Computer Graphics

Unit 3 – Part 5

-By Manjula. S

Light and Matter

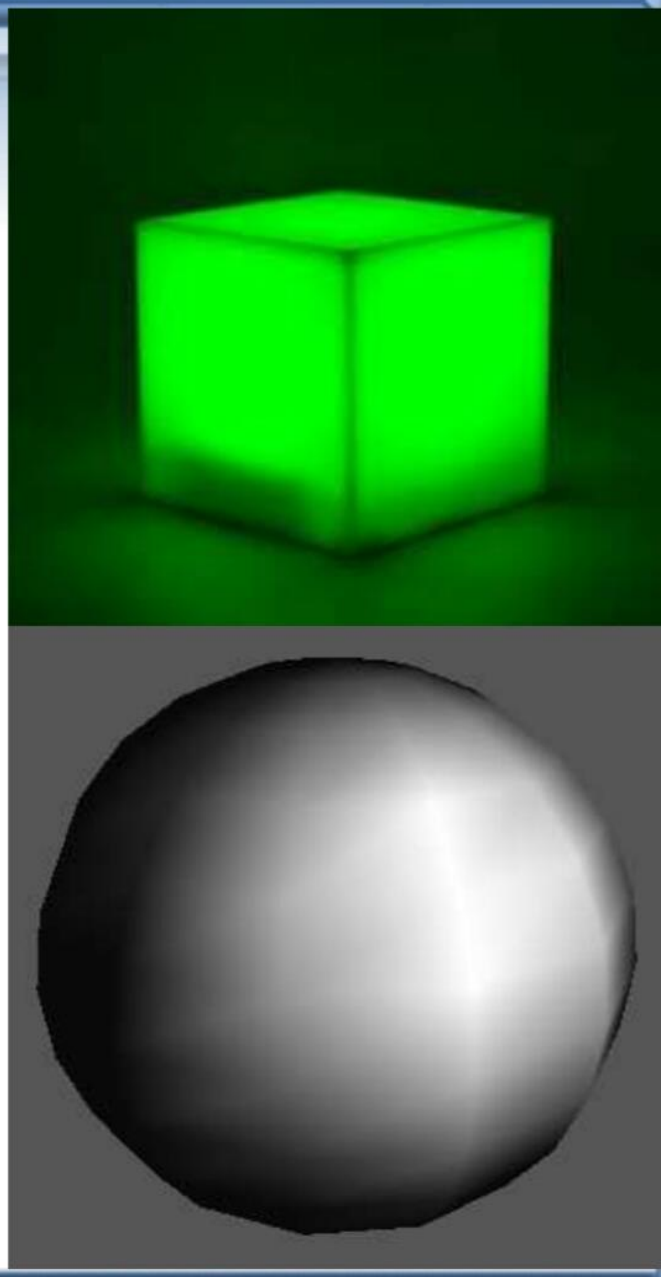
Lighting

Process of computing the luminous intensity reflected from a specific 3-D point

Shading

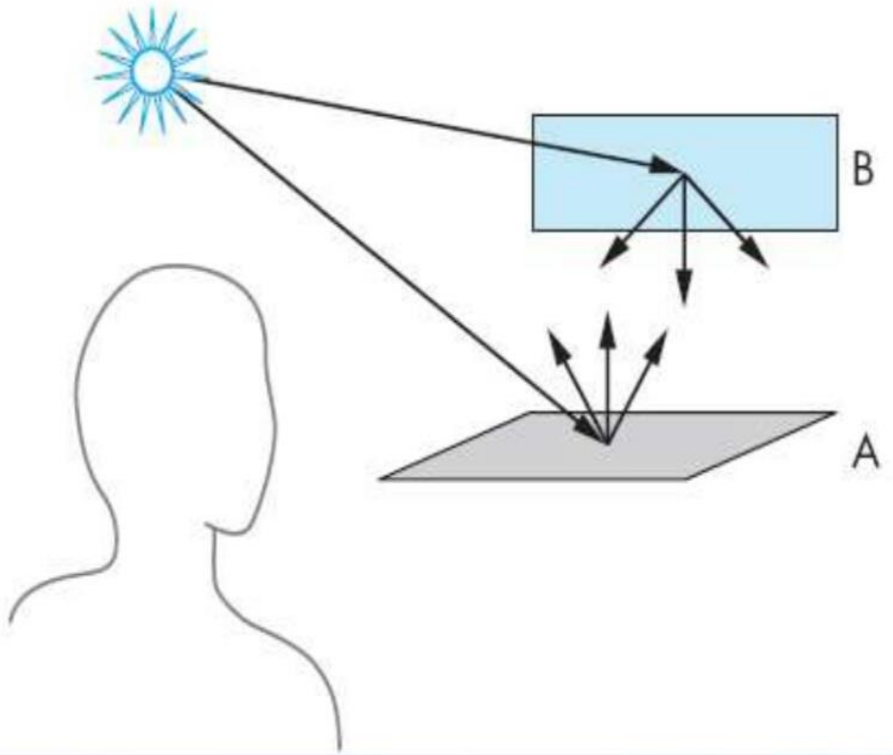
Process of assigning colors to pixels

Shading model dictates how light is scattered or reflected from a surface



Light and Matter

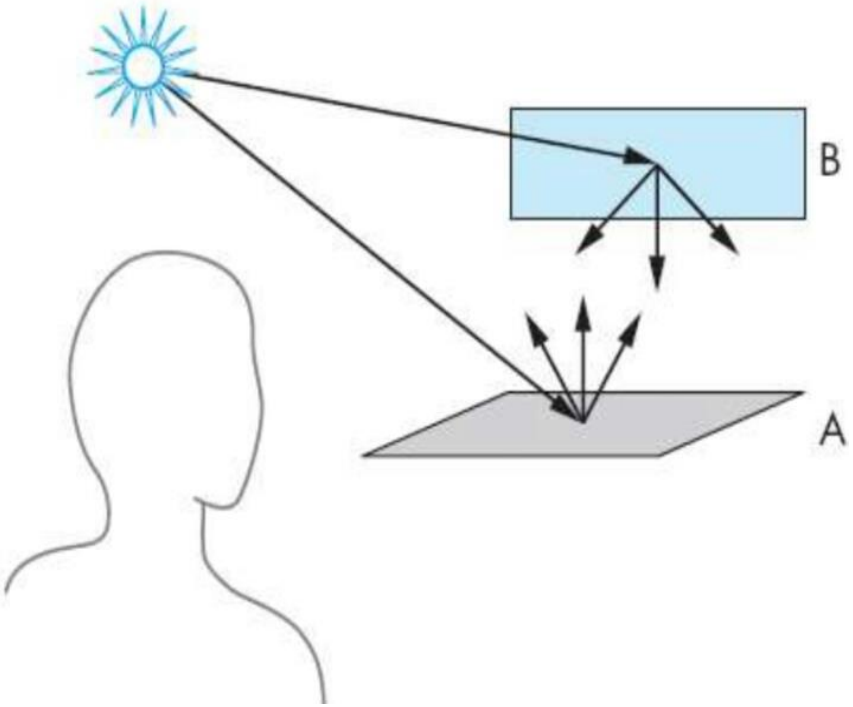
- A surface can either emit light by self emission (light bulb) or reflect light from other surfaces that illuminate it(moon).
- Some surfaces may both reflect light and emit light from internal physical processes.



-By Manjula. S

Light and Matter

- When we look at a point on an object, the color that we see is determined by multiple interactions among light sources and reflective surfaces.
- These interactions can be viewed as a recursive process

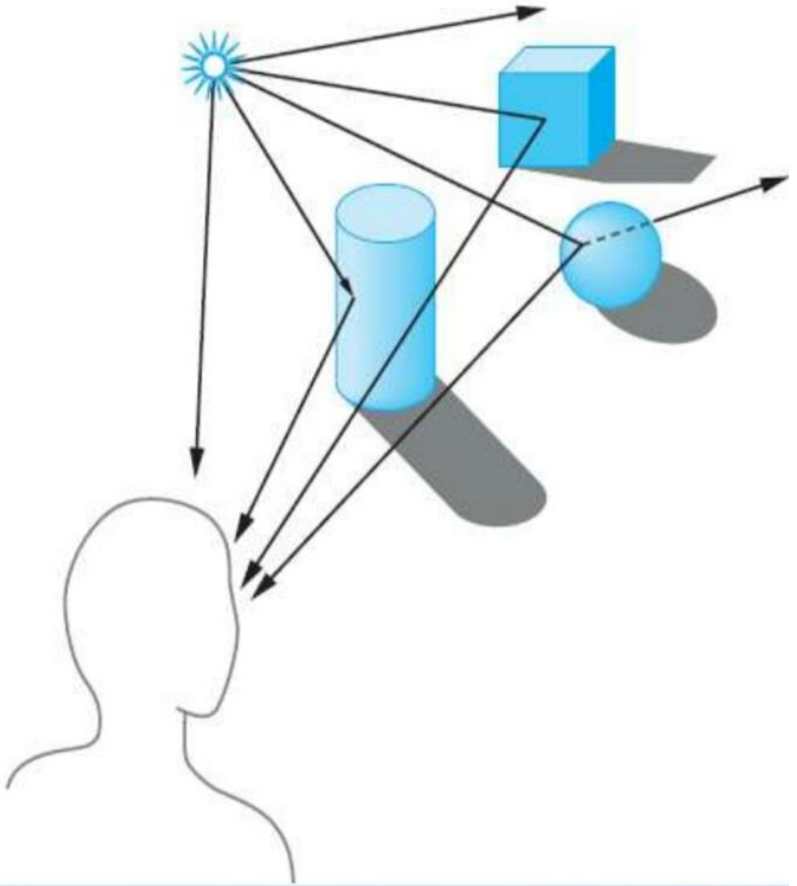


-By Manjula. S

Interactions between materials and light

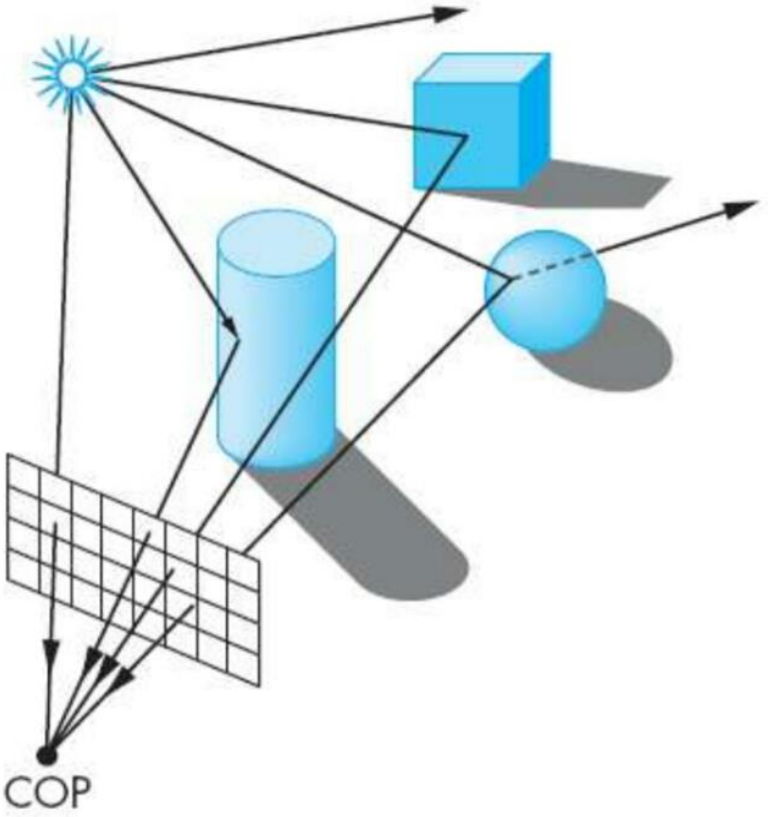
- We consider the light rays from light-emitting surfaces. These are called as light sources.

- We model what happens to these rays as they interact with reflecting surfaces in the scene



Interactions between materials and light

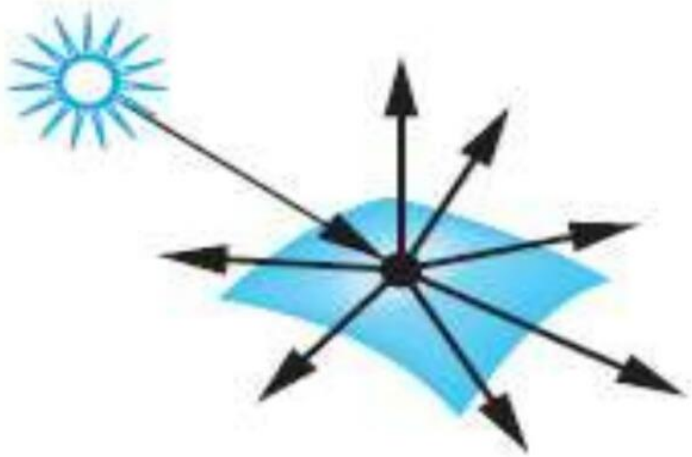
- In terms of Computer Graphics, Place the projection plane between the COP and objects.
- The clipping window in this plane is mapped to the display.
- We consider light that enters the camera by passing through COP, we can start at COP and follow projector



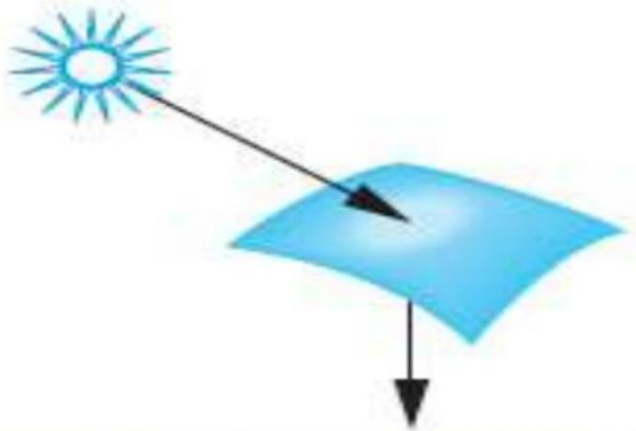
-By Manjula. S

Classification of Light and Material Interactions

- Specular surfaces



- Diffuse surfaces



- Translucent surfaces

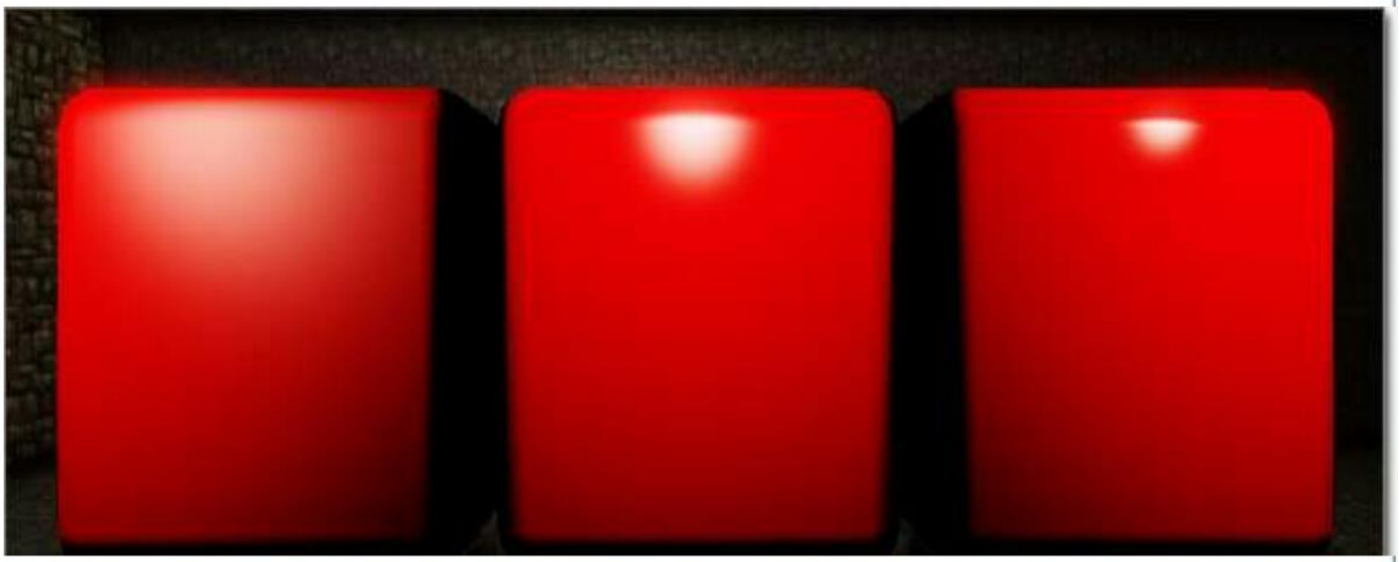
Classification of Light and Material Interactions

- Appear shiny because most of the light that is reflected is scattered in a narrow range of angles close to the angle of reflection.
- Mirrors are perfectly specular surfaces;

-By Manjula. S

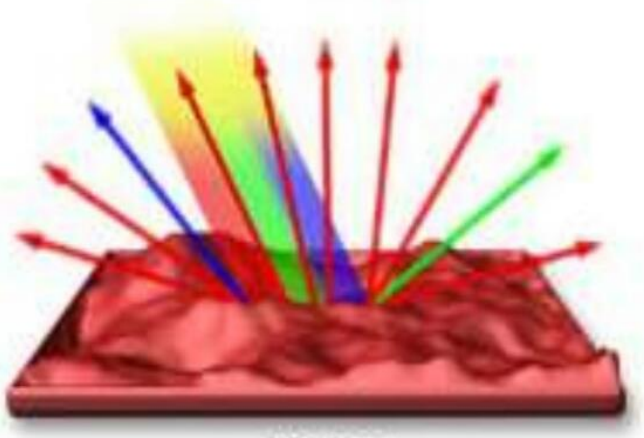
Classification of Light and Material Interactions

- The light from an incoming light ray may be partially absorbed, but all reflected light emerges at a single angle, obeying the rule that the **angle of incidence is equal to the angle of reflection.**



Classification of Light and Material Interactions

- They are characterized by reflected light being scattered in all directions.
- Walls painted with matte or flat paint are diffuse reflectors, as are many natural materials, such as terrain viewed from an airplane or a satellite.



Diffuse Reflection

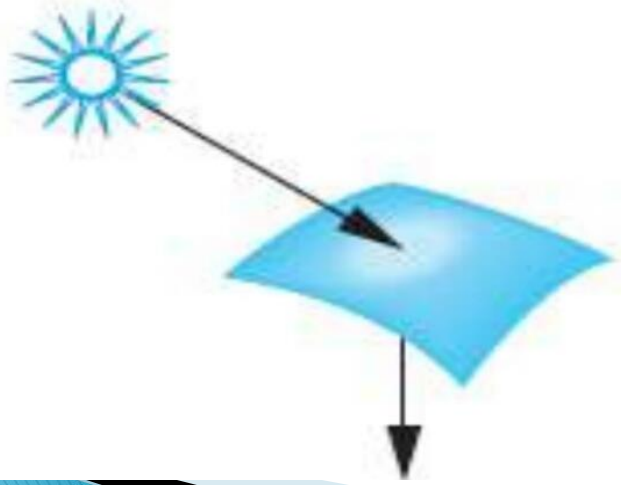
Classification of Light and Material Interactions

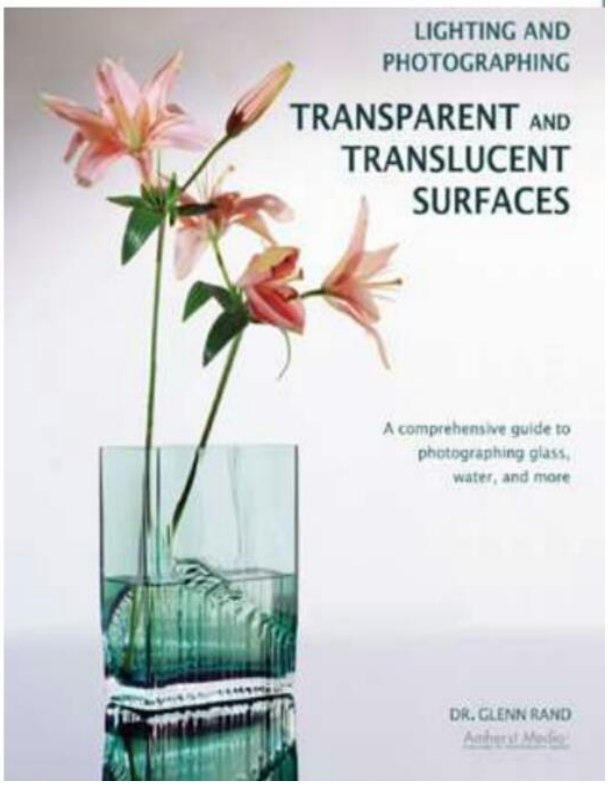
- Perfectly diffuse surfaces scatter light equally in all directions and thus appear the same to all viewers.

-By Manjula. S

Classification of Light and Material Interactions

- Allow some light to penetrate the surface and to emerge from another location on the object.
- This process of refraction characterizes glass and water
- Some incident light may also be reflected at the surface.





-By Manjula. S

Bidirectional Reflection Distribution Function

- Reflection, absorption, transmission of light at the surface of a material is described by a single function called BRDF
- BRDF is a function of 5 variables
- **Frequency** of light
- 2 angles required to describe the **direction of the input vector**
- 2 angles required to describe the **direction of the output vector**

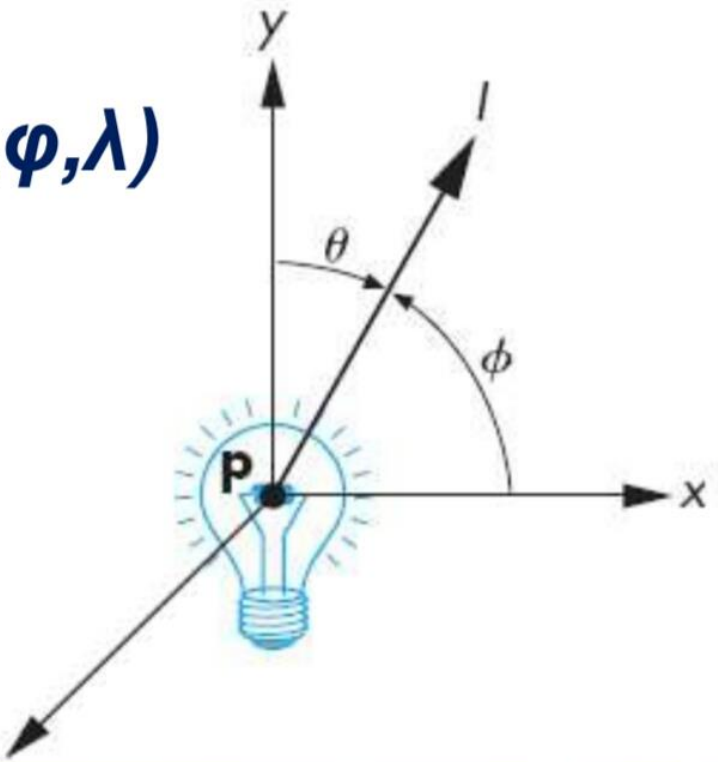
Light Sources

- Light leave surface through two fundamental process

  Self emission

  Reflection

- Illumination Function $I(x, y, z, \theta, \varphi, \lambda)$
- Point (x,y,z)
- Direction of emission
- Wavelength λ
- Intensity I

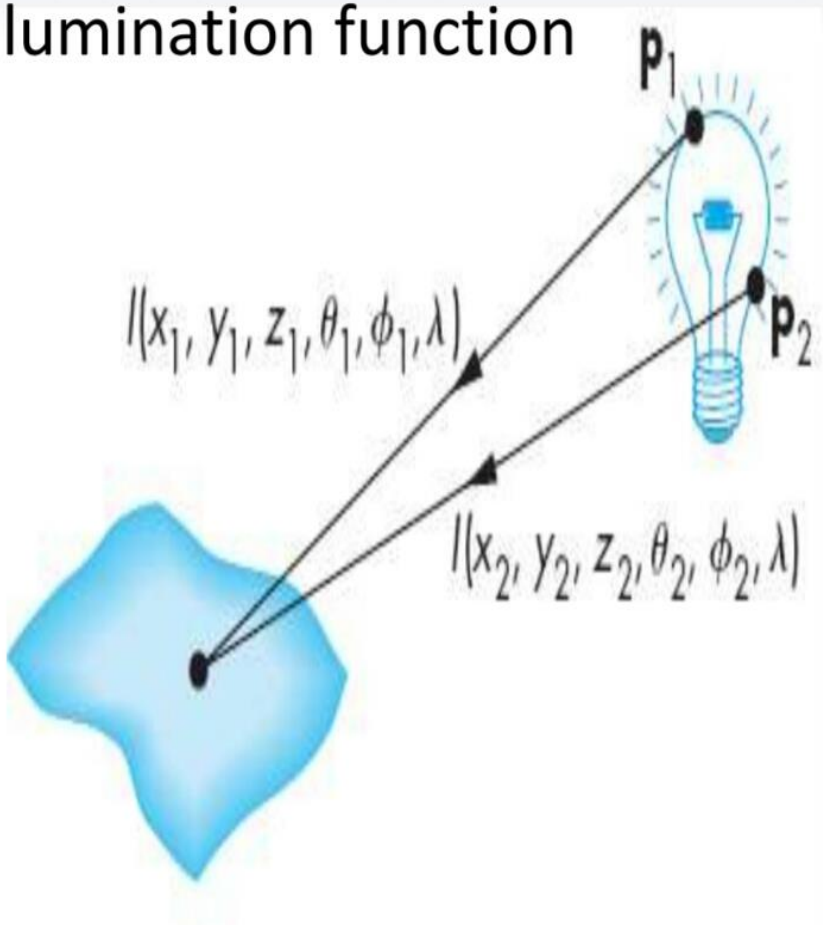


Light Sources

Three component intensity or illumination function

L=(Lr, Lg , Lb)

$$\mathrm{I} = \begin{bmatrix} I_{\mathrm{r}} \\ I_{\mathrm{g}} \\ I_{\mathrm{b}} \end{bmatrix}$$



-By Manjula. S

Light Sources

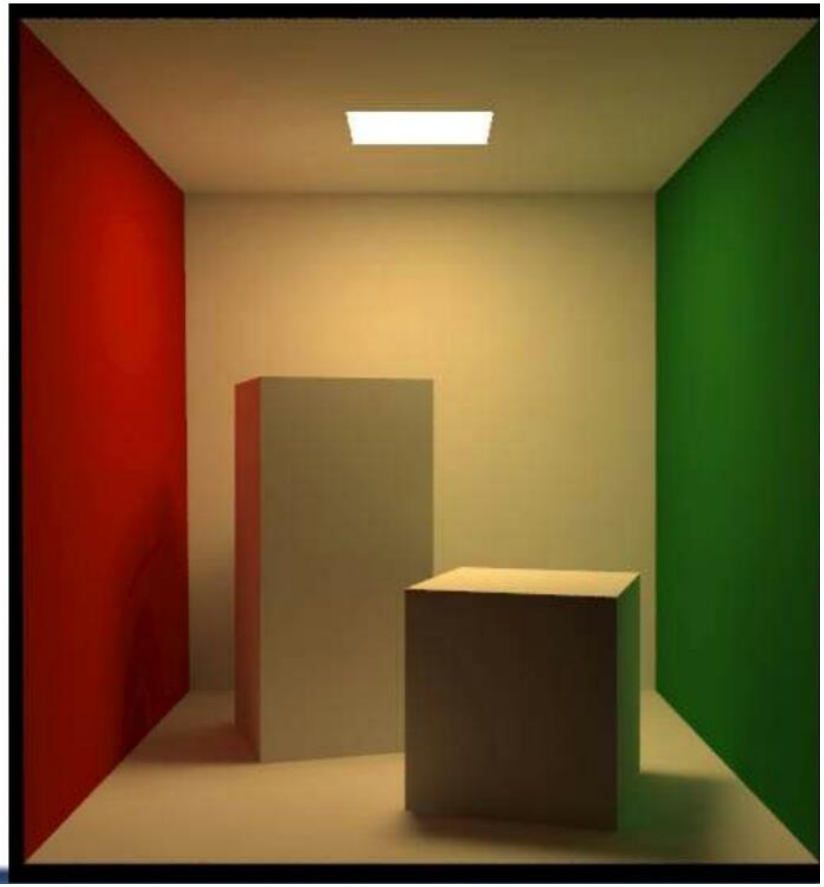
Types of Light Sources

- Ambient Lighting
- Point Sources
- Spot Lights
- Distant Lights

-By Manjula. S

ambient light

- In many situations, the lights have been designed and positioned to provide uniform illumination through out the room.
- Such illumination is achieved through large sources that have diffusers whose purpose is to scatter light in all directions.
- The uniform lighting is called ambient light.

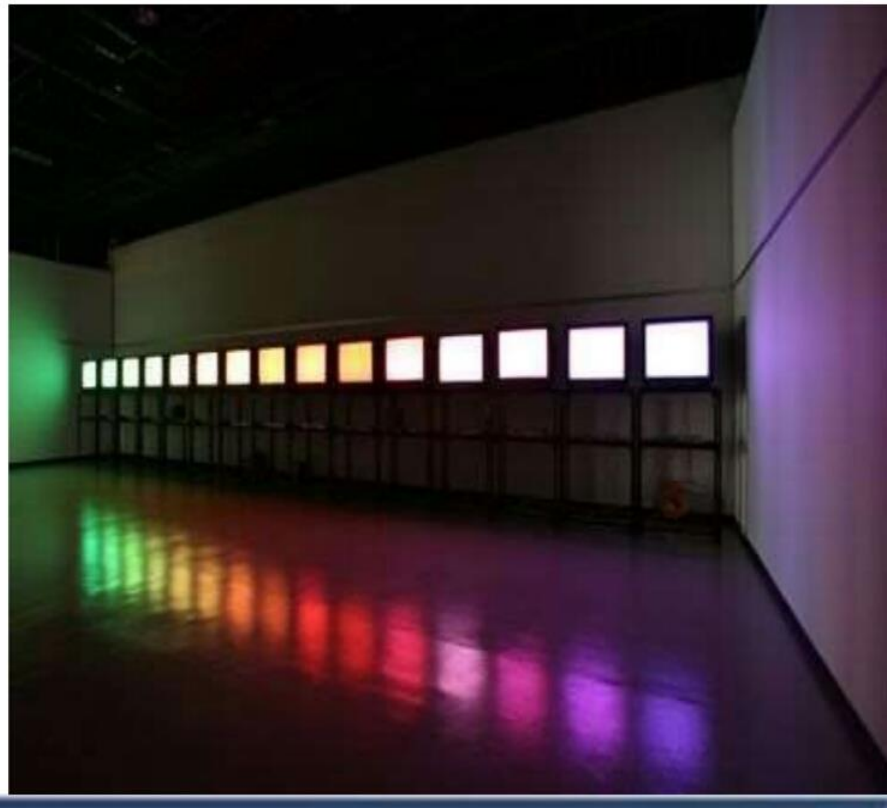


-By Manjula. S

Ambient Lighting

- Ambient light depends on the color of the light sources in the environment.
- a red light bulb in a white room creates red ambient light. Hence if we turn off the light, the ambient contribution disappears.

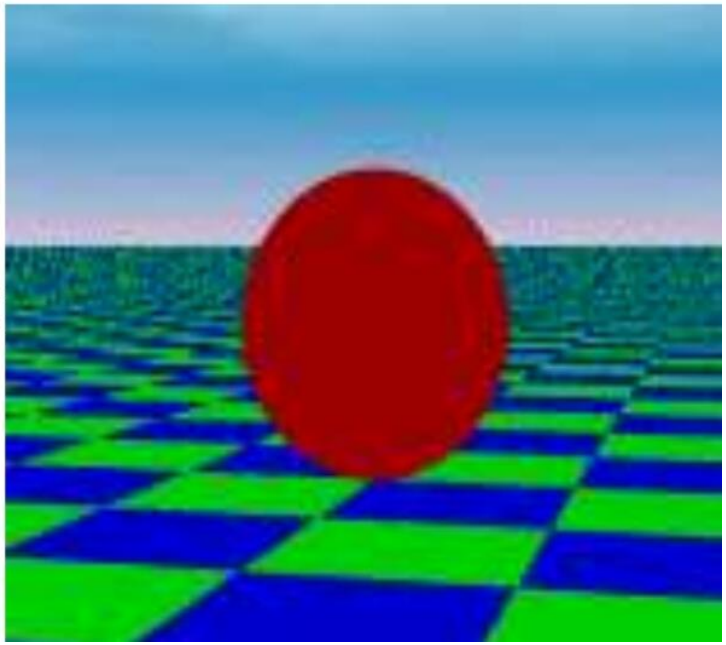
$$\mathrm{I_a} = \begin{bmatrix} I_{\mathrm{ar}} \\ I_{\mathrm{ag}} \\ I_{\mathrm{ab}} \end{bmatrix}$$



-By Manjula. S

Ambient Lighting

- OpenGL permits to add a **global ambient** term which does not depend up on any of the light sources and is reflected from surfaces.
- Advantage There will be always some light in the environment so that objects in the view volume that are not blocked by other objects will always appear in the image.

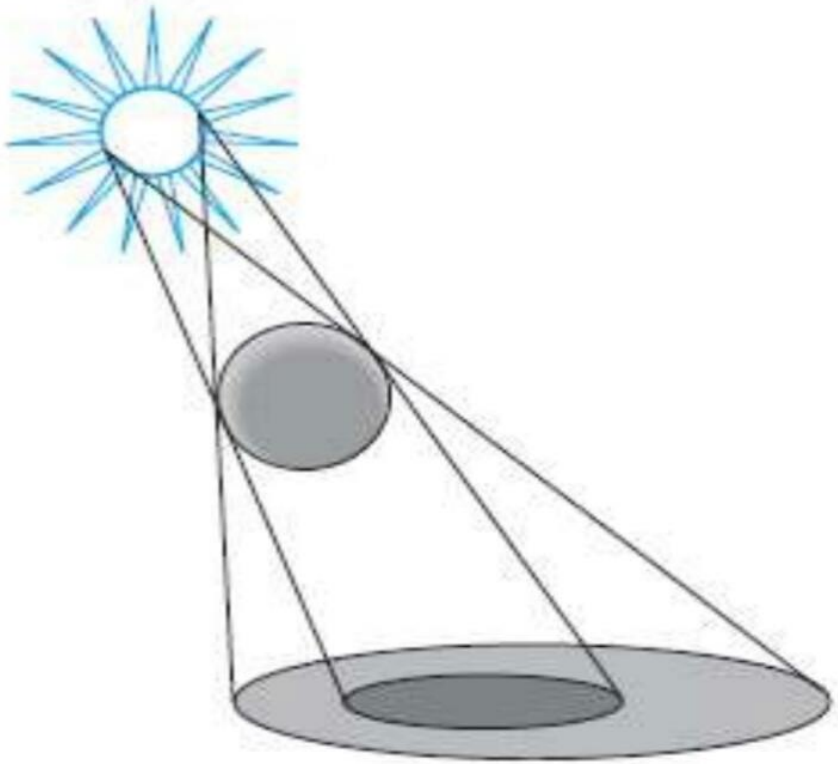


-By Manjula. S

Point Source

- An ideal point source emits light equally in all directions.
- We can characterize a point source located at point **Po** by 3 component color function.

$$\mathbf{I}(\mathbf{p}_0) = \begin{bmatrix} I_r(\mathbf{p}_0) \\ I_g(\mathbf{p}_0) \\ I_b(\mathbf{p}_0) \end{bmatrix}$$

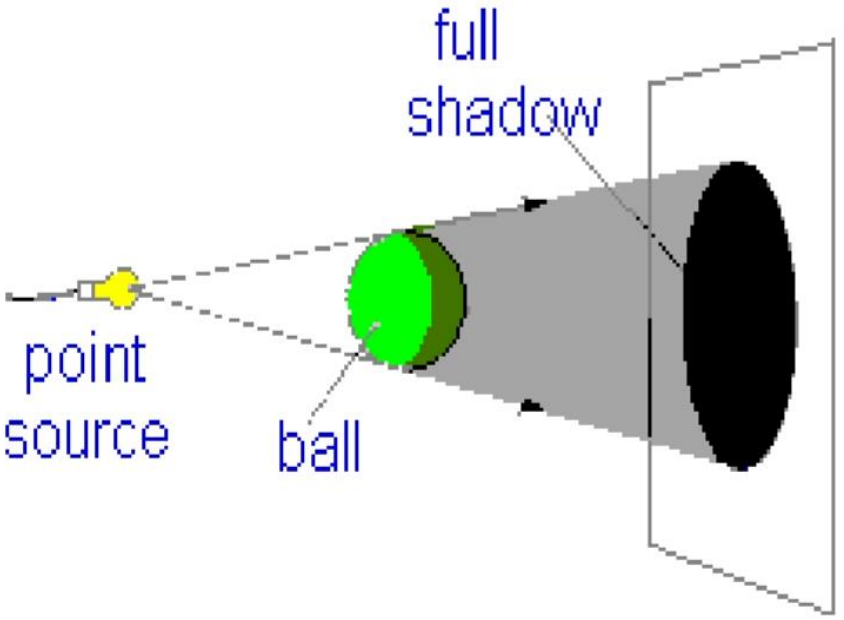


-By Manjula. S

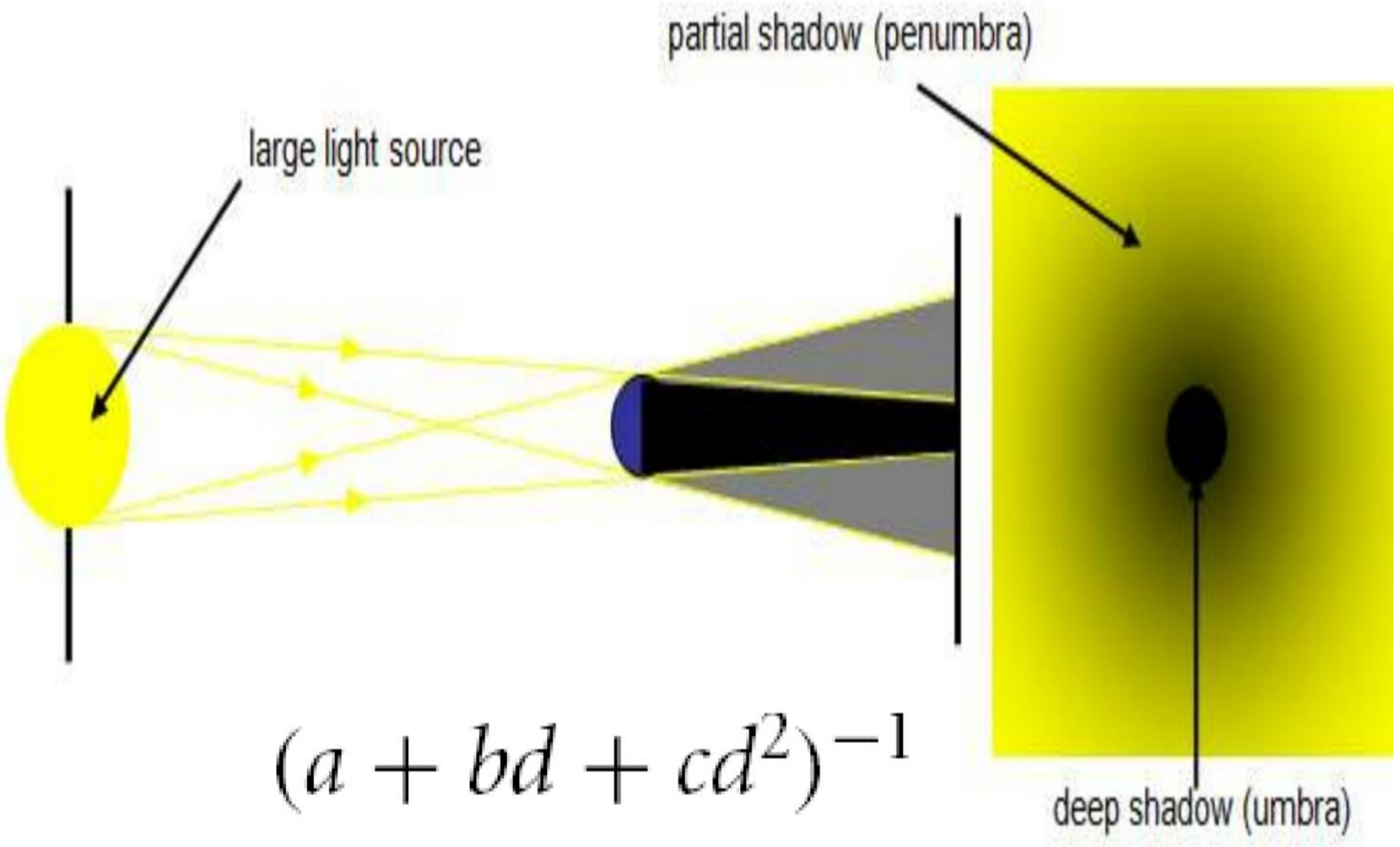
Point Source

- The intensity of illumination received from a point source located at P_0 at a point **P** is proportional to the inverse square of the distance from the source.

$$\mathbf{i}(\mathbf{p}, \mathbf{p}_0) = \frac{1}{|\mathbf{p} - \mathbf{p}_0|^2}\mathbf{I}(\mathbf{p}_0).$$



Point Source



$$(a + bd + cd^2)^{-1}$$

-By Manjula. S

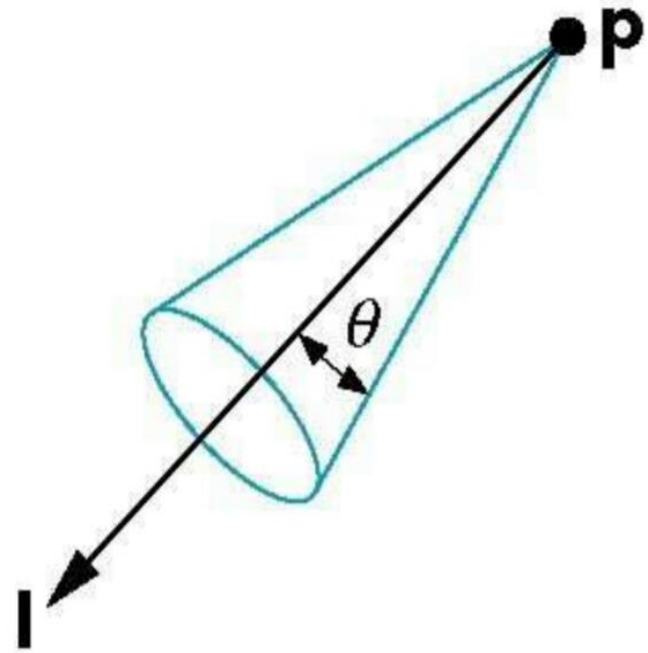
Spot light

- More realistic spot lights are characterized by the distribution of light with in the cone.
- Cosines are convenient functions for lighting calculations.
- Characterized by a narrow range of angles through which light is emitted.

Spot light

- Can be constructed from a point source by limiting the angles at which light from the source can be seen.

- We can use a cone whose apex is at p which points in the direction l whose width is determined by angle θ.

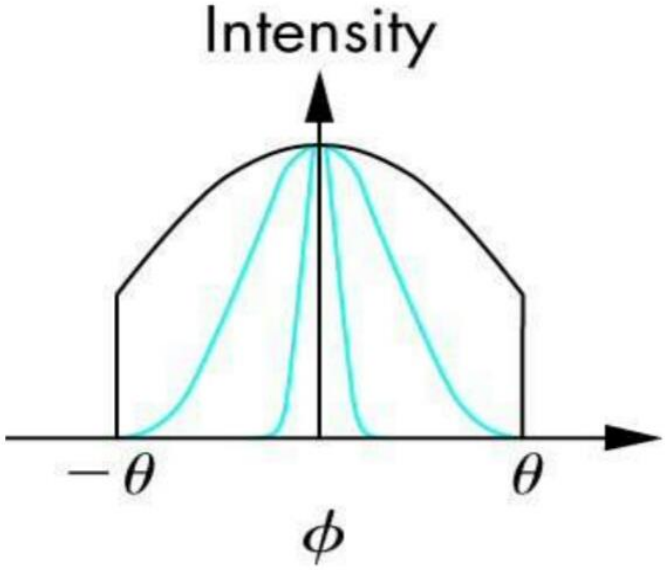


Spot light

- If θ=180 the spot light becomes a point source.
- Light is concentrated at the center of the cone

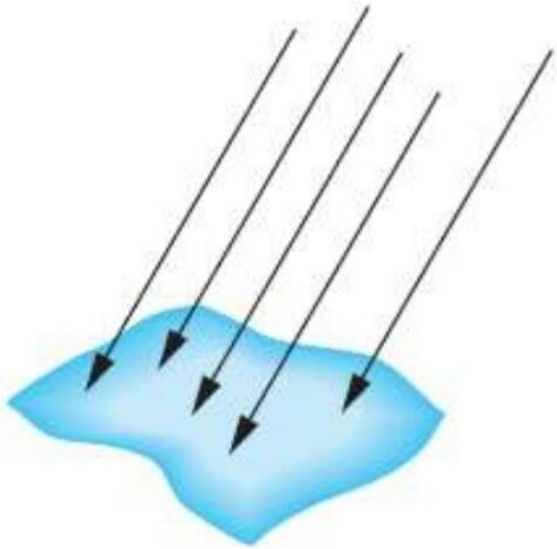
$$\cos^e \phi$$

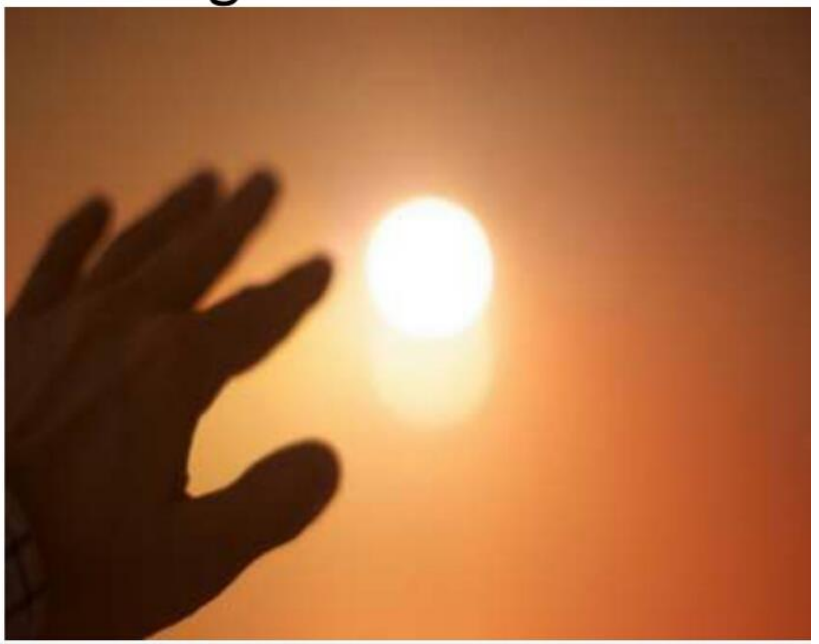
- If u and v are unit length vectors, we can compute cosine of the angle between them with dot product Cosθ= u•v



Distant Light Sources

- The calculations for distant light sources are similar to the calculations for parallel projections.
- The parallel projections replace the location of the light source with the direction of the light source.





-By Manjula. S

Distant Light Sources

- Hence, in homogeneous coordinates, the location of a point light source at Po is represented internally as a 4D column matrix.

$$po = \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

- The distant light source is described by a direction vector whose representation in homogeneous co-ordinates is the matrix

$$po = \begin{pmatrix} x \\ y \\ z \\ 0 \end{pmatrix}$$

Illumination Model

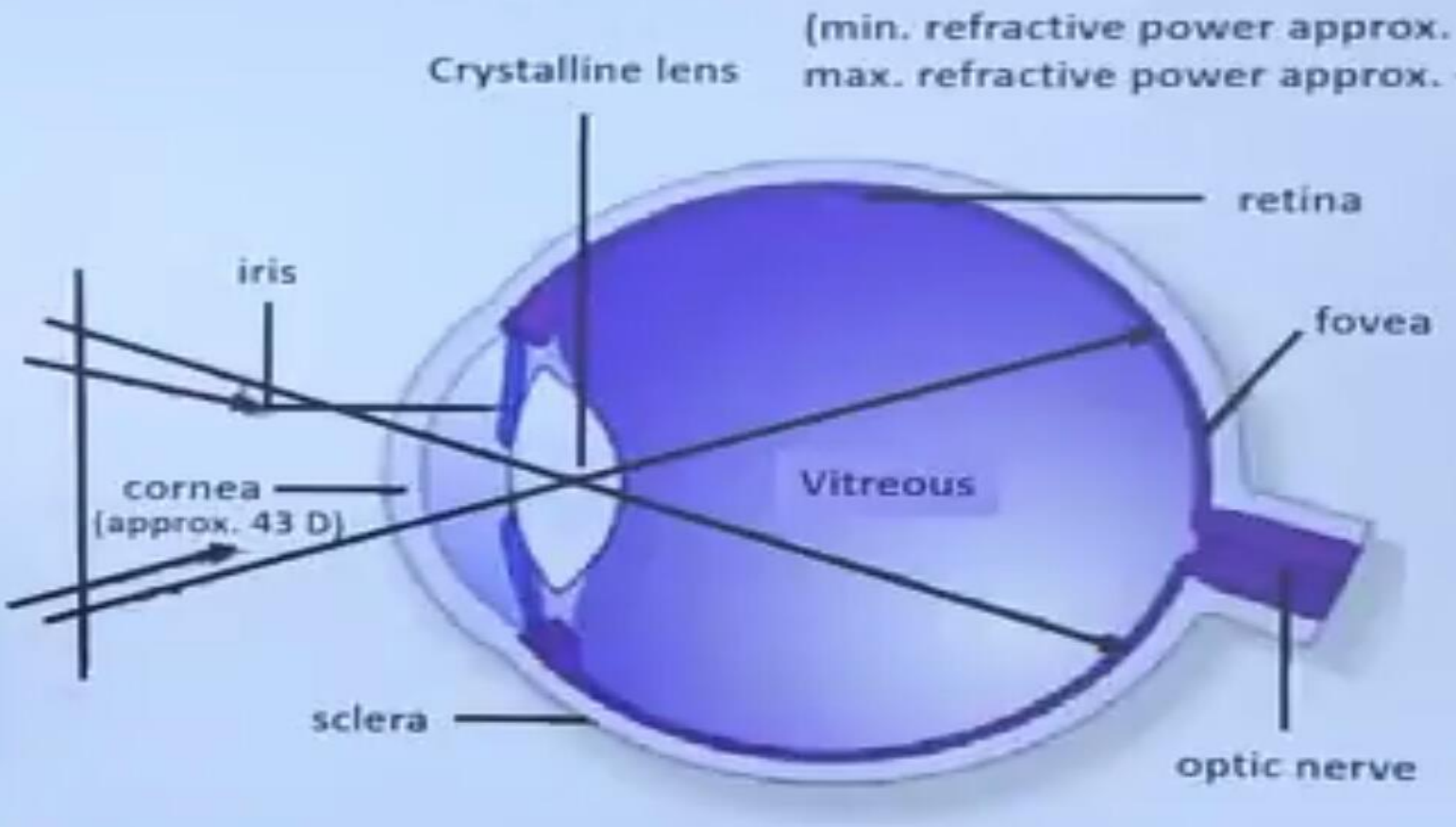
- **What is an Illumination Model?**
- An **Illumination Model** is a formula in variables associated to the surface properties and light conditions to calculate the intensity of light reflected from a point on a surface. Based on standard lighting conditions in a scene, following basic models are -
  - ✓ **Ambient Light (Model 1)** : A simple way to model indirect reflection. All surfaces in all positions and orientations are illuminated equally.
  - ✓ **Diffused Light (Model 2)** : The diffuse shading produced by dull, smooth objects.
  - ✓ **Specularly Reflected Light (Model 3)** : The bright spots appearing on smooth shiny (e.g., metallic or polished) surfaces.
- To compute the color of objects according to the position of the light, normal vector and camera position we normally use an illumination model known as Phong Illumination Model.

-By Manjula. S

Background of Illumination

What is the background of Illumination?

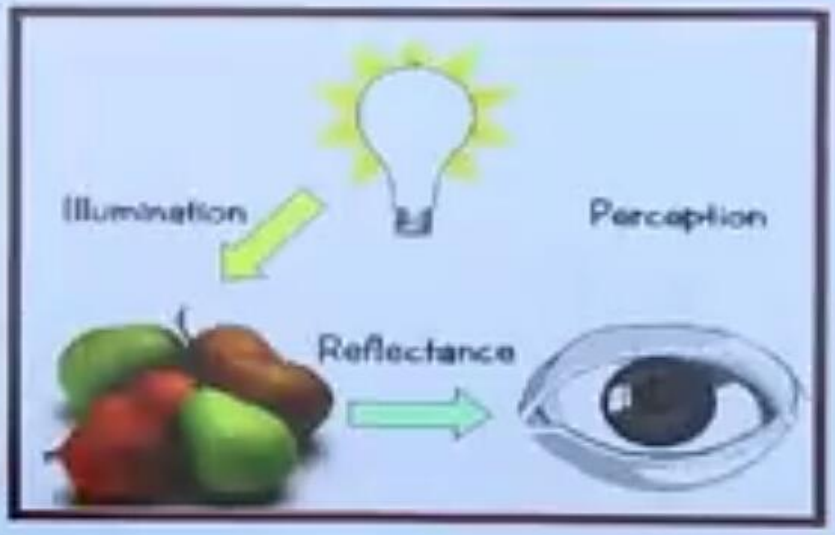
The eye works like a camera. Lots of photo sensors at the back of the eye. Sensing the amount of light coming from different directions.

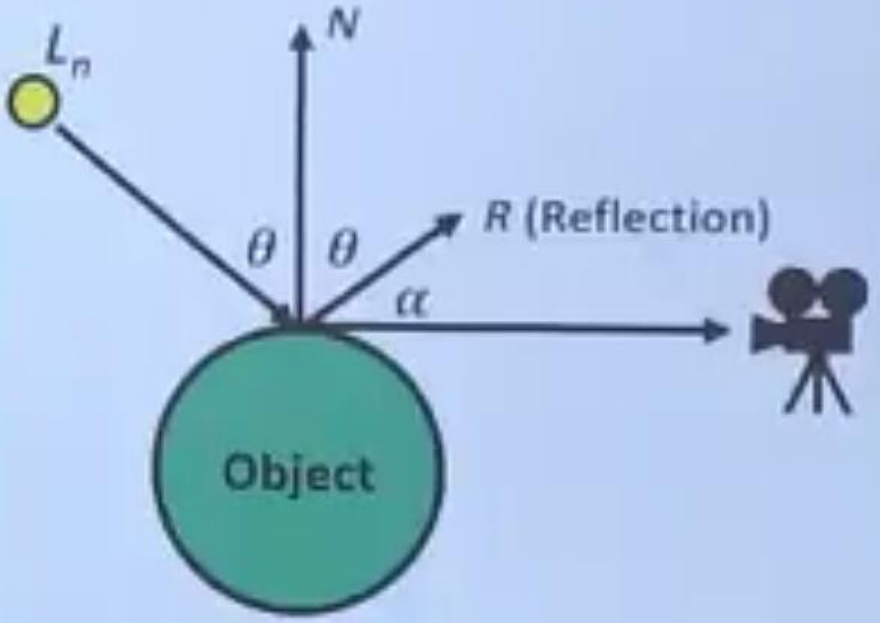


-By Manjula. S

Background of Illumination

- **What affects the light that comes into the eyes?**
  - The position of the point
  - The position of the light
  - Color and intensity of the light
  - Camera vector
  - Normal vector of the surface at the vertex
  - Physical characteristics of the object (reflectance model, color etc.)

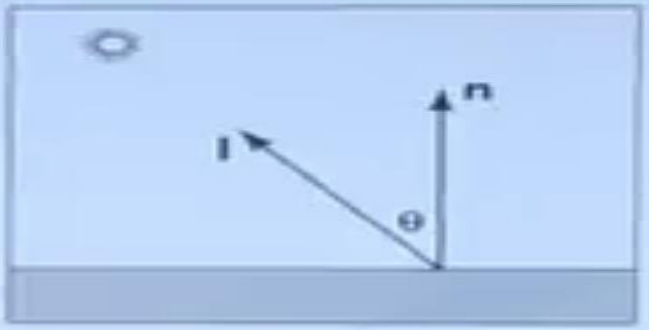




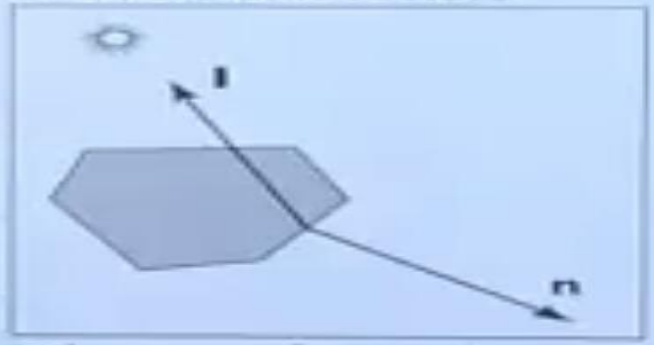
-By Manjula. S

Diffuse Reflection(Lambertian Reflection)

Lambert's Cosine Law



The geometry of Lambert's Law. Both **n** and **l** are unit vectors



When a surface points away from the light, it should receive no light. This case can be verified by checking whether the dot product of **l** and **n** is negative

A Lambertian object obeys *Lambert's cosine law*, which states that the color c of a surface is proportional to the cosine of the angle between the surface normal and the direction to the light source.

$c \propto \cos\Theta$,

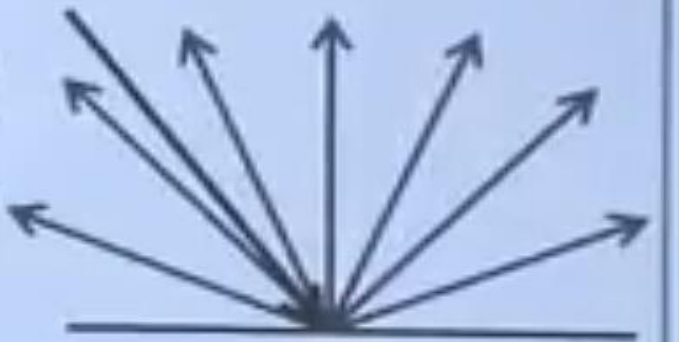
or in vector form, $c \propto \mathbf{n} \cdot \mathbf{l}$,

Thus, the color on the surface will vary according to the cosine of the angle between the surface normal and the light direction, assuming the light is ***"distant"*** relative to object size.

-By Manjula. S

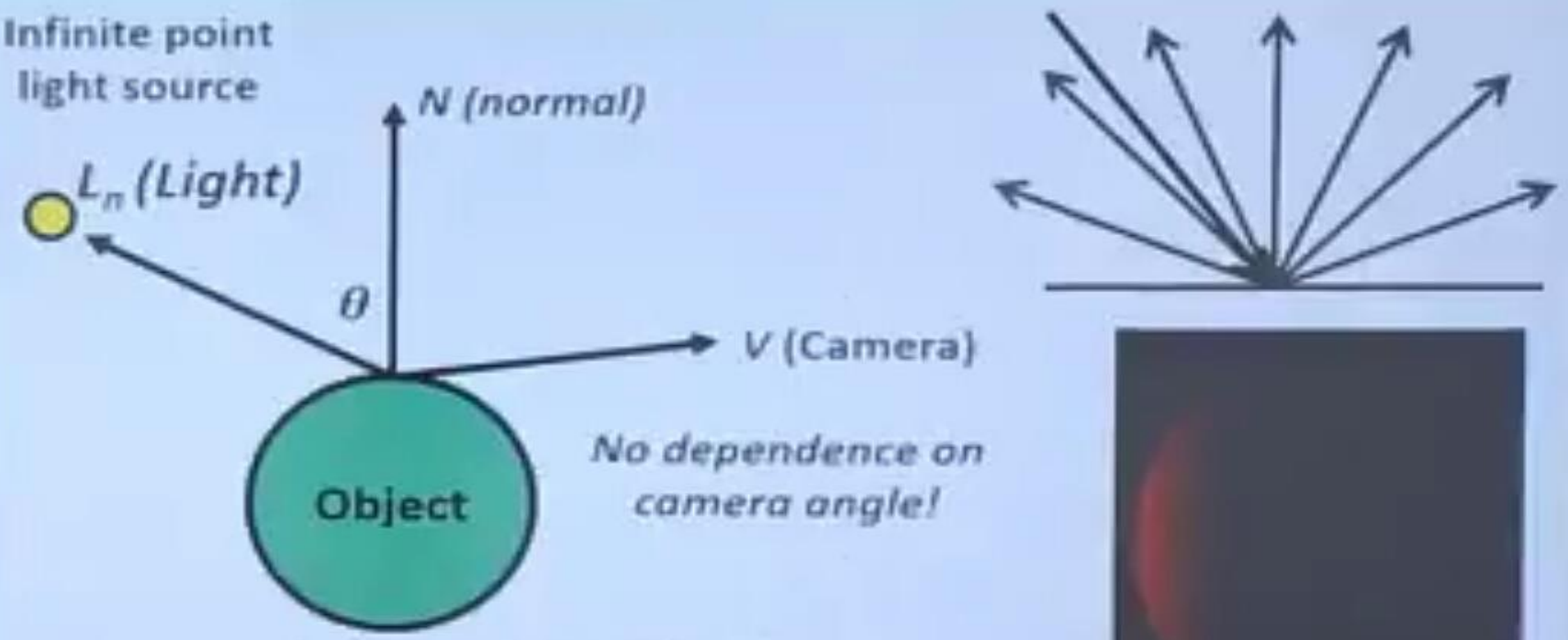
Diffuse Reflection

- **Diffuse Reflection (Lambertian Reflection)**
  - ✓ When light hits an object and If the object has a rough surface, then it is reflected to various directions.
  - ✓ Result: Light reflected to all directions.
  - ✓ The smaller the angle between the incident vector and the normal vector is, the higher the chance that the light is reflected back.
  - ✓ When the angle is larger, the reflection light gets weaker because the chance the light is shadowed/masked increases.



-By Manjula. S

Diffuse Reflection



Example: Sphere (light from left)

$I = I_p k_d COS\ \theta$

I_p: Light Intensity

θ: the angle between the normal vector direction towards the light

K_d: diffuse reflectivity

-By Manjula. S

Specular Reflection

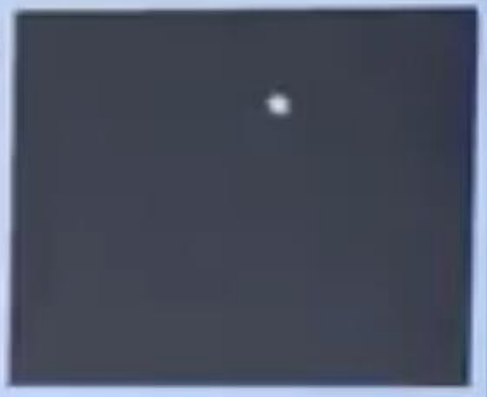
- ✓ Direct reflections of light source off shiny object.
- ✓ The object has a very smooth surface.

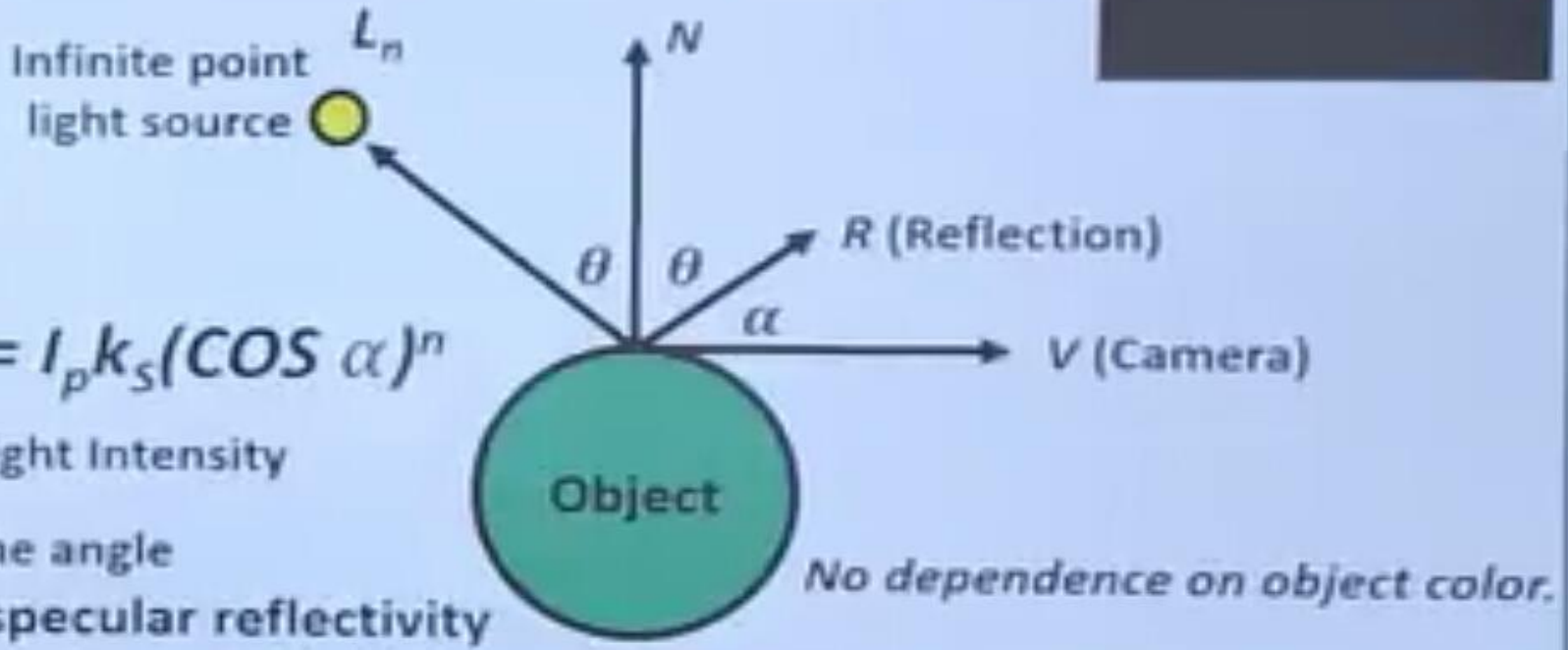
specular highlight on object

-By Manjula. S

Specular Reflection

- **Direct reflection of light source off shiny object**
  - Specular intensity n = shiny reflectance of object
  - Result : **specular highlight on object**





$$I = I_p k_s (\cos \alpha)^n$$

I_p: Light Intensity

α: the angle

K_s: specular reflectivity

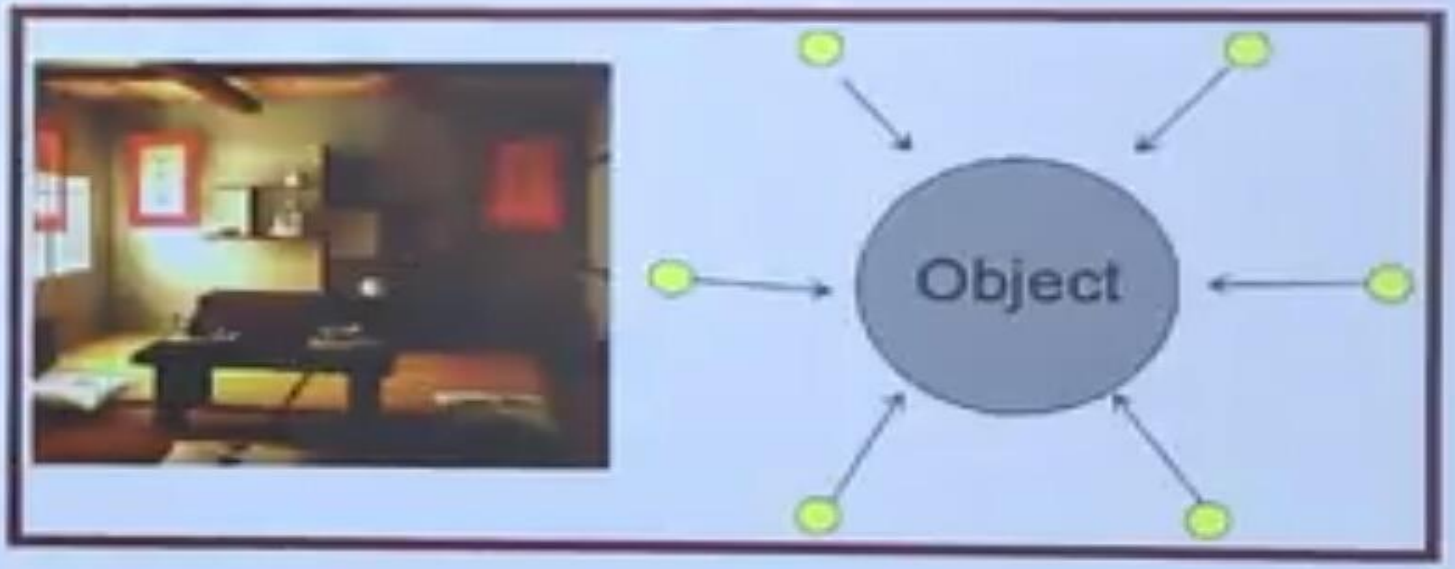
No dependence on object color.

-By Manjula. S

Ambient Lighting

- **Features of Ambient Lighting:**

  - ✓ Light from the environment
  - ✓ Light reflected or scattered from other objects
  - ✓ Coming uniformly from all directions and then reflected equally to all directions
  - ✓ A precise simulation of such effects requires a lot of computation.



-By Manjula. S

Ambient Lighting

- Simple approximation to complex 'real world' process

  Result : **globally uniform color for object**

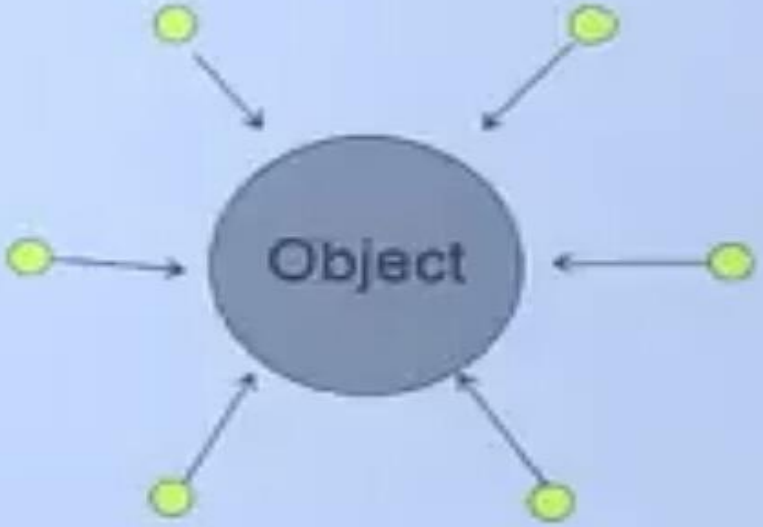
  I = resulting intensity

  I_a = light intensity

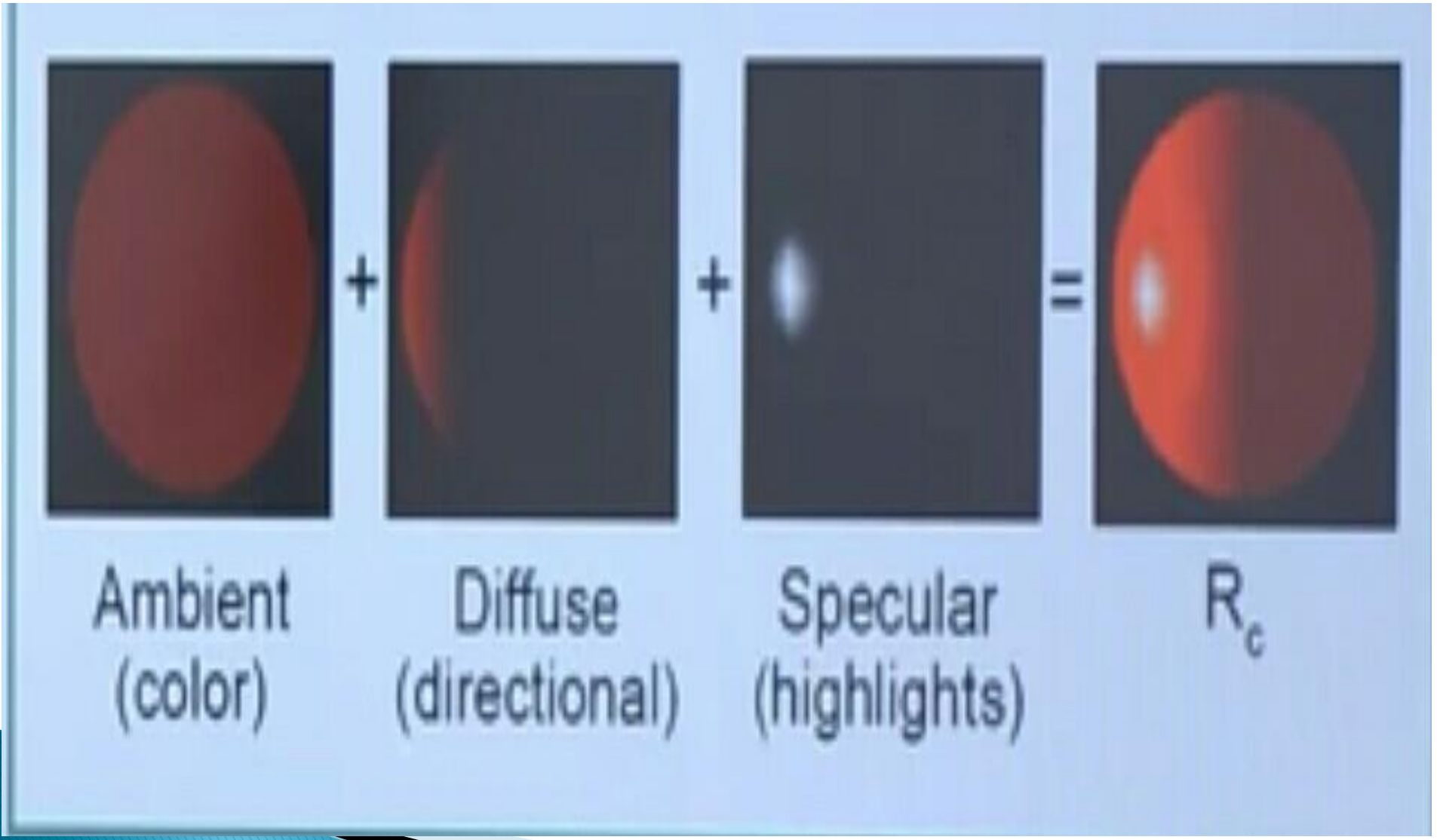
  K_a = reflectance

$$I = k_a I_a$$

Example: sphere



Illumination Model



-By Manjula. S

Phong Illumination Model

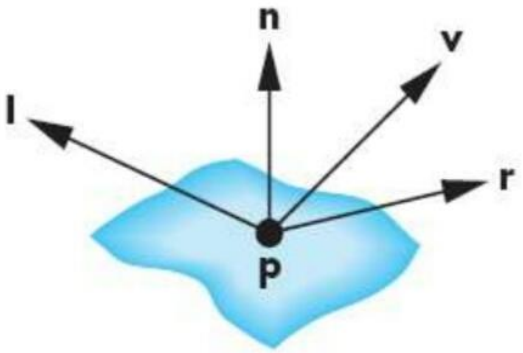
- **What is Phong Illumination Model?**
  - Components of Phong illumination or reflection model using RGB model: OpenGL allows us to break this light's emitted intensity into 3 components: ambient L_a, diffuse L_d, and specular L_s. Each type of light component consists of 3 color components, so, for example, L_{rd} denotes the intensity of the red component of diffuse illumination.

- It is a simple 3 parameter model i.e. the sum of 3 illumination terms:
  - ✓ **Diffuse** : non-shiny illumination and shadows
  - ✓ **Specular** : bright, shiny reflections
  - ✓ **Ambient** : 'background' illumination

Diffuse (directional) | Specular (highlights) | Ambient (color) | R_c

The Phong Lighting Model

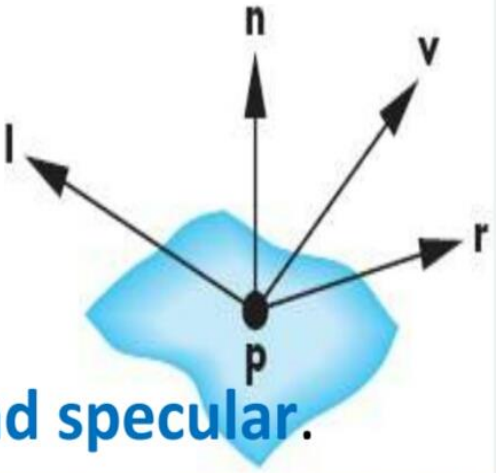
- The phong model is introduced by **Phong** and modified by **Blinn** .
- It is the basis for lighting and shading in graphics APIs and is implemented on virtually all graphics cards.
- The phong lighting model uses 4 vectors to calculate a **color** for an arbitrary point **P** on a surfac
- If the surface is curved, all 4 vectors can change as we move from point to point.



-By Manjula. S

The Phong Lighting Model

- Vector **n** is the normal at p.
- Vector **v** is in the direction from **p** to the viewer or **COP**.
- Vector l is in the direction of a line from p to an arbitrary point on the source for a distributed light source.
- Vector **r** is in the direction that a perfectly **reflected ray** from **l** would take.
- R is determined by **n** and **l.**
- Phong model supports 3 types of light material interactions – **ambient, diffuse and specular**.



-By Manjula. S

The Phong Lighting Model

- If the light source model has separate ambient, Diffuse and specular terms, we need 9coefficients to characterize a light source at any point **p** on the surface.

$$\mathbf{L}_i = \begin{bmatrix} L_{ira} & L_{iga} & L_{iba} \\ L_{ird} & L_{igd} & L_{ibd} \\ L_{irs} & L_{igs} & L_{ibs} \end{bmatrix}$$

First row contains the **ambient intensities** for RGB terms from source i.

Similarly, second and third rows contain **diffuse and specular intensities.**

The Phong Lighting Model

- We build the lighting model by summing the contributions for all the light sources at each point we wish to light.
- For each light source, we have to compute the amount of light reflected for each of the 9 terms in the illumination array.
- For each point, 9 coefficients, we can place in an array of reflection terms

$$\mathbf{R}_i = \begin{bmatrix} R_{ira} & R_{iga} & R_{iba} \\ R_{ird} & R_{igd} & R_{ibd} \\ R_{irs} & R_{igs} & R_{ibs} \end{bmatrix}$$

The Phong Lighting Model

Red intensity $$I_{ir} = R_{ira}L_{ira} + R_{ird}L_{ird} + R_{irs}L_{irs}$$

$$= I_{ira} + I_{ird} + I_{irs}.$$

- Total intensity $$I_r = \sum_i (I_{ira} + I_{ird} + I_{irs}) + I_{ar}$$

I_{ar} Red component of global ambient light.

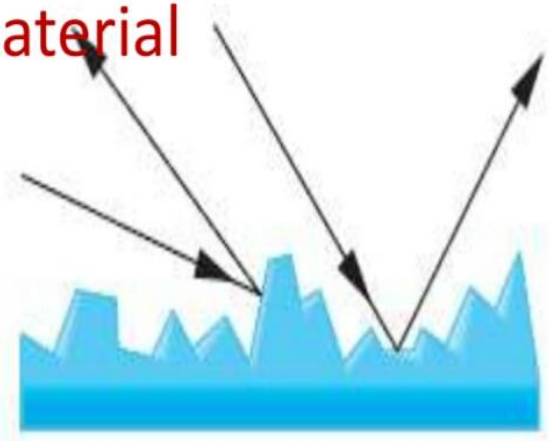
$$I = I_a + I_d + I_s = L_aR_a + L_dR_d + L_sR_s$$

-By Manjula. S

Ambient Reflection Diffuse Reflection

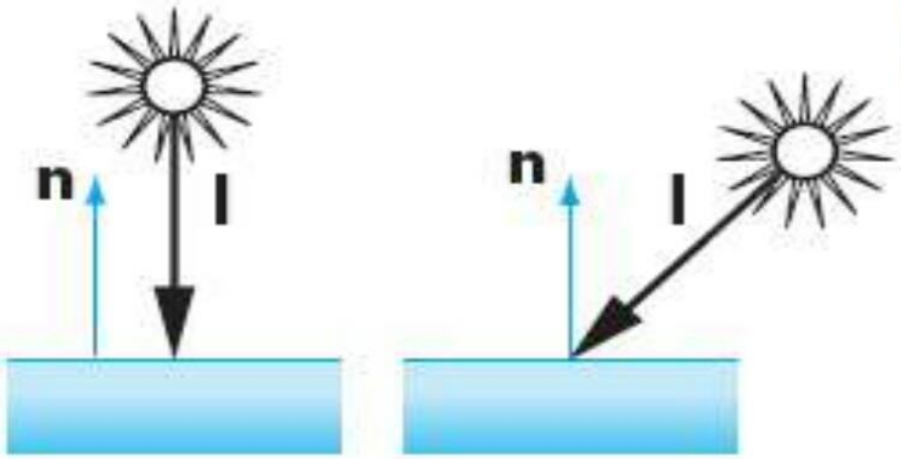
- The intensity of ambient light I_a is the same on every point the surface.
- The light ambient reflection co efficient $R_a = K_a$ $(0 <= K_a <= 1)$

  $I_a = K_a L_a$
- **Diffuse Reflection**
- Diffuse reflectors scatter light equally in all directions
- Amount of light reflected depends on material
- Characterized by rough surfaces
- **Lambertian Surfaces**



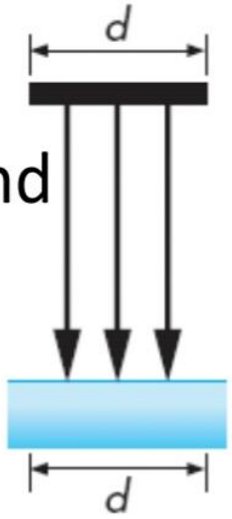
-By Manjula. S

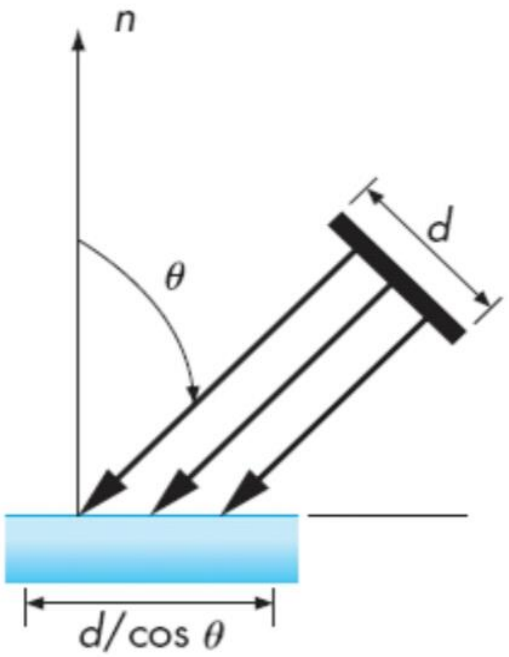
Diffuse Reflection



$$R_d \propto \cos\theta$$

$$\cos\theta = \mathbf{l} \cdot \mathbf{n}.$$





Θ is the angle between normal and point **n** and the direction of light source **l**

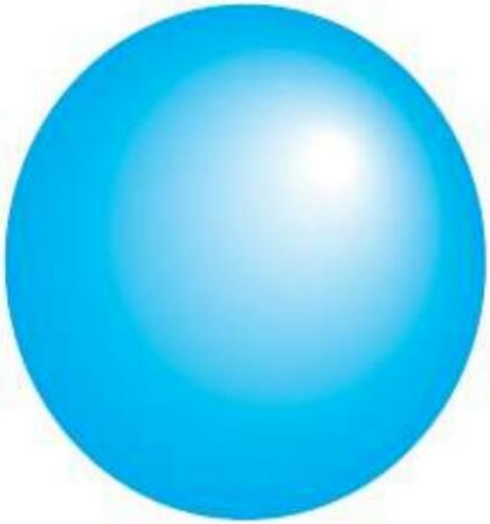
$$I_d = k_d(\mathbf{l} \cdot \mathbf{n})L_d.$$

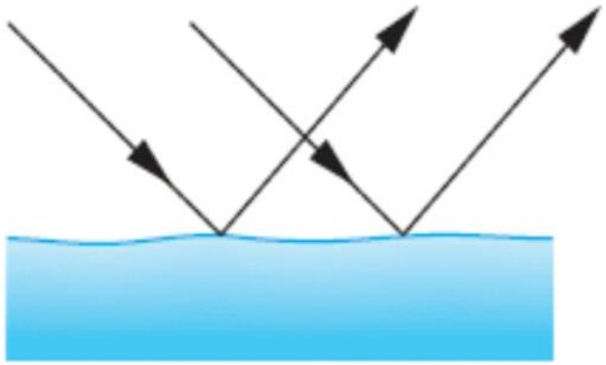
$$I_d = \frac{k_d}{a + bd + cd^2}(\mathbf{l} \cdot \mathbf{n})L_d.$$

-By Manjula. S

Specular Reflection

- High light from shiny object
- Smooth surface resembles mirror
- Light is reflected in small angles





-By Manjula. S

Specular Reflection

- The phong model uses the equation to find amount of the light viewer sees.

$$I_s = k_s L_s \cos^{\alpha} \phi.$$

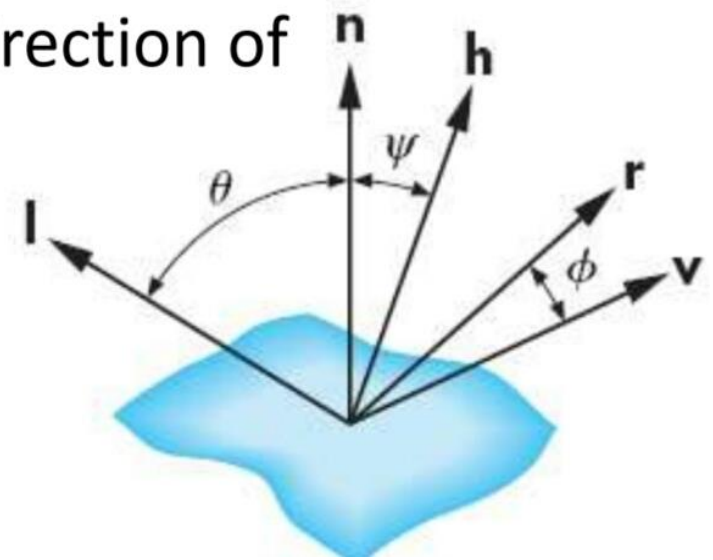
$$I_s = k_s L_s \max((\mathbf{r} \cdot \mathbf{v})^{\alpha}, 0).$$

$$I = \frac{1}{a + bd + cd^2}(k_d L_d \max(\mathbf{l} \cdot \mathbf{n}, 0) + k_s L_s \max((\mathbf{r} \cdot \mathbf{v})^{\alpha}, 0)) + k_a L_a.$$

-By Manjula. S

Modified Phong model

- Phong model with specular reflection in rendering
- Dot product of r and v recalculated at every point
- Half way vector $$\mathbf{h} = \frac{\mathbf{l} + \mathbf{v}}{|\mathbf{l} + \mathbf{v}|}$$
- If normal is in the direction of halfway vector then maximum reflection is in the direction of viewer
- $2\psi = \varphi$
- Replace r,v by n, h



Modified Phong model

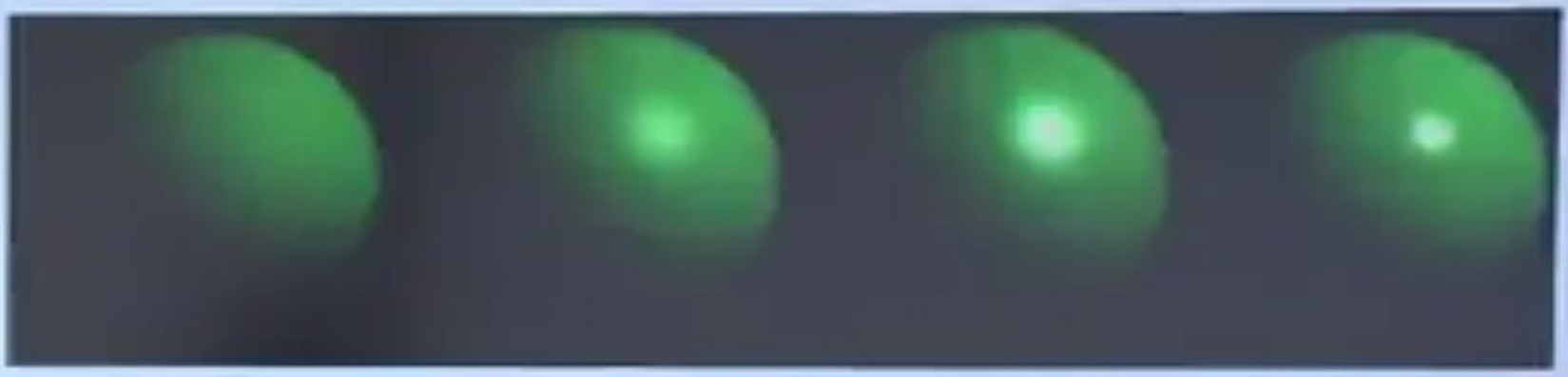
In the Phong Model replace $(v \cdot r)^{\alpha}$ by $(n \cdot h)^{\beta}$, β is chosen to match shininess.

Note that Halfway Angle is half of angle between r and v if vectors are coplanar.

Resulting model is known as the Modified Phong or Blinn-Phong Lighting model which is Specified as a standard in OpenGL.

What is Missing?

- **Combining Diffuse and Specular Reflections:**

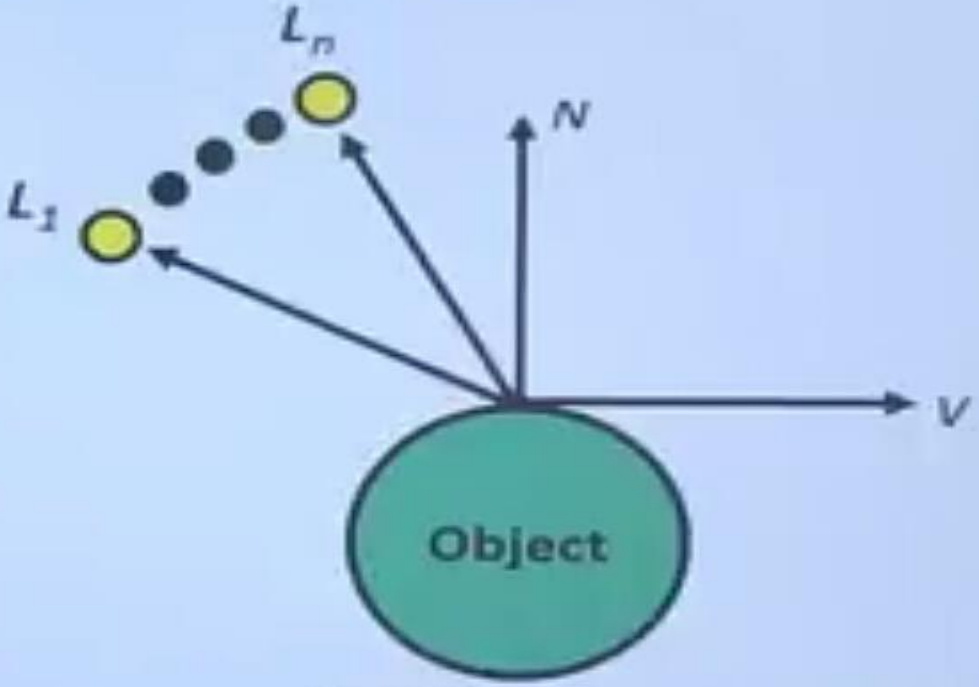


- ✓ Only the side that is lit by the light appears brighter.
- ✓ The other side of the object appears very dark, as if it is in the space.

-By Manjula. S

Multiple Light Sources

- If there are multiple light sources, we need to do the lighting computation for each light source and sum them altogether

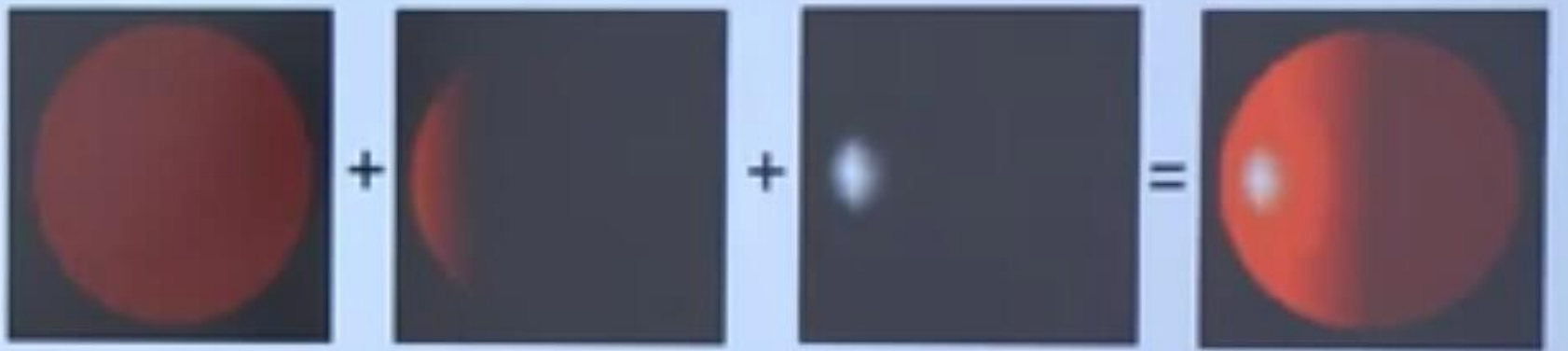


$$I_\lambda = I_a k_a + \sum_{p=1}^{lights} I_p \, [k_d \cos\theta + k_s \cos^n\alpha]$$

Combined Lighting Models

- **Summing it altogether : Phong Illumination Model**

$$I_\lambda = I_a k_a + I_p [k_d \cos\theta + k_s \cos^n \alpha]$$



Ambient (color) | Diffuse (directional) | Specular (highlights) | R_c

Using the Dot Products

- Use dot product of the vector instead of calculating the angles θ, α

$$I_\lambda = I_a k_a + \sum_{p=1}^{lights} I_p \left[k_d (\overline{N} \bullet \overline{L})\, k_s (\overline{V} \bullet \overline{R})^n\right]$$

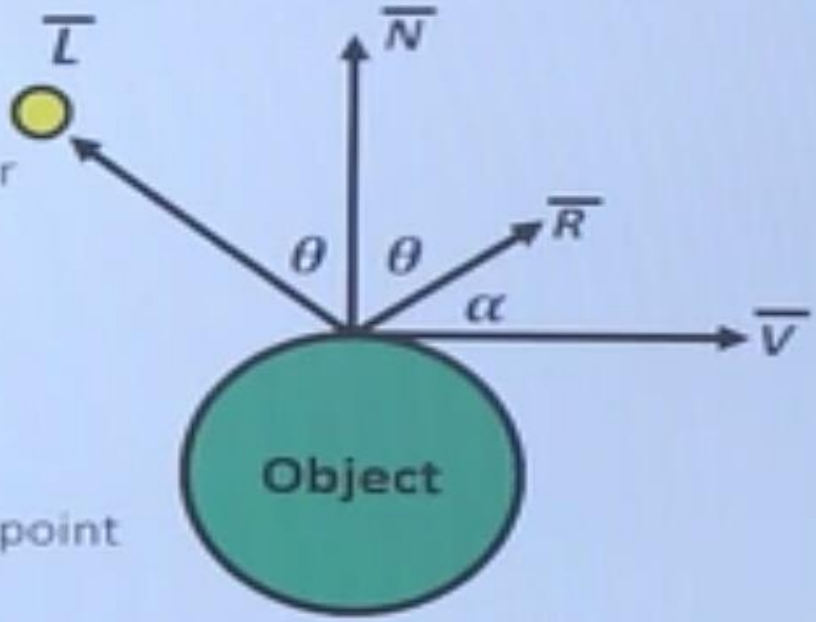
($\cos\theta$ and $\cos\alpha$ are the dot products)

$\overline{V}$: Vector from the surface to the viewer

$\overline{N}$: Normal vector at the colored point

$\overline{R}$: Normalized reflection vector

$\overline{L}$: Normalized vector from the colored point towards the light source



Specifying Light Sources in OpenGL

- For each light source, we can set an RGBA for the diffuse, specular, and ambient components, and for the position

```
GL float diffuse0[]={1.0, 0.0, 0.0, 1.0};
GL float ambient0[]={1.0, 0.0, 0.0, 1.0};
GL float specular0[]={1.0, 0.0, 0.0, 1.0};
Glfloat light0_pos[]={1.0, 2.0, 3,0, 1.0};

glEnable(GL_LIGHTING);
glEnable(GL_LIGHT0);
glLightv(GL_LIGHT0, GL_POSITION, light0_pos);
glLightv(GL_LIGHT0, GL_AMBIENT, ambient0);
glLightv(GL_LIGHT0, GL_DIFFUSE, diffuse0);
glLightv(GL_LIGHT0, GL_SPECULAR, specular0);
```

Specifying Materials in OpenGL

General form:

glMaterialf(face, parameter,value);

glMaterialfv(face, parameter,*array);

face is GL_FRONT, GL_BACK, GL_FRONT_AND_BACK

Specifying Materials in OpenGL

face is GL_FRONT, GL_BACK, GL_FRONT_AND_BACK

parameter is:

GL_AMBIENT four values that specify the ambient RGBA reflectance of the material. **(0.2,0.2,0.2,1.0)**

GL_DIFFUSE four values that specify the diffuse RGBA reflectance of the material. **(0.8,0.8,0.8,1.0)**

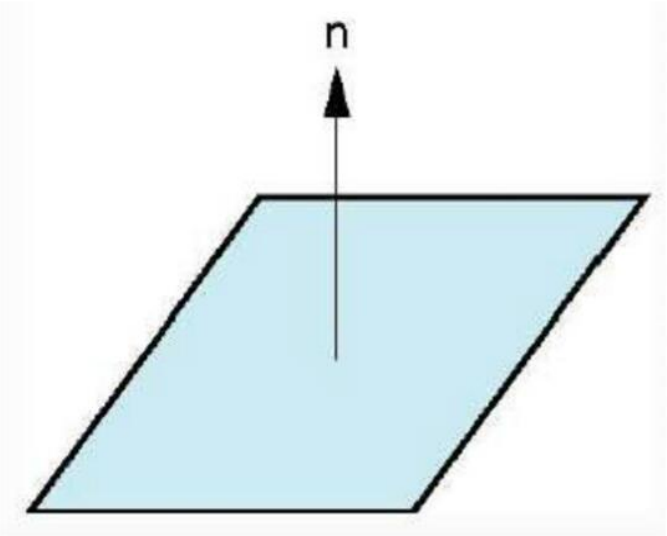
GL_SPECULAR four values that specify the ambient RGBA reflectance of the material. **(0.0,0.0,0.0,1.0)**

GL_SHININESS specifies the specular reflectance exponent of the material. **0.0**

Plane Normals

- Equation of plane: ax+by+cz+d= 0
- we know that plane is determined by three points p0, p2, p3 or normal n and p0
- Normal can be obtained by

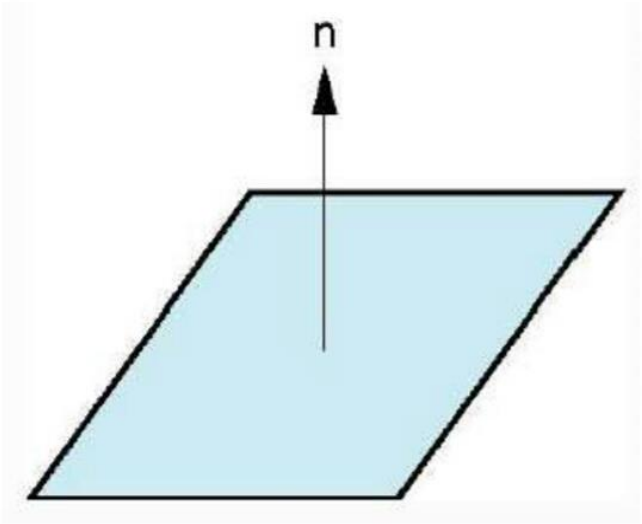
  n= (p2-p0) ×(p1-p0)



Plane Normals

- Equation of plane: ax+by+cz+d= 0
- we know that plane is determined by three points p0, p2, p3or normal n and p0
- Normal can be obtained by

  n= (p2-p0) ×(p1-p0)



Normal to Sphere

- Implicit function f(x,y.z)=0
- Normal given by gradient
- Sphere f(p)=p·p-1
- n = [.f/.x, .f/.y, .f/.z]T=p

Shading Models

- **There are different types of shading models –**

- ***Flat Shading*** or ***Constant Shading*** or ***Faceted Shading*** – Less computation needed.

- ***Interpolated Shading*** – The constant shading scheme is a kind of trade-off between computational expense and realistic shading quality. To yield more realistic shading with intensity varying from point to point on a surface needs due consideration to change of curvature, direction of incident light and position of viewpoint from one point on a surface to another.
  - ***Gouraud Shading*** – Once per vertex.
  - ***Phong Shading*** – Once per pixel and thus requires heavy computations..

Polygon shading

1. Flat Shading
2. Smooth (gauraud) or interpolative shading
3. Phong Shading

Flat Shading

- **Flat Shading or Constant Shading or Faceted Shading:**
  - ✓ Compute the color at the middle of the polygon.
  - ✓ All pixels in the same polygon are colored by the same color.
  - ✓ Works well for objects really made of flat faces.

Flat Shading

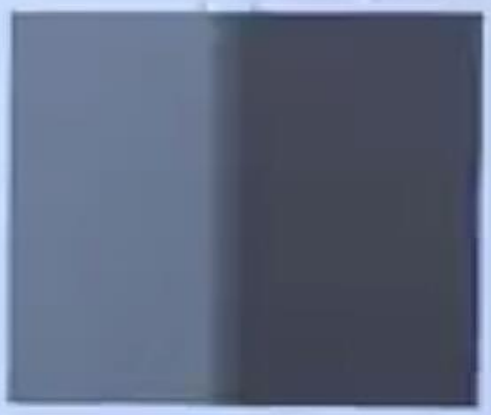
Flat Shading or Constant Shading or Faceted Shading (Contd.):

- ✓ Suffers from Mach band effect
- ✓ Humans are very sensitive to the sudden change of the brightness
- ✓ The artefact remains although the polygon number is increased

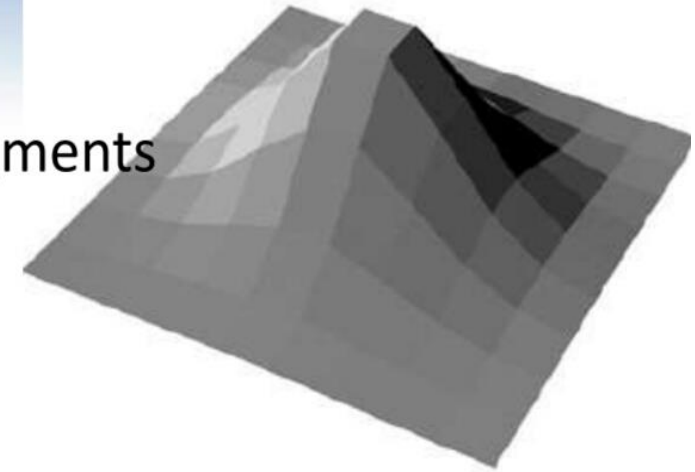
Note: **Mach** bands is an optical illusion named after the physicist Ernst **Mach**. It magnifies the contrast between edges of the slightly differing shades of gray, as soon as they contact one another, by triggering edge-detection in the human visual system.

Mach Band (by Ernst Mach)

An Optical Illusion



Flat shading



- Pre vertex lightening calculations

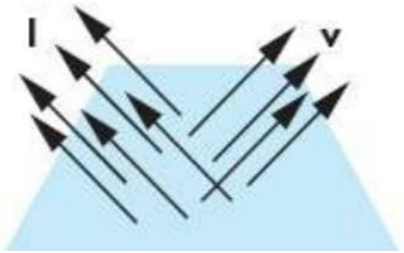
  Shades are computed and Assigned to fragments

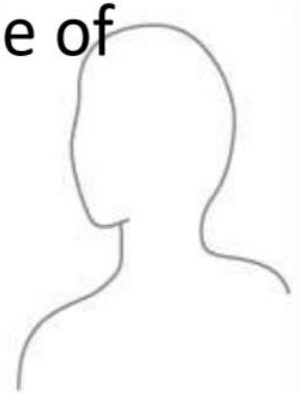
  Shade is computed

- Consider vector **l, v** and **n**
- Shading once for each polygon

  glShadeModel(GL_FLAT);

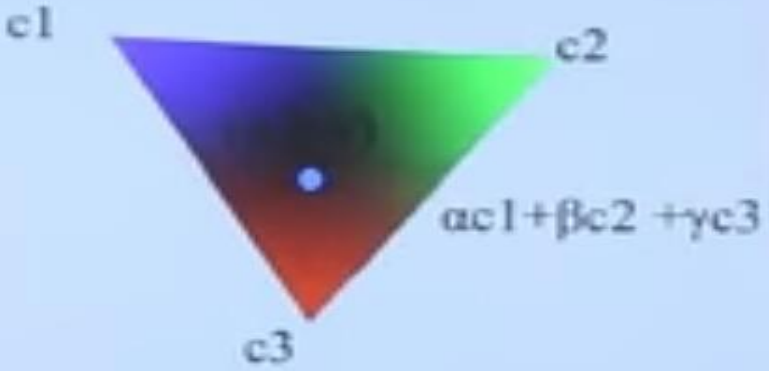
The color at the first vertex will determine the shade of the whole polygon





Gouraud Shading

- **Gouraud Shading:**
  - ✓ Proposed by Henri Gouraud.
  - ✓ Computing the color per vertex by local illumination model.
  - ✓ Then, interpolating the color within the polygons.

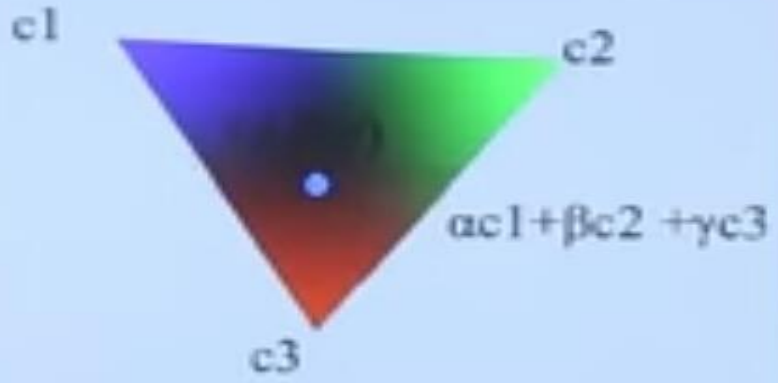


We can interpolate the color by Barycentric coordinates. **Barycentric coordinates** are triples of numbers corresponding to masses placed at the vertices of a reference triangle . These masses then determine a point , which is the geometric centroid of the three masses and is identified with **coordinates** .

-By Manjula. S

Gouraud Shading

- **Gouraud Shading:**
  - ✓ Proposed by Henri Gouraud.
  - ✓ Computing the color per vertex by local illumination model.
  - ✓ Then, interpolating the color within the polygons.



We can interpolate the color by Barycentric coordinates. **Barycentric coordinates** are triples of numbers corresponding to masses placed at the vertices of a reference triangle . These masses then determine a point , which is the geometric centroid of the three masses and is identified with **coordinates** .

-By Manjula. S

Gouraud Shading

- Following steps are required to carry out for each polynomial face of the surface mesh for this shading scheme –

  - **Step 1:** Determine the unit normal vector at each vertex of a polygon. If a vertex is shared by a number of polygon faces the average of the normals to all the faces at that vertex gives the desired normal vector.

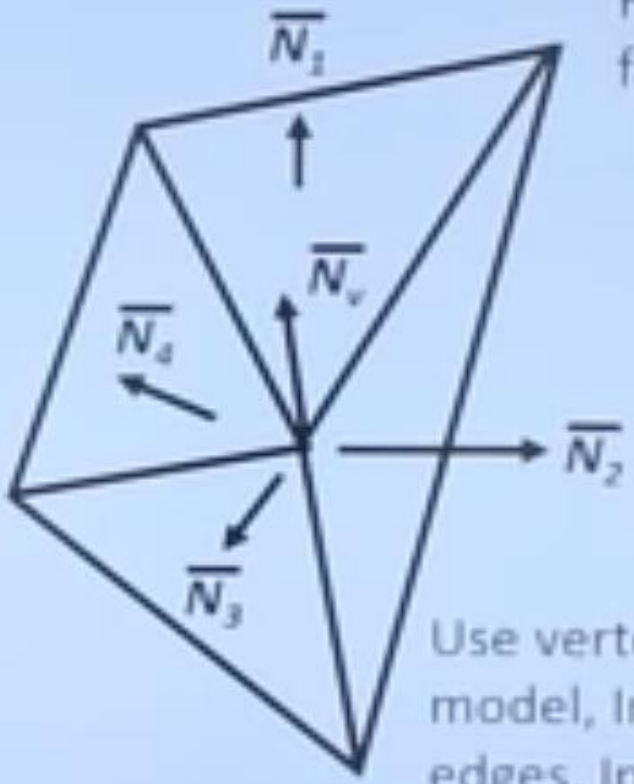
  - **Step 2:** Find the intensities at each vertex by applying illumination models separately using respective vertex normals

  - **Step 3:** For each scan line, calculate intensities at the intersection of a scan line with an edge by linear interpolation of intensities at the edge end points (vertices).

  - **Step 4:** Calculate intensities at points along a scan line between the edges by linear interpolation of intensities at the intersections of the scan line with the edges.

Gouraud Shading

Computing the Vertex Normals:



Find vertex normals by averaging face normal

$$\overline{N}_V = \frac{\sum_{1 \le i \le n} \overline{N}_i}{\left| \sum_{1 \le i \le n} \overline{N}_i \right|}$$

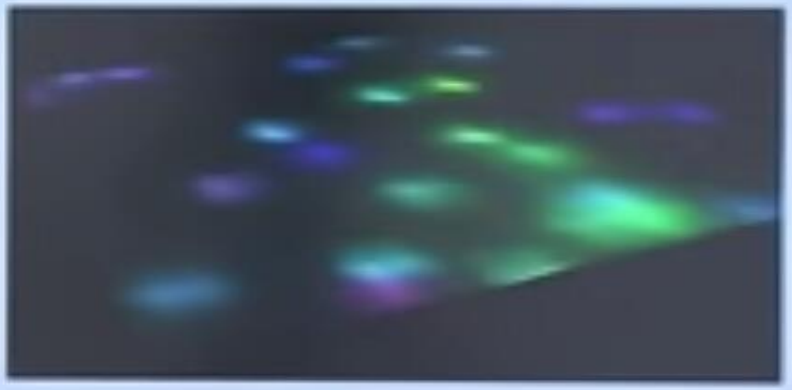
Use vertex normals with desired shading model, Interpolate vertex intensities along edges. Interpolate edge values across a scanline.

-By Manjula. S

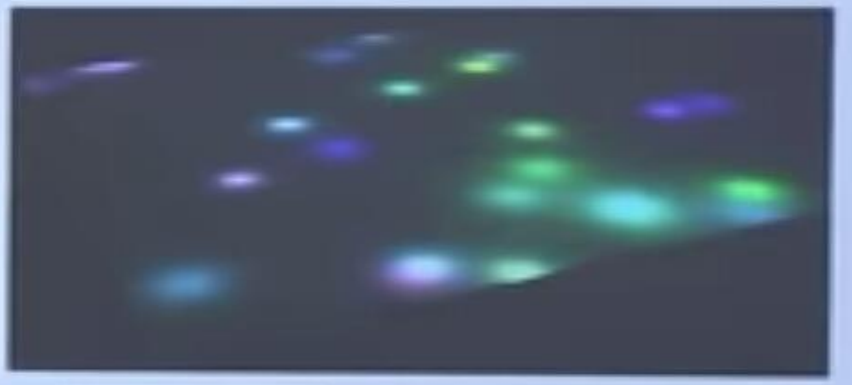
Gouraud Shading

- **Problems with Gauraud Shading:**
  - One of the major disadvantages of Gouraud shading is that it avoids finding surface normal at each point on every scan-line. As a result the change in curvature from point to point is overlooked.

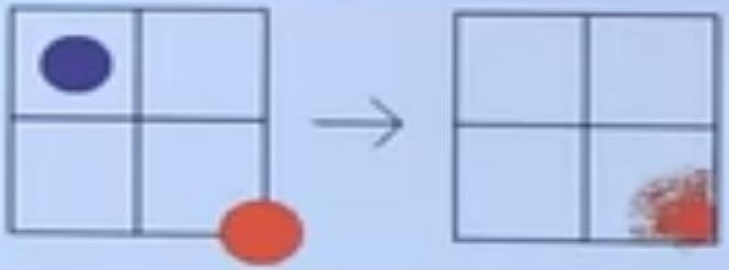
Gouraud Shaded Floor



Phong Shaded Floor



Gouraud shading is not good when the polygon count is low



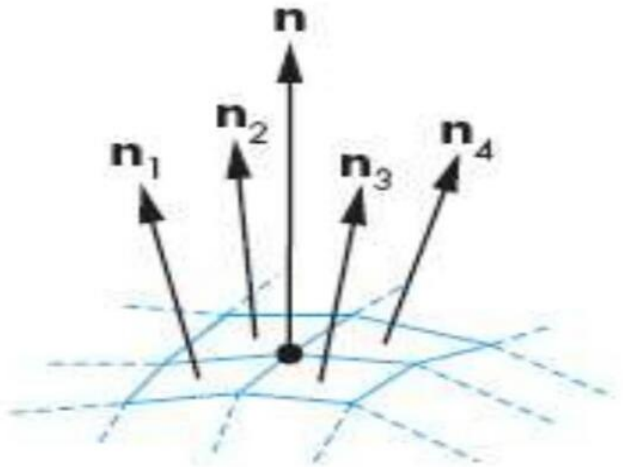
Smooth shading

- By default in OpenGL
- Shading calculations are done for each vertex

  Vertex colors become vertex shades
- By default, vertex shades are interpolated across the polygon

  glShadeModel(GL_SMOOTH);

$$\mathbf{n} = \frac{\mathbf{n}_1 + \mathbf{n}_2 + \mathbf{n}_3 + \mathbf{n}_4}{|\mathbf{n}_1 + \mathbf{n}_2 + \mathbf{n}_3 + \mathbf{n}_4|}.$$



-By Manjula. S

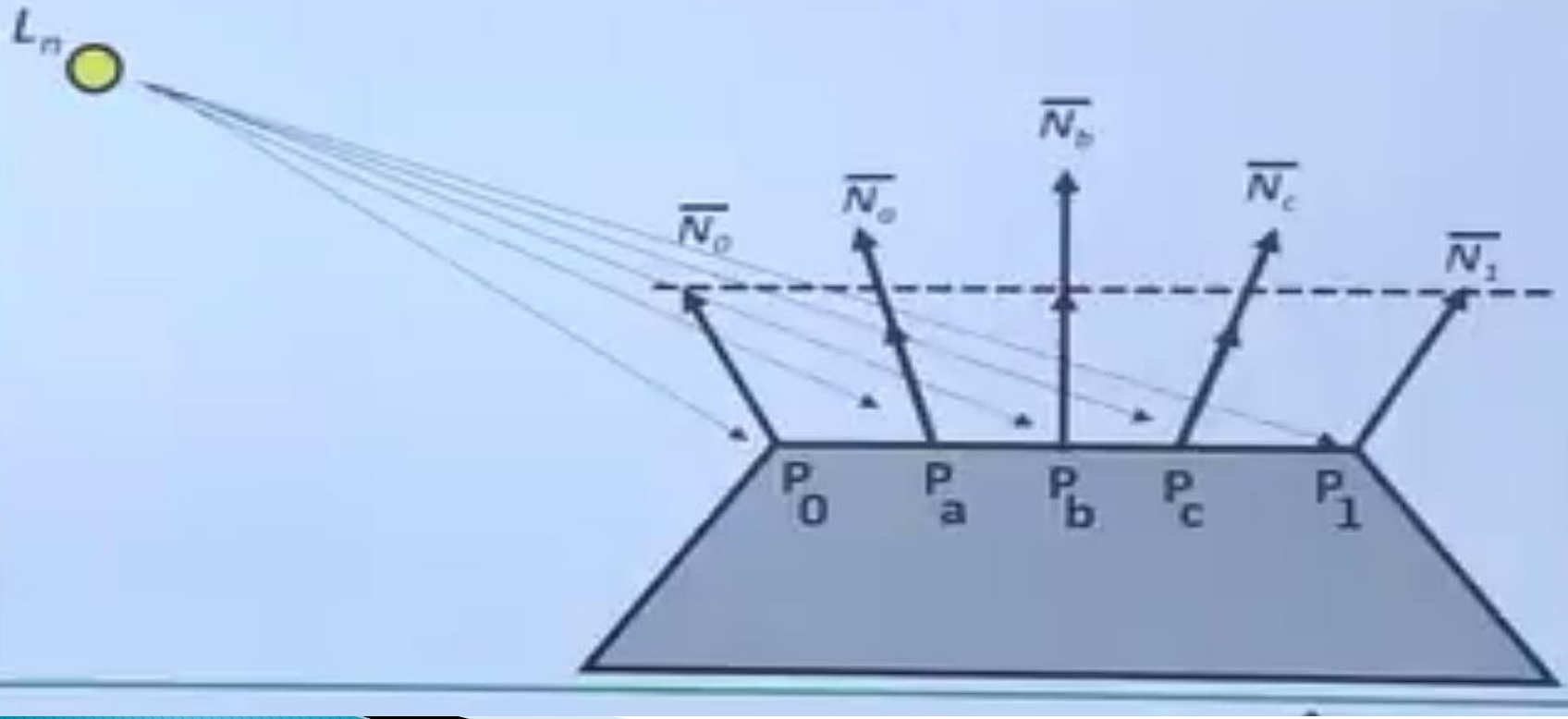
Phong Shading

Phong Shading was proposed by Bui Tuong Phong.
This shading achieves better approximation to the true shape of a surface and thus a better rendering of the surface by determining surface normals at each point using linear interpolation technique.
And the process is computationally expensive.

-By Manjula. S

Phong Shading

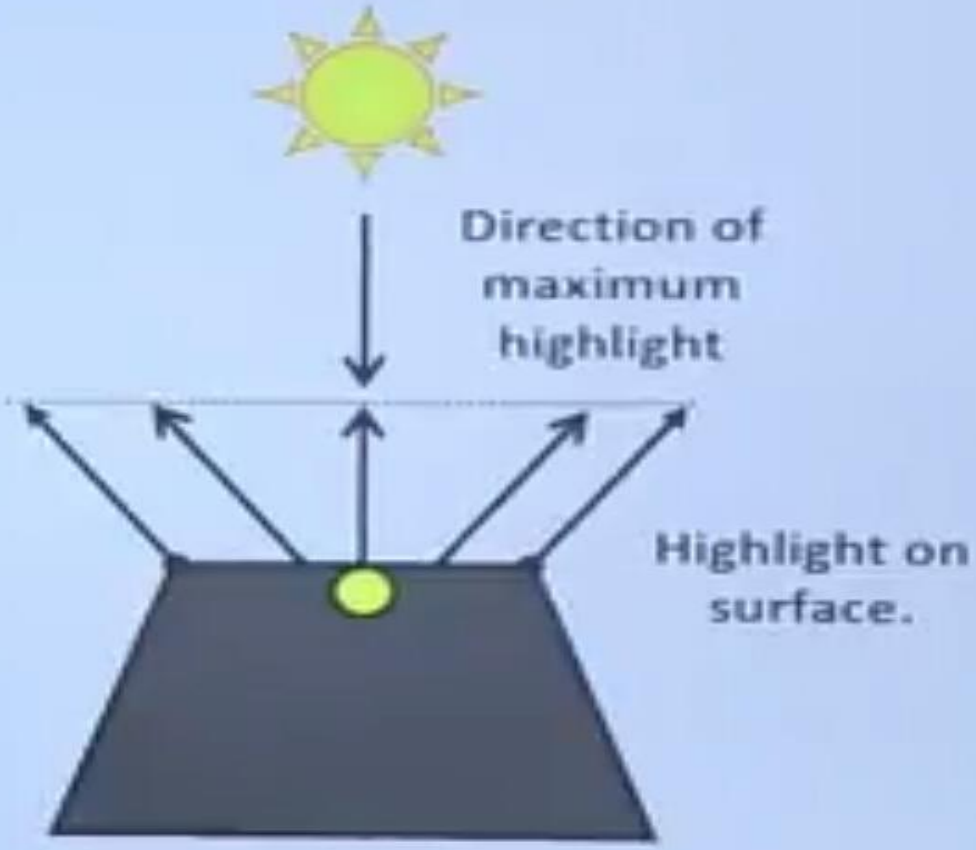
- **Phong Shading features:**
  - ✓ In Phong shading doing the lighting computation at every pixel during rasterization.
  - ✓ Interpolating the normal vectors at the vertices (again using Barycentric coordinates).



-By Manjula. S

Phong Shading

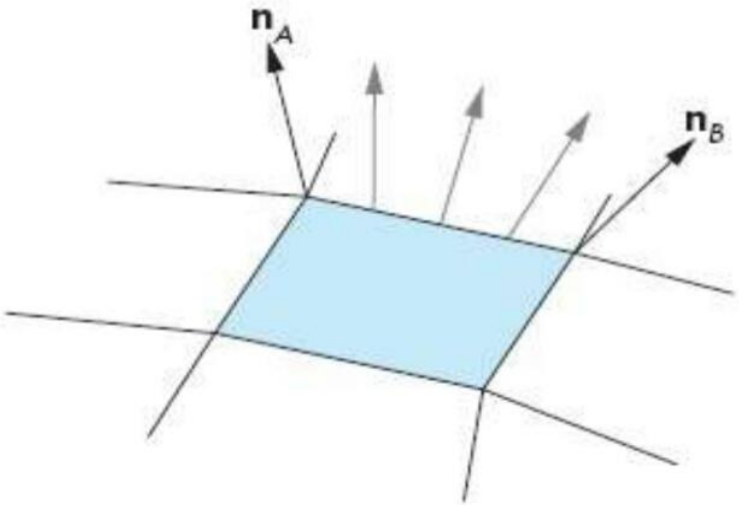
- **Phong Shading features (Contd.):**
  - ✓ For specular reflection, highlight falls off with $\cos^n\alpha$.
  - ✓ Can well produce a highlight that occurs in the middle of the face.

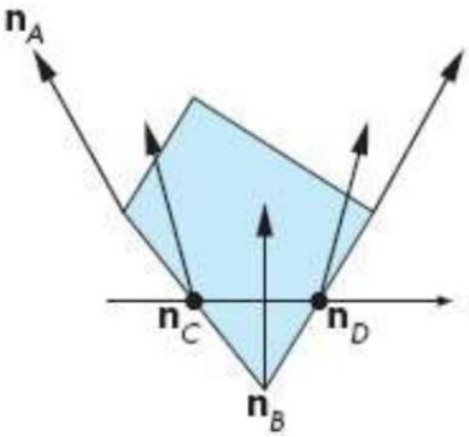


Phong shading

- Interpolating vertex intensities
- Compute the vertex normal by interpolating over the normals of the polygon that share the vertex

$$\mathbf{n}_C(\alpha) = (1-\alpha)\mathbf{n}_A + \alpha\mathbf{n}_B. \qquad \mathbf{n}(\alpha,\beta) = (1-\beta)\mathbf{n}_C + \beta\mathbf{n}_D.$$





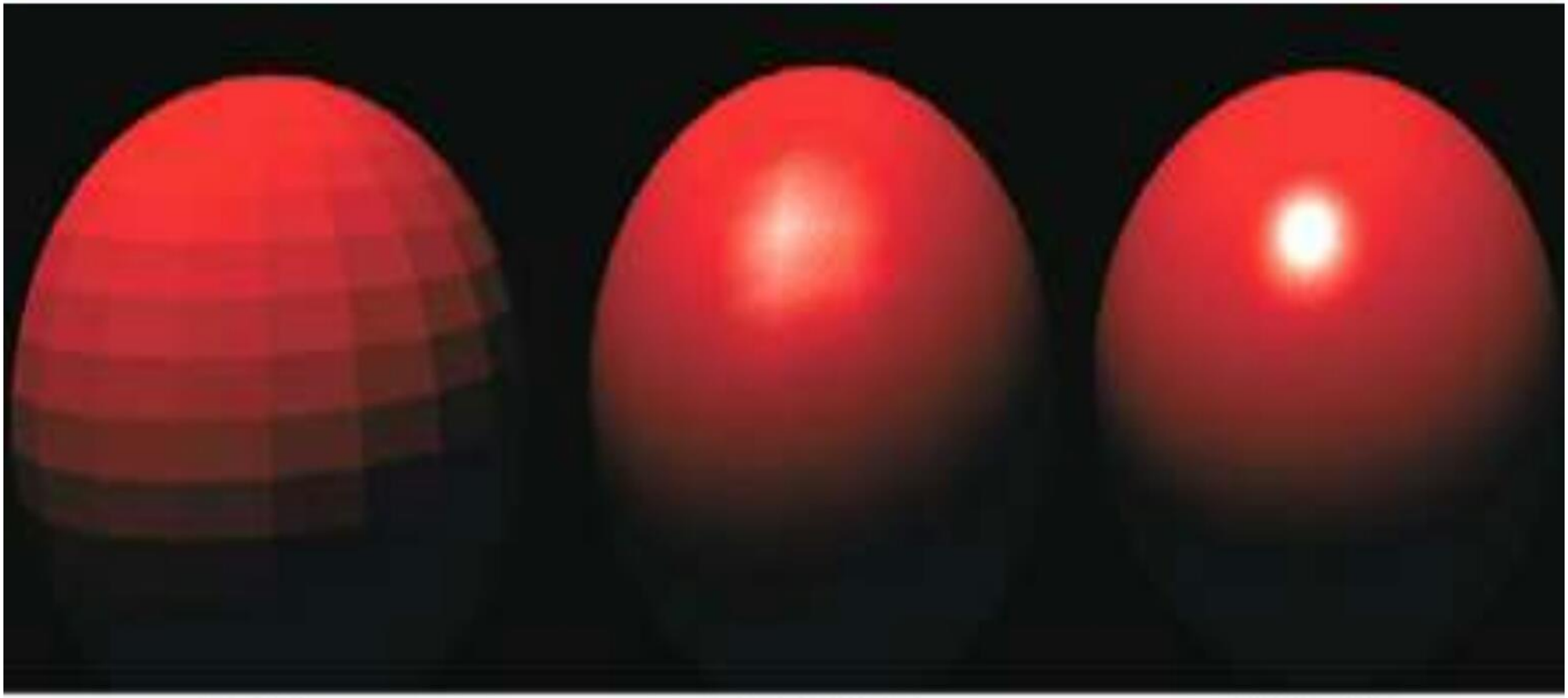
-By Manjula. S

Phong Shading

- **Phong Shading example:**
  - ✓ The following picture depicts why Phong shading is better that Flat and Gouraud shading models.

Flat Gouraud Phong

shading



-By Manjula. S

Thank You

-By Manjula. S