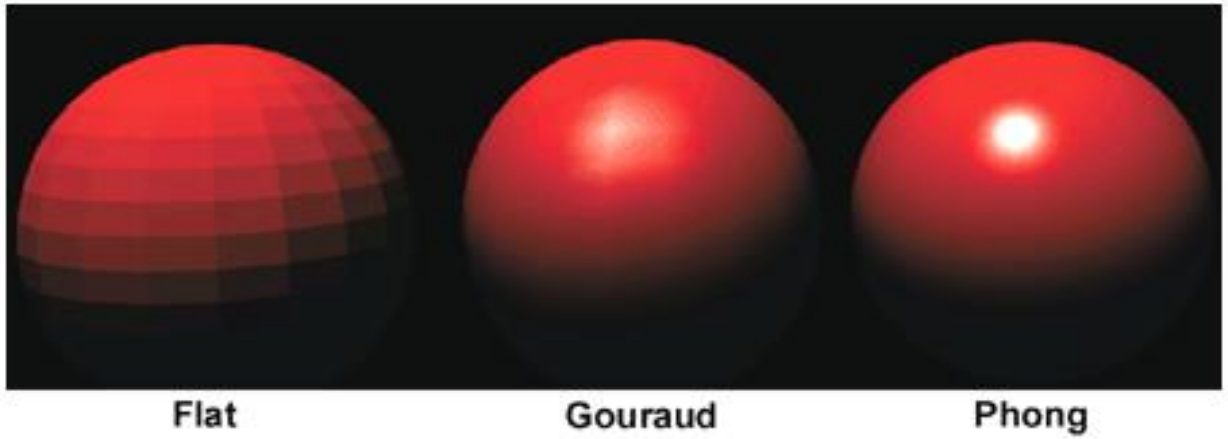


NTUST. CIIC.

How to render a scene

- Three types of famous shading algorithm
  - Flat shading
  - Gouraud Shading
  - Phong Shading

  (Major difference in “diffuse” and “specular”)



NTUST. CIIC.

Flat shading

- What if a faceted object is illuminated only by directional light sources and is either diffuse or viewed from infinitely far away

NTUST.CIIC.

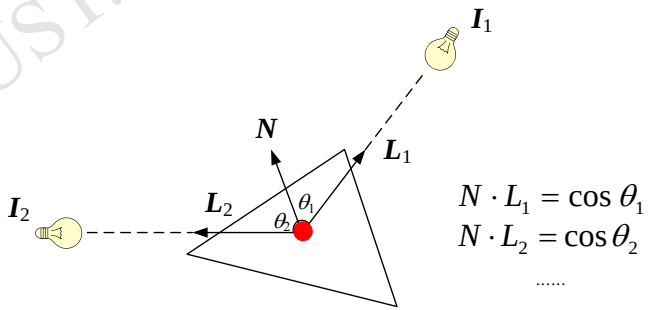
Flat shading

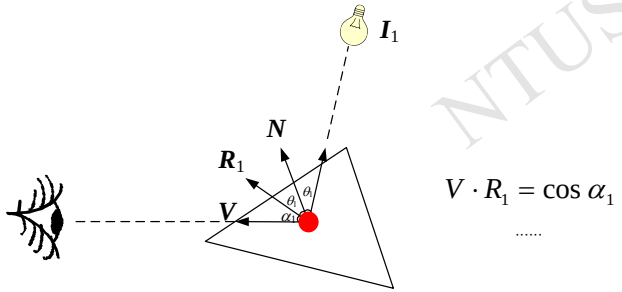
- The normal vectors of all pixels on a polygon are the same.
  - Hint: you need to know the normal vector of a polygon

$$I = I_E + K_A I_{AL} + \sum_i [K_D (N \cdot L_i) I_{Di} + K_S (V \cdot R_i)^n I_{Si}]$$

constant for a polygon

almost constant for a polygon

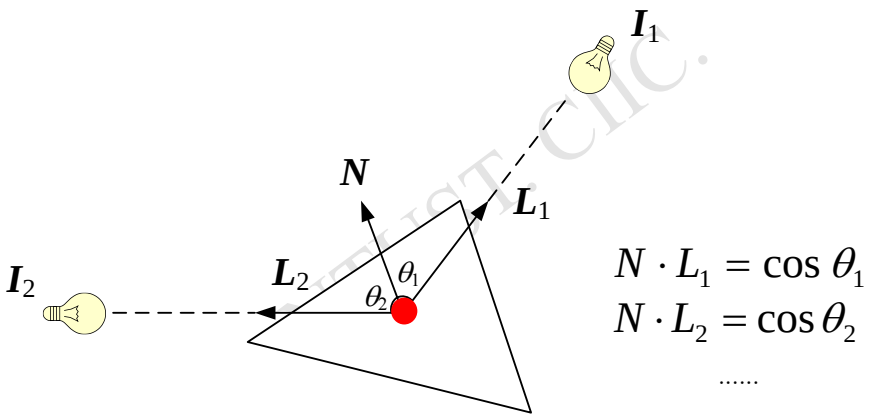




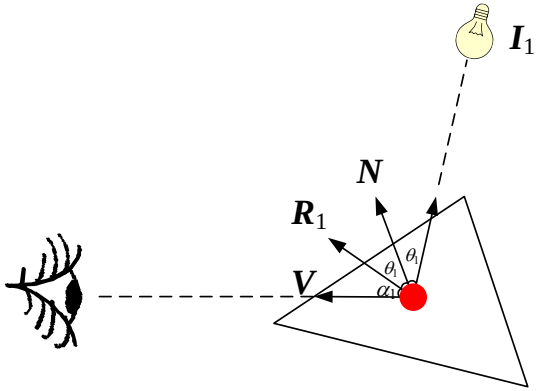
NTUST.CIIC.

Gouraud shading

- Developed by “Henri Gouraud” 1971
  - It is a method to create continuous shading effect by linearly interpolation from the “corners” of a polygon. In other words, for a triangle, calculate the intensities of three vertexes, then interpolate all other pixels of the triangle.



$$N \cdot L_1 = \cos\theta_1$$
$$N \cdot L_2 = \cos\theta_2$$
.....



$$V \cdot R_1 = \cos\alpha_1$$
.....

NTUST. CIIC.

Gouraud shading

- Vertex shading
  - Step 1: Determine the intensity of all vertexes of a polygon
  - Step 2: Linearly interpolate all intensities of the rest pixels in a polygon

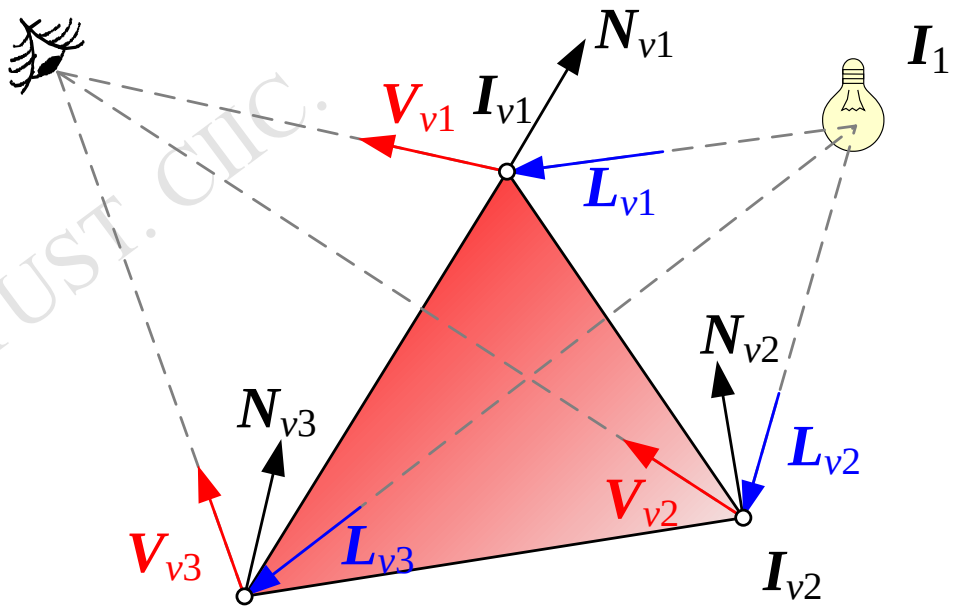
$$I = I_E + K_A I_{AL} + \sum_i [K_D (N \cdot L_i) I_i + K_S (V \cdot R_i)^n I_i]$$

NTUST. CIIC.

Gouraud shading

- Vertex shading
  - Step 1: Determine the intensity of all vertexes of a polygon → calculate intensities I for tree vertexes, says Iv1, Iv2 and Iv3 from equation:

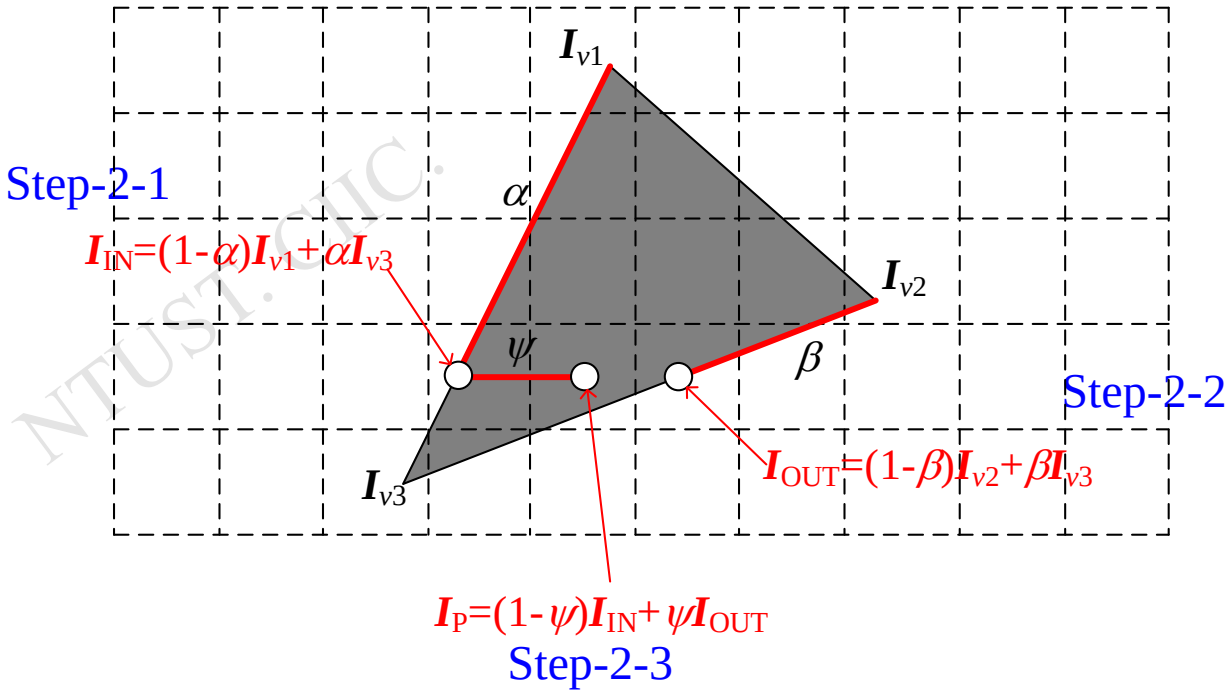
$$I = I_E + K_A I_{AL} + \sum_i [K_D (N \cdot L_i) I_i + K_S (V \cdot R_i)^n I_i]$$



NTUST. CIIC.

Gouraud shading

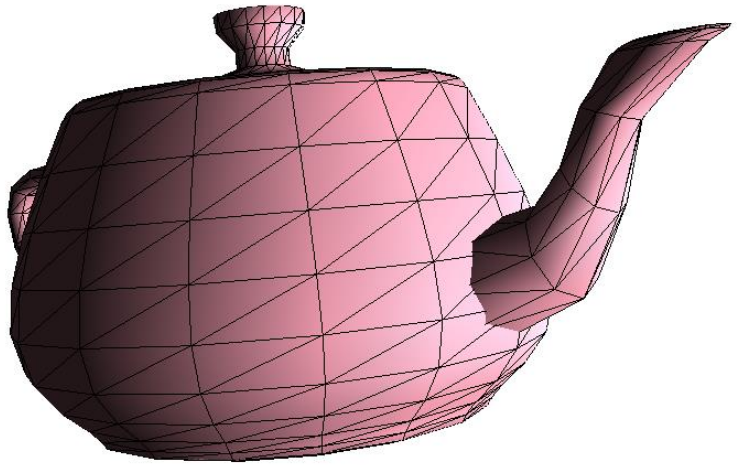
- Vertex shading
  - Step 2: Linearly interpolate all intensities of the rest pixels in a polygon. Normally, integrate with scan line algorithm. Interpolate boundary pixels, then internal pixels.



NTUST. CIIC.

Gouraud shading

- Phenomenon
  - Since all internal pixels are interpolated from the vertexes of the polygon, the brightest intensity should be on either vertex of edge.



NTUST. CIIC.

Phong shading

- Developed by “Bui Tuong Phong” in 1973.
  - This algorithm mainly interpolates pixel’s normal from tree normals of vertexes in a triangle. By using the interpolated normal, the pixel intensity can be determined by

$$I = I_E + K_A I_{AL} + \sum_i [K_D (N \cdot L_i) I_i + K_S (V \cdot R_i)^n I_i]$$

NTUST. CIIC.

Phong shading

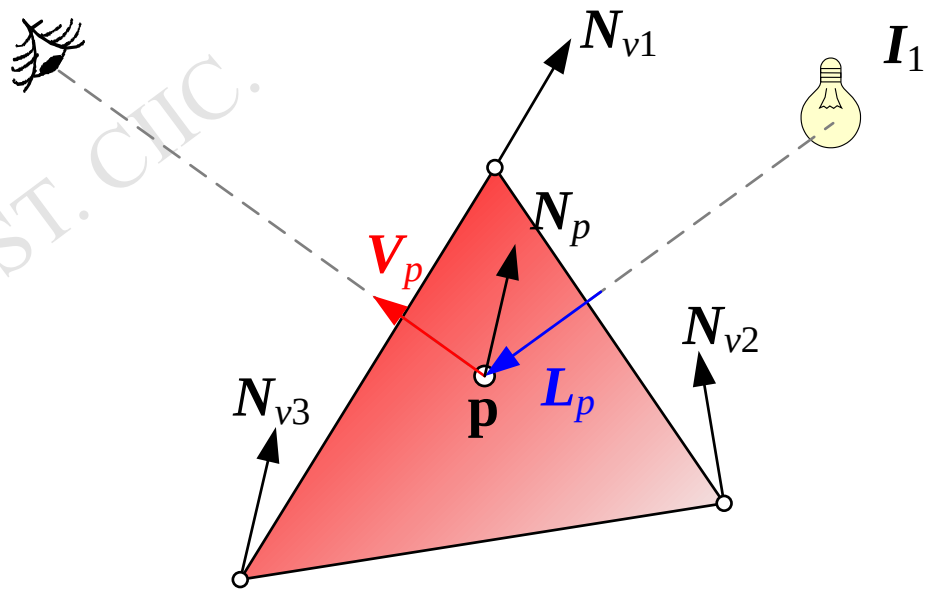
- Fragment shading
  - Summary:
  - Step 1: Interpolate the normal vector for every pixel on a triangle by three vertexes' normal
  - Step 2: By using the interpolated normal to calculate the pixel intensity by following equation:

$$I = I_E + K_A I_{AL} + \sum_i [K_D (N \cdot L_i) I_i + K_S (V \cdot R_i)^n I_i]$$

NTUST. CIIC.

Phong shading

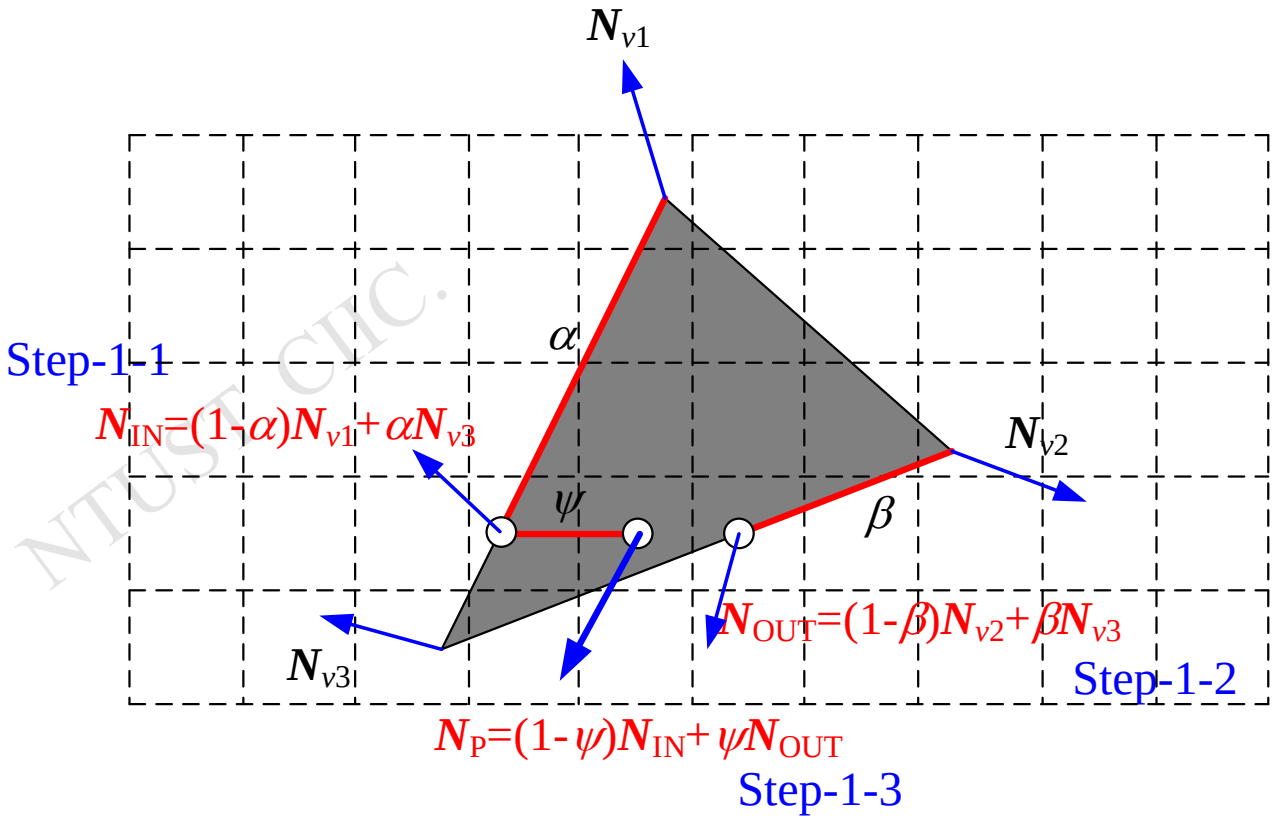
- Example:
  - Np is the normal vector of a desired pixel on a triangle (to be determined). Note: Nv1, Nv2 and Nv3 are known for interpolating Np. Then, Vp and Lp are also known for further determining the intensity of pixel p.



NTUST. CIIC.

Phong shading

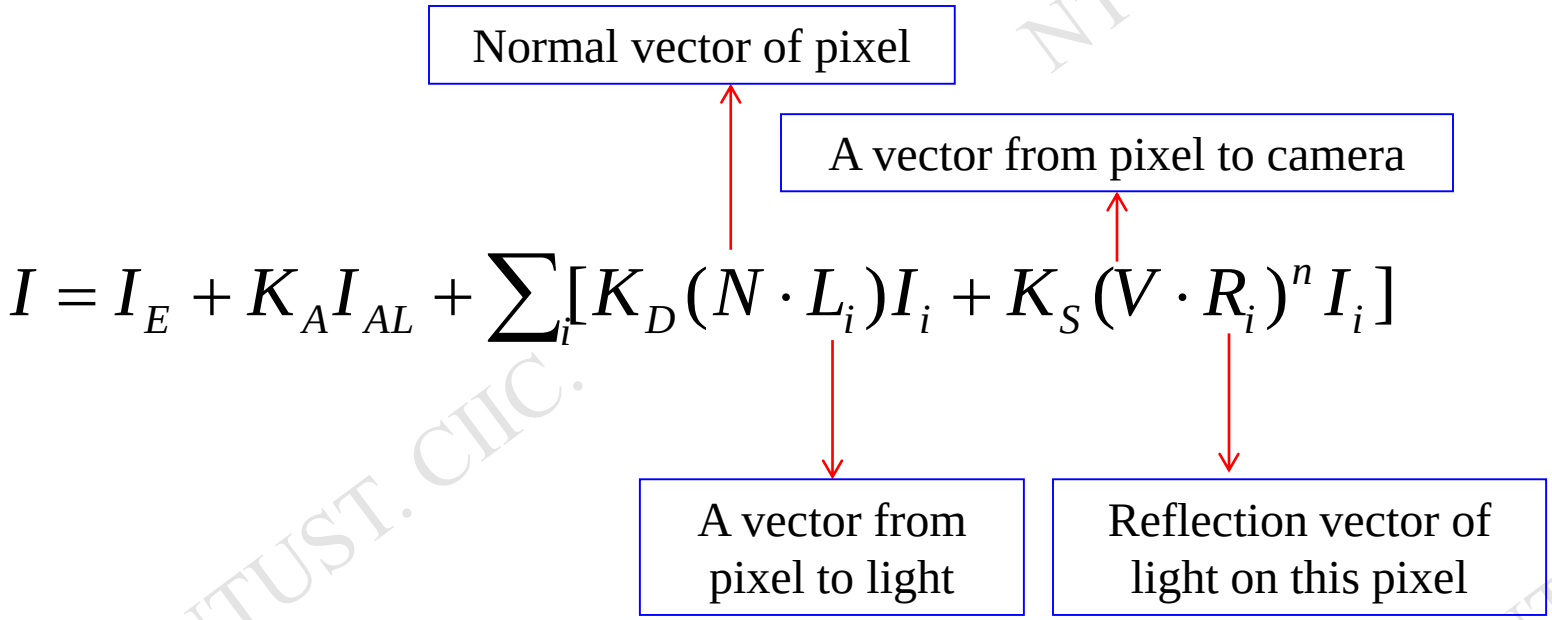
- Example: cont.
  - Linearly interpolate the vector on boundaries, then interpolate the internal normal Np.



NTUST. CIIC.

Phong shading

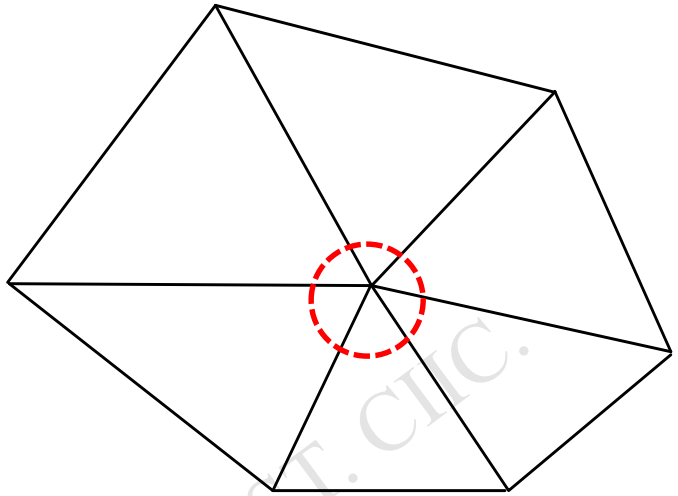
- Finally, calculate every pixel's intensity



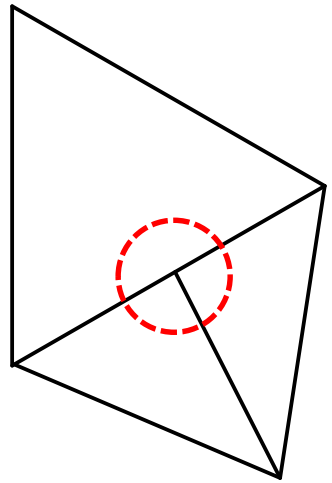
NTUST. CIIC.

Data format

- Manifold vs non-manifold



Continuous



Discontinuous

NTUST. CIIC.

3D data format—what we need in shading

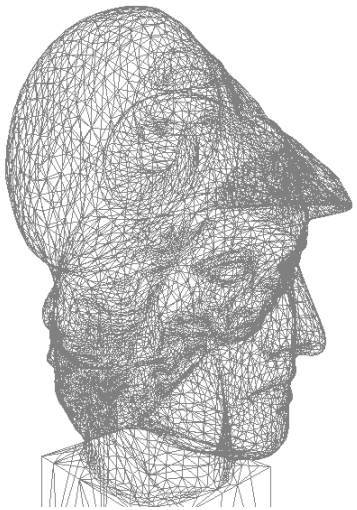
- In summary
- Flat shading → surface normal
- Gouraud shading → vertex normal
- Phong shading → vertex normal

NTUST. CIIC.

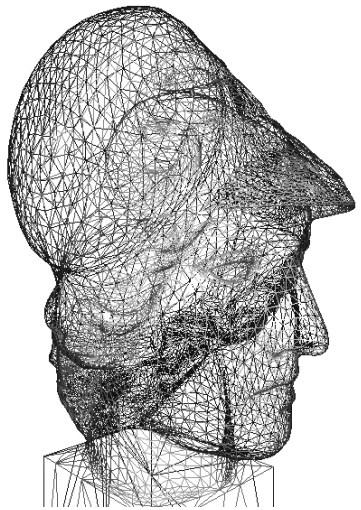
Shading effect

Constant color vertex

Shading vertex



Constant color line



Shading line

NTUST. CIIC.

3D data format—remind

File format	Color resolution	Surface color representation	Note
PLY	32 bit on vertex	Interpolate in triangle	**PLY**(Stanford triangle format
OBJ	32 bit on vertex 32 bit texture map	Interpolate in triangle Interpolate in image space	**OBJ**(wavefront object)
STL	16 bit on facet	Uniform color in a triangle	**STL**(stereo lithography)
VRML	Similar to OBJ	Similar to OBJ	Virtual Reality Makeup Language

NTUST. CIIC.

Graphics of game in previous decades

NTUST. CIIC.

Flat shading

Sega game

Gouraud Shading (Bezier surface)

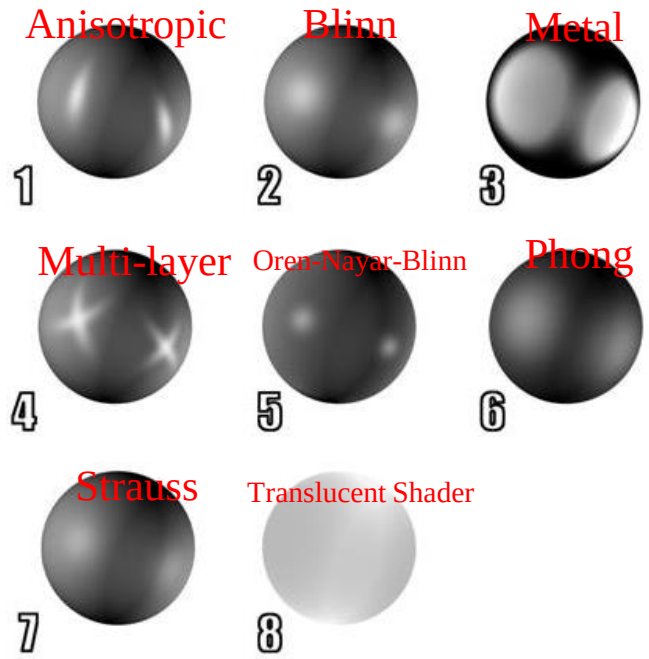
Nintendo N64 Game

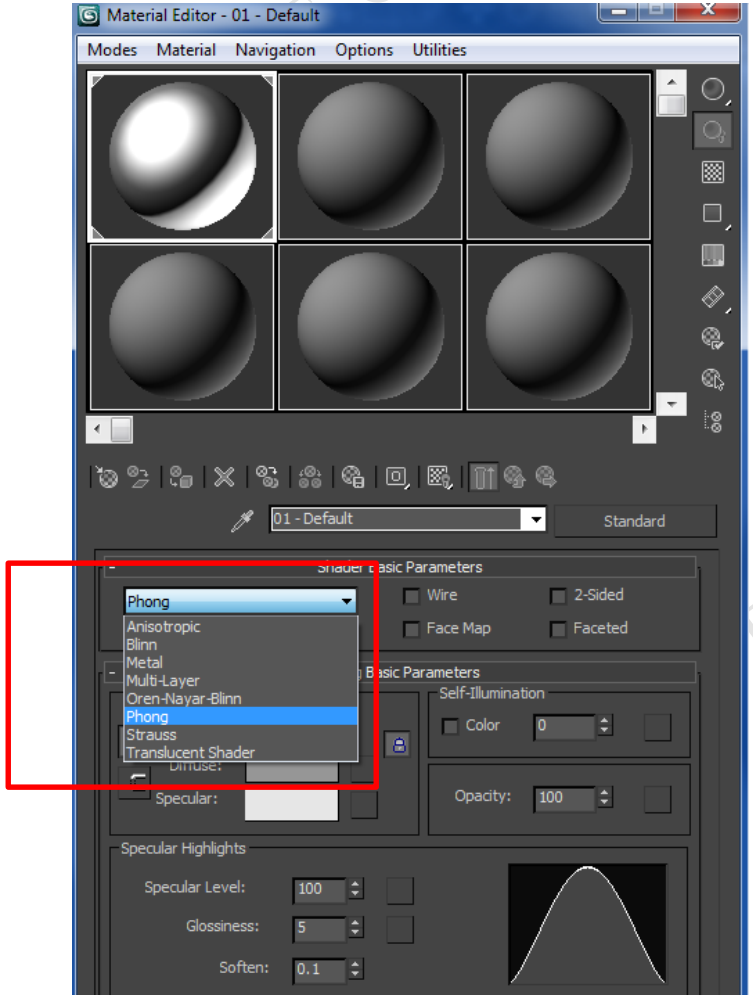
Phong shading or fragment shading

Image from EA game

Rendering method in commercial product

- Phong shading
- Modified Phong shading
- …





Rendering engine

- VFX
- Metal Ray
- iRay
- POV Ray
- Keyshot
- Render farm
- ….

NTUST. CIIC.

CIT 色彩與照明科技研究所
Graduate Institute of
Color and Illumination Technology

Find us on: facebook

NTUST. CIIC.

This photo is licensed under CC BY-ND