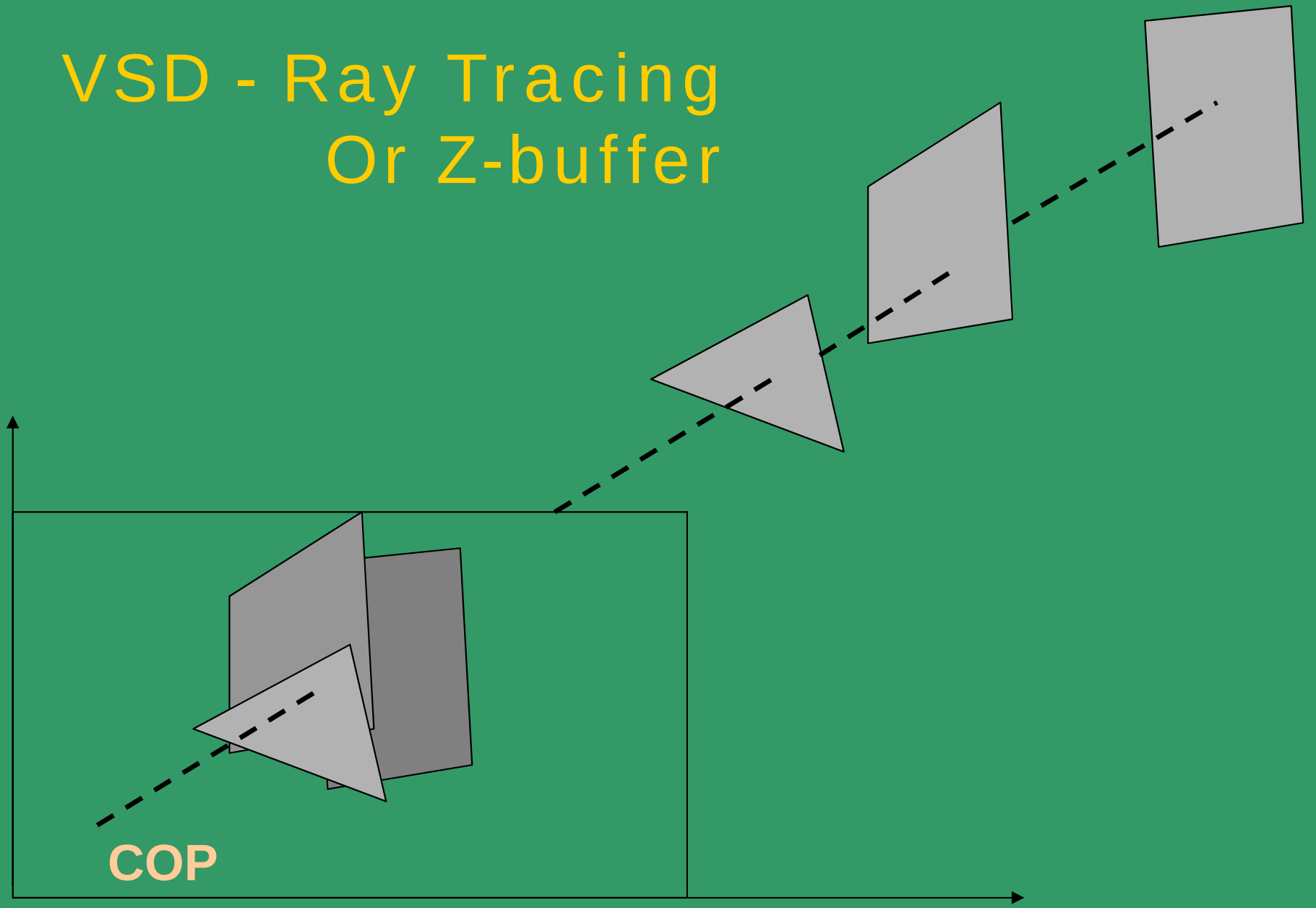
Illumination

and

Shading

VSD - Ray Tracing
Or Z-buffer



Illumination and Shading

Shading is a HARD problem.

Creating a virtual reality (say, a classroom) of a real scene, involves:

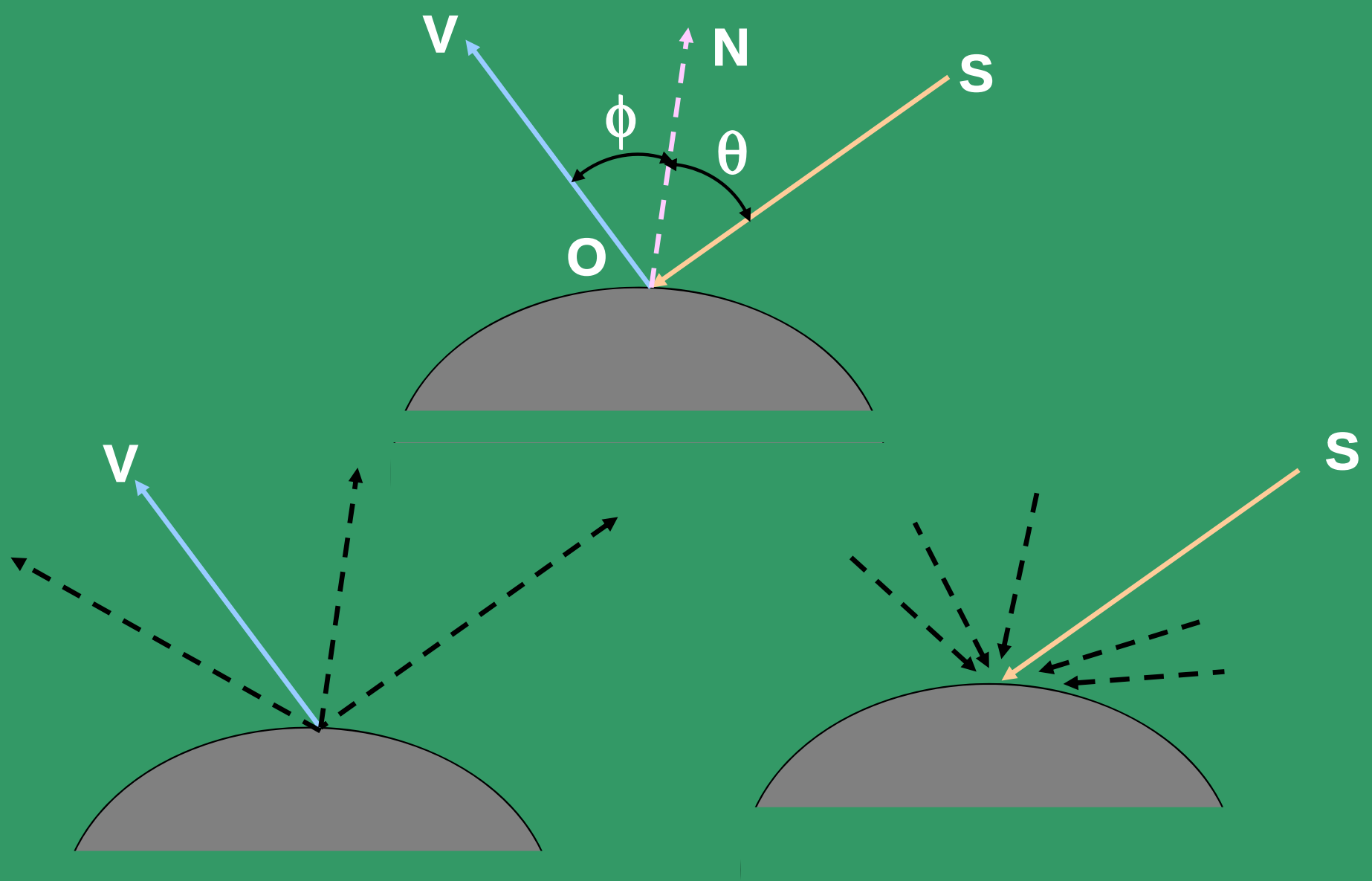
- **Modeling and positioning of several complex objects.**
- **Determine the VSD, project the view w.r.t. the viewer**
- **Obtain shading using surface normals, surface properties and light sources.**
- **Obtain shadows from occlusion**

In a real world environment, light rays flow in almost infinite directions, some direct from the source and some reflected from shiny surfaces of objects.

A real world image taken using a digital camera will only capture a small subset of the light rays (or light energy) passing through a small area.

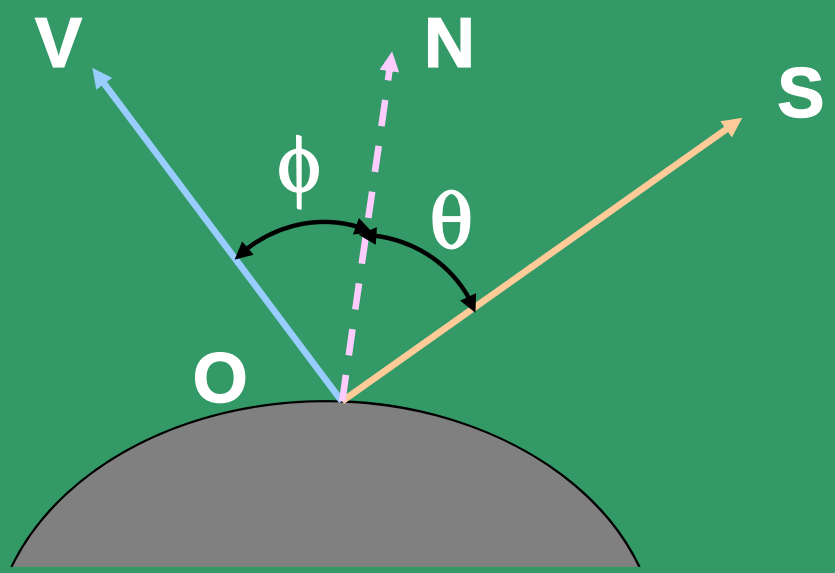
To accurately construct a picture of this room via computer graphics, we have to simulate this illumination process and be able to calculate the shading at each point of each surface in our scene.

Often an approximated view is generated using many complex formulations and algorithms.



The process of Illumination, involves surface normal, source and viewer directions

$$I(V_{x,y}) = F(\phi, \theta, \rho, S_l) + A(V)$$



The process of Illumination, involves surface normal, source and viewer directions

Typical compromises and simplifications:

- **Uniform Media**
- **Opaque objects**
- **No inter-reflections (approximated by ambient light)**
- **Point light sources**
- **Simple color model**

Illumination consists of three parts:

- **Ambient**
- **Specular**
- **Diffuse**

Ambient Light

This is a case of diffuse, non-directional source of light. This results from the effect of multiple reflections of light from many surfaces present in the environment.

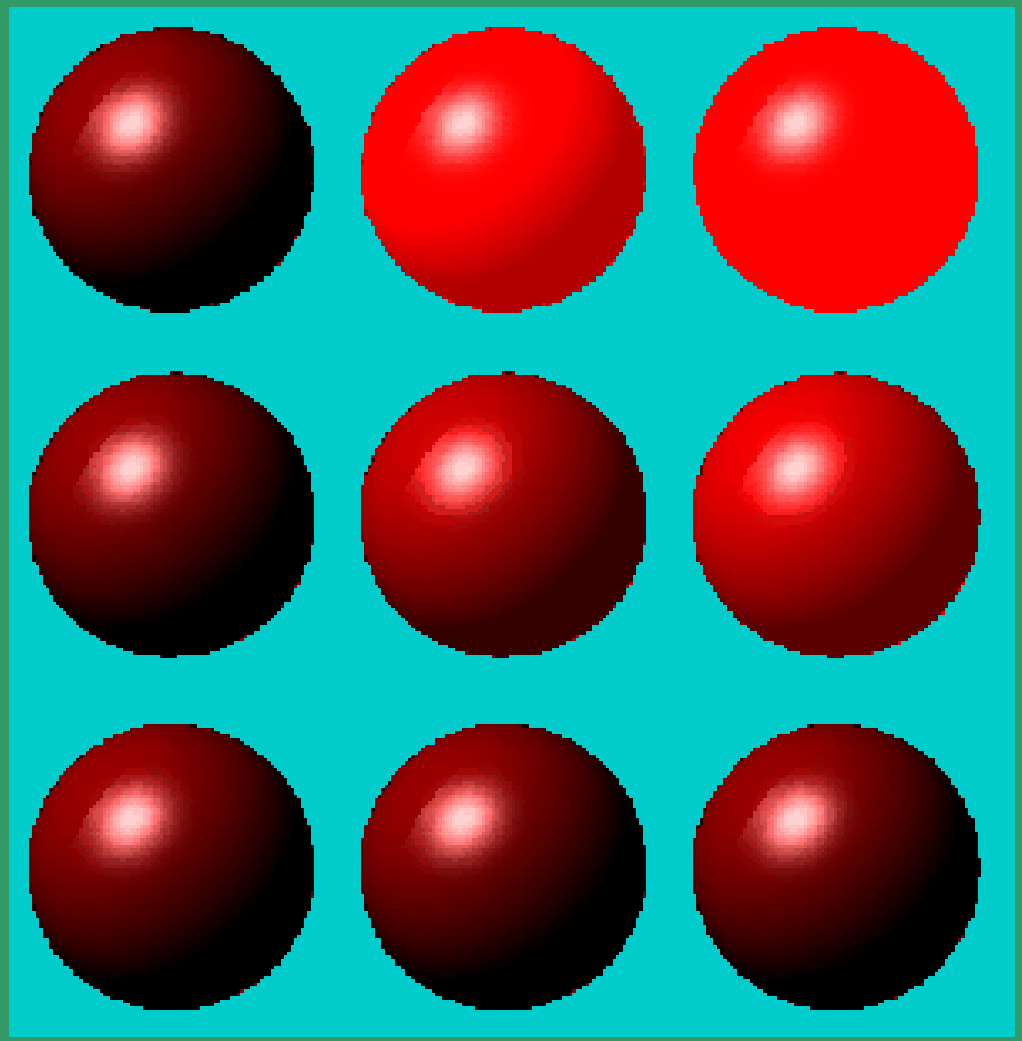
Ambient light is assumed to impinge equally on all surfaces from all directions.

The illumination equation is: $$I = I_a K_a$$

where, I_a is the intensity of ambient light and is assumed to be constant for all objects.

K_a is the *ambient-reflection coefficient* ($0 \leq K_a \leq 1$), a property of the material.

Effect of Ambient Light on a typical surface – object surface has red color



Diffuse Reflection

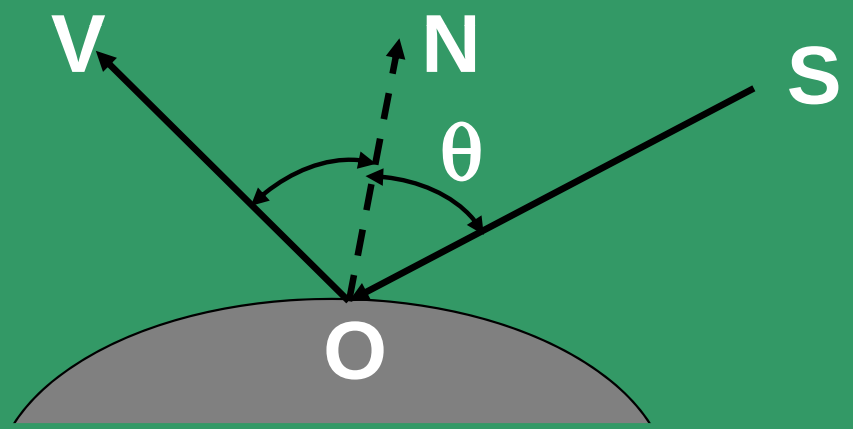
Assume a point light source, whose rays emanate uniformly in all directions from a single point.

The object's brightness varies from one place to another, depending on the direction of the light source (and also distance to some extent).

A *Lambertian surface* (examples: dull, matte surfaces, snow, chalk, projection and movie screens, uniformly painted walls etc.) exhibits diffuse reflection.

These surfaces appear equally bright from all viewing angles, because they reflect light with equal intensity in all directions.

For a Lambertian surface, the amount of light seen by the viewer is independent of the viewer's direction, as follows:



$$I = I_S K_d \cos(\theta)$$

The diffuse illumination equation:

$$I = I_S K_d \cos(\theta)$$

where, I_s is the point light source's intensity, K_d ($0 \leq K_d \leq 1$) is the material's *diffuse-reflection coefficient*, and $0 \leq \theta \leq 90^0$.

The above equation can be re-written as:

$$I = I_S K_d \vec{N}.\vec{S}$$

Unless you are in a perfect dark room (used for developing films), a more realistic illumination equation (using knowledge studied so far) is:

$$I = I_a K_a + I_S K_d \vec{N}.\vec{S}$$

Effect of distance (attenuation):

$$I = I_a K_a + f_{att} I_S K_d \vec{N}.\vec{S}$$

$$f_{att} = \frac{1}{d_L^2};(or)$$

$$f_{att} = min(\frac{1}{c_1 + c_2 d_L + c_3 d_L^2}, 1)$$

Atmospheric Attenuation

This is a process of simulation of the effect of depth cueing. Distant objects are rendered with lower intensity than the closer ones.

Use NPC to pre-define the parameters:

Scale factors – $0 \leq (S_t , S_b) \leq 1.$

Range of depths – $Z_t , Z_b.$

Depth-cue color – $I_{dc\lambda}$.

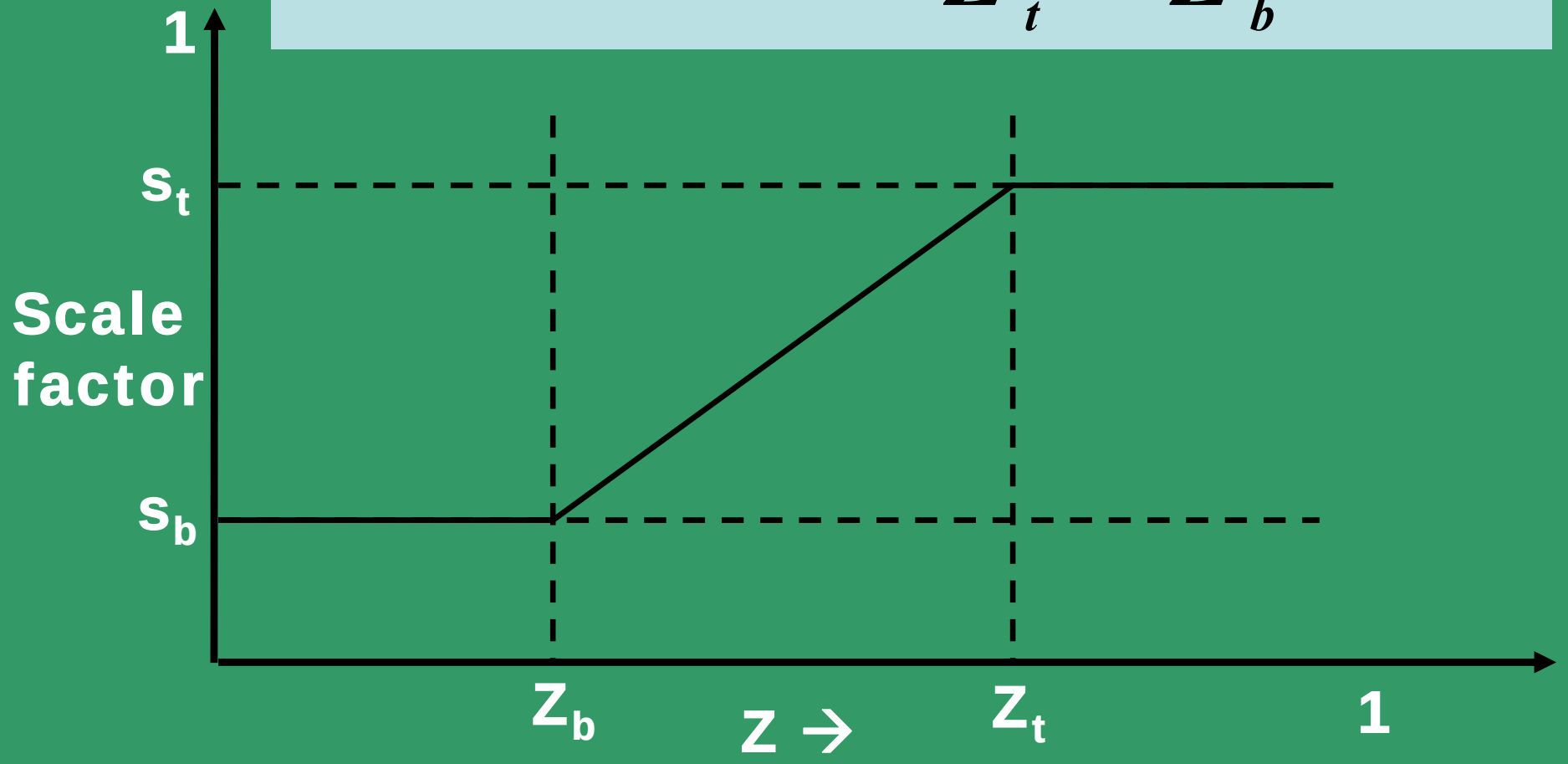
Original Intensity – I_λ .

Depth-cued value of intensity – I'_λ :

$$I'_\lambda = s_0 I_\lambda + (1 - s_0) I_{dc\lambda}$$

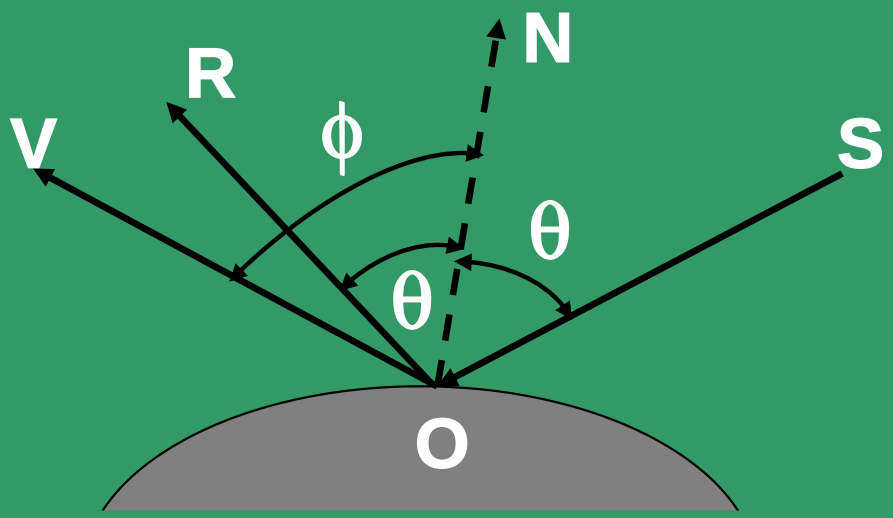
where

$$s_0 = s_b + \frac{(Z_0 - Z_b)(s_t - s_b)}{Z_t - Z_b}$$



Specular Reflection

Specular reflection is observed on a shiny surface.



Shiny surface reflects light unequally in different directions. Mirror is an example of a perfect shiny surface. Other examples of shiny (imperfect, non-mirror type) surfaces are; apple, shiny plastics, gold and silver coated metal surfaces etc,

Phong's Illumination Model:

$$I_\lambda = I_{a\lambda} K_a O_{d\lambda} + f_{att} I_{p\lambda} [K_d O_{d\lambda} \cos\theta + K_s O_{s\lambda} \cos^m \alpha]$$

$O_{d\lambda}$ – represents Object's diffuse color.

I_p – Point light Source Intensity.

K_s – Specular Reflection Coefficient;
$0 \leq K_s \leq 1$

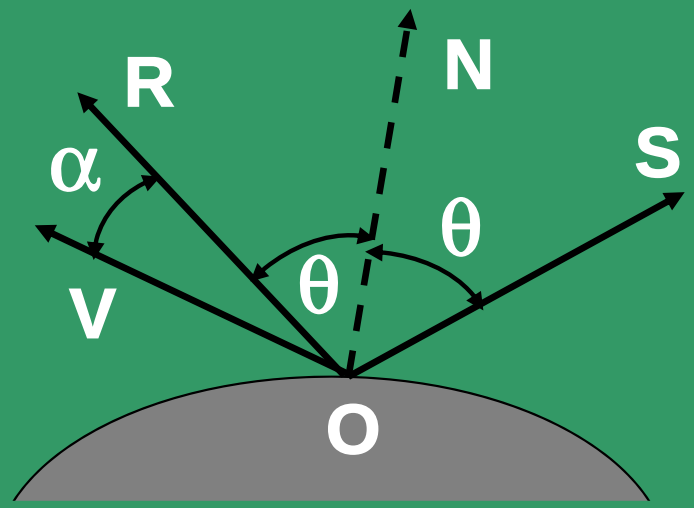
$O_{s\lambda}$ – represents Object's specular color.

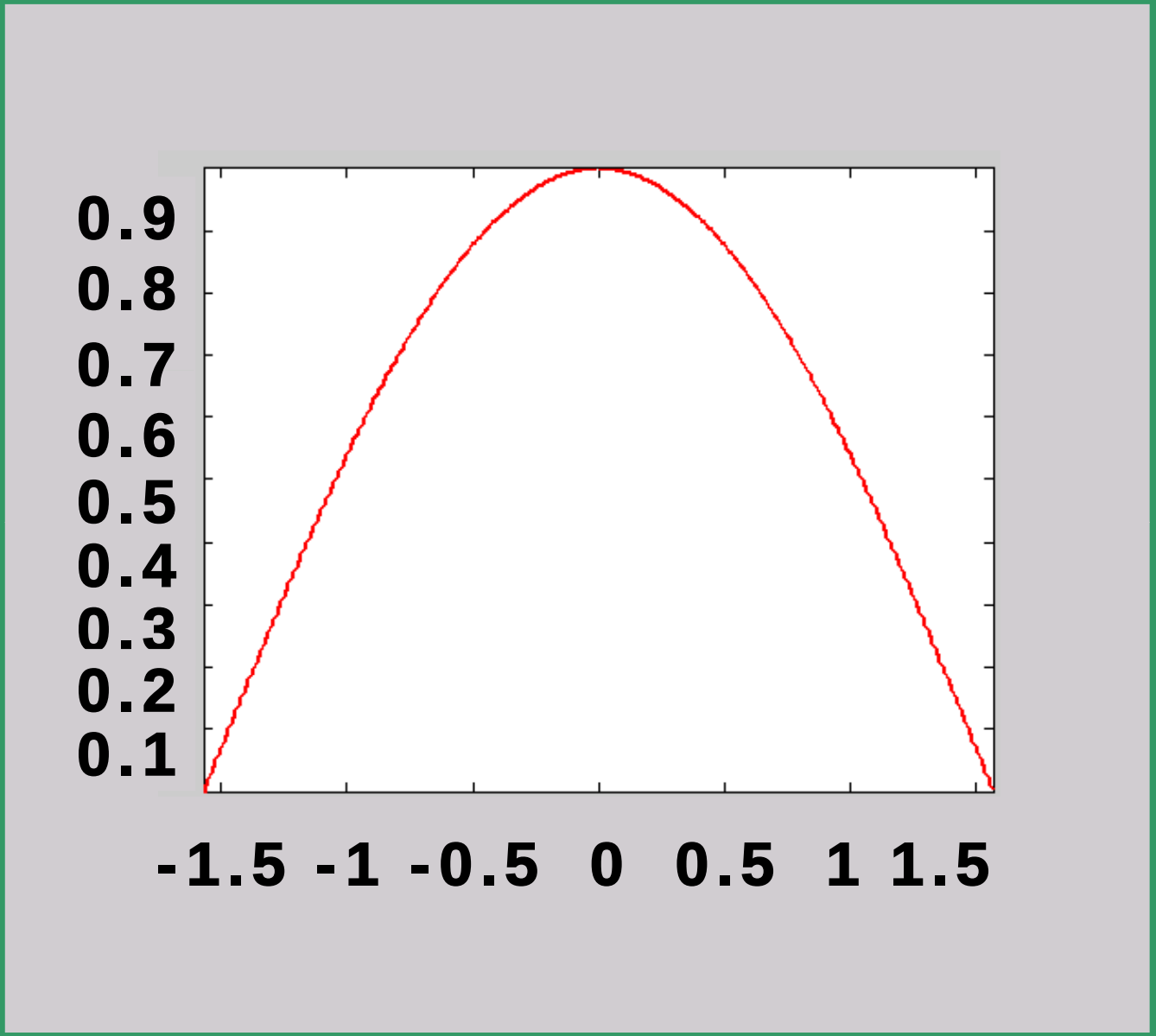
Obtaining α:

$$R + S = 2N(N.S);$$ *(work it out)*

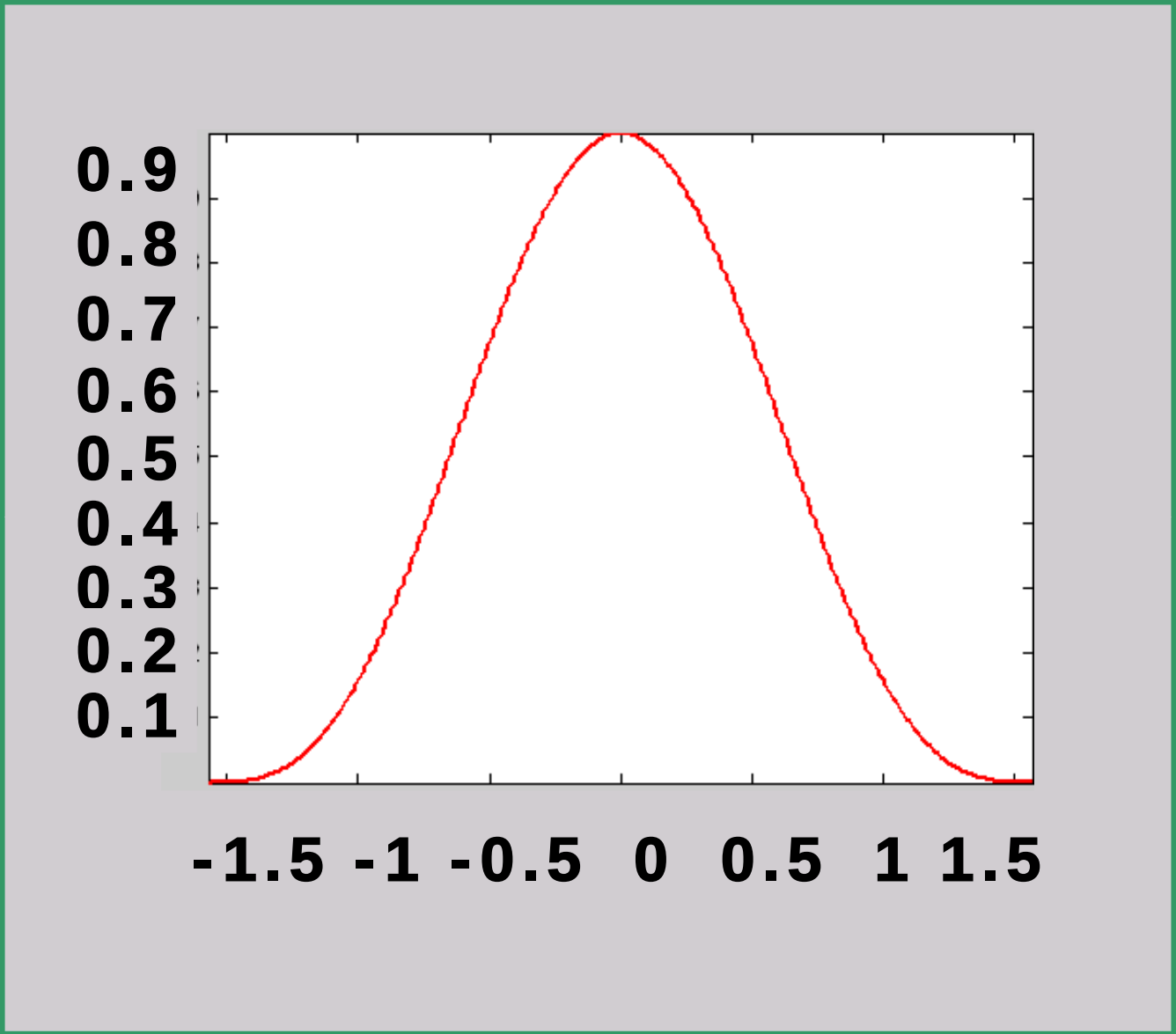
$$R = 2N(N.S) - S;$$

$$\cos\alpha = R.V = V.(2N(N.S) - S)$$

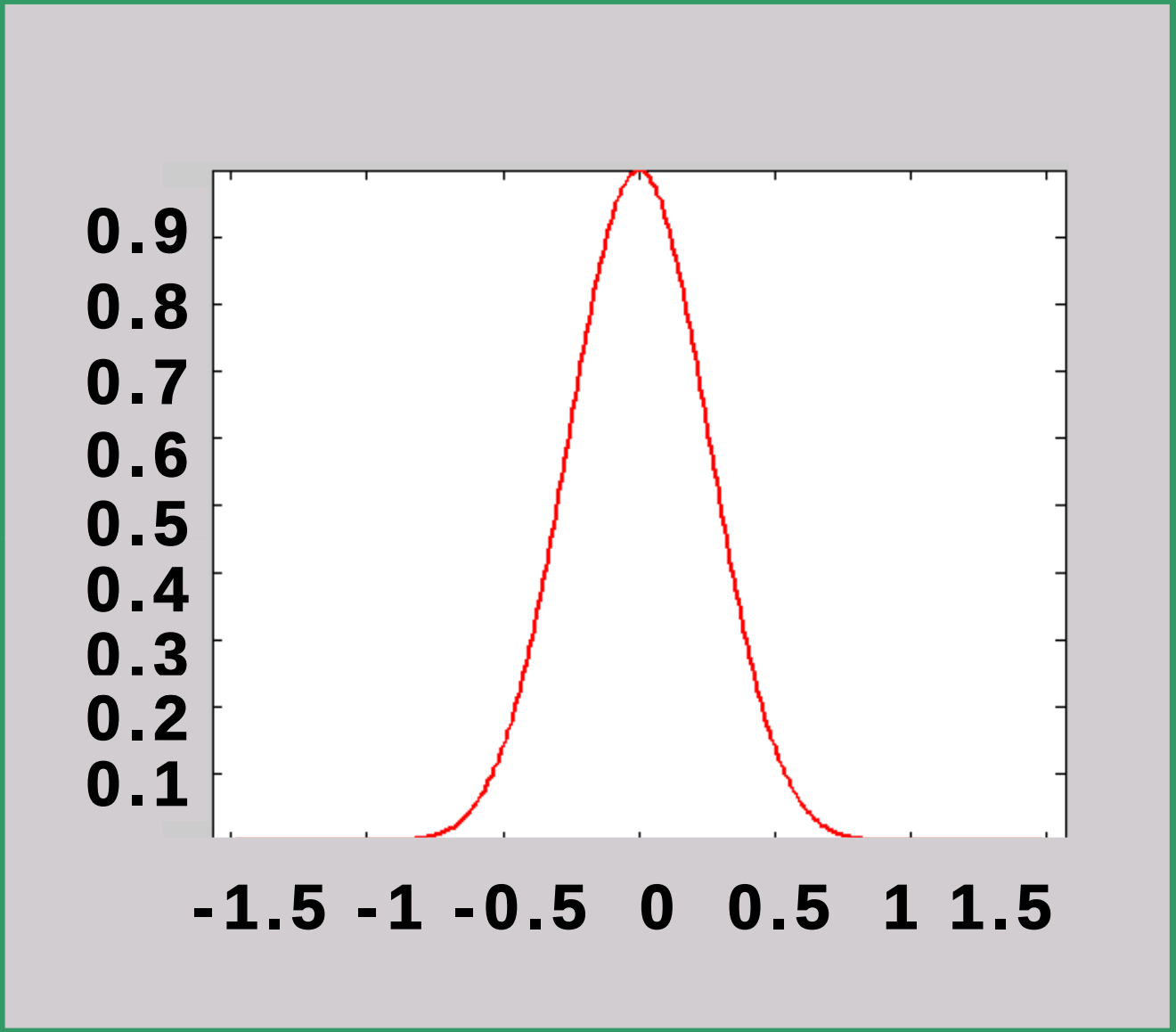




m = 1



m = 3

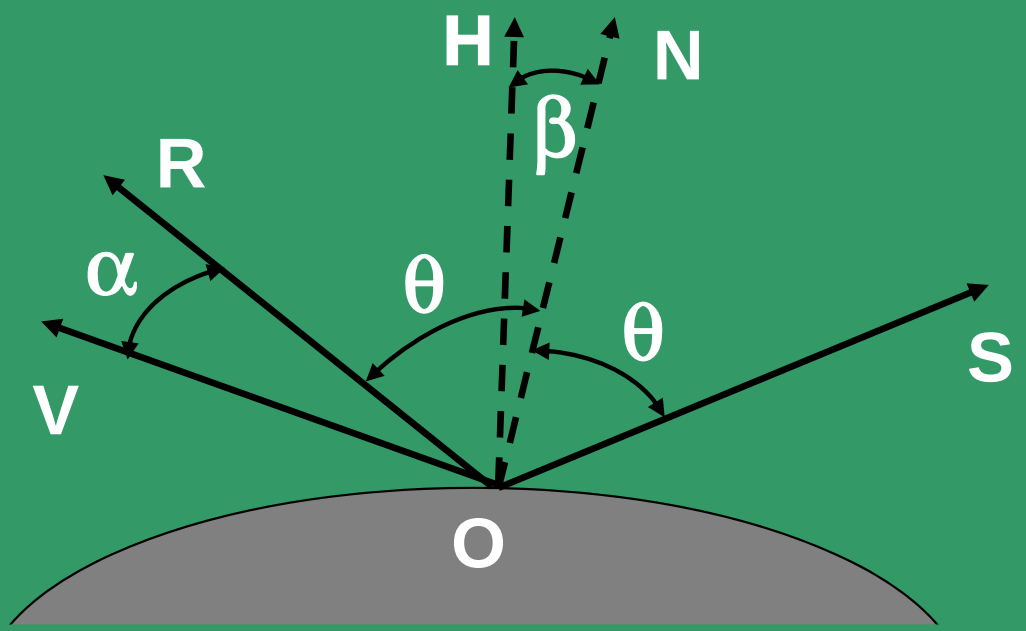


m = 11

If the light source is at infinity, the angle given by N.S is constant, whereas R.V varies across a polygon.

For curved surfaces and light source at a finite distance, both these factors vary across a surface.

Simplify by the use of ***HALF-WAY VECTOR***, ***H***, if the light source and viewer are both at infinity, using:



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

The specular term: $(N.H)^m$

If there are L light sources, then the formula is:

$$I_\lambda = I_{a\lambda} K_a O_{d\lambda} + \sum_{1 \le i \le L} f_{att} I_{p\lambda} [K_d O_{d\lambda} \cos\theta + K_s O_{s\lambda} \cos^m \alpha]$$

You may need to :

normalize the color/intensity value,
or clamp it,
or choose coefficients appropriately
or display, say, in log scale (when?).

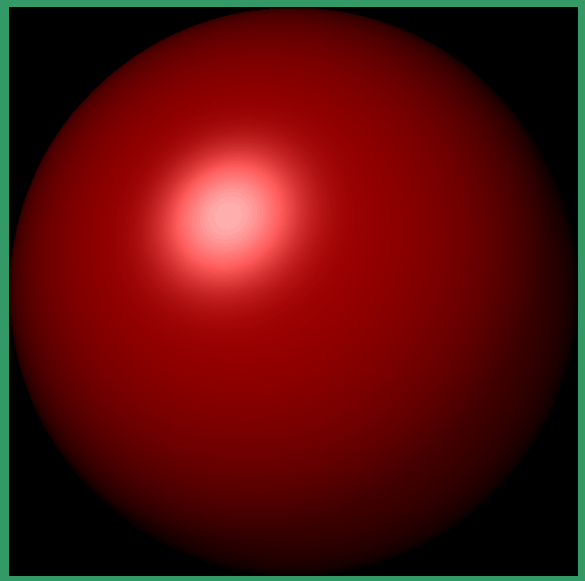
Reflectance Properties

LAMBERTIAN MODEL

$$\phi(N,S,V) = K_d \cos\theta$$

PHONG MODEL

$$I_\lambda = \phi(N,S,V) = K_d \cos\theta + K_s \cos^m \alpha$$

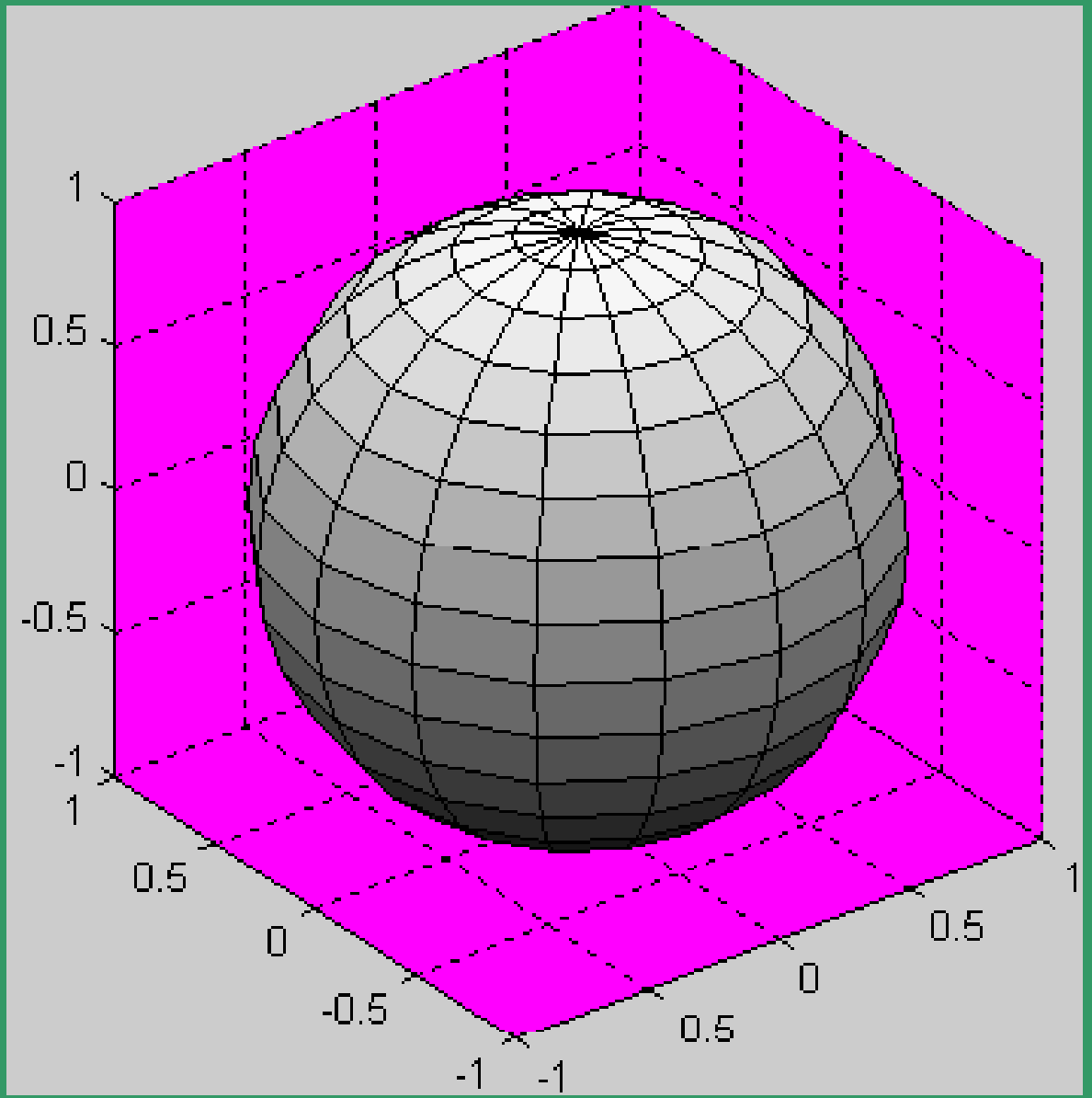


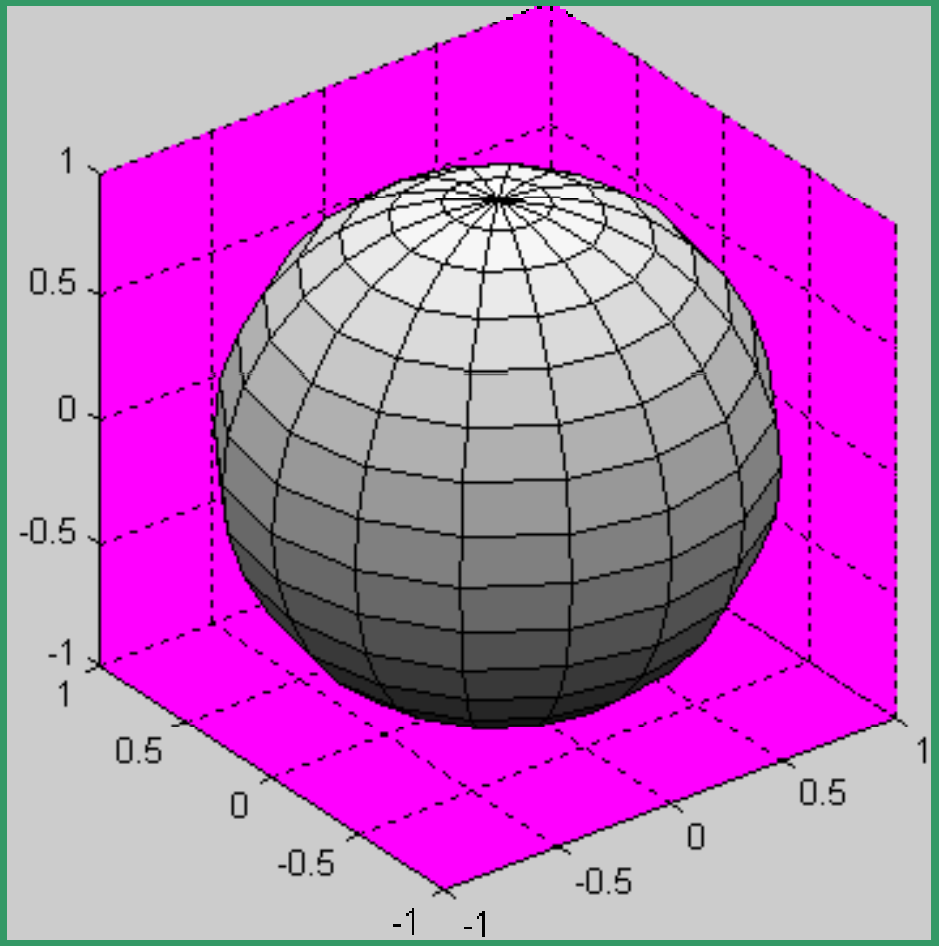
K_d=0.3, K_s =0.7, m=2; K_d =0.7, K_s =0.3, m=0.5

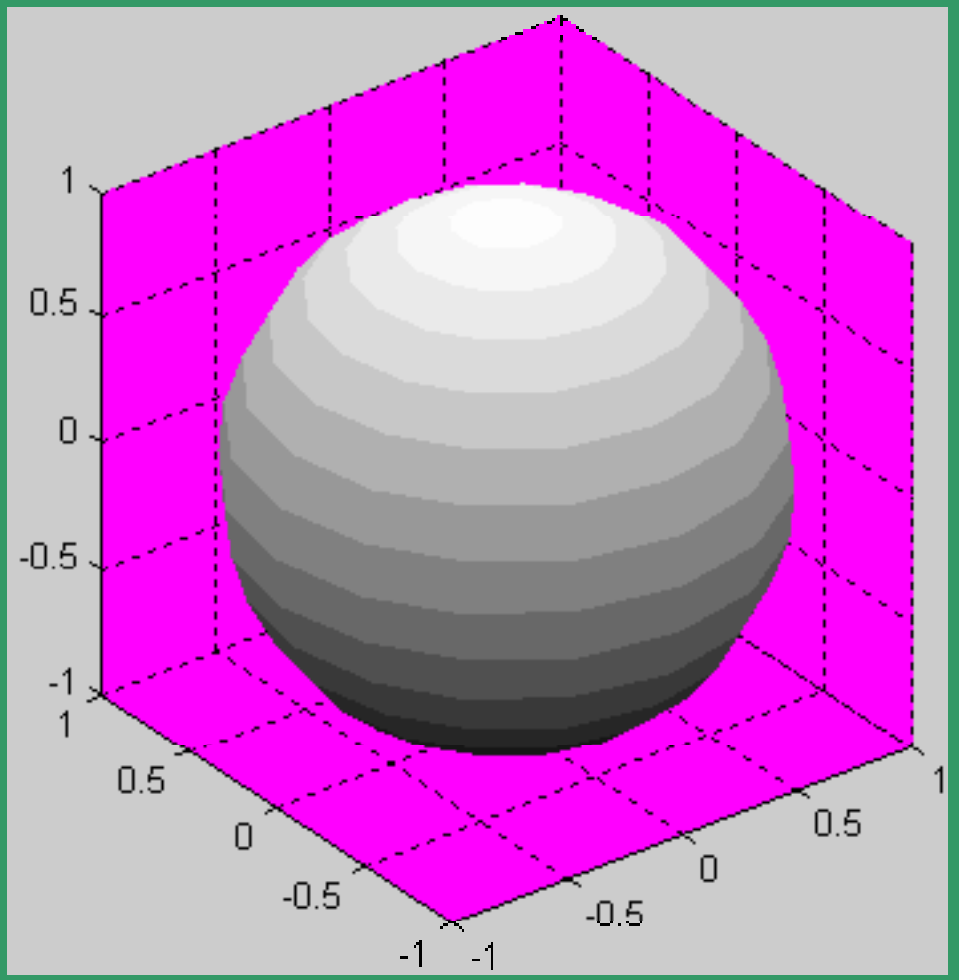
Shading Model for Polygon

Constant Shading

-Also called *faceted shading, or flat shading*.




1
0.5
0
-0.5
-1
1
0.5
0
-0.5
-1
1
0.5
0
-0.5
-1

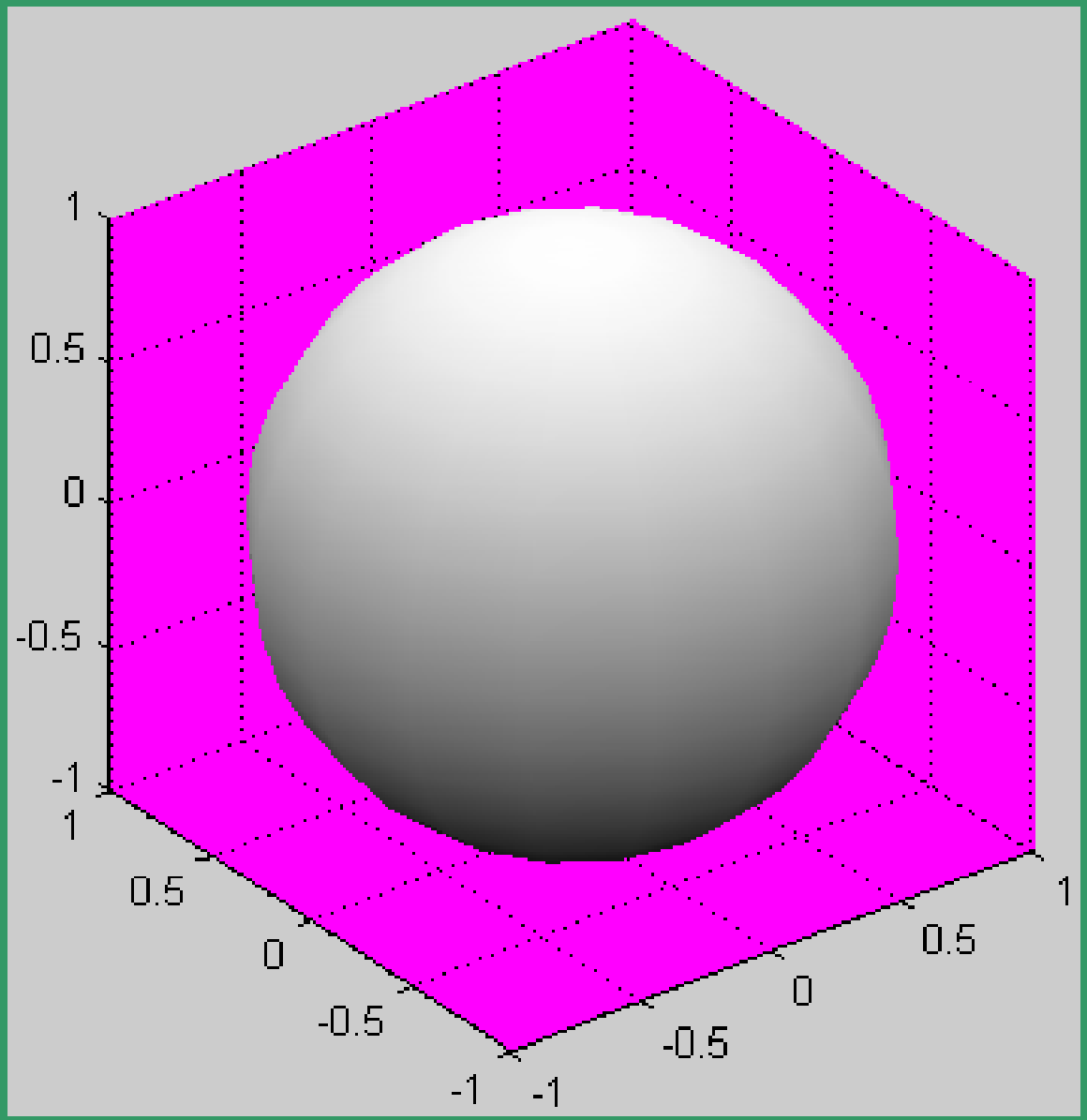

1
0.5
0
-0.5
-1
1
0.5
0
-0.5
-1
1
0.5
0
-0.5
-1

Faceted Shading:

Determine a single intensity value and use it to shade an entire polygon.

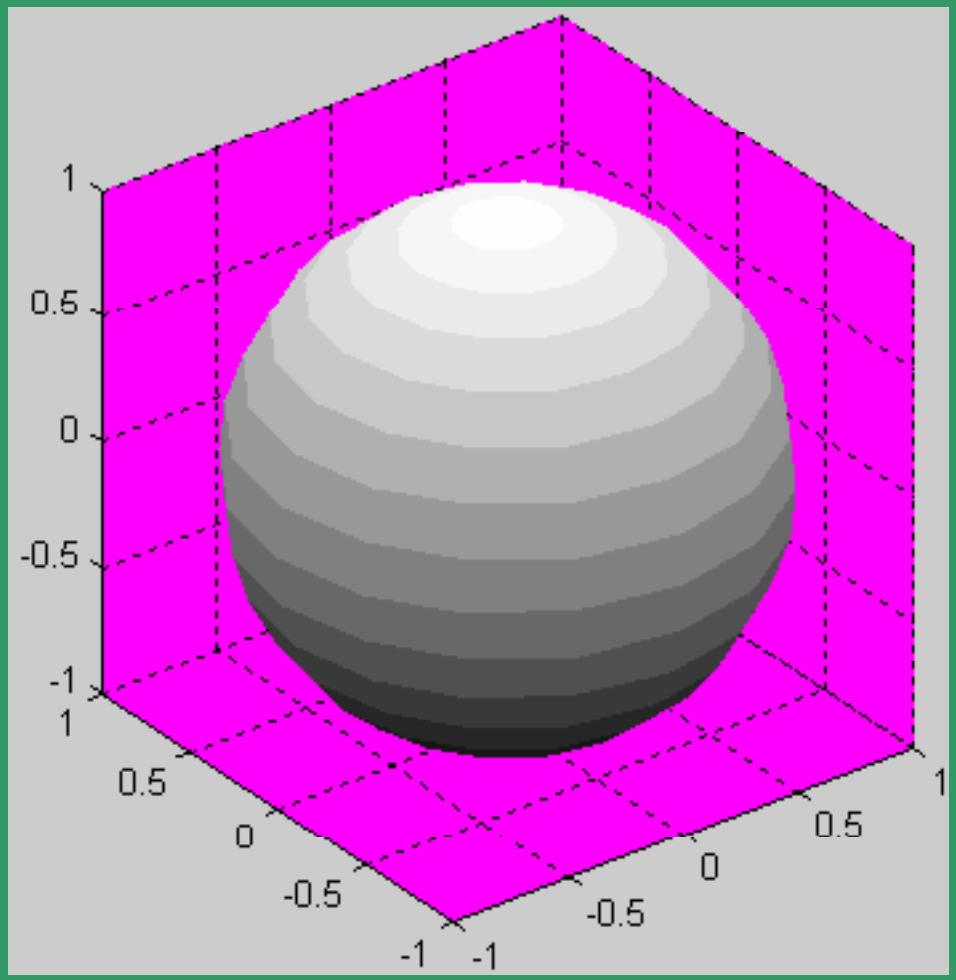
This approach is valid provided:

- The light source is at infinity – N.S is constant over the entire face
- The Viewer is at infinity – N.V is constant over the entire face
- The polygon is not an approximation of a curved surface.

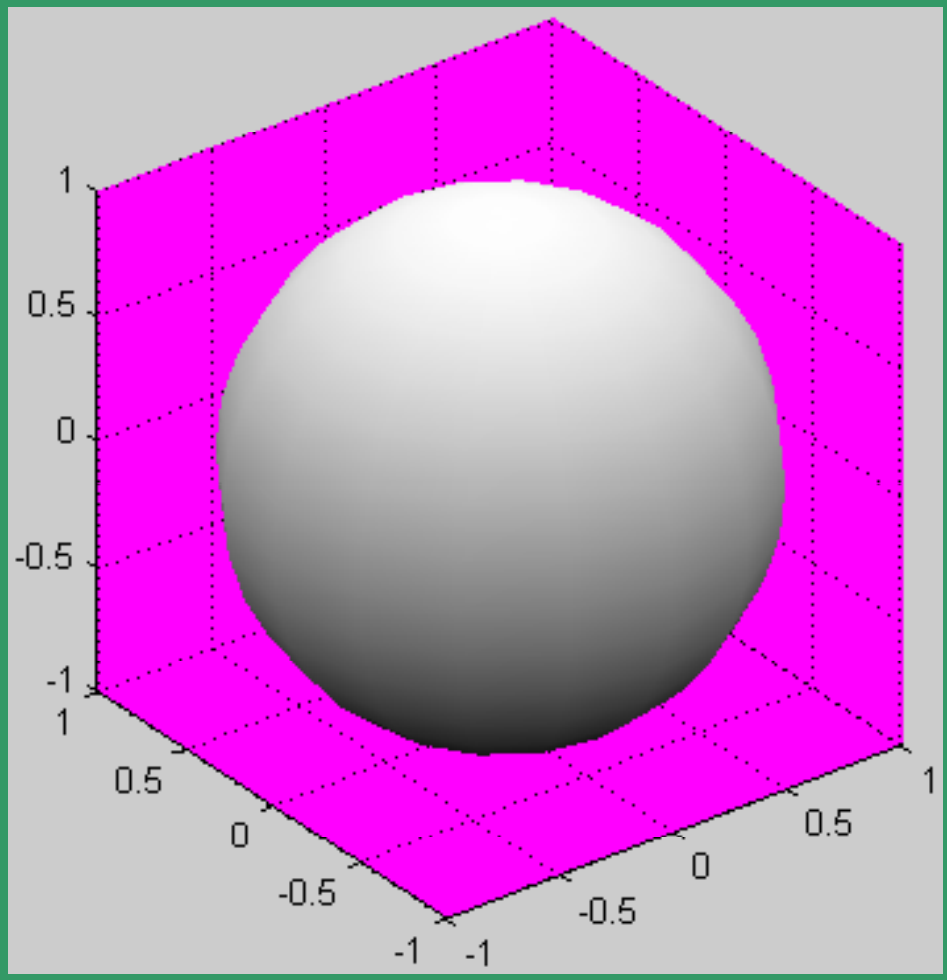


Gouraud Shading or Color / Intensity Interpolation shading

Illustrating the different shadings of a sphere.



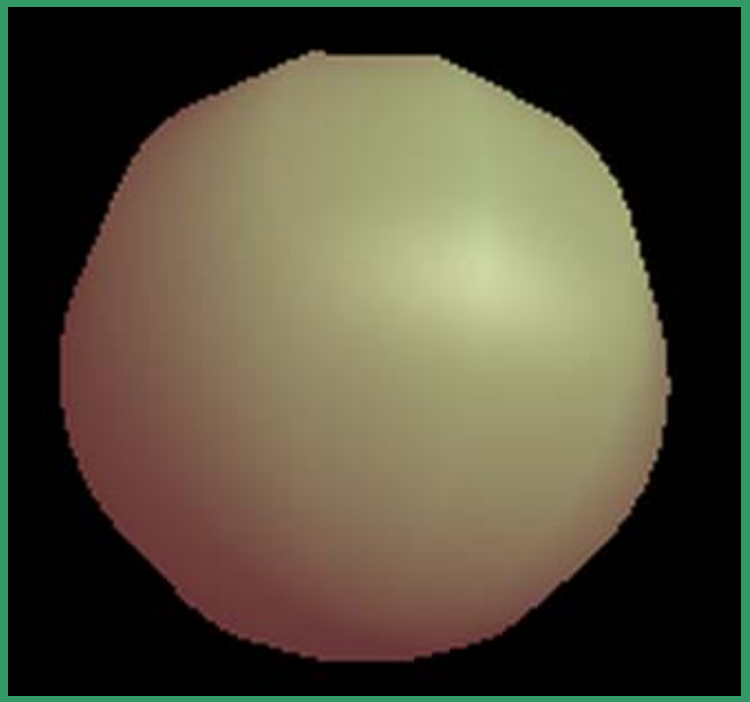
Faceted



Gouraud / Smooth / Interpolated

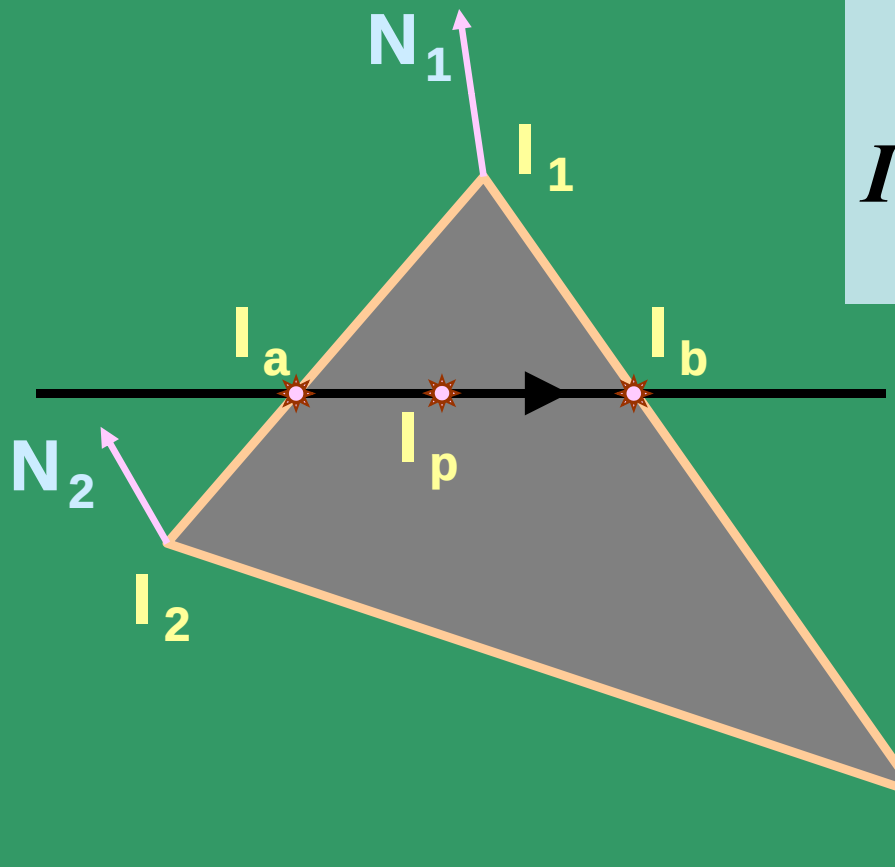
Illustrating the different shadings of a sphere.

Faceted



Gouraud/
Smooth/
Interpolated

Illustrating the principle of Gouraud Shading or Color / Intensity Interpolation shading

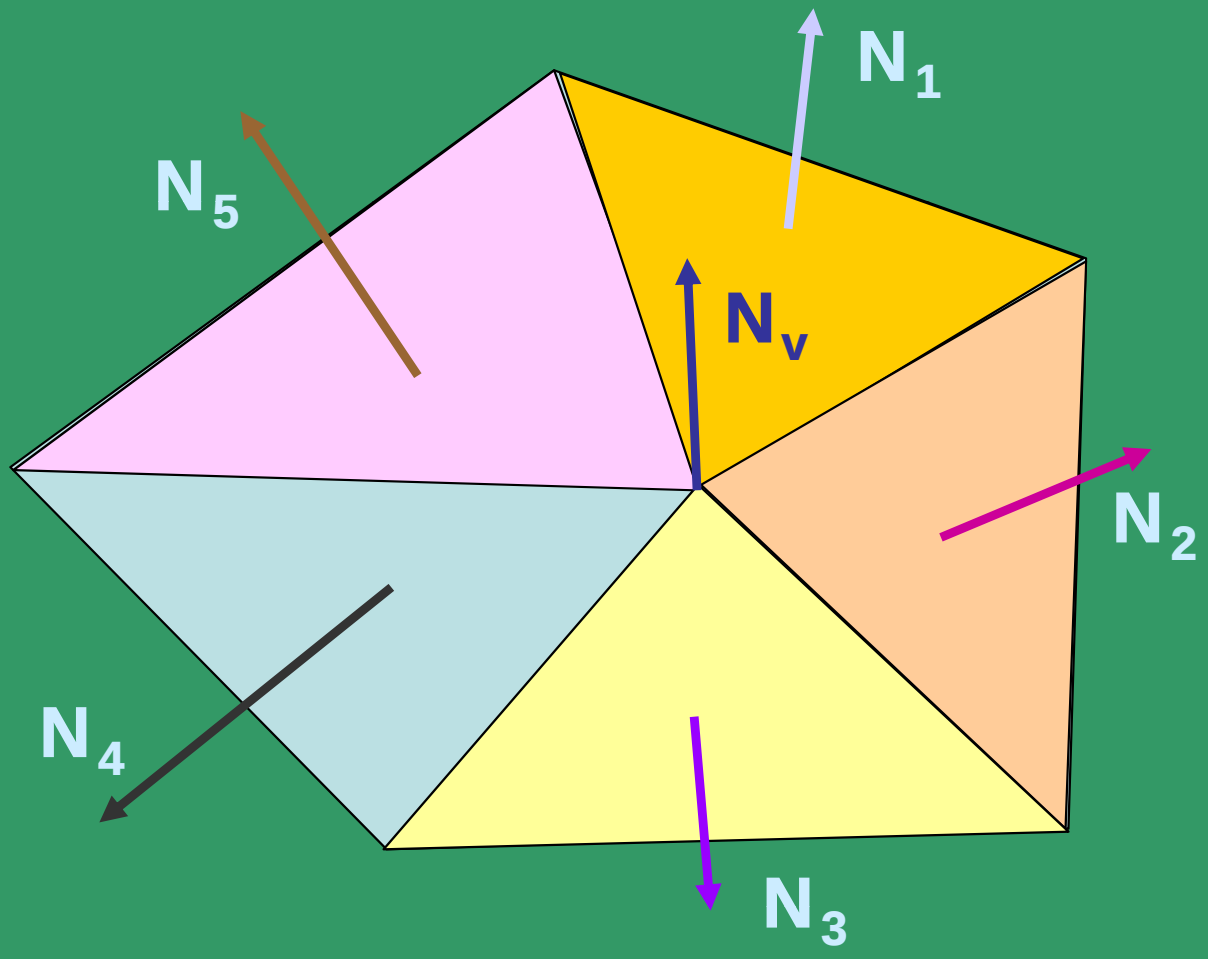


$$I_a = I_1 - (I_1 - I_2)\left(\frac{y_1 - y_s}{y_1 - y_2}\right);$$

$$I_b = I_1 - (I_1 - I_3)\left(\frac{y_1 - y_s}{y_1 - y_3}\right);$$

$$I_p = I_b - (I_b - I_a)\left(\frac{x_b - x_p}{x_b - x_a}\right)$$

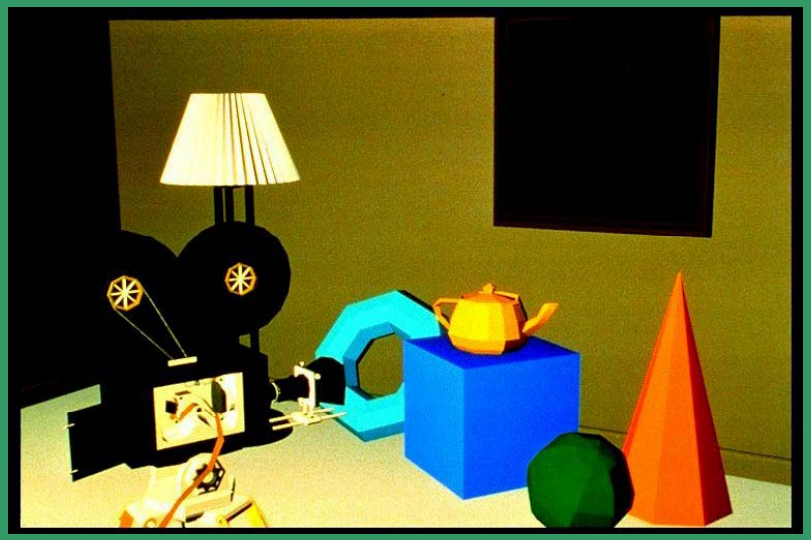
Illustrating the principle of Gouraud Shading or Color / Intensity Interpolation shading



Illustrating the difference in shadings in case of a tea-pot.

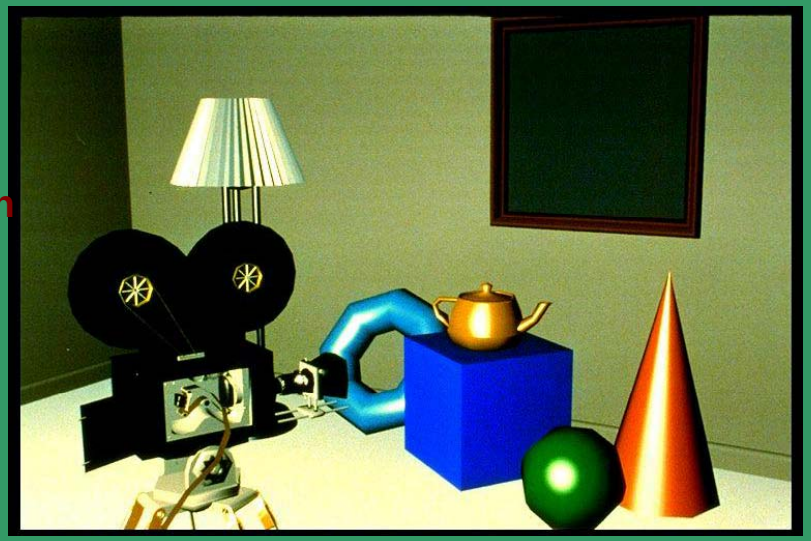
Photo-realistic Rendering

Flat or Faceted Shading: Constant intensity over each face



Gouraud Shading: Interpolation of intensity

Phong Shading: Interpolation of surface normals. Note the specular highlights



Global Illumination: Inter-object reflections, shadows, and texture mapping

End of lectures on

Illumination

and

Shading