

Unit 5

Illumination Models & Surface-Rendering Methods

Mrs. Deepali K. Jaadhav, KITCoEK

Illumination Models & Surface-Rendering Methods

- Light sources
- Basic illumination models
- Displaying light intensities
- Halftone patterns and Dithering Techniques
- Polygon Rendering methods
- Ray tracing methods

Light Sources

Seeing Object

How do you see an object?

Light from the object enters your eye.

Do you see all objects in the same way?

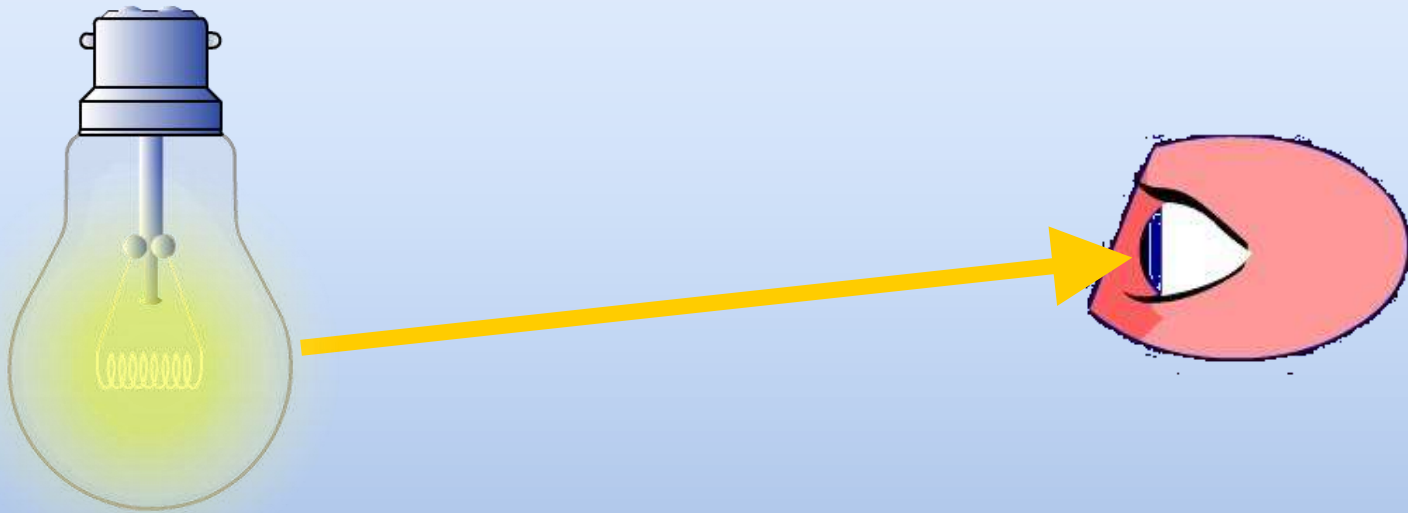
There are two ways you see objects:

- You see some objects because they are **light sources**.
- You see other objects by **reflected light**.

Seeing a luminous object

A **luminous** object gives out light and can also be called a light source.

How does light from a light bulb and other light sources reach your eye?

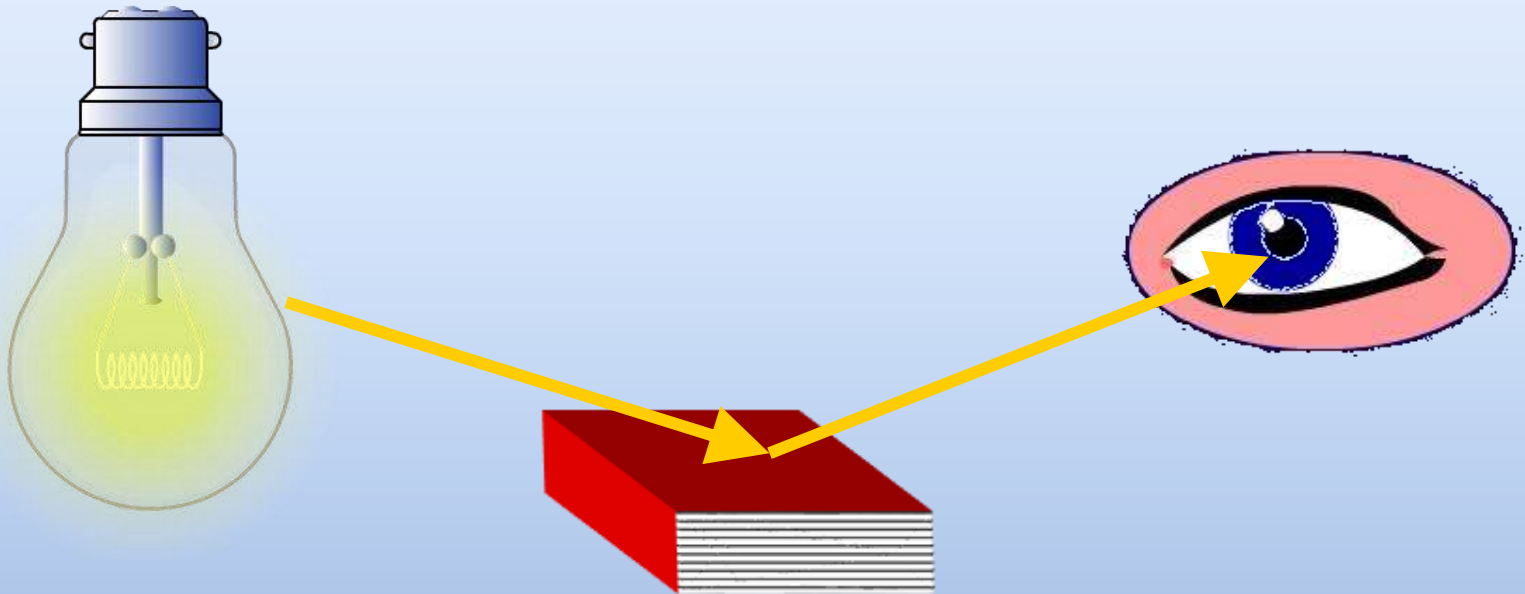


Light travels in a straight line directly into your eye.

Seeing a non-luminous object

Objects that do not give out light are **non-luminous**.

How does your eye see **non-luminous** objects such as a book?

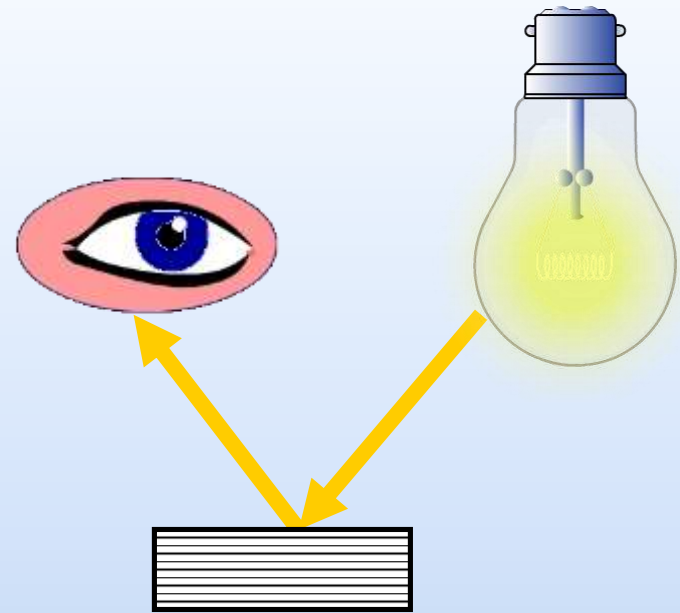


Light from the light source strikes the book and some of the light is reflected into your eye.

Seeing colours

How do you see non-luminous objects such as a book?

You see a non-luminous object when light hits the object and is then reflected into your eyes.



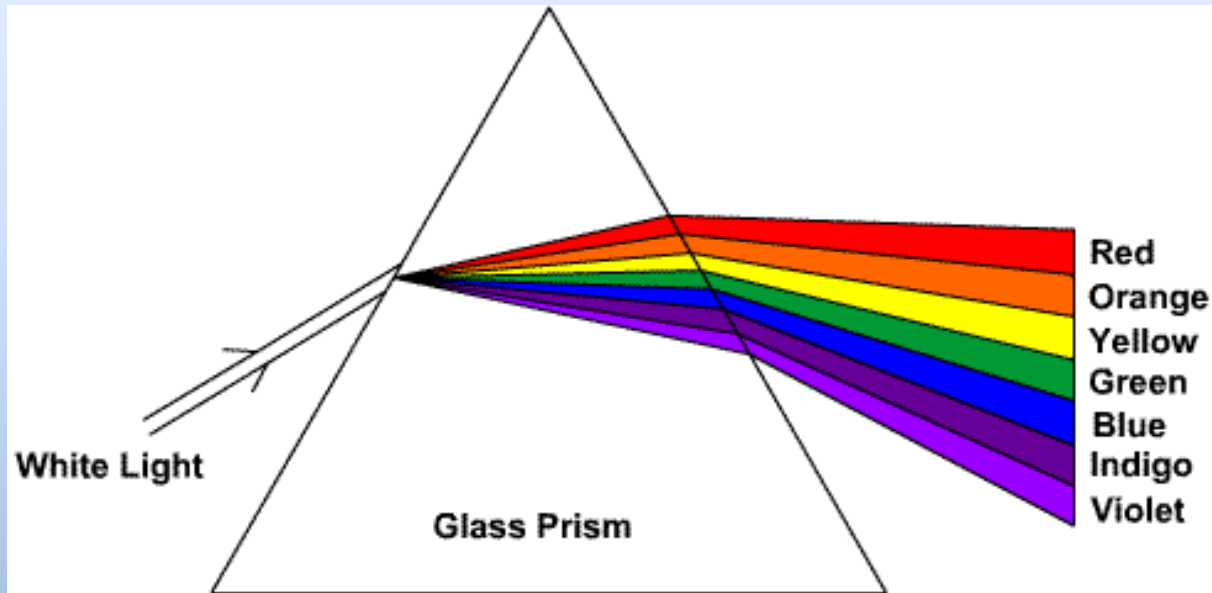
So how do we see different colours?

Why does a red dress look red?

Why does a green apple look green?

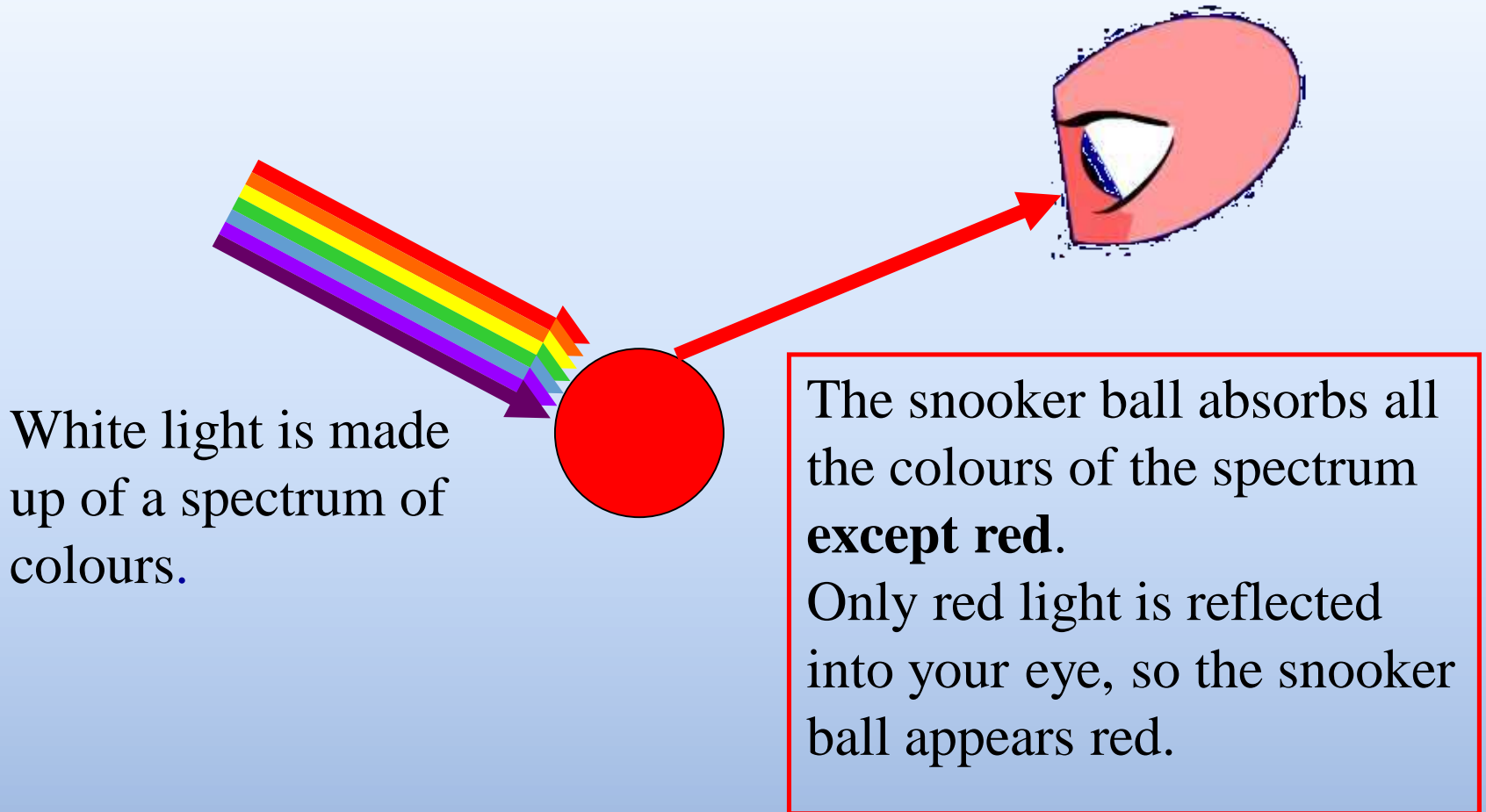
White Light

- White light is a mixture of seven colors (or seven colored light)



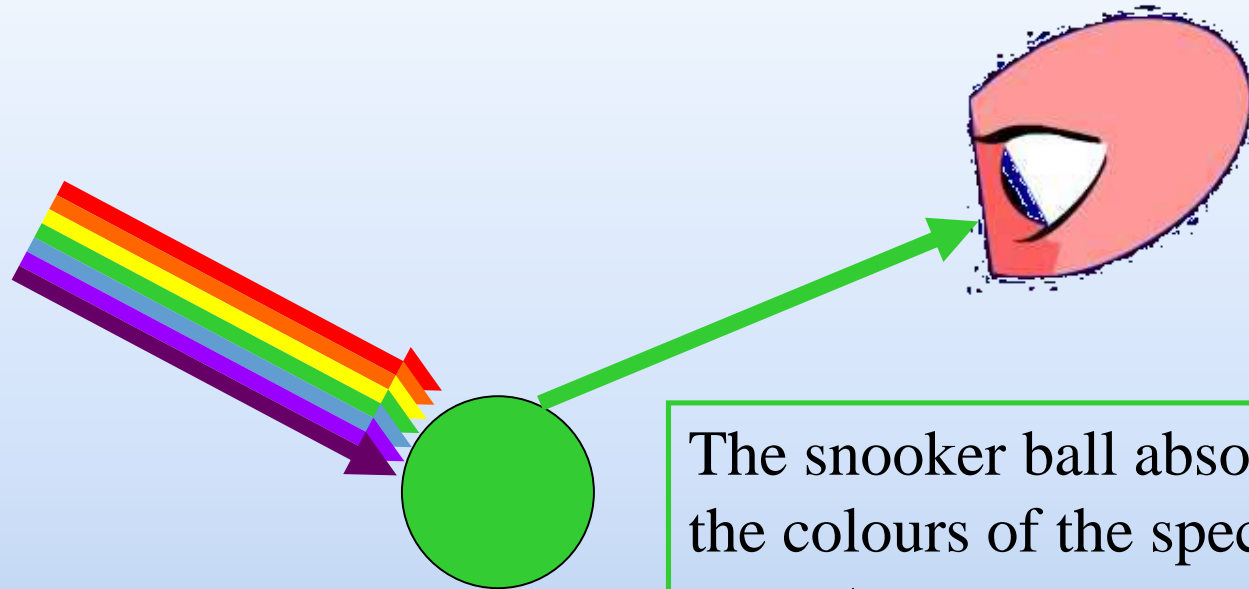
Seeing red

Why does a **red** snooker ball look **red** in white light?



Seeing green

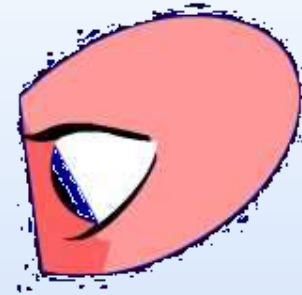
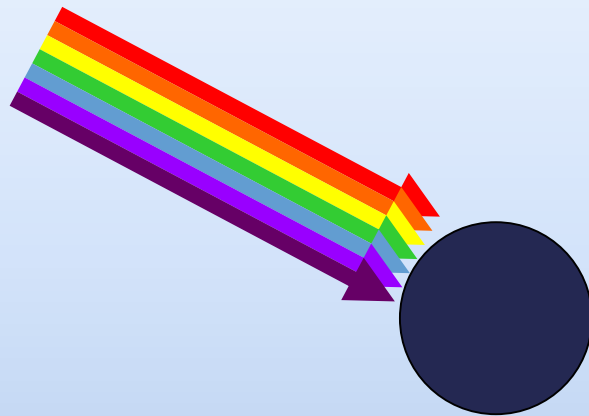
Why does a **green** snooker ball look **green** in white light?



The snooker ball absorbs all the colours of the spectrum **except green**.
Only green light is reflected into your eye, so the snooker ball appears green.

Seeing black

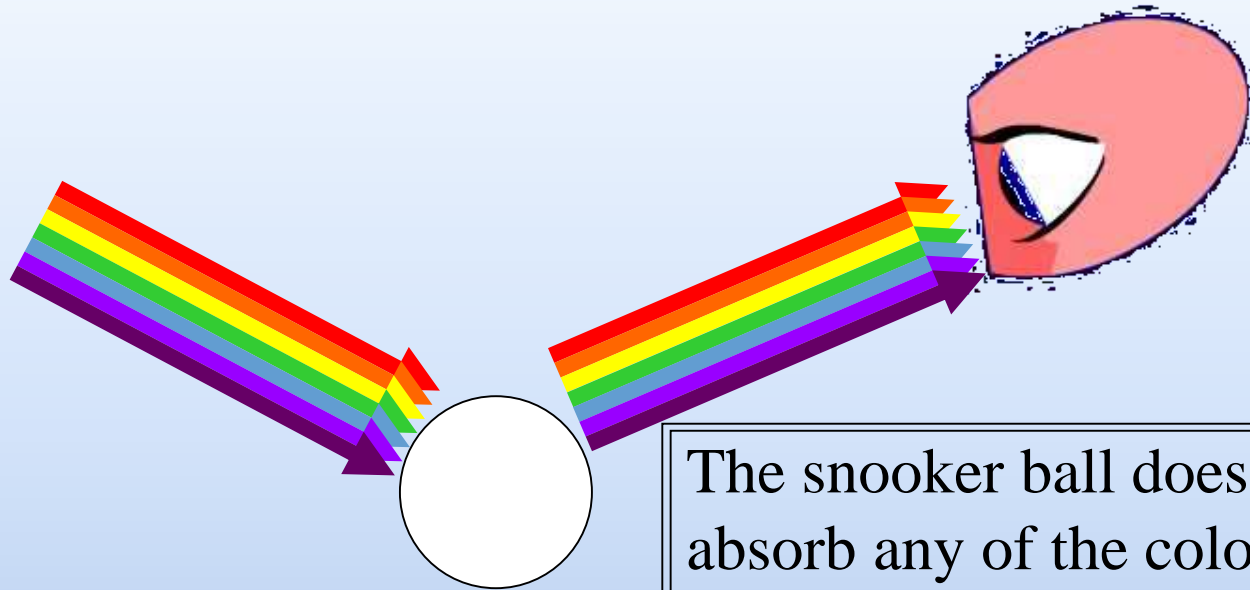
Why does a **black** snooker ball look **black** in white light?



The snooker ball absorbs all the colours of the spectrum. No light is reflected into your eye, so the snooker ball appears black.

Seeing white

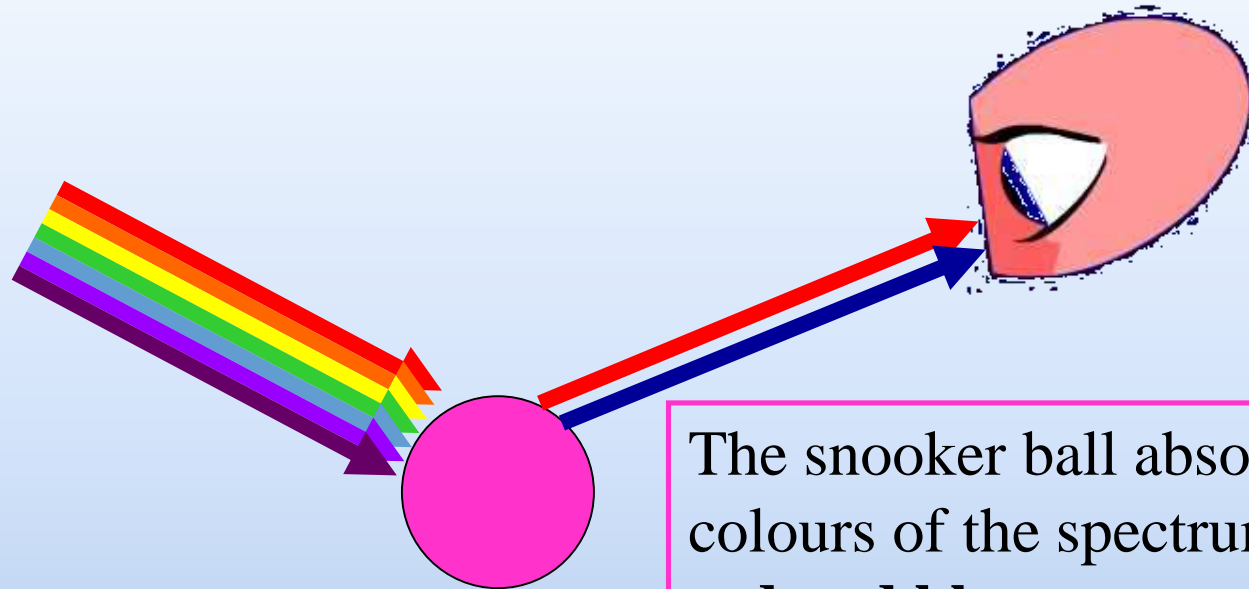
Why does a white snooker ball look white in white light?



The snooker ball does **not** absorb any of the colours of the spectrum.
The whole spectrum of light is reflected into your eye, so the snooker ball appears white.

Seeing magenta

Why does a magenta ball look magenta in white light?



The snooker ball absorbs all the colours of the spectrum **except red and blue.**

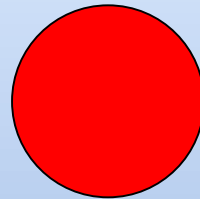
Red and blue light are reflected into your eye, so the snooker ball appears magenta.

Seeing colours in coloured light

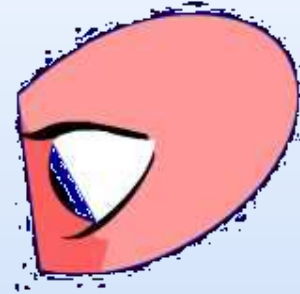
Why do colours look different in different coloured light?

Consider a red ball in red light.

The red light
shines on the
ball.



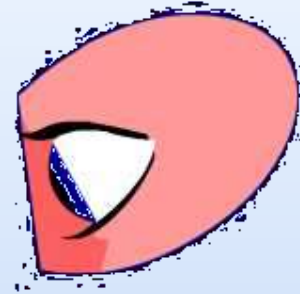
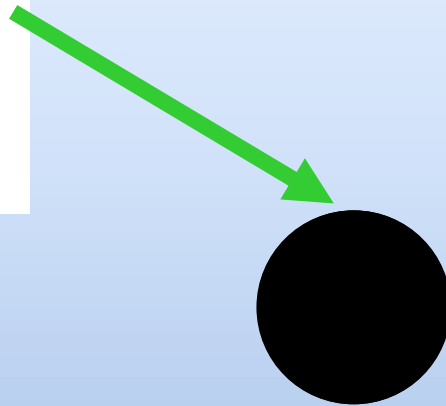
The red ball reflects
the red light and so
appears red.



Seeing colours in coloured light

What colour does a red ball appear in green light?

The green light shines on the ball.



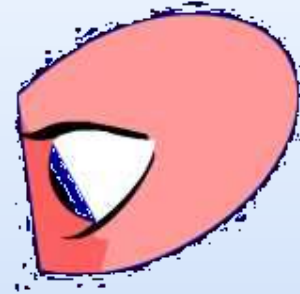
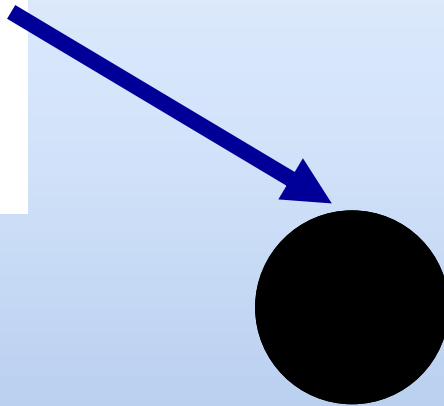
The red ball only reflects red light and so it absorbs the green light.

So in green light, this ball does not reflect any light and so **appears black**.

Seeing colours in coloured light

What colour does a green ball appear in blue light?

The blue light
shines on the
ball.



The green ball only reflects
green light and so it absorbs
the blue light.

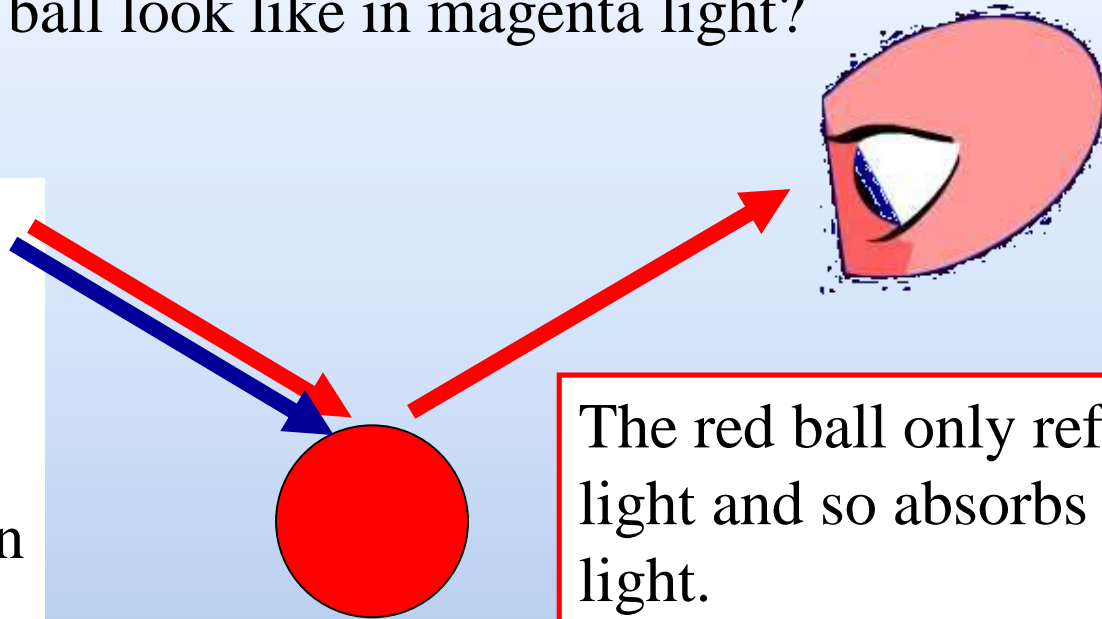
So in blue light, this ball does
not reflect any light and so
appears black.

Seeing colours in coloured light

What happens when using a coloured filter which lets through more than one type of light?

What will a red ball look like in magenta light?

The magenta light, which is a mixture of red and blue light, shines on the ball.

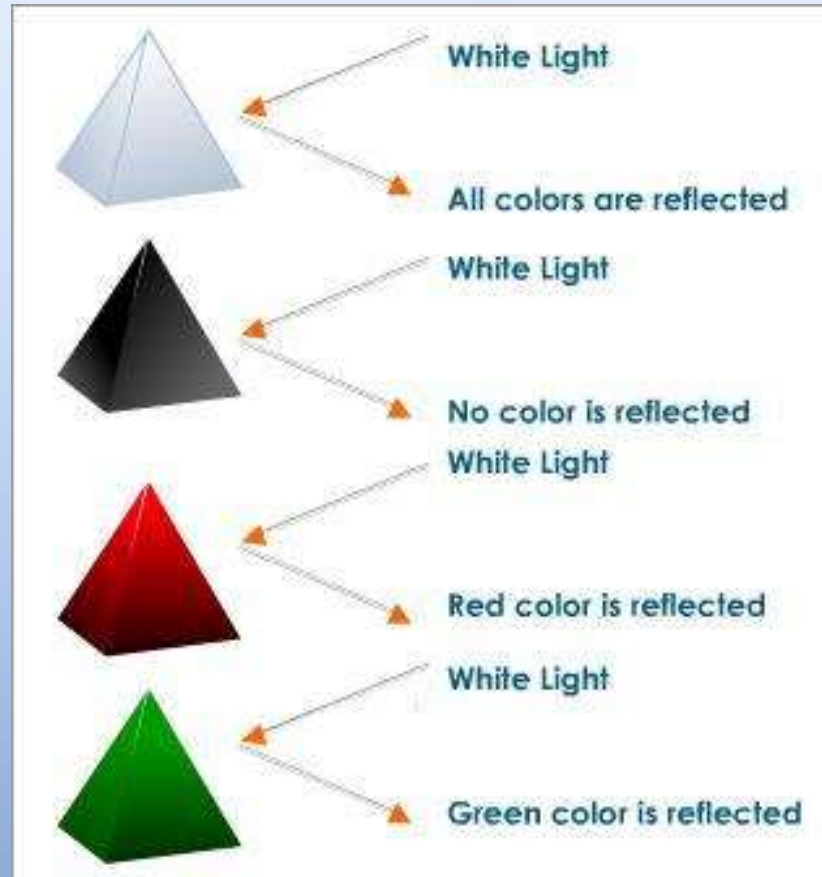


The red ball only reflects red light and so absorbs the blue light.

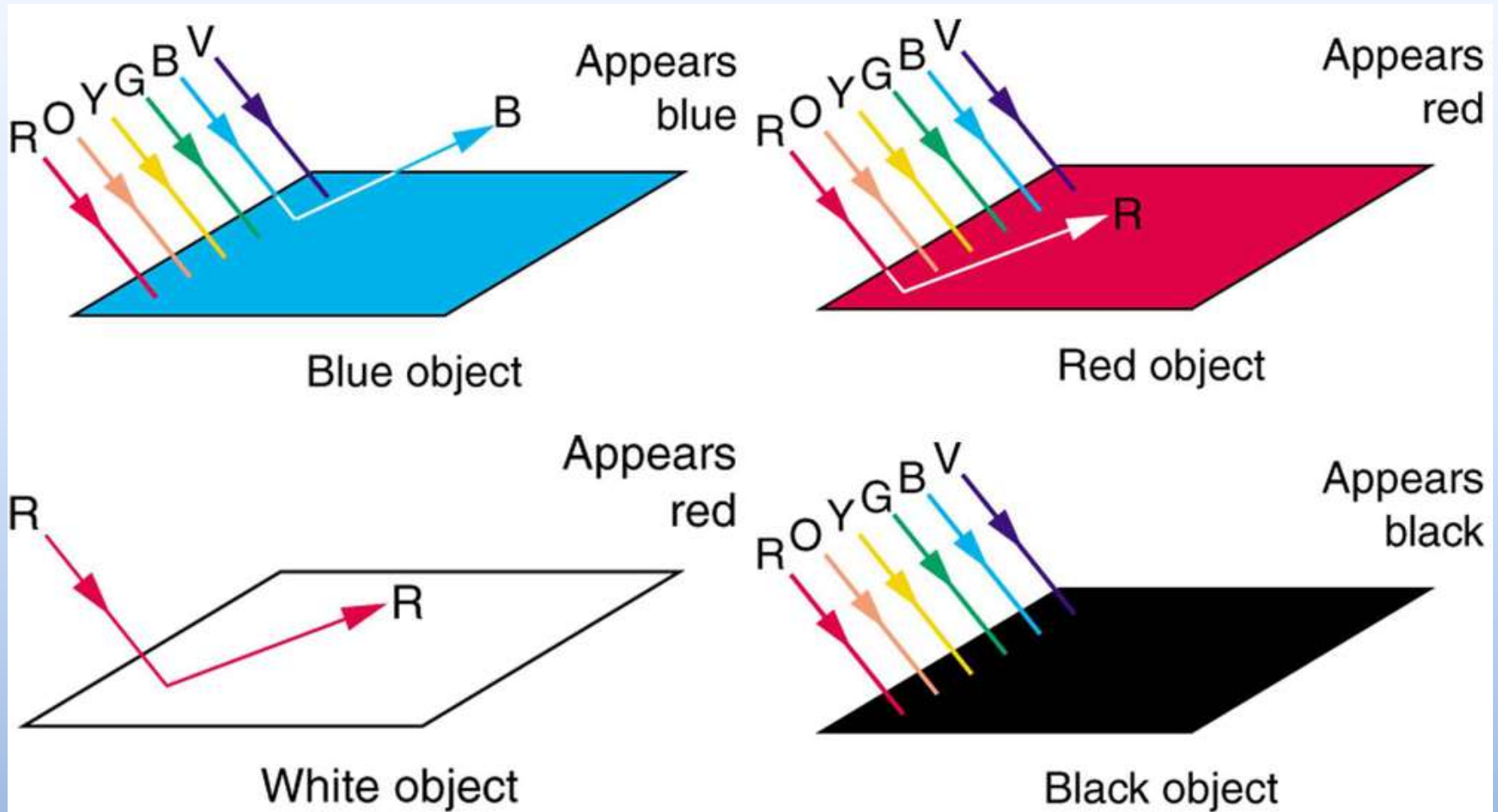
So in magenta light, this ball reflects the red light and **appears red.**

Seeing color of Object






- The color of a non-luminous, opaque object depends upon the color of the light reflected by it.
- The color of the reflected light depends upon the color of the incident light.



Seeing color of Object

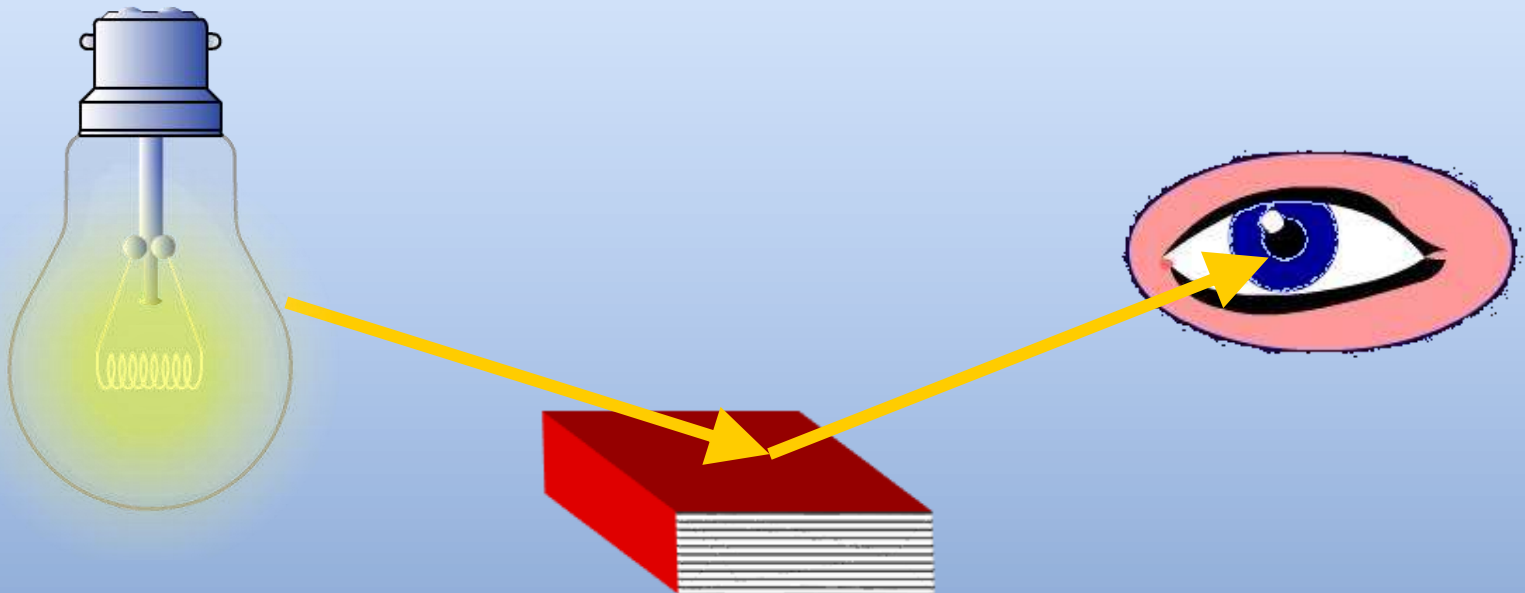


Effect of Light Source on color of object

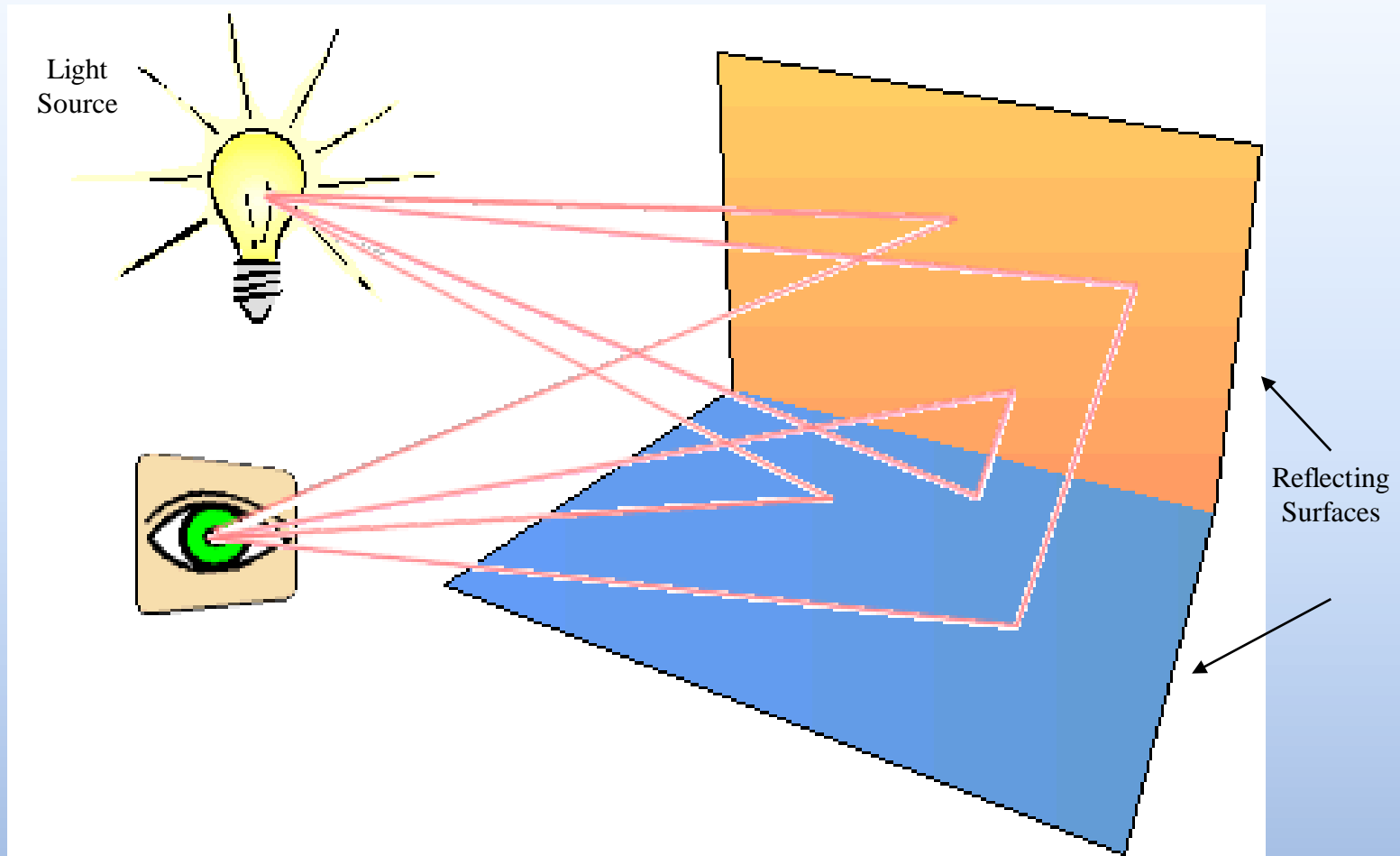
| ball | light | image |
|--------|-------|--|
| white | red |  |
| red | white |  |
| red | green |  |
| purple | blue |  |
| yellow | aqua |  |

Light Sources

- When we view an **opaque nonluminous object**, we see reflected light from the surfaces of the object.
- The **total reflected light** is the sum of the contributions from *light sources* and other reflecting surfaces in the scene.
- **Light sources** = *light-emitting sources*.
- **Reflecting surfaces** = *light-reflecting sources*.



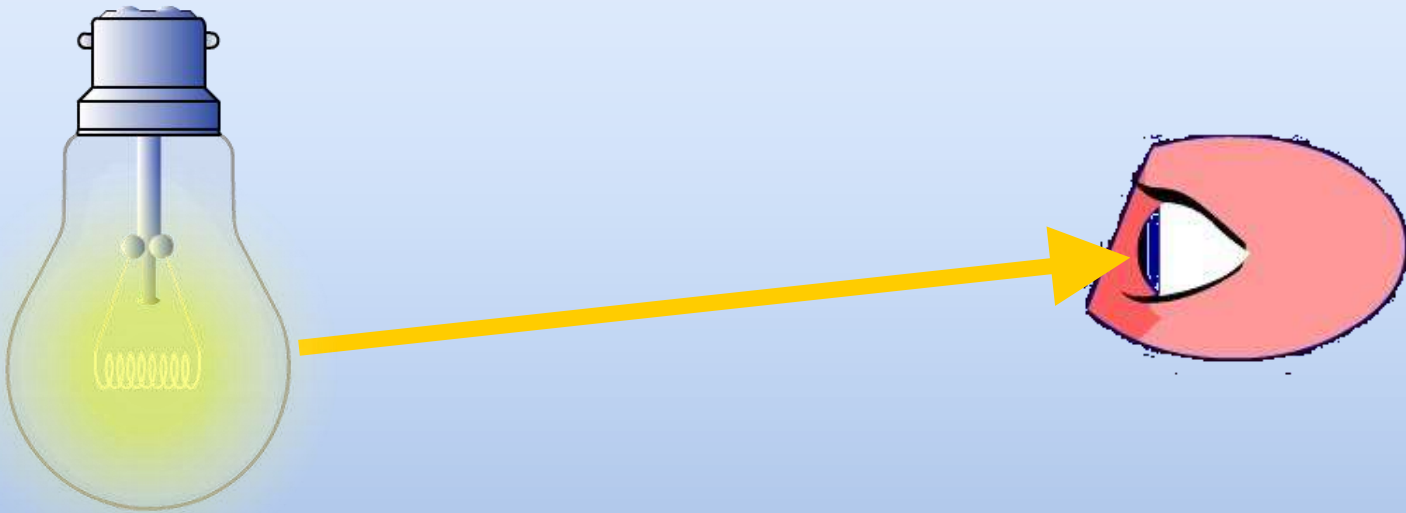
Light Sources



Light viewed from an opaque surface is in general a combination of reflected light from a light source and reflections of light reflections from other surfaces.

Light Sources

- A **luminous object**, can be both a **light source** and a **light reflector**.
- Example a **plastic globe with a light bulb inside**.



Point Light Source

- The simplest model for a light emitter is a point source.
- The rays emitted from a point light radially diverge from the source.
- This model is approximation for sources whose **dimensions are small** compared to the size of objects in the scene.
- A point light source is a fair approximation to a local light source such as a **light bulb**.
- The direction of the light to each point on a surface changes when a point light source is used.

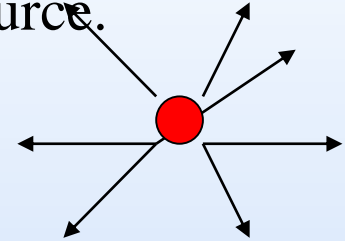


Fig. 2
Diverging ray paths from a point light source.

Distributed Light Source

- A nearby source, such as the **long fluorescent light**.
- **All of the rays** from a directional/distributed light source **have the same direction**, and **no point of origin**.
- It is as if the light source was infinitely far away from the surface that it is illuminating.
- **Sunlight** is an example of an **infinite light source**.

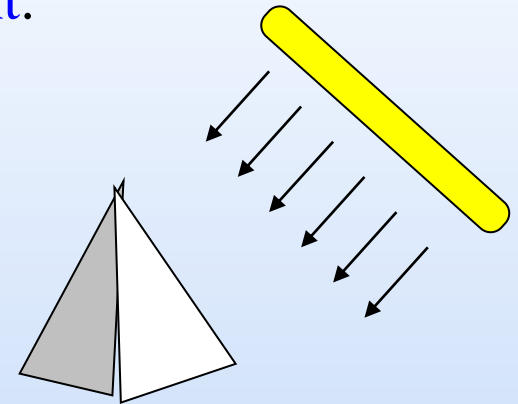


Fig. 3
An object illuminated with a distributed light source.

Materials

Opaque Surface:

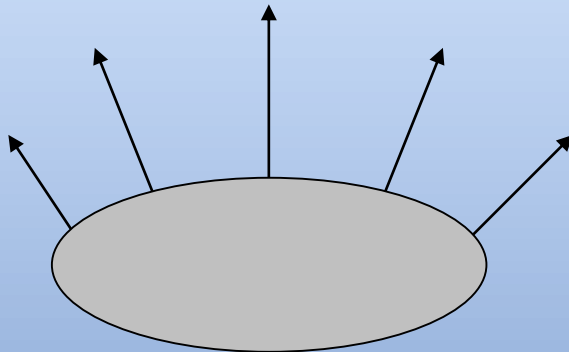
- When light is incident on an **opaque surface**, part of it is **reflected** and part is **absorbed**.
- **Shiny** materials **reflect more** of the incident light
- **Dull** surface **absorb more** of the incident light.

Transparent Surface:

- For an illuminated **transparent surface**, some of the incident light will **be reflected** and some will **be transmitted** through the material.

Diffuse reflection

- **Rough or Grainy** surfaces **scatter the reflected light** in all directions. This scattered light is called *diffuse reflection*.
- The surface appears equally bright from all viewing directions.
- In that case the color of an object is the color of the diffuse reflection of the incident light.
- A blue object illuminated by white light source, reflects the blue component of the white light and totally absorbs all other components.
- If a blue object is viewed under a red light, it appears black since all of the incident red light is absorbed nothing is reflected.

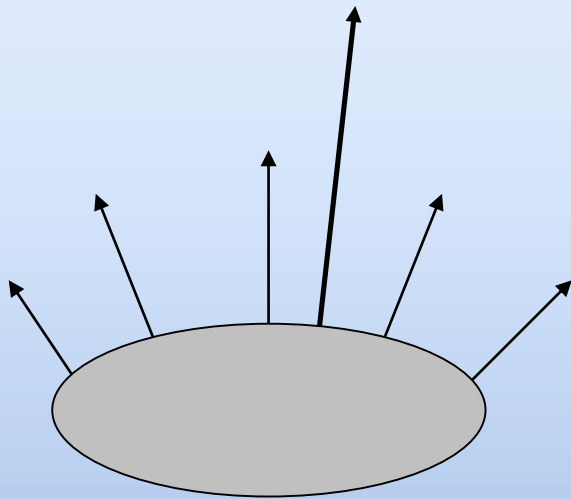


Diffuse reflection from a surface.

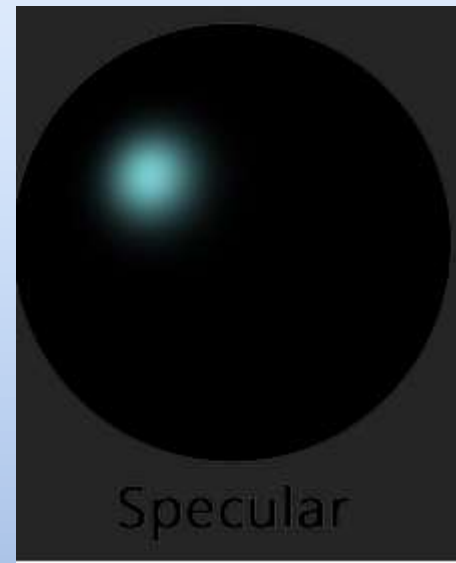


Specular reflection

- Light sources **create highlights**, bright spots, called *specular reflection*.
- More pronounced on **shiny surfaces** than on dull.



Specular reflection superimposed
on diffuse reflection vectors.



Illumination Models & Surface-Rendering Methods

- *Illumination model or a lighting model or shading model* is the model for calculating light intensity at a single surface point.
- *Surface rendering:* Uses the intensity calculations from an illumination model to determine the light intensity for all projected pixel positions for the various surfaces in the scene.

Illumination models

The parameters required for intensity calculations:

- The optical properties of surfaces (opaque/transparent, shiny/dull, surface-texture);
- The relative positions of the surfaces in a scene;
- The color and positions of the light sources;
- The position and orientation of the viewing plane.

Illumination models calculate the intensity projected from a particular surface point in a specified viewing direction.

Basic Illumination Models

- Simplified methods for calculating light intensities.
- Lighting calculations are based on:
 - ❖ Optical properties of surfaces, such as glossy, matte, opaque, and transparent. This controls the amount of reflection and absorption of incident light.
 - ❖ The background lighting conditions.
 - ❖ The light-source specifications.
 - ❖ All light sources are considered to be point sources, specified with a coordinate position and intensity value (color).

Basic Illumination Models

- Ambient Light
- Diffuse Reflection
- Specular Reflection and the Phong Model
- Warn Model

Ambient Light

- A surface that is not exposed directly to a light source still will be visible if nearby objects are illuminated.
- This is a simple way to model the combination of light reflections from various surfaces to produce a uniform illumination called the ambient light, or background light.
- Ambient light has no spatial or directional characteristics.
- The amount of ambient light incident on each object is a constant for all surfaces and over all directions.
- The amount of ambient light that is reflected by an object is constant and independent of the object's position or orientation and depends only on the optical properties of the surface.

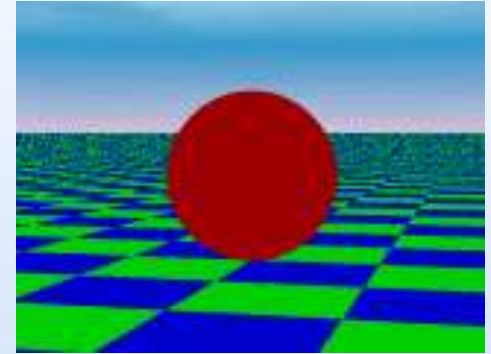


Fig. 6
Ambient light shading.

Ambient Light

- The level of ambient light in a scene is a parameter I_a , and each surface illuminated with this constant value.
- Illumination equation for ambient light is

$$I = k_a I_a$$

where

I is the resulting intensity

I_a is the incident ambient light intensity

k_a is the object's basic intensity (surface reflectivity), *ambient-reflection coefficient*.

Ambient Light - Example



An ambient illumination only.

Diffuse Reflection

- Diffuse reflections are constant over each surface in a scene, independent of the viewing direction.
- The amount of the incident light that is diffusely reflected can be set for each surface with parameter k_d , the *diffuse-reflection coefficient*, or *diffuse reflectivity*.

$$0 \leq k_d \leq 1;$$

k_d near 1 – highly reflective surface;

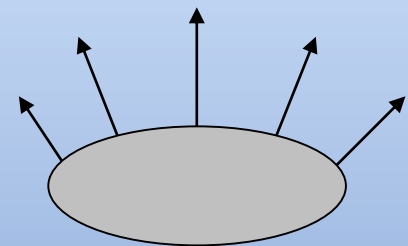
k_d near 0 – surface that absorbs most of the incident light;

k_d is a function of surface color; assume constant.

- If a surface is exposed only to ambient light, we can express the intensity of the diffuse reflection at any point on the surface as

$$I_{ambdiff} = k_d I_a$$

I_a is the incident ambient light intensity



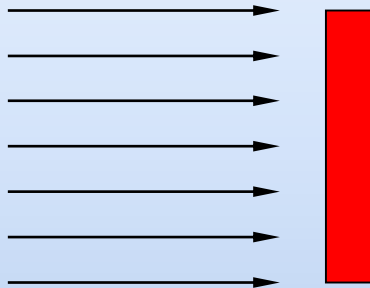
Diffuse reflection
from a surface.

Diffuse Reflection

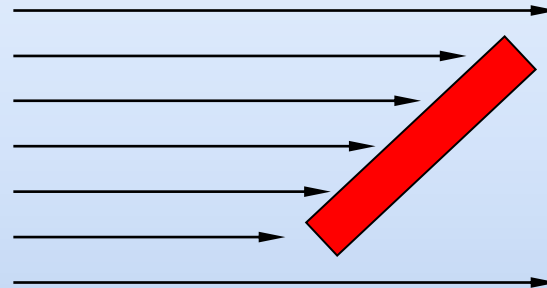
- *Ideal diffuse reflectors or Lambertian reflectors:*
 - Surfaces from which diffuse reflections are scattered with equal intensity in all directions, irrespective of viewing directions.
- The radiant light energy from any point on such surfaces is governed by Lamberts cosine law.
- *Lamberts cosine law:* the radiant energy from any small surface area dA in any direction θ_N relative to the surface normal is proportional to $\cos \theta_N$
- Thus for Lambertian reflection, the intensity of light is the same over all viewing directions.

Diffuse Reflection

- Even though there is equal light scattering in all direction from a surface, the brightness of the surface does depend on the orientation of the surface relative to the light source:



(a)

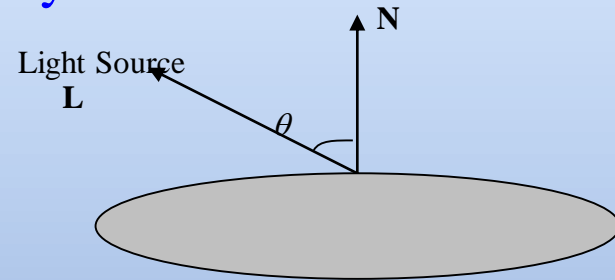


(b)

A surface perpendicular to the direction of the incident light (a) is more illuminated than an equal-sized surface at an oblique angle (b) to the incoming light direction.

Diffuse Reflection

- As the angle between the surface normal and the incoming light direction increases, less of the incident light falls on the surface.
- We denote the *angle of incidence* between the incoming light direction and the surface normal as θ . Thus, the amount of illumination depends on $\cos \theta$.
- If the incoming light from the source is perpendicular to the surface at a particular point, that point is fully illuminated.
- As the *angle of illumination moves away* from the surface normal, the *brightness of the surface drops off*.



Angle of incidence θ between the unit light-source direction vector \mathbf{L} and the unit surface normal \mathbf{N} .

Diffuse Reflection

If I_l is the intensity of the point light source, then the diffuse reflection equation for a point on the surface can be written as

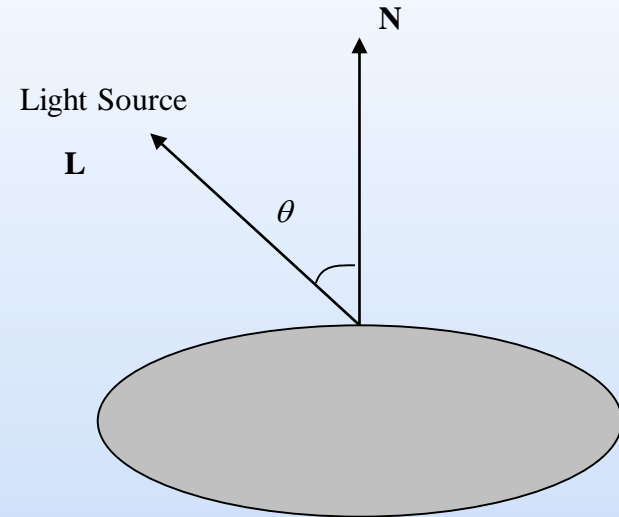
$$I_{l,diff} = k_d I_l \cos \theta$$

or

$$I_{l,diff} = k_d I_l (\mathbf{N} \cdot \mathbf{L})$$

where

\mathbf{N} is the unit normal vector to a surface and \mathbf{L} is the unit direction vector to the point light source from a position on the surface.



Angle of incidence θ between the unit light-source direction vector \mathbf{L} and the unit surface normal \mathbf{N} .

Diffuse Reflection

Following Figure illustrates the illumination with diffuse reflection, using various values of parameter k_d between 0 and 1.



Fig. Series of pictures of sphere illuminated by diffuse reflection model only using different k_d values (0.4, 0.55, 0.7, 0.85, 1.0).

Diffuse Reflection

- We can combine the ambient and point-source intensity calculations to obtain an expression for the total diffuse reflection.

$$I_{diff} = k_a I_a + k_d I_l (\mathbf{N} \cdot \mathbf{L})$$

where both k_a and k_d depend on surface material properties and are assigned values in the range from 0 to 1.

k_a is the object's basic intensity (surface reflectivity), *ambient-reflection coefficient*.



Series of pictures of sphere illuminated by ambient and diffuse reflection model.

$I_a = I_l = 1.0$, $k_d = 0.4$ and k_a values (0.0, 0.15, 0.30, 0.45, 0.60).

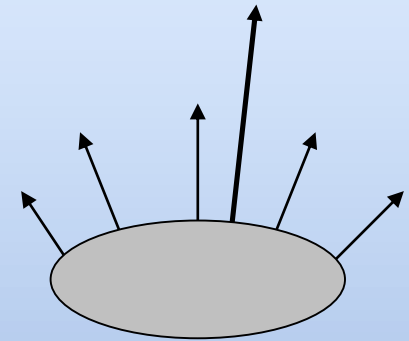
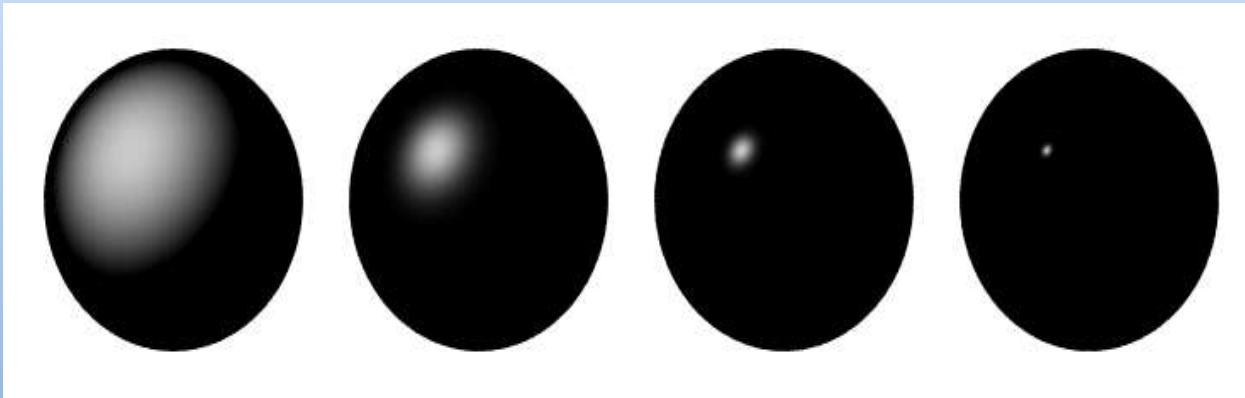
Diffuse Reflection - Example



Individually shaded polygons with diffuse reflection.

Specular Reflection and the Phong Model

- *Specular reflection* is the result of total, or near total, reflection of the incident light in a concentrated region around the *specular-reflection angle*.
- Shiny surfaces have a narrow specular-reflection range.
- Dull surfaces have a wider reflection range.

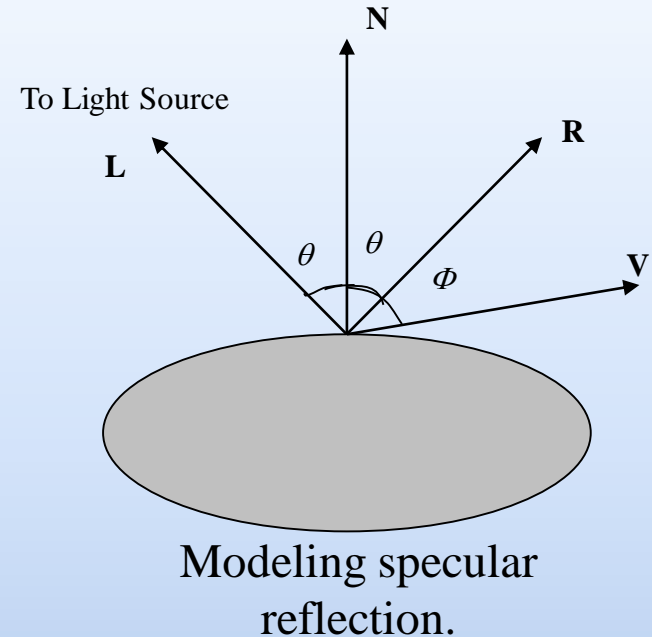


Specular reflection
superimposed on
diffuse reflection
vectors.

Specular Reflection

Figure shows the specular reflection direction at a point on the illuminated surface. In this figure,

- **R** - represents the unit vector in the direction of specular reflection;
- **L** – unit vector directed toward the point light source;
- **V** – unit vector pointing to the viewer from the surface position;
- Angle Φ is the viewing angle relative to the specular-reflection direction **R**.

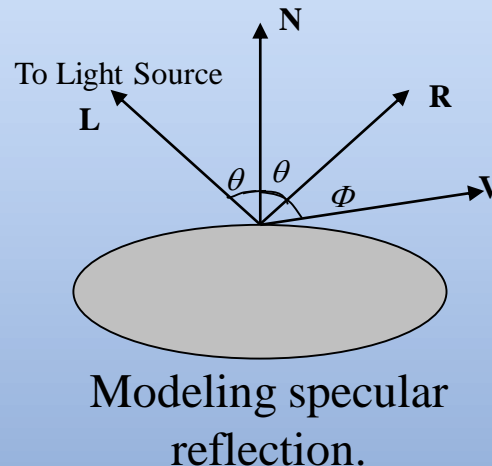


Ideal reflectors (perfect mirror): incident light is reflected only in specular reflection direction. ($\Phi=0$)

Phong Model

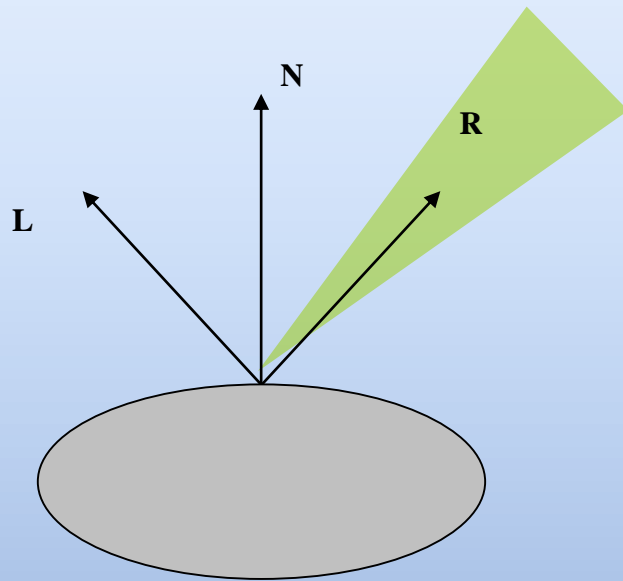
Phong model is an empirical model for calculating the specular-reflection range:

- Sets the intensity of specular reflection proportional to $\cos^{n_s} \Phi$;
- Angle Φ assigned values in the range 0° to 90° , so that $\cos \Phi$ values from 0 to 1;
- *Specular-reflection parameter* n_s is determined by the type of surface,
- *Specular-reflection coefficient* k_s equal to some value in the range 0 to 1 for each surface.

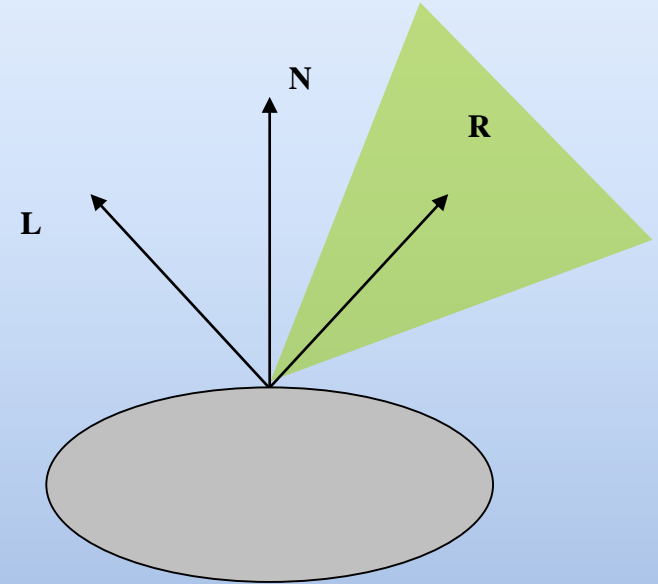


Phong Model

- Very shiny surface is modeled with a large value for n_s (say, 100 or more);
- Small values (near 1) are used for dull surfaces.
- For perfect reflector (perfect mirror), n_s is infinite;



Shiny Surface (Large n_s)



Dull Surface (Small n_s)

Modeling specular reflection with parameter n_s .

Phong Model

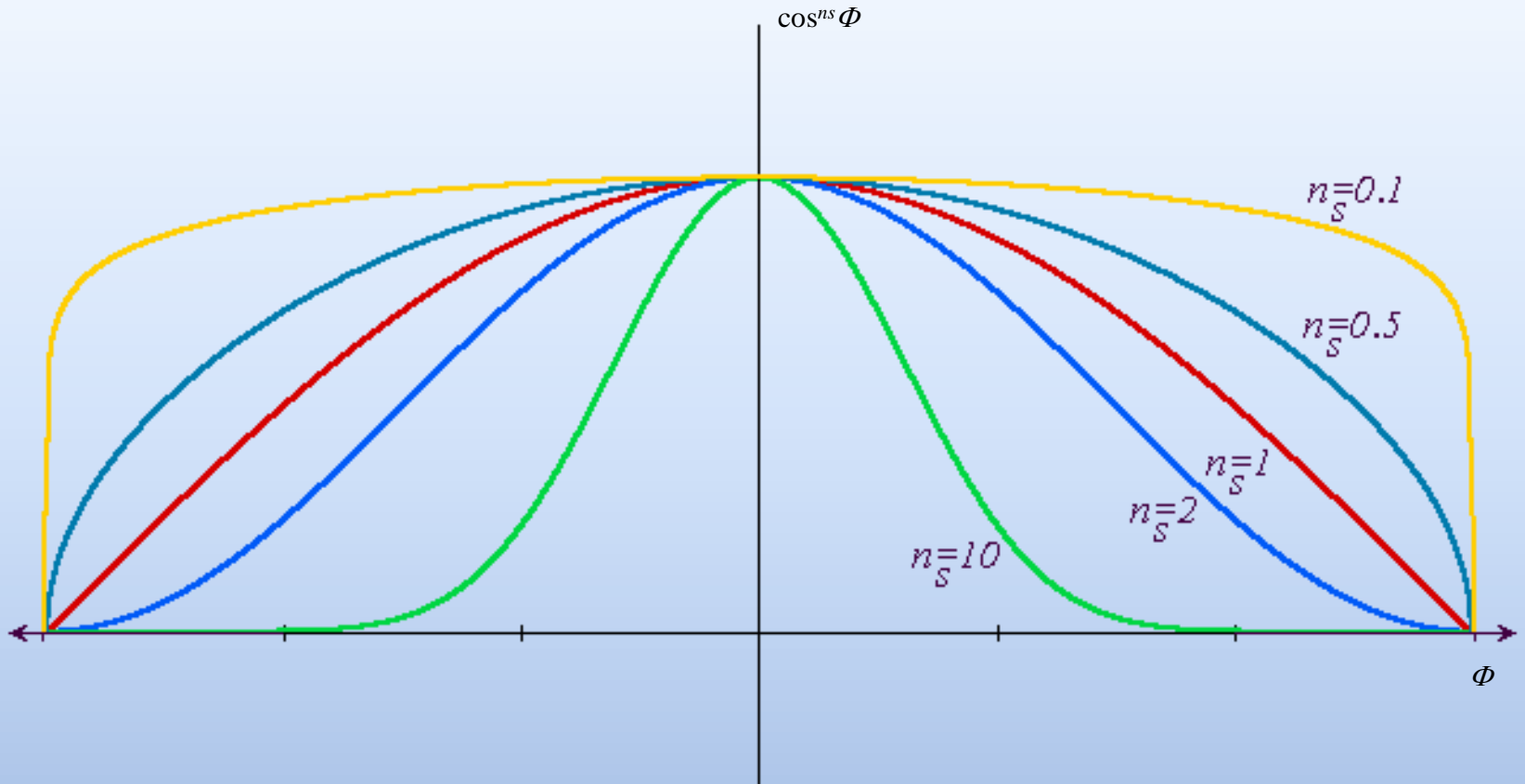


Fig. 15

Plots of $\cos^{n_s} \phi$ for several values of specular parameter n_s .

Phong Model

Phong specular-reflection model:

$$I_{spec} = k_s I_l \cos^{ns} \Phi$$

Since **V** and **R** are unit vectors in the viewing and specular-reflection directions, we can calculate the value of $\cos^{ns} \Phi$ with the dot product **V**·**R**.

$$I_{spec} = k_s I_l (\mathbf{V} \cdot \mathbf{R})^{ns}$$

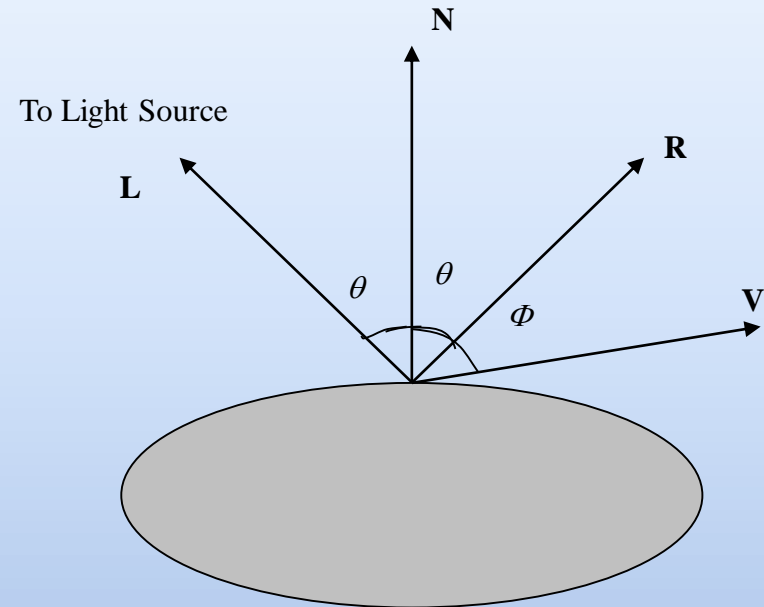


Fig. 13
Modeling specular reflection.

Specular Reflection - Example



Fig. 18

Phong shading polygons with specular reflection.

Combine Diffuse & Specular Reflections

For a single point light source, we can model the combined diffuse and specular reflections from a point on an illuminated surface as

$$I = I_{diff} + I_{spec}$$

$$= k_a I_a + k_d I_l (\mathbf{N} \cdot \mathbf{L}) + k_s I_l (\mathbf{N} \cdot \mathbf{H})^{ns}$$

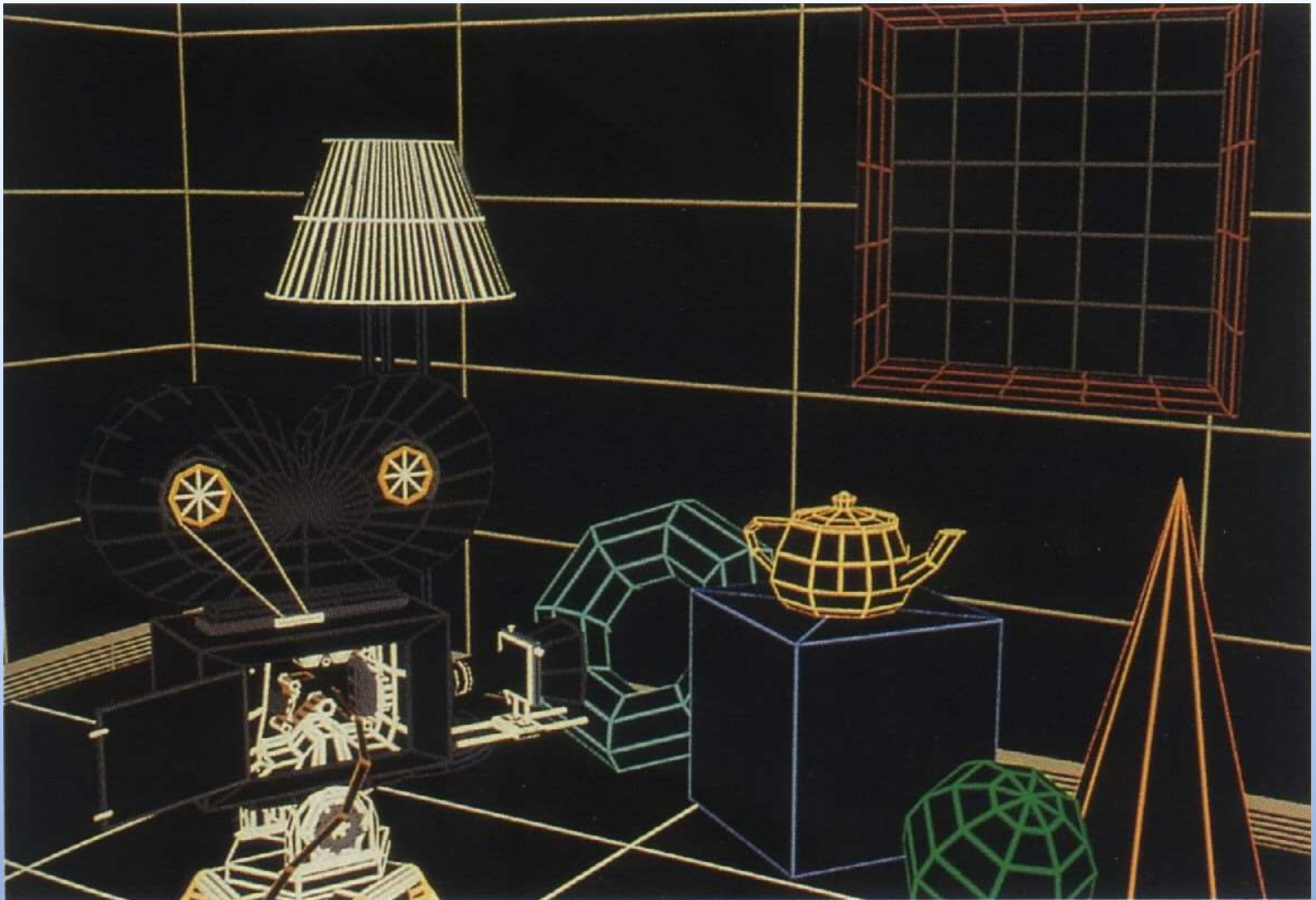
H – halfway vector

Combine Diffuse & Specular Reflections with Multiple Light Sources

If we place **more than one point source in a scene**, we obtain the light reflection at any surface point by summing the contributions from the individual sources:

$$I = k_a I_a + \sum_{i=1}^n I_{li} [k_d (\mathbf{N} \cdot \mathbf{L}_i) + k_s (\mathbf{N} \cdot \mathbf{H}_i)^{ns}]$$

Visible-line determination



Visible-surface determination with ambient illumination only



Individually shaded polygons with diffuse reflection



Gouraud shaded polygons with diffuse reflection



Gouraud shaded polygons with specular reflection



Phong shaded polygons with specular reflection



Curved surfaces with specular reflection



Warn Model

- Provides method for **simulating studio lighting effects** by controlling light intensity in different directions
- Light sources are modeled as points on a reflecting surface, using the Phong model for the surface points.
- Then the intensity in different directions is controlled by selecting values for the Phong exponent.
- In addition, light controls, such as **"barn doors"** and **spotlighting**, used by studio photographers can be simulated in the Warn model.
- **Flaps** are used to control the amount of light emitted by a source in various directions. Two flaps are provided for each of the **x, y, and z directions**.
- ***Spotlights*** are used to control the amount of light emitted within a cone with apex at a point-source position.

Light controls



Barn doors



Spot Lighting



Reflective Umbrellas

Studio lighting effects



STUDIO LIGHTS | How high should they be?

Choose where you want the shadows to fall



01 High

In most cases you'll want to have your main light positioned above the model. Notice how the shadow from the nose falls down the face, elongating the features. Ideally you want the shadow of the nose to point towards the end of the lips. The triangle of light on the cheek on the shadow side is often referred to as 'Rembrandt' lighting; get your model to move their head slightly to achieve this.



02 Eye level

With the flashlight to the side and at the same height as the model the light falls across the face, causing a shadow that widens the facial features. If this light is balanced with one of equal strength on the other side it can be quite effective, but as a sculpting technique height would be better. Keep your flashlight's modelling lights switched on so you can see how the shadows will lie.



03 Low

There are unlikely to be too many situations when a low light is going to work well as your primary light source. It gives a spooky look, so Halloween is probably the only time you're even going to think about using this technique. As you can see from our example, underlighting is not very flattering even on a young model. With underlighting the nose shadow is clumpy and any bags under the eyes will be amplified.



Studio lighting effects



Surface rendering

- Surface rendering can be performed by applying the illumination model to every visible surface point or by interpolating (inserting) intensities across the surface from a small set of illumination-model calculations.
- Scan-line algorithms use interpolation schemes.
- Ray tracing algorithms invoke the illumination model at each pixel position.
- Surface-rendering procedures are also termed as *surface-shading methods*.

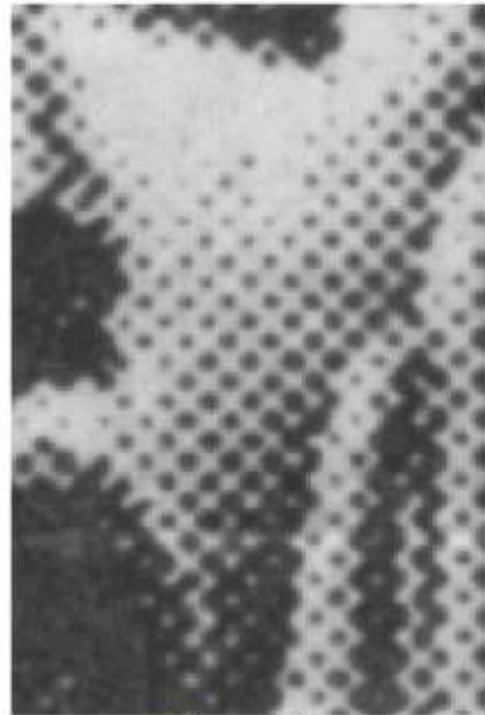
Halftoning

- When an output device has a limited intensity range, we can create an apparent **increase in the number of available intensities** by incorporating multiple pixel positions into the display of each intensity value
- Many image rendering technologies only have binary output. For example, printers can either “fire a dot” or not.
- Halftoning is a method for **creating the illusion of continuous tone** output with a binary device thus improving the quality of rendered images at minimal cost.
- Continuous-tone photographs are reproduced for publication in **newspapers, magazines, and books** with a printing process called halftoning, and the reproduced pictures are called halftones.

Continuous Half Toning



Original

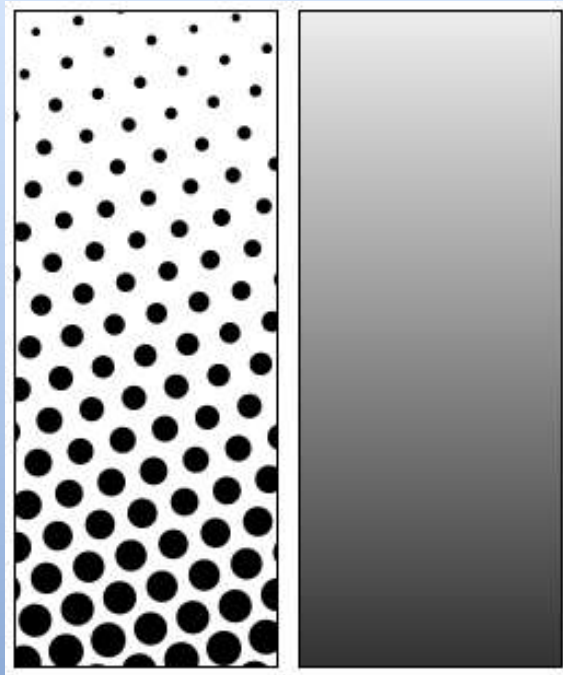


Zoom-in

Halftoning

- For a black-and-white photograph, each intensity area is reproduced as a series of black circles on a white background.
- The diameter of each circle is proportional to the darkness required for that intensity region.
- Darker regions are printed with larger circles, and lighter regions are printed with smaller circles (more white area).

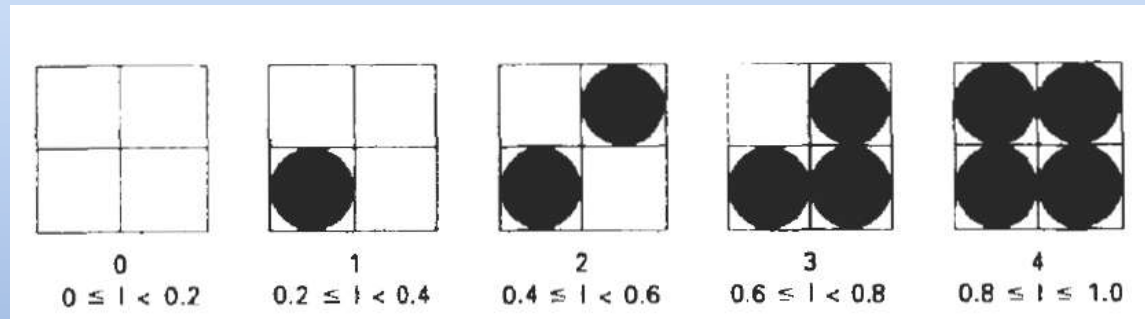
Halftone dots →



← How the human eye would see this sort of arrangement from a sufficient distance.

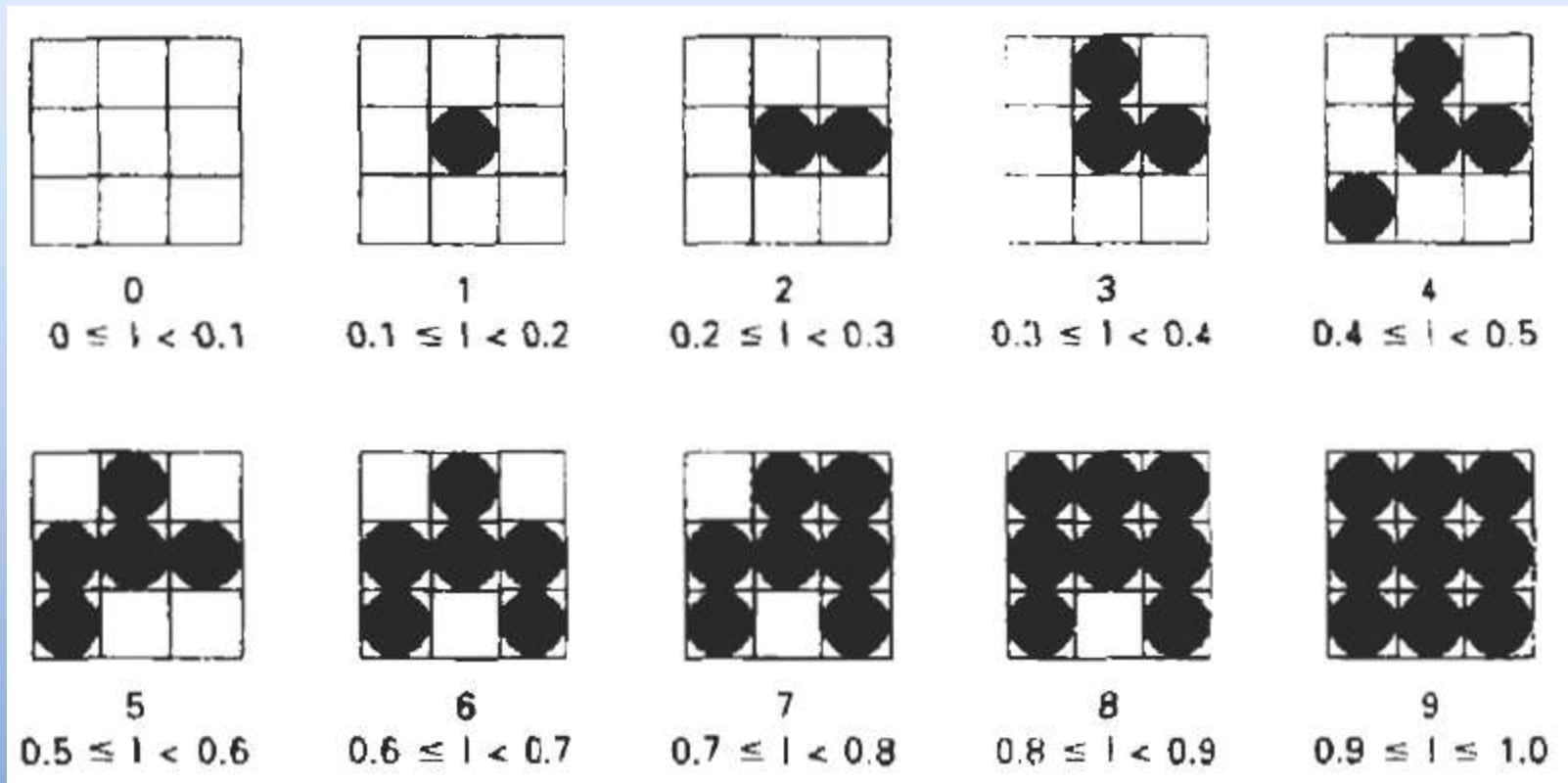
Halftone Approximations

- Halftone reproductions are approximated using rectangular pixel regions, called **halftone patterns or pixel patterns**.
- The number of intensity levels that we can display with this method depends on how many pixels we include in the rectangular grids and how many levels a system can display.
- **With n by n pixels for each grid on a bilevel system, we can represent $n^2 + 1$ intensity levels.**



Halftone Approximations

- Pixel positions are chosen at each level so that the patterns approximate the increasing circle sizes used in halftone reproductions. That is, the "on" pixel positions are near the center of the grid for lower intensity levels and expand outward as the intensity level increases

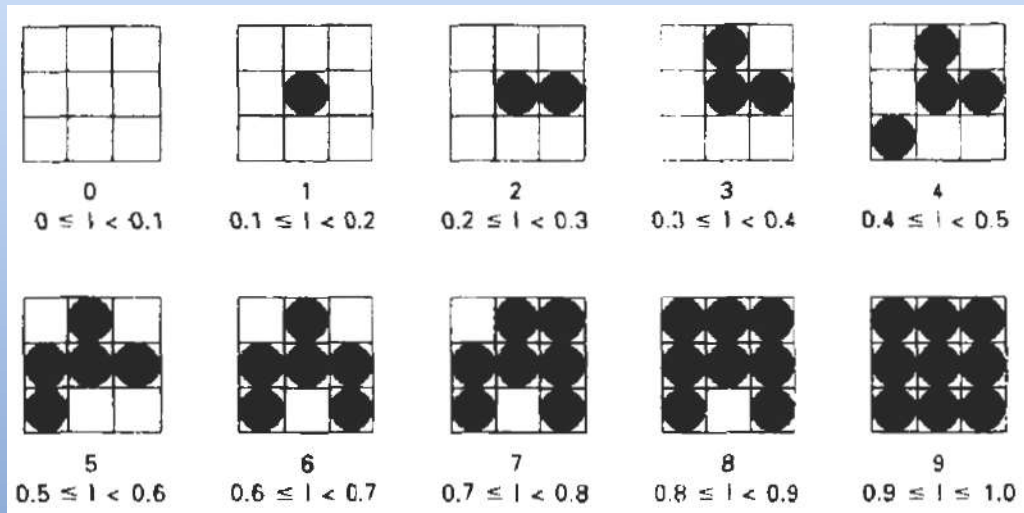


Halftone Approximations

- For any pixel-grid size, we can represent the pixel patterns for the various possible intensities with a "mask of pixel position numbers."
- Mask to generate nine 3 by 3 grid patterns

| | | |
|---|---|---|
| 8 | 3 | 7 |
| 5 | 1 | 2 |
| 4 | 9 | 6 |

- To display a particular intensity with level number k , we turn on each pixel whose position number is less than or equal to k .

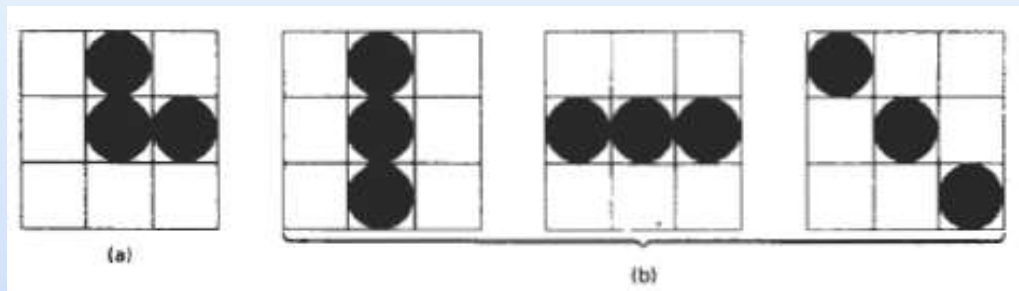


Halftone Approximations

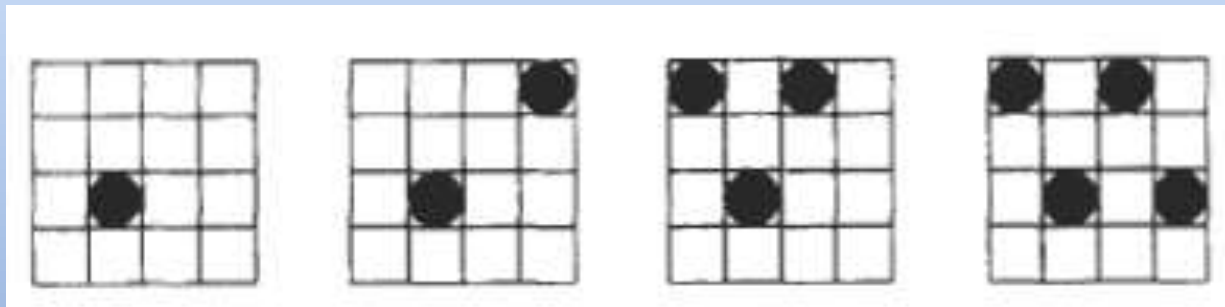
- Increases the number of intensities that can be displayed, but reduce the resolution of the displayed picture by a factor of $1/n$ along each of the *x and y axes*.
- A 512 by 512 screen area, for instance, is reduced to an area containing 256 by 256 intensity points with 2 by 2 grid patterns. And with 3 by 3 patterns, we would reduce the 512 by 512 area to 128 intensity positions along each side.

Halftone Approximations

Avoid symmetrical patterns: would produce vertical, horizontal, or diagonal streaks



Avoid a single "on" pixel or isolated "on" pixels:



Halftone Approximations

- Can also increase the number of intensity options on systems that are capable of displaying more than two intensities per pixel.
- On a system that can display four intensity levels per pixel, we can *use 2 by 2 pixel grids to extend the available intensity levels from 4 to 13*

| | |
|---|---|
| 0 | 0 |
| 0 | 0 |

0

| | |
|---|---|
| 0 | 1 |
| 0 | 0 |

1

| | |
|---|---|
| 0 | 1 |
| 1 | 0 |

2

| | |
|---|---|
| 1 | 1 |
| 1 | 0 |

3

| | |
|---|---|
| 1 | 1 |
| 1 | 1 |

4

| | |
|---|---|
| 1 | 2 |
| 1 | 1 |

5

| | |
|---|---|
| 1 | 2 |
| 2 | 1 |

6

| | |
|---|---|
| 2 | 2 |
| 2 | 1 |

7

| | |
|---|---|
| 2 | 2 |
| 2 | 2 |

8

| | |
|---|---|
| 2 | 3 |
| 2 | 2 |

9

| | |
|---|---|
| 2 | 3 |
| 3 | 2 |

10

| | |
|---|---|
| 3 | 3 |
| 3 | 2 |

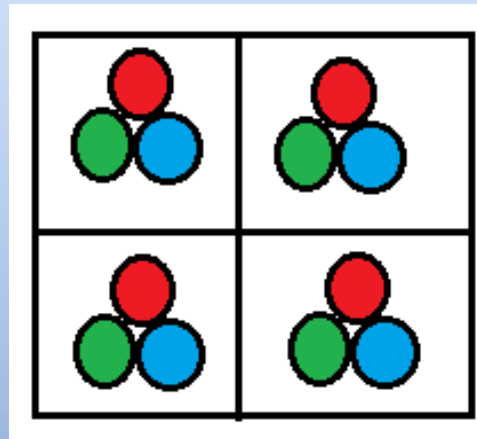
11

| | |
|---|---|
| 3 | 3 |
| 3 | 3 |

12

Halftone Approximations

- Similarly, we can use pixel-grid patterns to increase the number of intensities that can be displayed on a [color system](#).
- As an example, suppose we have a three-bit per pixel RGB system. Using 2 by 2 pixel-grid patterns, we now have 12 phosphor dots that can be used to represent a particular color value.
- Each of the three RGB colors has four phosphor dots in the pattern, which allows five possible settings per color. This gives us a total of 125 different color combinations.



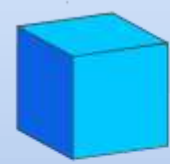
POLYGON-RENDERING METHODS

- Given a free form surface, one usually approximates the surface as a polyhedral (3D shapes).
- How do we calculate in practice the illumination at each point on the surface?
- Applying the illumination model at each surface point is computationally expensive
- Three methods:
 - **Constant-Intensity (flat) Shading**
 - **Gouraud Shading**
 - **Phong Shading**

Regular Polyhedrons



Tetrahedron



Cube



Octahedron



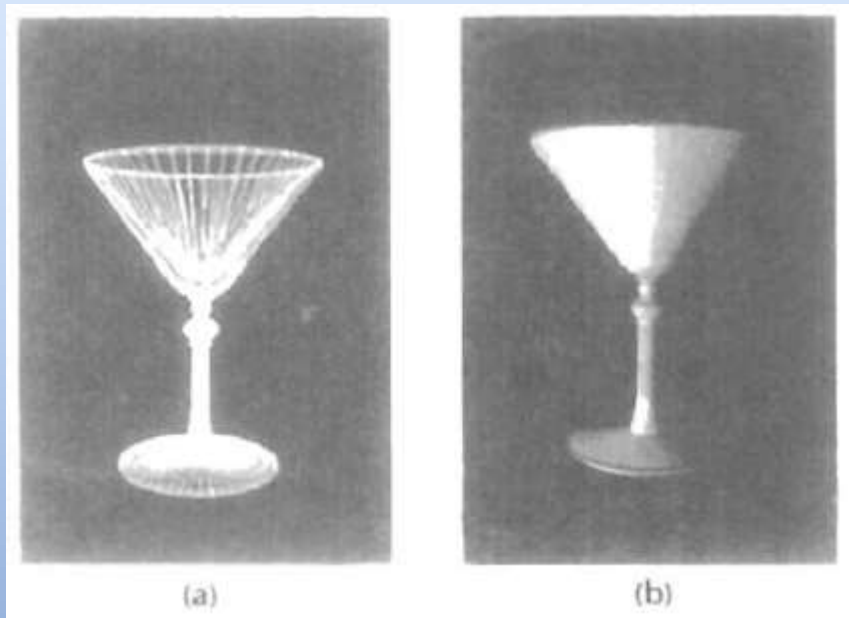
Dodecahedron



Icosahedron

Constant-Intensity Shading

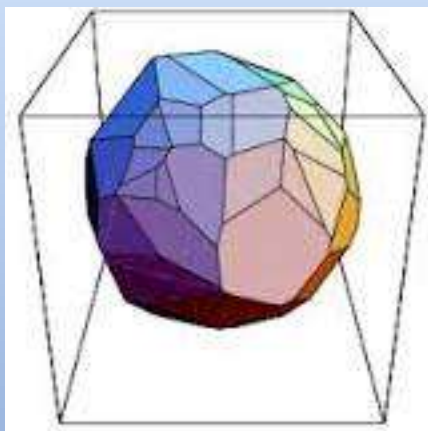
- A fast and simple method for rendering an object with polygon surfaces also called **flat shading**.
- A single intensity is calculated for each polygon. All points over the surface of the polygon are then displayed with the same intensity value.
- Constant shading can be useful for quickly displaying the general appearance of a curved surface



Constant-Intensity Shading

Assumptions (Requirements):

- The object is a polyhedron and is not an approximation of an object with a curved surface.
- All light sources illuminating the object are sufficiently far from the surface so that $N \cdot L$ and the attenuation function are constant over the surface.
- The viewing position is sufficiently far from the surface so that $V \cdot R$ is constant over the surface.

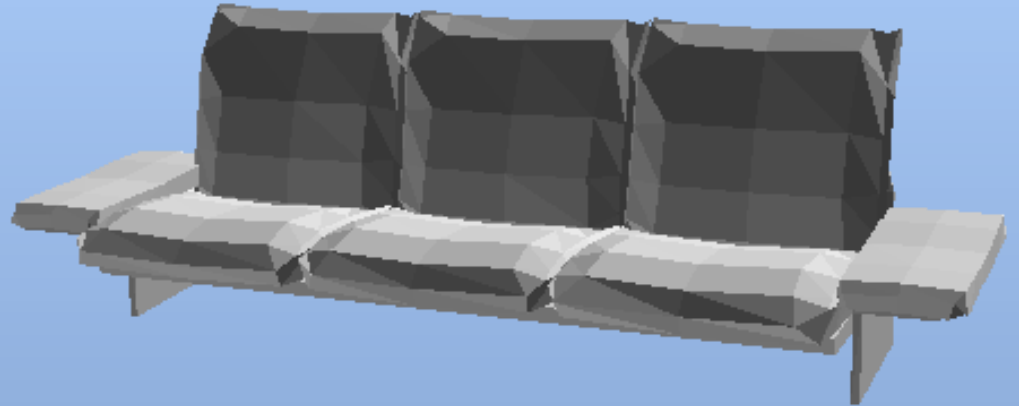
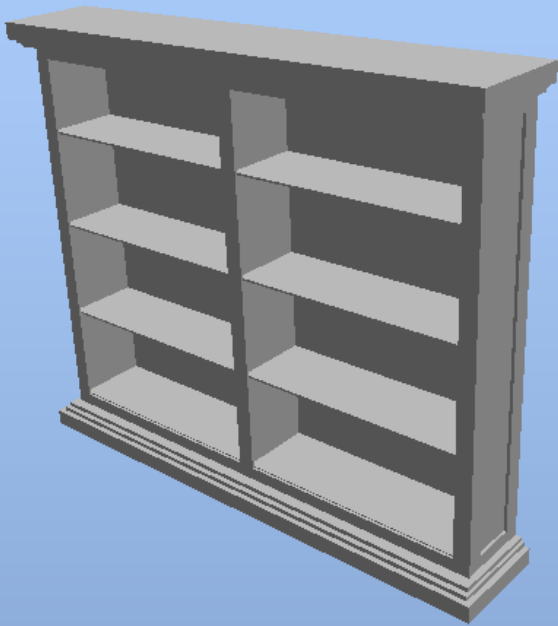


Flat Shading

Objects look like they are composed of polygons

OK for polyhedral objects

Not so good for ones with smooth surfaces



Gouraud Shading

- This intensity-interpolation scheme, renders a polygon surface by linearly interpolating intensity values across the surface.
- Intensity values for each polygon are matched with the values of adjacent polygons along the common edges, thus eliminating the intensity discontinuities that can occur in flat shading

Each polygon surface is rendered with Gouraud shading by performing the following **calculations (Steps)**:

- Determine the average unit normal vector at each polygon vertex.
- Apply an illumination model to each vertex to calculate the vertex intensity.
- Linearly interpolate the vertex intensities over the surface of the polygon.

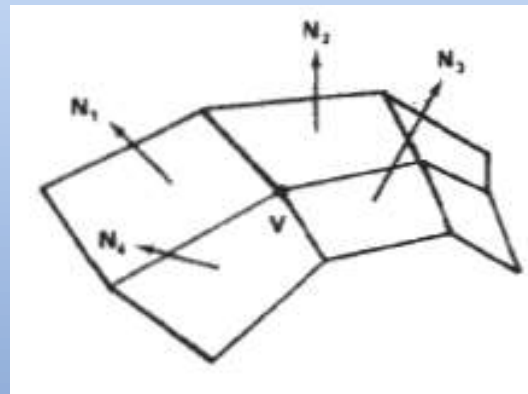


Gouraud Shading

- At each polygon vertex, we obtain a normal vector by averaging the surface normals of all polygons sharing that vertex, as illustrated in Fig.
- Thus, for any vertex position V , we obtain the unit vertex normal with the calculation

$$\mathbf{N}_V = \frac{\sum_{k=1}^n \mathbf{N}_k}{\left| \sum_{k=1}^n \mathbf{N}_k \right|}$$

Once we have the vertex normals, we can determine the intensity at the vertices from a lighting model.



Gouraud Shading



(a)



(b)



(c)

Figure 14-47

A polygon mesh approximation of an object (a) is rendered with flat shading (b) and with Gouraud shading (c).

Gouraud Shading

Advantages:

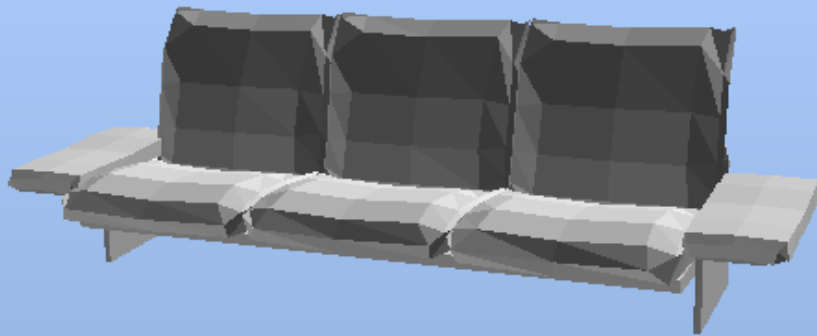
- Gouraud shading removes the intensity discontinuities associated with the constant-shading model

Disadvantages:

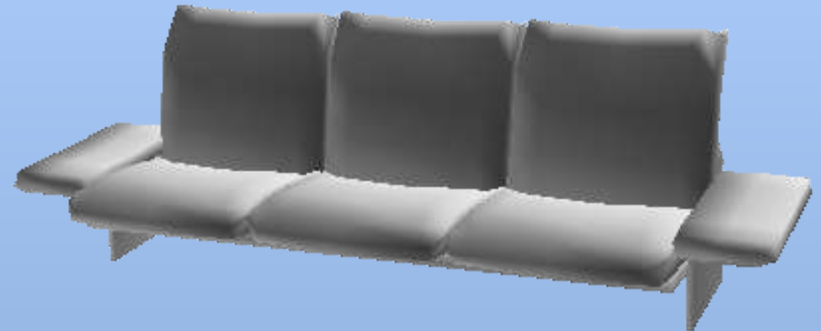
- Highlights on the surface are sometimes displayed with anomalous shapes, and the linear intensity interpolation can cause bright or dark intensity streaks, called **Mach bands**, to appear on the surface

Gouraud Shading

- Produces smoothly shaded polygonal mesh
 - Piecewise linear approximation
 - Need fine mesh to capture subtle lighting effects



Flat Shading



Gouraud Shading

Phong Shading

- Normal vector interpolation shading.
- It displays more realistic highlights on a surface and greatly reduces the Mach-band effect.
- A polygon surface is rendered using Phong shading by carrying out the following steps:
 - Determine the average unit normal vector at each polygon vertex.
 - Linearly interpolate the vertex normals over the surface of the polygon.
 - Apply an illumination model along each scan line to calculate projected pixel intensities for the surface points.

Phong Shading

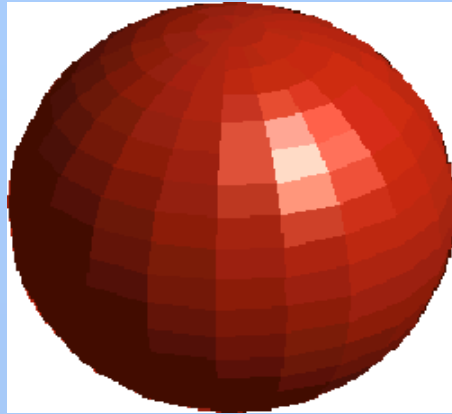
- Incremental methods are used to evaluate normals between scan lines and along each individual scan line.
- At each pixel position along a scan line, the illumination model is applied to determine the surface intensity at that point.

Advantage:

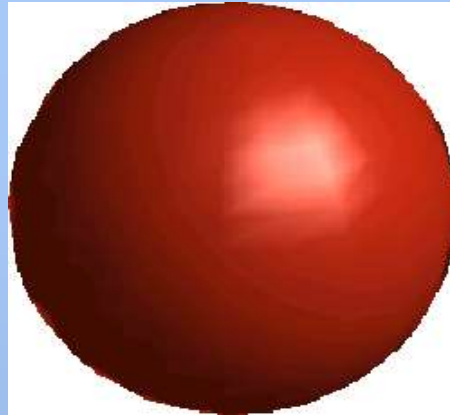
- Produce more accurate results than the direct interpolation of intensities, as in Gouraud shading.

Disadvantage:

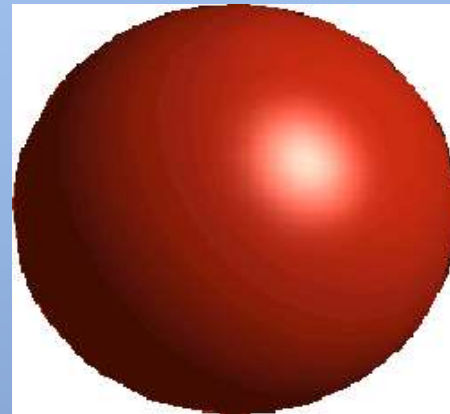
- The trade-off, however, is that Phong shading requires considerably more calculations.



Flat



Gouraud

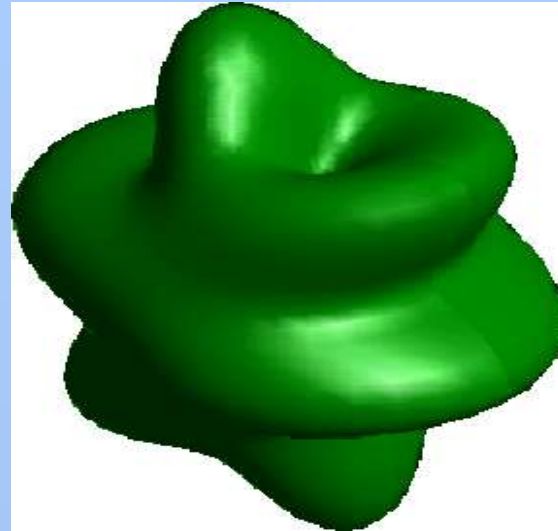


Phong

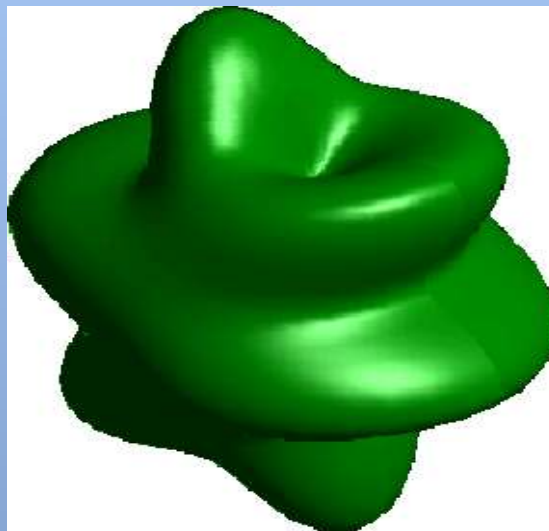
Flat Shading



Gouraud Shading

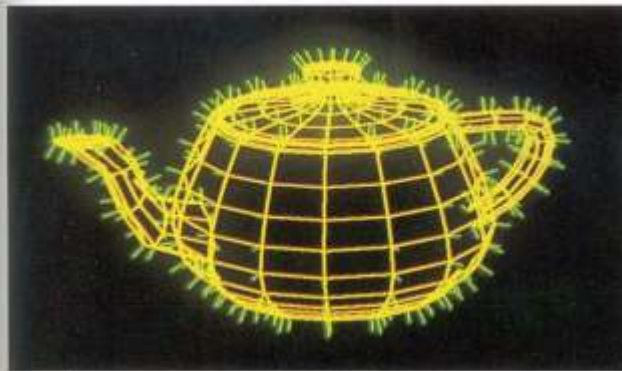


Phong Shading



Polygon Shading Algorithms

Wireframe



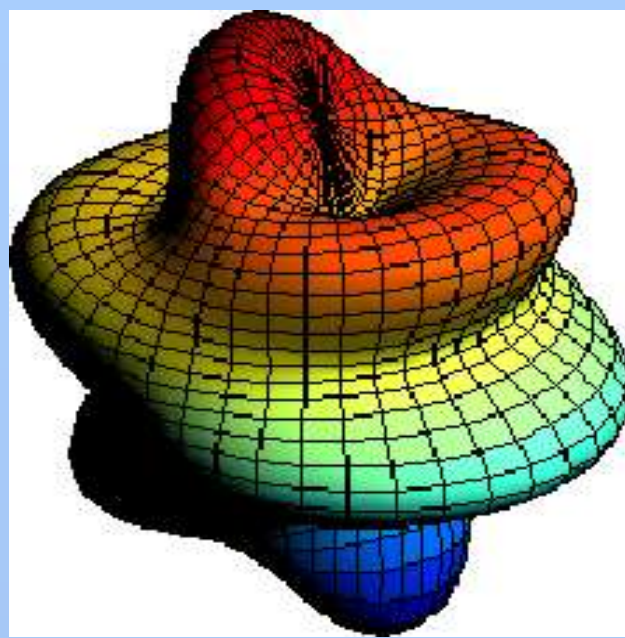
Flat



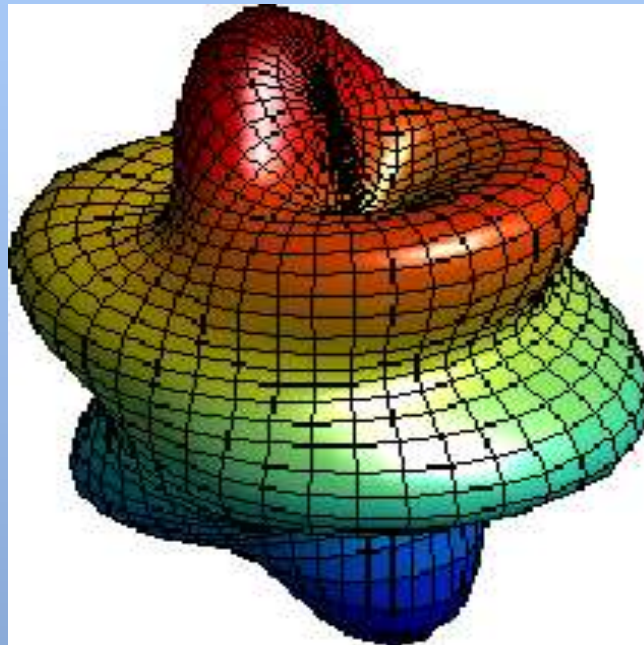
Gouraud



Phong



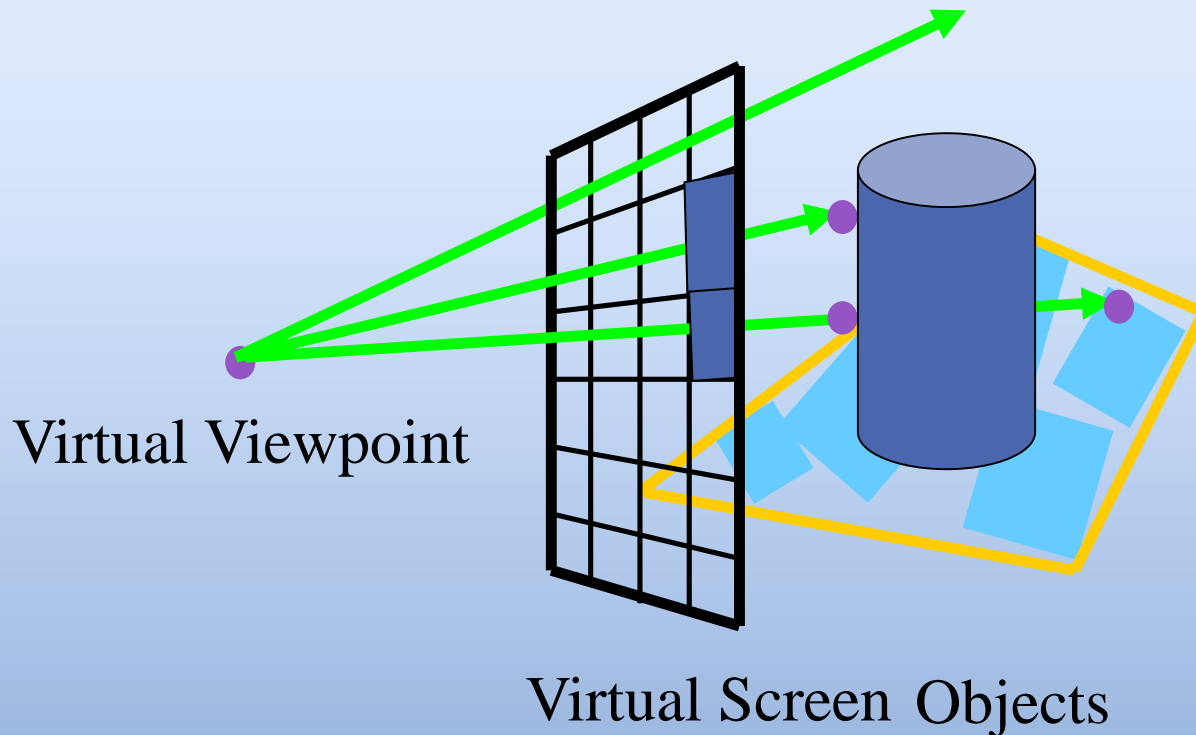
Diffuse
surface



With additional
specular
component

RAY-TRACING METHODS

Ray casting: *a ray is sent out from each pixel position to locate surface intersections for object, a method for determining visible surfaces in a scene.*



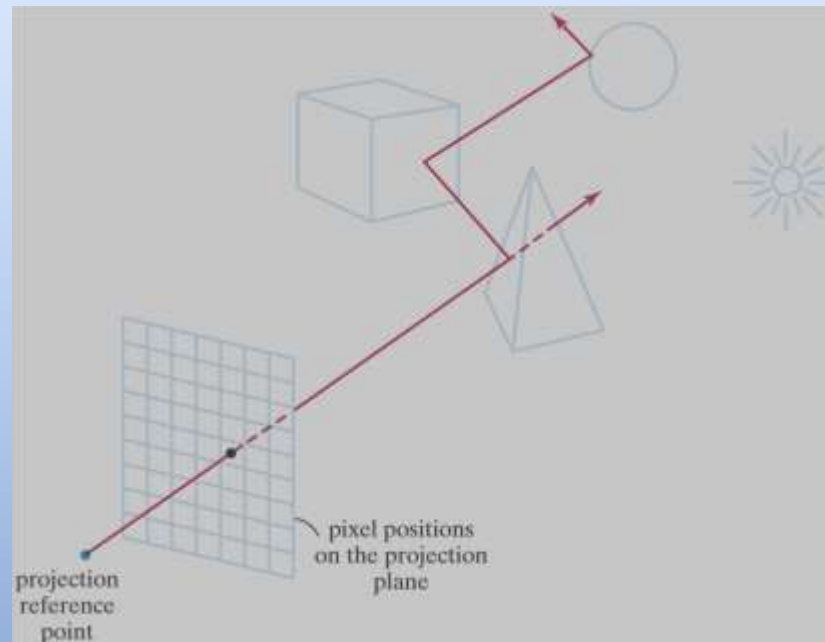
Ray intersects
object: shade
using color, lights,
materials

Multiple
intersections: Use
closest one (as does
OpenGL)

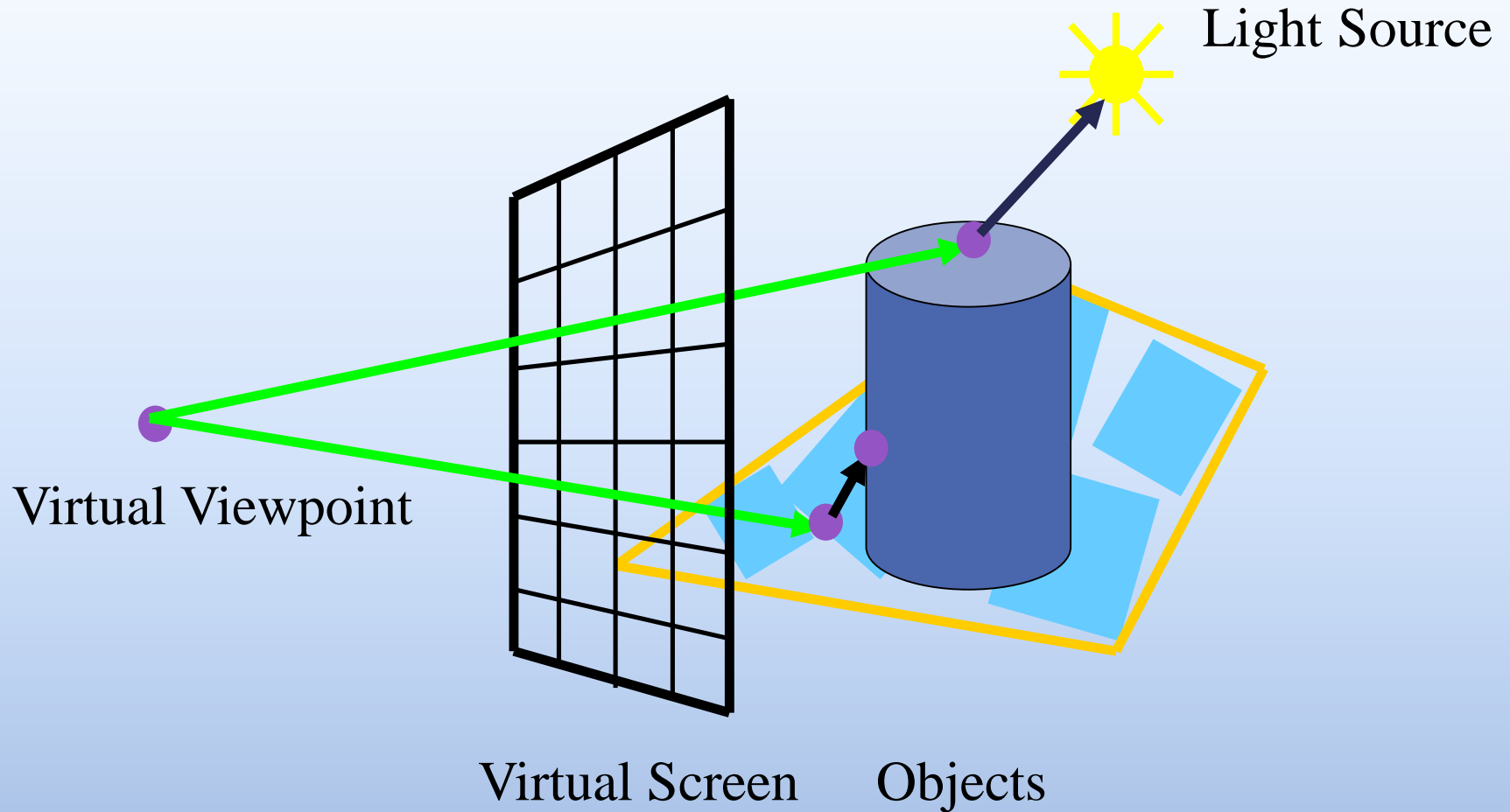
Ray misses all
objects: Pixel
colored black

RAY-TRACING METHODS

- **Ray tracing** is an extension of this basic idea.
- Instead of merely **looking for the visible surface** for each pixel, we continue to **bounce the ray around the scene** collecting intensity contributions.
- The basic ray-tracing algorithm also provides for **visible-surface detection, shadow effects, transparency, and multiple light-source illumination**



Ray Tracing

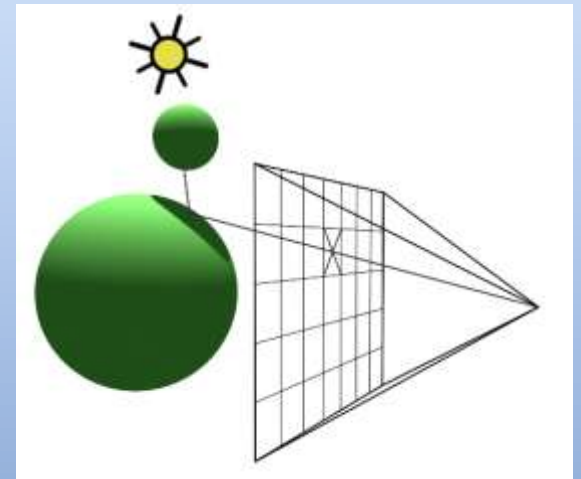


Shadow ray to light is blocked: object in shadow

Shadow ray to light is unblocked: object visible

Basic Ray-Tracing Algorithm

- Fire a single ray from each pixel position into the scene along the projection path, which is equivalent to viewing the scene through a pinhole camera.
- Determine which surfaces the ray intersects and order these by distance from the pixel.
- The nearest surface to the pixel is the visible surface for that pixel
- Reflect a ray off the visible surface along the specular reflection angle.
- For transparent surfaces also send a ray through the surface in the refraction direction.
- Repeat the process for these secondary rays



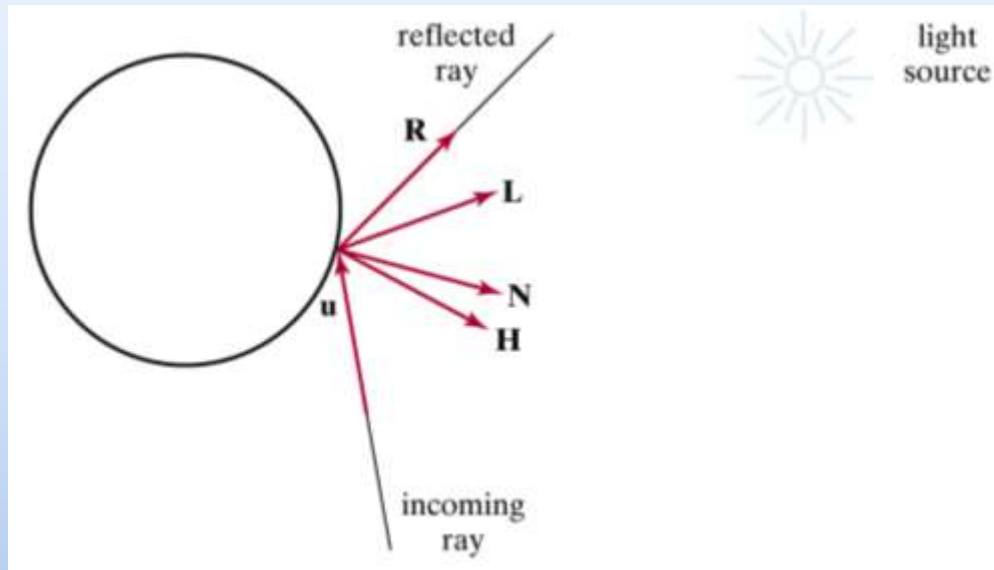
Basic Ray-Tracing Algorithm

We terminate a ray-tracing path when any one of the following conditions is satisfied:

- The ray intersects no surfaces
- The ray intersects a light source that is not a reflecting surface
- A maximum allowable number of reflections have taken place

Basic Ray-Tracing Algorithm

At each surface intersection the illumination model is invoked to determine the surface intensity contribution



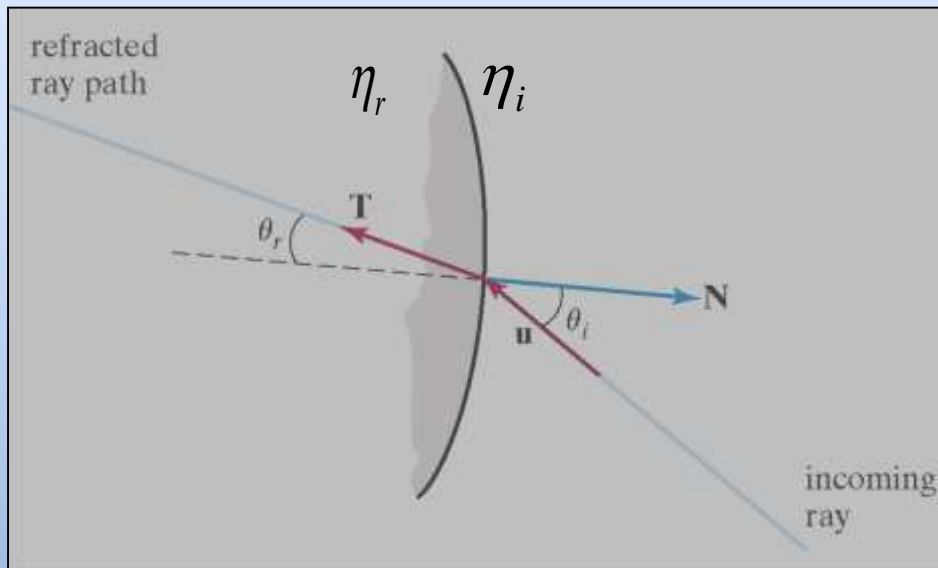
$$R = u - (2u \cdot N)N$$

- The path from the intersection to the light source is known as the **shadow ray**
- If any object intersects the shadow ray between the surface and the light source then the surface is in shadow with respect to that source

Basic Ray-Tracing Algorithm

For transparent surfaces we need to calculate a ray to represent the light refracted through the material

The direction of the refracted ray is determined by the refractive index of the material



$$T = \frac{\eta_i}{\eta_r} u - \left(\cos \theta_r - \frac{\eta_i}{\eta_r} \cos \theta_i \right) N$$

Angle of refraction θ_r *can be calculated from* Snell's law

$$\cos \theta_r = \sqrt{1 - \left(\frac{\eta_i}{\eta_r} \right)^2 (1 - \cos^2 \theta_i)}$$

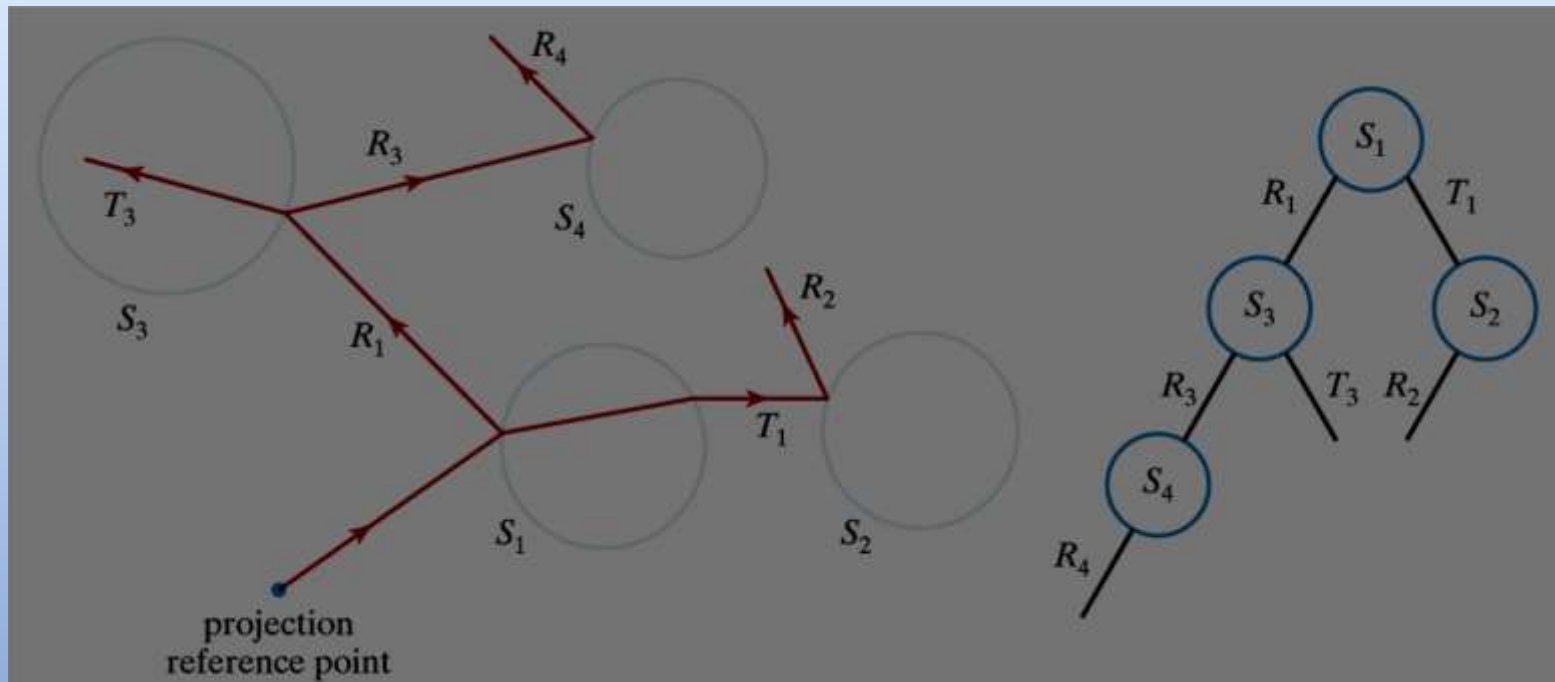
Basic Ray-Tracing Algorithm

As the rays around the scene ricochet (fly away from a surface after hitting it) each intersected surface is added to a binary **ray-tracing tree**

- The left branches in the tree are used to represent reflection paths
- The right branches in the tree are used to represent transmission paths

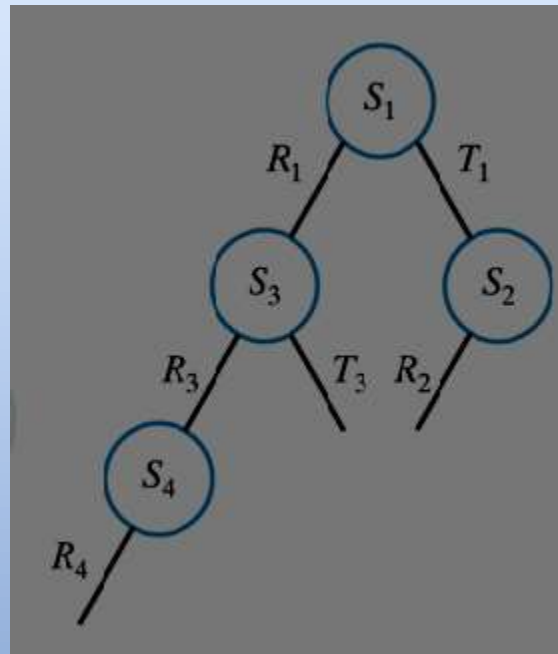
The tree's nodes store the intensity at that surface

The tree is used to keep track of all contributions to a given pixel



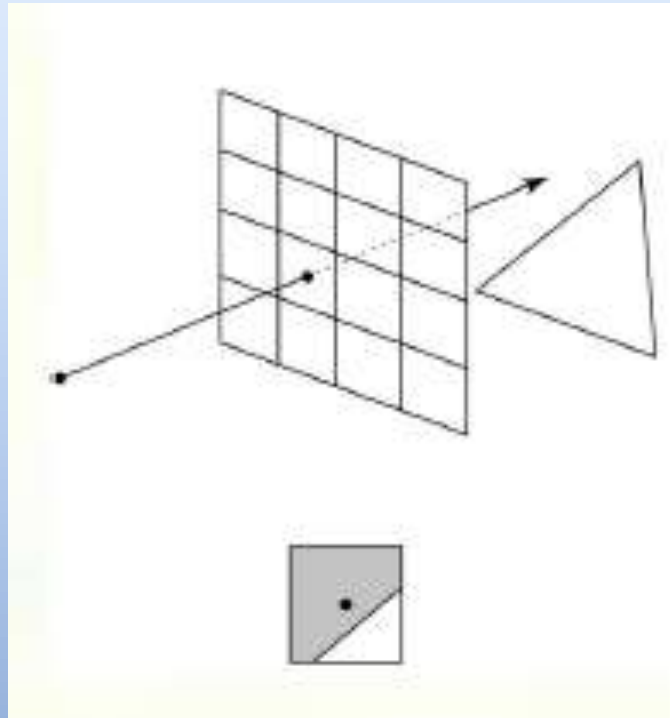
Basic Ray-Tracing Algorithm

- After the ray-tracing tree has been completed for a pixel the intensity contributions are accumulated
- We start at the terminal nodes (bottom) of the tree
- The surface intensity at each node is attenuated by the distance from the parent surface and added to the intensity of the parent surface
- The sum of the attenuated intensities at the root node is assigned to the pixel



Antialiased Ray tracing

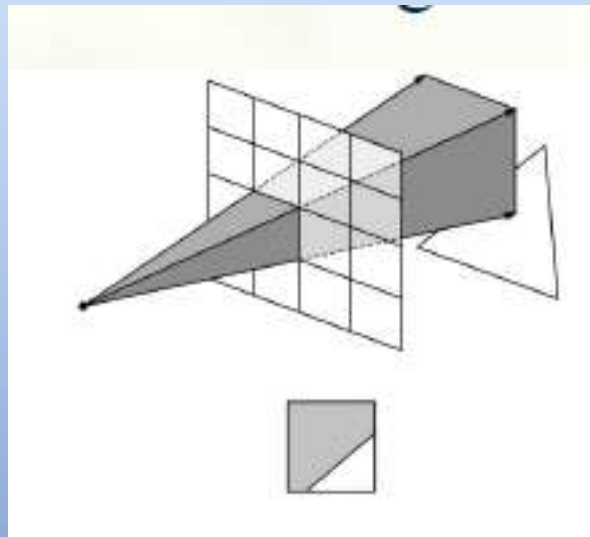
- When casting one ray per pixel, we are likely to have aliasing objects.
- Solution is to use multiple rays.
- Two basic techniques for antialiasing in ray-tracing algorithms are *supersampling* and *adaptive sampling*.



Antialiased Ray tracing

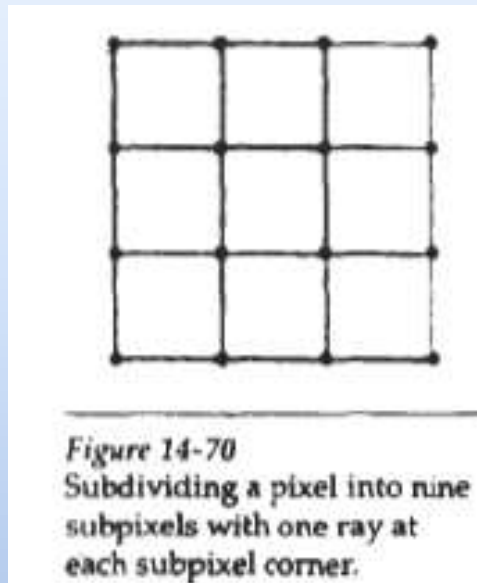
- The pixel is treated as a finite square area instead of a single point.
- **Super sampling** uses multiple, evenly spaced rays (samples) over each pixel area.
- **Adaptive sampling** uses unevenly spaced rays in some regions of the pixel area.
- When multiple rays per pixel are used, the intensities of the pixel rays are averaged to produce the overall pixel intensity.

Super sampling :one ray is generated through each corner of the pixel.



Antialiased Ray tracing

If the intensities for the four rays are not approximately equal, or if some small object lies between the four rays, we divide the pixel area into subpixels and repeat the process.



This subdivision process can be continued until each subpixel has approximately equal-intensity rays or an upper bound, say, 256, has been reached for the number of rays per pixel.

Antialiased Ray tracing

Instead of passing rays through pixel **corners**, we can generate rays through subpixel centers

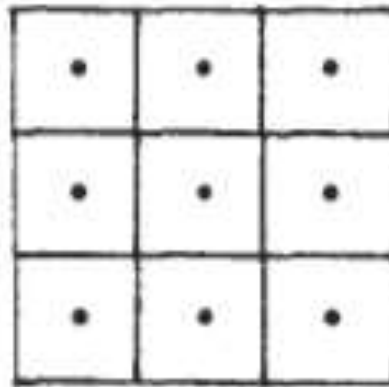


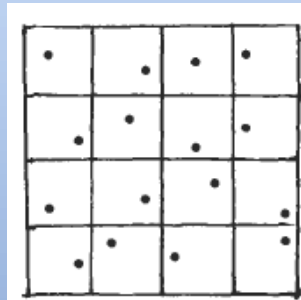
Figure 14-71
Ray positions centered on
subpixel areas.

Distributed Ray Tracing

- Pixel sampling is accomplished by randomly distributing a number of rays over the pixel surface.
- Choosing ray positions completely at random, however, can result in the rays clustering together in a small region of the pixel area, and leaving other parts of the pixel unsampled

Distributed Ray Tracing

- A better approximation is obtained by using a technique called **jittering** on a regular subpixel grid.
- This is usually done by initially dividing the pixel area (a unit square) into the 16 subarea and **generating a random jitter position** in each subarea.
- The random ray positions are obtained by jittering the center coordinates of each subarea by small amounts, δ_x , and δ_y , where both δ_x , and δ_y , are assigned values in the interval $(-0.5, 0.5)$.
- We then choose the ray position in a cell with center coordinates (\mathbf{x}, \mathbf{y}) as the jitter position $(\mathbf{x} + \delta_x, \mathbf{y} + \delta_y)$.



Distributed Ray Tracing

- Integer codes 1 through 16 are randomly assigned to each of the 16 rays.
- Each subpixel ray is then processed through the scene to determine the intensity contribution for that ray.
- The 16 ray intensities are then averaged to produce the overall pixel intensity.
- If the subpixel intensities vary too much, the pixel is further subdivided

End of Unit 5