SCS1302 COMPUTER GRAPHICS & MULTIMEDIA SYSTEMS

(Common to CSE & IT)

UNIT IV METHODS AND MODELS

Visible surface detection methods – Illumination models – Halftone patterns – Dithering techniques – Polygon rendering methods – Ray tracing methods – Color models and color application

VISIBLE SURFACE DETECTION METHODS

Introduction

- To generate realistic graphics displays.
- To determine what is visible within a scene from a chosen viewing position.
- For 3D worlds, this is known as visible surface detection or hidden surface elimination.
- So many methods and algorithms are available.
- Some require more memory space, some require more processing time.
- Which method? For which application? Depends on which object to be displayed, available equipment, complexity of the scene, etc..

Basic classification

- Two basic classifications –based on either an object or projected image , which is going to be displayed

i) **Object-space Methods-** compares objects and parts of objects to each other within a scene definition to determine which surfaces are visible

ii) **Image-space Methods-** visibility is determined point-by-point at each pixel position on the projection plane

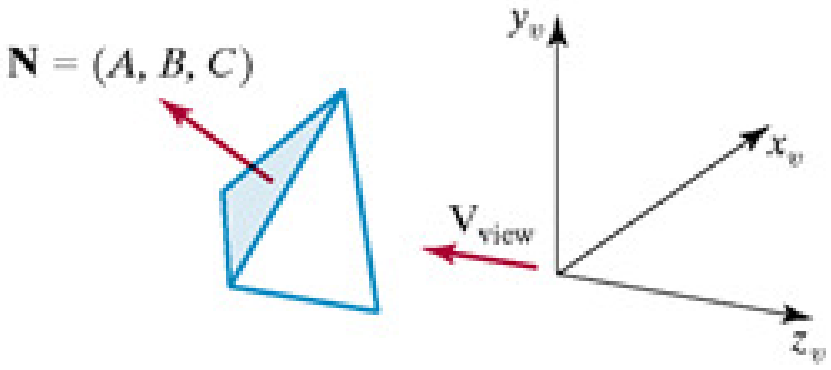
Image-space Method is the most commonly used method

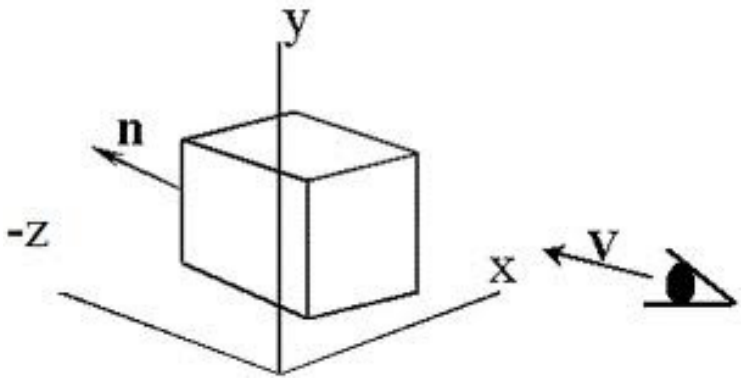
Back-face detection

- Object Space Method.
- To find the back faces of a polyhedron.
- Consider a polygon surface with parameters A,B,C and D.
- A point (x,y,z) is inside the polyhedrons' backface only if

$Ax+By+Cz+D<0$

- We can simply say that the z component of the polygon's normal is less than zero, then the point is on the back face.





- If we take V as the vector in the viewing direction from the eye and N as the normal vector to the polygon's surface, then we can state the condition as

$$V.N>0$$

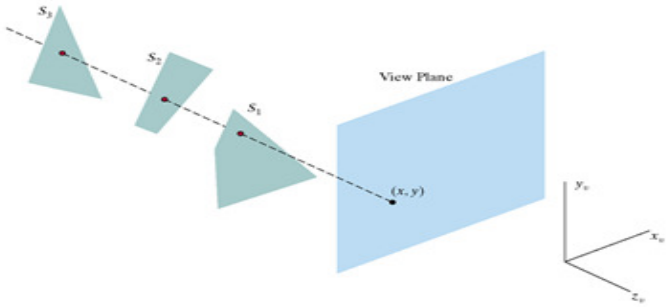
then that face is the back face.

- It eliminates about half of the polygon surfaces in a scene from further visibility tests.

Depth buffer method or z-buffer method

- Image Space Method.
- Compares surface depths at each pixel position throughout the scene on the projection plane.
- It is usually applied to images containing polygons.

- Very fast method.



Depth buffer algorithm

- For each projected (x, y) pixel position of a polygon, calculate the depth z (if not already known)
- If z < depth(x, y), compute the surface colour at that position and set

  depth(x, y) = z

  frameBuff(x, y) = surfColour(x, y)

After all surfaces are processed depthBuff and frameBuff will store correct values

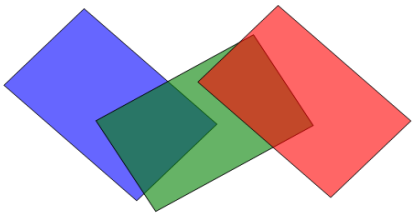
$$z = \frac{-Ax - By - D}{C}$$

$$z' = \frac{-A(x+1) - By - D}{C}$$

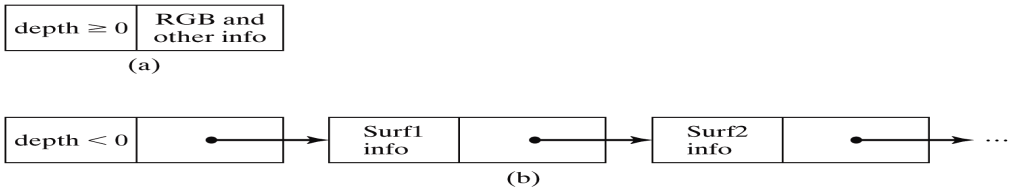
- The depth-buffer algorithm proceeds by starting at the top vertex of the polygon
- Then we recursively calculate the *x*-coordinate values down a left edge of the polygon
- The *x* value for the beginning position on each scan line can be calculated from the previous one.

A-buffer Method

- Extension of Depth-buffer -> **accumulation buffer(A-Buffer)**
- **A-** Antialiased, Area-averaged, Accumulation – Buffer.
- Drawback of Depth-buffer-can find one visible surface at each pixel position. Cannot accumulate intensity values for more than one surface.
- Each buffer position can reference a linked-list of surfaces.



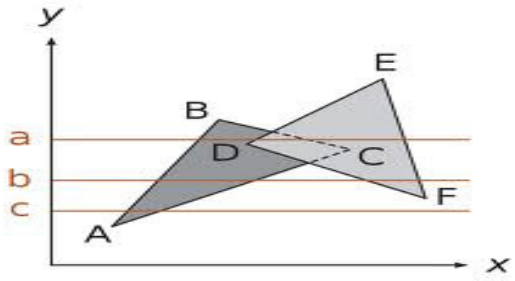
- Each position has 2 fields.
- Depth field – stores a positive or negative real number
- Intensity field- stores surface intensity information or a pointer value.



- If depth is >= 0(single surface), then the surface data field stores the depth of that pixel position as before
- If depth < 0 (multiple surfaces)then the data filed stores a pointer to a linked list of surface data.

Scan-line method

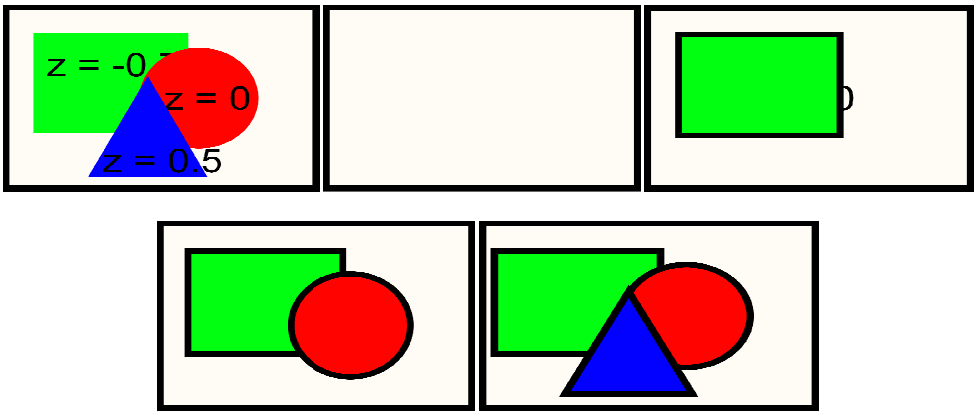
- Idea is to intersect each polygon with a particular scanline. Solve hidden surface problem for just that scanline.
- Requires a depth buffer equal to only one scanline
- The cost of tiling scene is roughly proportional to its depth complexity
- Efficient way to tile shallowly-occluded scenes



Depth- sorting or painter's algorithm

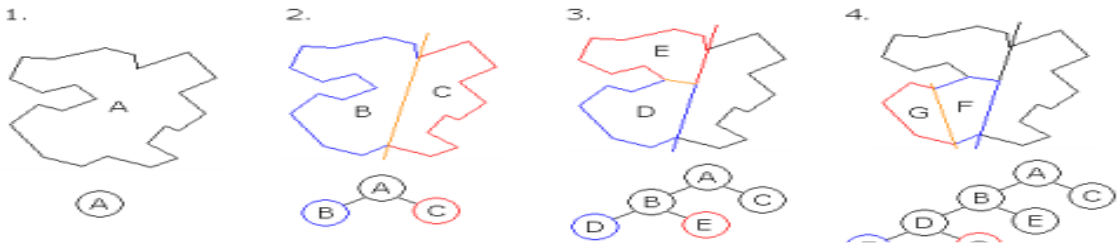
- Sort the objects by distance from the viewer.
    - Draw objects in order from farthest to nearest.

- Nearer objects will “overwrite” farther ones.
- If 2 objects DO overlap
- Need to find a plane to split one polygon by so that each new polygon is entirely in front of or entirely behind the other
- Polygons may actually intersect, so then need to split each polygon by the other



BSP Tree Method

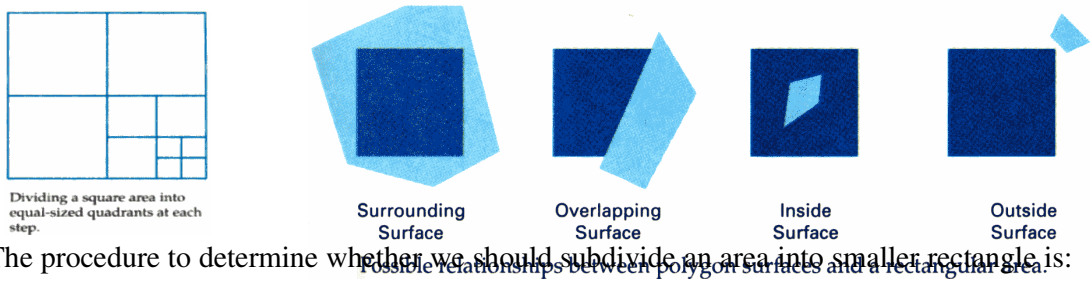
- A BSP (Binary Space-Partitioning) tree is formed by first choosing a triangle from the set of all triangles in the scene.
- The plane that contains this triangle is called P. Classify all other triangles into two groups: One group in front of P, and the other group behind P. All triangles that are intersected by P are split into multiple smaller triangles, each of which is either in front of P or behind P.
- Within each group, recursively pick another triangle and partition all the triangles in this group into two sub-groups.
- Do this until there is only one triangle in each group.
- The result is a tree.



Area Subdivision Method

The area-subdivision method takes advantage of area coherence in a scene by locating those view areas that represent part of a single surface.
The total viewing area is successively divided into smaller and smaller rectangles

until each small area is simple, ie. it is a single pixel, or is covered wholly by a part of a single visible surface or no surface at all.



The procedure to determine whether we should subdivide an area into smaller rectangle is:

1. We first classify each of the surfaces, according to their relations with the area:
Surrounding surface - a single surface completely encloses the area
Overlapping surface - a single surface that is partly inside and partly outside the area
Inside surface - a single surface that is completely inside the area
Outside surface - a single surface that is completely outside the area.

To improve the speed of classification, we can make use of the bounding rectangles of surfaces for early confirmation or rejection that the surfaces should be belong to that type.

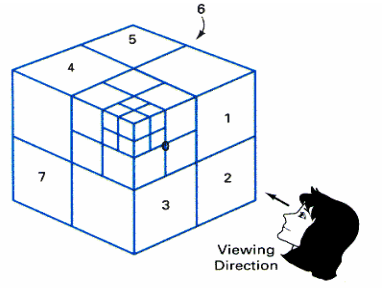
2. Check the result from 1., that, if any of the following condition is true, then, no subdivision of this area is needed.

a. All surfaces are outside the area.
b. Only one surface is inside, overlapping or surrounding surface is in the area.
c. A surrounding surface obscures all other surfaces within the area boundaries. For cases b and
c, the color of the area can be determined from that single surface.

Octree Methods

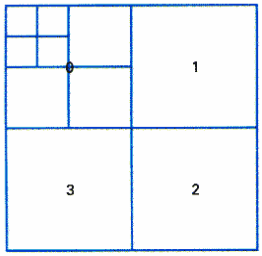
In these methods, octree nodes are projected onto the viewing surface in a front-to-back order. Any surfaces toward the rear of the front octants (0,1,2,3) or in the back octants (4,5,6,7) may be hidden by the front surfaces.

With the numbering method (0,1,2,3,4,5,6,7), nodes representing octants 0,1,2,3 for the entire region are visited before the nodes representing octants 4,5,6,7. Similarly the nodes for the front four sub- octants of octant 0 are visited before the nodes for the four back sub- octants.



Octants in Space

When a colour is encountered in an octree node, the corresponding pixel in the frame buffer is painted only if no previous color has been loaded into the same pixel position.



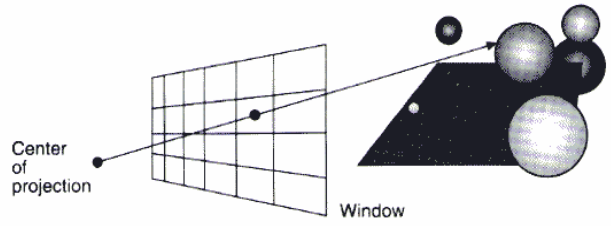
Quadrants for
the View Plane

In most cases, both a front and a back octant must be considered in determining the correct color values for a quadrant. But

- If the front octant is homogeneously filled with some color, we do not process the back octant.
- If the front is empty, it is necessary only to process the rear octant.
- If the front octant has heterogeneous regions, it has to be subdivided and the sub-octants are handled recursively.

Ray Casting Method

The intensity of a pixel in an image is due to a ray of light, having been reflected from some objects in the scene, pierced through the centre of the pixel.



A ray is fired from the center of projection through each pixel to which the window maps, to determine the closest object intersected.

So, visibility of surfaces can be determined by tracing a ray of light from the centre of projection
(viewer's eye) to objects in the scene. (backward-tracing).

- Find out which objects the ray of light intersects.
- Then, determine which one of these objects is closest to the viewer.
- Then, set the pixel color to this object.

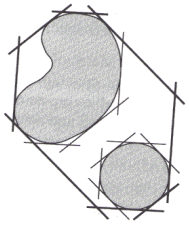
The ray-casting approach is an effective visibility-detection method for scenes with curved surfaces, particularly spheres.

Speeding up the intersection calculation in ray tracing

For 1024x1024 pixels image and 100 objects in the scene, total number of object intersection calculations is about 100 millions.

1. Bounding Volume Approach

- Test for intersection of ray with the bounding volume of the object.
- Typical bounding volumes are sphere, ellipsoid, rectangular solid. The intersection calculation for these bounding volumes are usually faster than the displayed object.
- If ray does not intersect bounding volume, no further processing of the object is needed.
- Other bounding volumes include convex polygons formed by a set of planes.

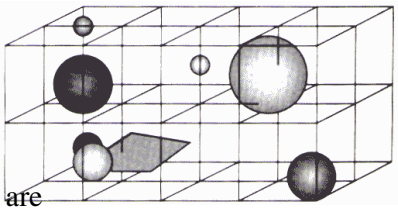


2. Using Hierarchies

- If a parent bounding volume does not intersect with a ray, all its children bounding volumes do not intersect with the ray and need not be processed
- Thus reduce the number of intersection calculations.

3. Space Partitioning Approach

- Partition the space into a regular grid of equal-size volumes.
- Each volume has associated with it a list of objects which are contained within or intersect the volume.
- Intersection calculation is only applied to those objects that are contained within the partition through which the ray passes.
- Objects lying within the partitions which do not intersect with the ray are not processed.



Summary and Comparison

The most appropriate algorithm to use depends on the scene

- depth-sort is particularly suited to scene with objects which are spread out along z-axis and/or with a small number of objects => rarely overlap in depth

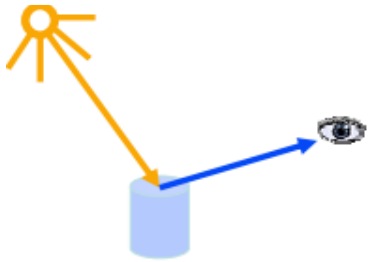
- scan-line and area subdivision algorithms are suitable to scene where objects are spread out horizontally and/or scene with small number of objects (about several thousand surfaces).
- Z-buffer and subdivision algorithms perform best for scene with fewer than a few thousand surfaces.
- Octree is particularly good because it does not require any pre-sorting or intersection calculations.
- If parallel processing hardware is available, ray tracing would be a good choice (each processor handles a ray).

ILLUMINATION MODELS

Introduction

• Motivation: In order to produce realistic images, we must simulate the appearance of surfaces under various lighting conditions

• Illumination Model: Given the illumination incident at a point on a surface, quantifies the reflected light



Illumination Model Parameters

• Lighting effects are described with models that consider the interaction of light sources with object surfaces

• The factors determining the lighting effects are:

– The light source parameters:

• Positions

• Electromagnetic Spectrum

• Shape

– The surface parameters

• Position

• Reflectance properties

• Position of nearby surfaces

– The eye (camera) parameters

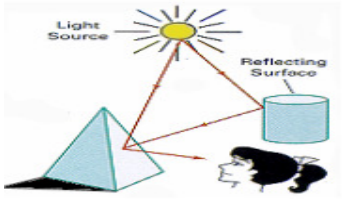
• Position

• Sensor spectrum sensitivities

Illumination Models and Rendering

• An illumination model is used to calculate the intensity of the light that is reflected at a given point on a surface

• A rendering method uses intensity calculations from the illumination model to determine the light intensity at all pixels in the image

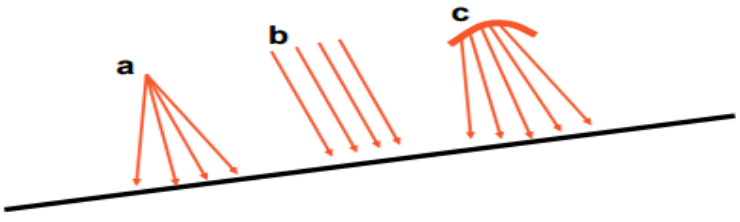


Light Source Models

• Point Source (a): All light rays originate at a point and radially diverge. A reasonable approximation for sources whose dimensions are small compared to the object size

• Parallel source (b): Light rays are all parallel. May be modeled as a point source at infinite distance (the sun)

• Distributed source (c): All light rays originate at a finite area in space. It models a nearby source, such as a fluorescent light



• Simplified and fast methods for calculating surfaces intensities, mostly empirical

• Calculations are based on optical properties of surfaces and the lighting conditions (no reflected sources nor shadows)

• Light sources are considered to be point sources

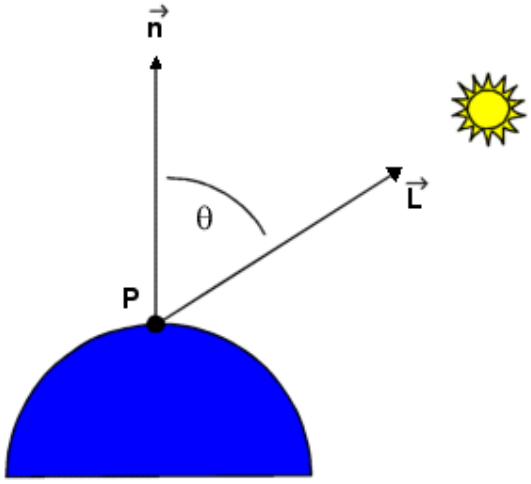
• Reasonably good approximation for most scenes

Simple Illumination Model

Surfaces in real world environments receive light in 3 ways:

1. Directly from existing light sources such as the sun or a lit candle
2. Light that passes and refracts through transparent objects such as water or a glass vase
3. Light reflected, bounced, or diffused from other exisiting surfaces in the environment

Diffuse illumination



Lambert's cosine law of reflection as shown in the above diagram:

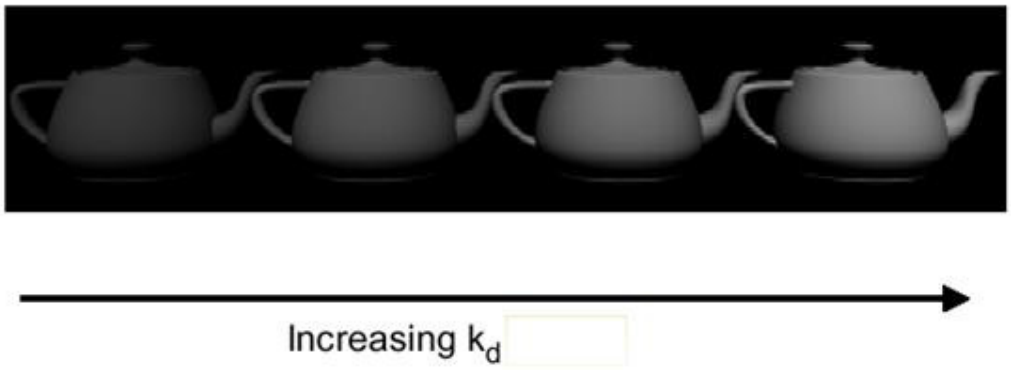
1. **n**; a normal vector to the surface to be illuminated.
2. **L,** a vector from the surface position that points towards the light source.
3. I_l, an intensity for a point light source.
4. k_d , a diffuse reflection constant.

Equation gives the brightness of a surface point in terms of the brightness of a light source and its orientation relative to the surface normal vector, **n**,

$$I = I_l k_d \hat{\mathbf{n}} \bullet \hat{\mathbf{L}} = I_l k_d \cos\theta \qquad 0 \le \theta \le \pi/2$$

- *I* is the reflected intensity
  - Measures how bright the surface is at that point.

- Surface brightness varies as a function of the angle between **n** and **L**
    - When **n** and **L** coincide, the light source is directly overhead.
- ***I*** is at a maximum and $\cos_q = 1$.
    - As the angle increases to 90^o, the cosine decreases the intensity to 0.
- All the quantities in the equation are normalized between 0 and 1.
- *I* is converted into frame buffer intensity values by multiplying by the number of shades available.
- With $2^8 = 256$ possible shades, we have 1 * 255, the brightest frame buffer intensity.
- For **n** and **L** at an angle of 45^o, I = cos 45^o * 256 = 181.



- An image rendered with a Lambertian shader exhibits a dull, matte finish.
- It appears as if it has been viewed by a coal miner with a lantern attached to his helmet.
- In reality, an object is not only subjected to direct illumination from the primary light source I_l, but secondary scattered light from all remaining surfaces.

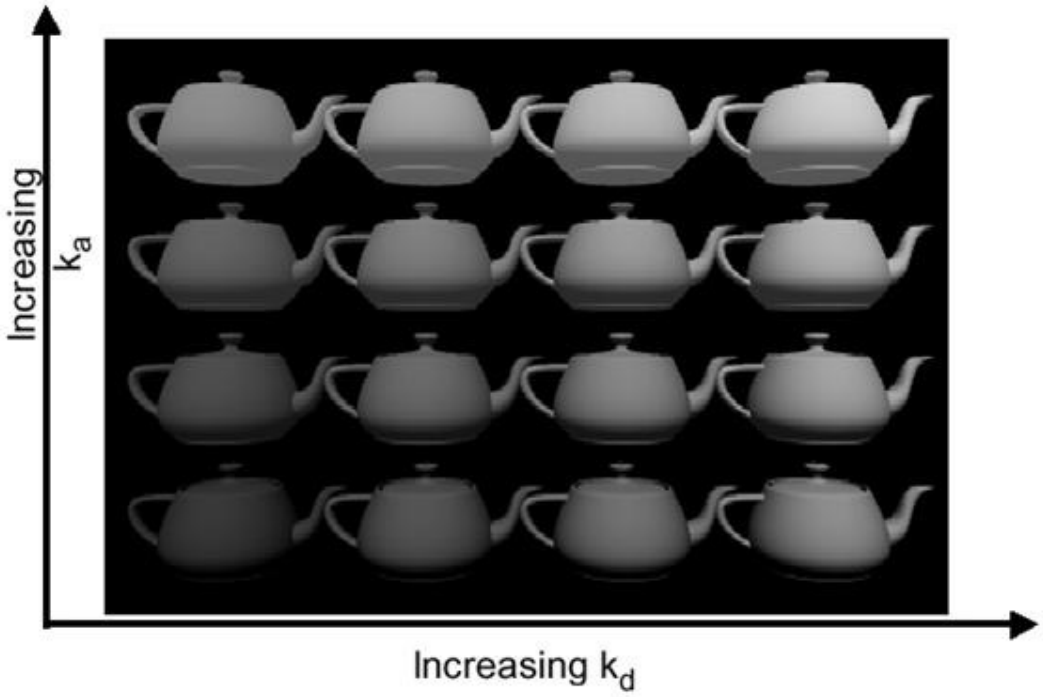
Ambient illumination

- Simple illuminated model is unable to directly accommodate all scattered light
- It is grouped together as independent intensity, I_a.
- The formula becomes

$$I = I_a k_a + I_l k_d \cos\theta \qquad 0 \le \theta \le \pi/2$$

- $I_a k_a$ is the ambient illumination term, taking into account the additional environmental illumination, I_a, and the ability of the object to absorb it, k_a.

Only ambient illumination



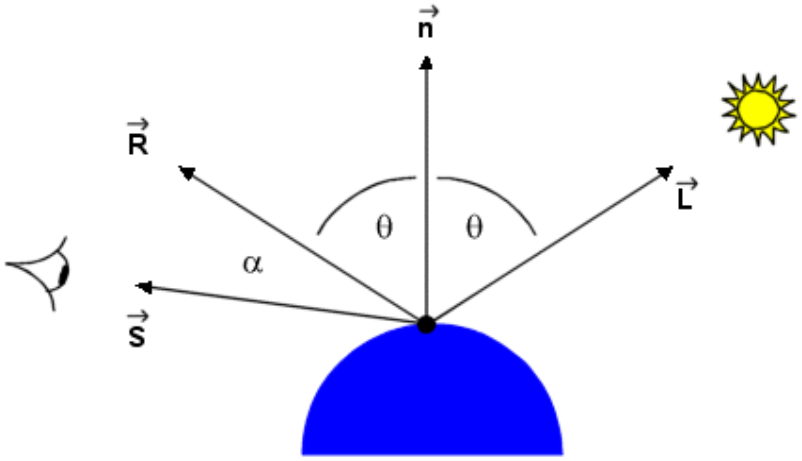
Diffuse + ambient illumination

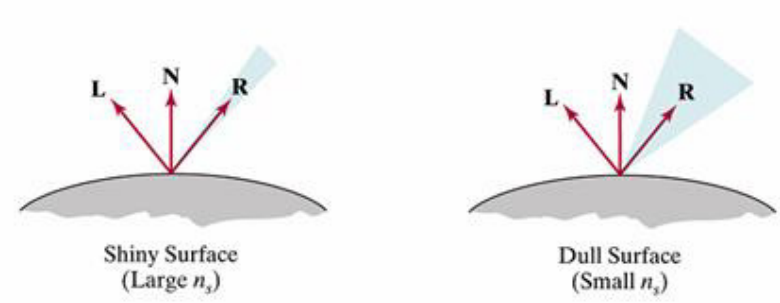
- Illumination decreases from the light source by I/d^2.
- Objects at a greater distance from the light source receive less illumination and appear darker.
- Above equation is distance independent.
- Dividing the Lambertian term by d^2 would seem to get the physics right, but it makes the intensity vary sharply over short distance.
- Modified distance dependence is employed, giving the following equation, where d is the distance from the light source to the object.

$$I = I_a k_a + \frac{I_i k_d \cos\theta}{d + K} \qquad 0 \le \theta \le \pi/2$$

Specular Highlights – (Phong Reflection Model)

- Regions of significant brightness, exhibited as spots or bands, characterize objects that specularly reflect light.
- Specular highlights originate from smooth, sometimes mirrorlike surfaces
- Fresnel equation is used to simulate this effect.
- The Fresnel equation states that for a perfectly reflecting surface the angle of incidence equals the angle of reflection.





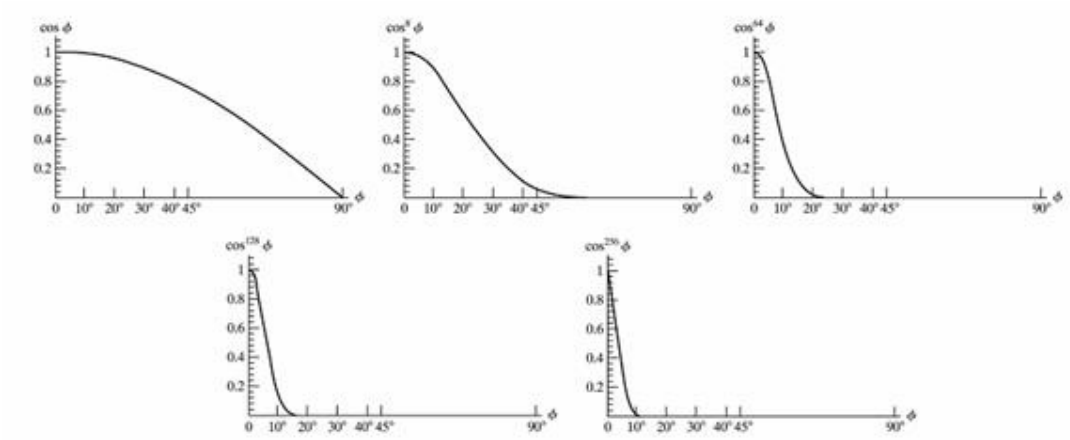
Modeling specular reflections (shaded area) with parameter n_s.

- Most objects are not perfect mirrors.
  - some angular scattering of light.
  - If the viewer increases the angle (a) between himself, the line of sight vector (**S**), and the reflectance vector (**R**), the bright spot gradually disappears.
  - Smooth surfaces scatter light less then rough surfaces.
  - This produces more localized highlights.

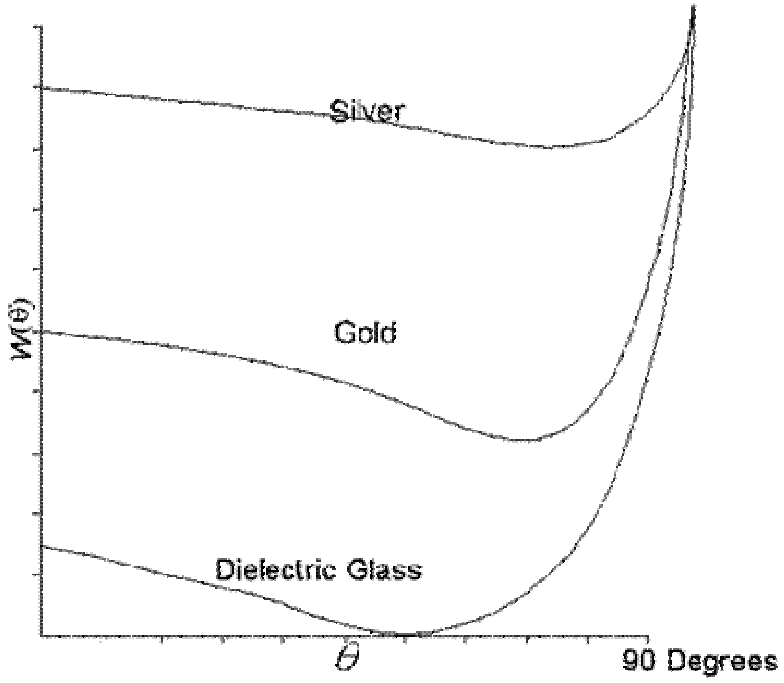
$$I_{l,specular} = W(\theta) I_l \cos^n \alpha$$

$$W(\theta) = \text{specular reflection coefficient}$$

$$n = \text{specular reflection exponent}$$



Plots of $\cos^{n_s} \phi$ using five different values for the specular exponent n_s.

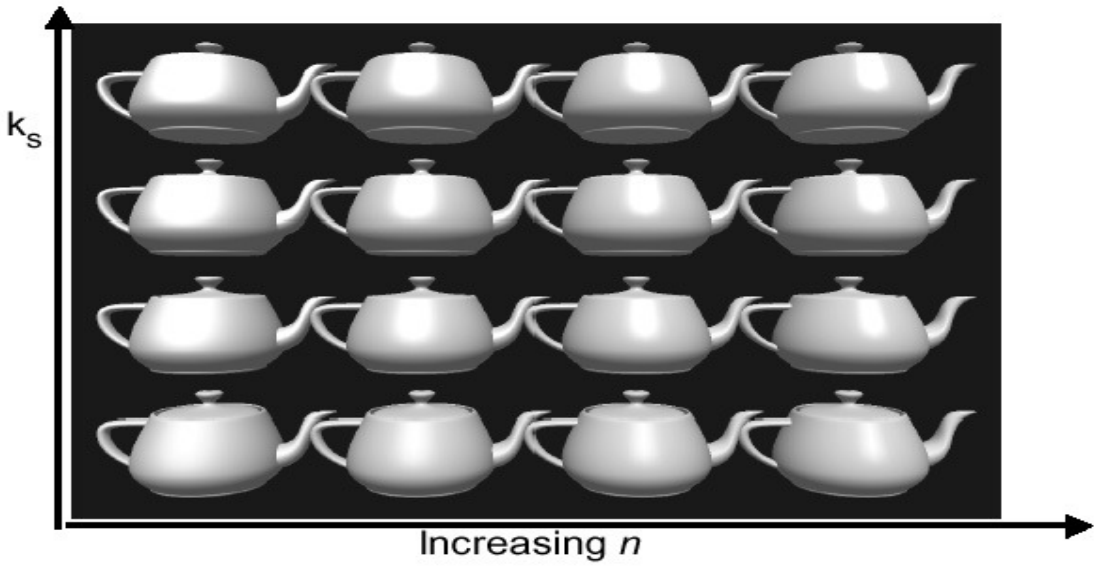


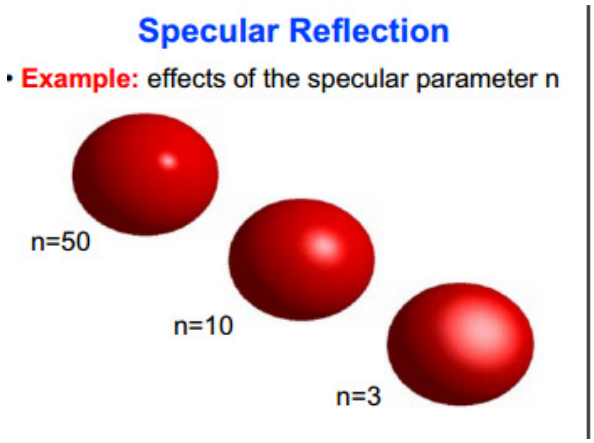
- Building this effect into the lighting model gives

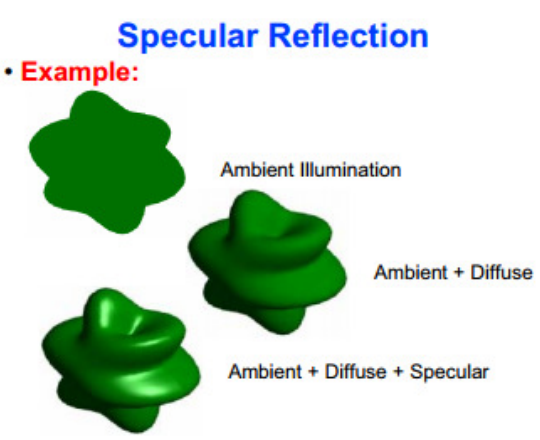
$$I = I_a k_a + \frac{I_l}{d + K}\left(k_d \cos\theta + k_s \cos^n \alpha\right)$$

- Specular reflectance term possesses a specular reflectance constant, k_s.
- The cosine term is raised to the nth power.

    - Small values of n (e.g. 5) distribute the specular highlights, characteristic of glossy paper.
    - High values of n (e.g. 50) are characteristic of metals.







Summary - Simple Illumination Model

- Deficiencies
  - point light source
  - no interaction between objects
  - ad hoc, not based on model of light propagation
- Benefits
  - fast
  - acceptable results
  - hardware support

POLYGON RENDERING METHODS

Flat Shading (also known as faceted shading)

Shades each polygon of an object based on the angle between the polygon's surface normal and the direction of the light source.

• A single intensity is calculated for each surface polygon

• Fast and simple method

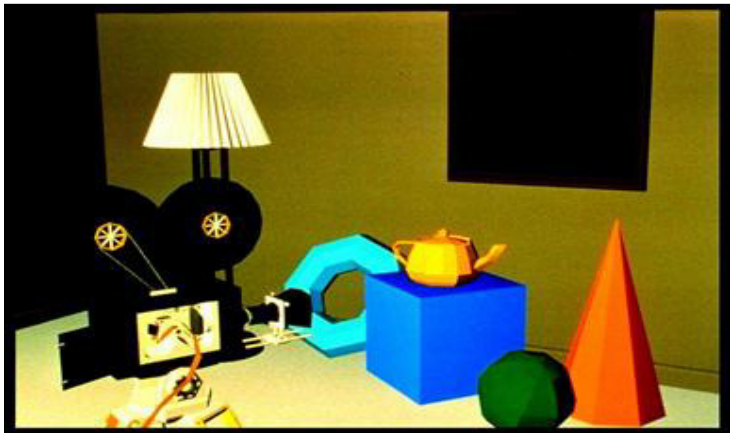
• Gives reasonable result only if all of the following assumptions are valid:

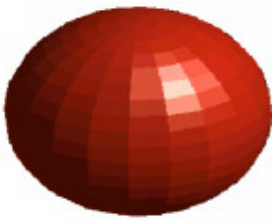
– The object is a polyhedron

– Light source is far away from the surface so that N•L is constant over each polygon

– Viewing position is far away from the surface so that V•R is constant over each polygon

Disadvantage - it gives low-polygon models a faceted look. Sometimes this look can be advantageous though, such as in modelling boxy objects. Artists sometimes use flat shading to look at the polygons of a solid model they are creating.





Gouraud Shading

Gouraud shading (Henri Gouraud, 1971) is used to achieve smooth lighting on low-polygon surfaces using the Lambertiandiffuse lighting model.

1. Calculates the surface normals for the polygons.

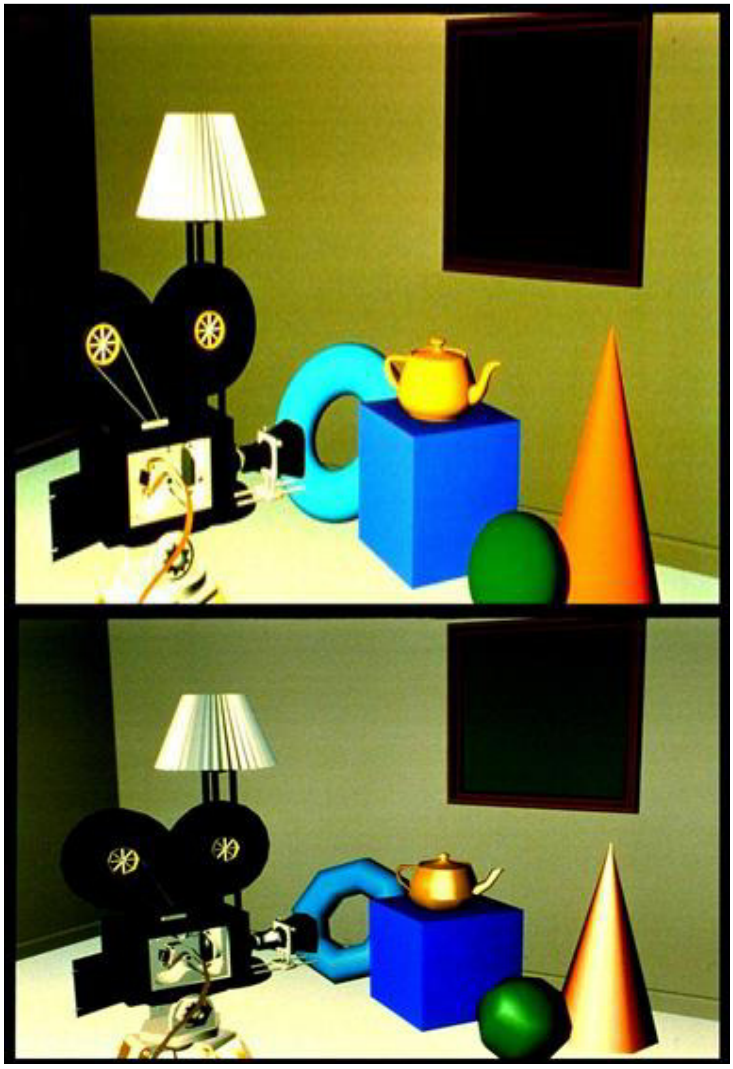
2. Normals are then averaged for all the polygons that meet at each vertex to produce a vertex normal.

3. Lighting computations are then performed to produce intensities at vertices.

4. These **intensities are interpolated** along the edges of the polygons.

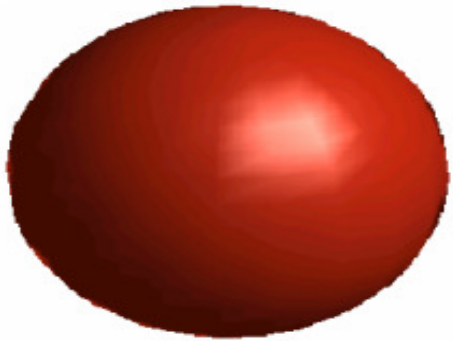
5. The polygon is filled by lines drawn across it that interpolate between the previously calculated edge intensities.

Advantage - Gouraud shading is superior to flat shading which requires significantly less processing than Gouraud, but gives low-polygon models a sharp, faceted look.



Gouraud Shading

- **Example:** Gouraud shading of a sphere



Phong Shading (Phong Interpolation Model)

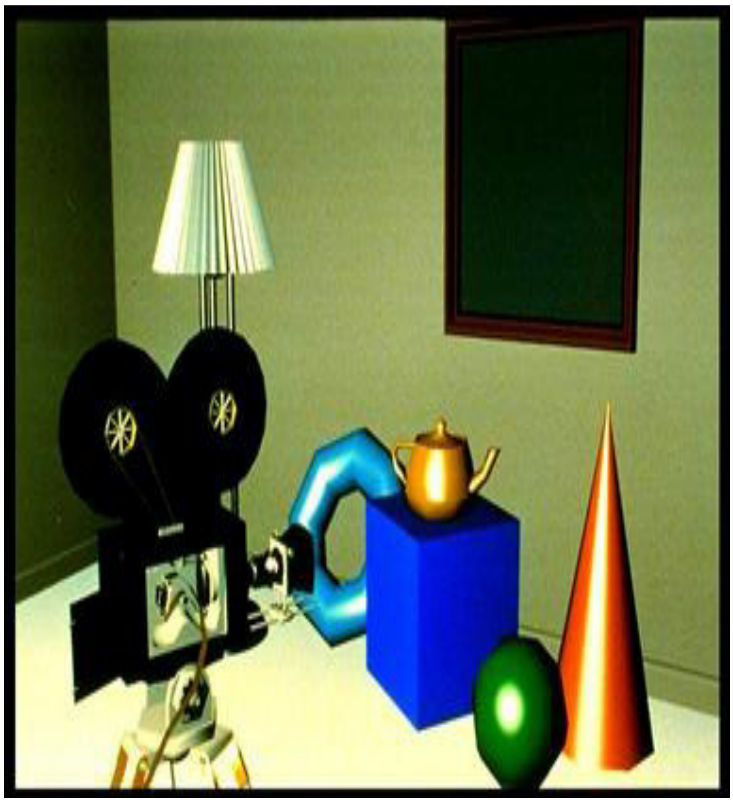
- An improved version of Gouraud shading that provides a better approximation to the Phong shading model.
- Main problem with Gouraud shading - when a specular highlight occurs near the center of a large triangle, it will usually be missed entirely. Phong shading fixes the problem.

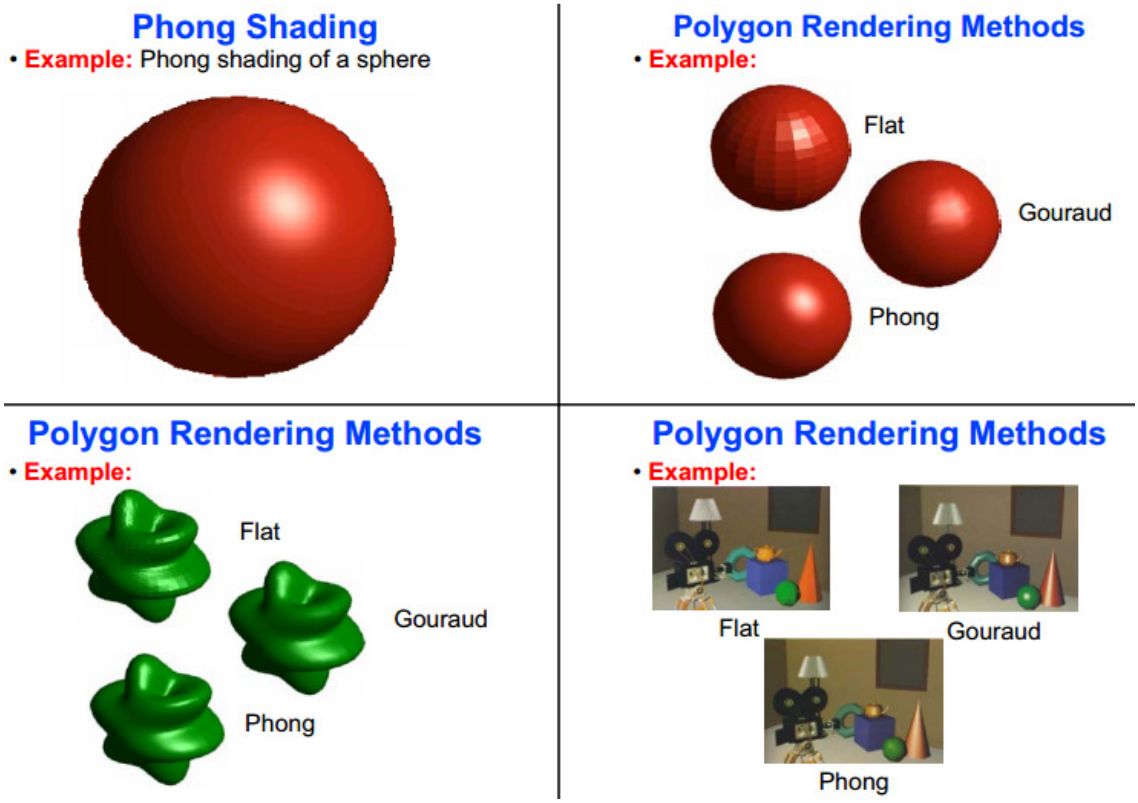
1. Calculate the surface normals at the vertices of polygons in a 3D computer model.

2. Normals are then averaged for all the polygons that meet at each vertex.

3. These **normals are interpolated** along the edges of the polygons.

4. Lighting computations are then performed to produce intensities at positions along scanlines.





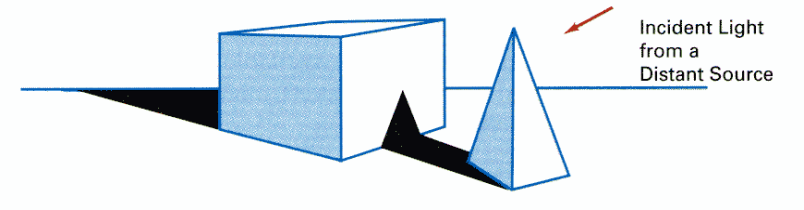
Other Effects

SHADOW

- Shadow can help to create realism. Without it, a cup, eg., on a table may look as if the cup is floating in the air above the table.

- By applying hidden-surface methods with pretending that the position of a light source is the viewing position, we can find which surface sections cannot be "seen" from the light source => shadow areas.

- We usually display shadow areas with ambient-light intensity only.



TEXTURE MAPPING

Since it is still very difficult for the computer to generate realistic textures, a method

called texture mapping is developed in which a photograph of real texture is input into the computer and mapped onto the object surface to create the texture for the object.

- Texture pattern is defined in a MxN array of *texels* or a *texture map* indiced by (u,v) coordinates.

- For each pixel in the display:

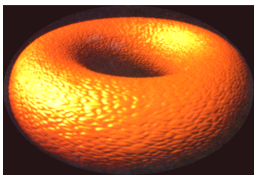
  - Map the 4 corners of pixel back to the object surface (for cureved surfaces, these 4 points define a surface patch)

  - Map the surface patch onto the texture map (this mapping computes the source area in the texture map)

  - The pixel values is modified by weighted sum of the texels' color.

BUMP MAPPING

- To simulate surface roughness within the geometric description of the object, surface roughness can be generated by perturbing surface normals.

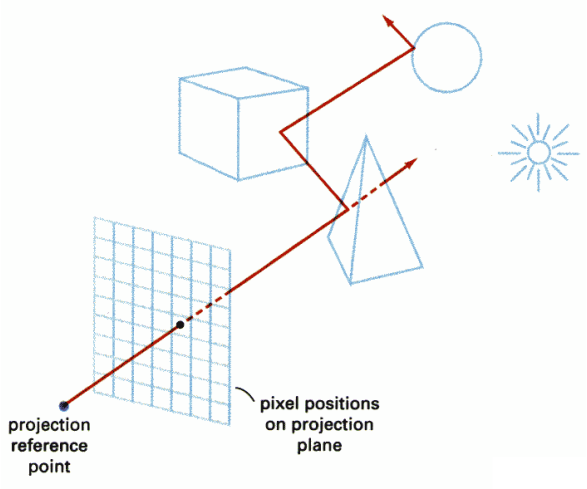


FOG EFFECT

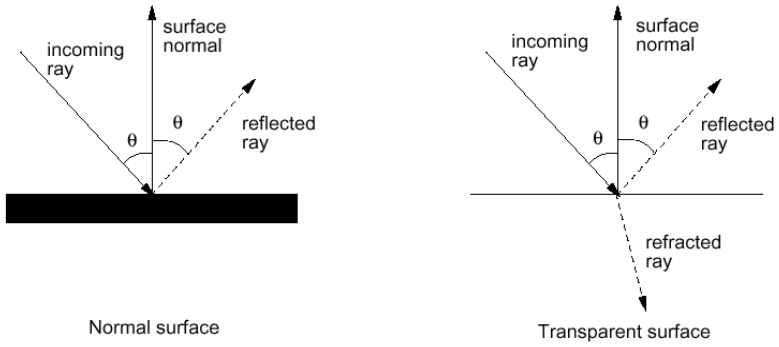
- As an object is further away from the observer, the color of the object fades.
- Fog is a general term that describes similar forms of atmospheric effects. It can be used to simulate haze, mist, smoke, or pollution.

RAY TRACING METHODS

- For each pixel on the image plane, a ray is projected from the center of projection through the pixel into the scene.
- The first object that the ray intersects is determined.
- The point at which the ray hits the object (ie. the ray intersection point) is also determined. The color value is then calculated according to the directions of the light sources to the surface normal.
- However, if there is another object located between a particular light source and the ray intersection point, then the point is in shadow and the light contribution from that particular light source is not considered.



- The ray is then reflected and projected from the object until it intersects with another object. (If the surface is a transparent surface, the ray is refracted as well as reflected.)



- The point at which the reflected ray hits the second object are determined and the color value is again calculated in a similar way as the first object.

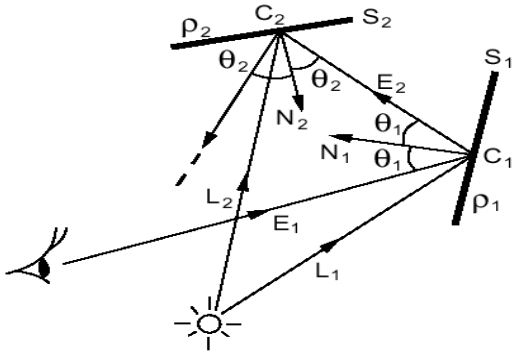
- The reflected ray is then reflected again from the second object. This process will continue until the color contribution of an intersected object is too small to be considered.

- In practical situation, se would specify the maximum number of reflections that a pixel can make to prevent spending too much processing at a particular pixel.

- All the color values calculated from the intersected objects will be weighted by the attenuation factors (which depend on the surface properties of the objects) and added up to produce a single color value. This color value becomes the pixel value.

Because ray-tracing method calculates the intensity value for each pixel independently, we can consider specular reflection and refraction in the calculation. Hence the method can generate very realistic images. The major problem of this method, however, is that it requires a lot of computations and therefore this method is slow.

To calculate the color of a pixel, consider the following diagram:



- Assume that surfaces S_1 and S_2 have reflective indices $_1$ and $_2$ respectively.

- Given the surface normal vector, N_1, and the light vector, L_1, of surface S_1, we can calculate the color, C_1, of the surface at the point where the eye ray, E_1 intersects the surface.

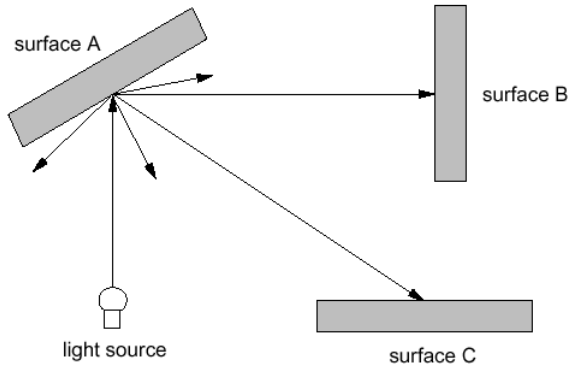
- Similarly, given the surface normal vector, N_2, and the light vector, L_2, of surface S_2, we can calculate the color, C_2, of the surface at the point where the eye ray, E_2 intersects the surface.

- The color for the pixel is then calculated as: $C_p = C_1 + C_2 + \ldots$

Radiosity

Ray-tracing can model specular reflection very well, but not diffuse reflection. Why?
Recall that in diffuse reflection, although light is reflected with equal intensity in all directions, the amount of light energy received by another surface depends on the orientation of the surface relative to the source. Hence surfaces of different orientations may receive different amount of light.

Consider the diagram, assuming that all 3 surfaces are matte. Although A reflects light with equal intensity to B and C, B would receive more light energy than C because B has a smaller angle of incident (ie. higher cos).



Radiosity is developed to model this kind of light interaction between surfaces.

- The algorithm is based on the theory of the conservation of energy.

- In a closed environment such as a room, the rate at which energy leaves a surface, called its radiosity, is the sum of the rates at which the surface emits energy and it reflects (or transmits) energy from other surfaces.

- To simplify the calculation, all surfaces in the scene are broken into small patches. Each of which is assumed to be of finite size, emitting and reflecting light uniformly over its entire area.

- The radiosity of a patch i, B_i, can be calculated as follows:

$$B_i = E_i + \rho_i \sum_{j \in all\ \ patches} L_{j \to i}$$

where $L_{j \to i}$ is the amount of light from patch j reaching patch i,

E_i is the light emitted from patch i, and

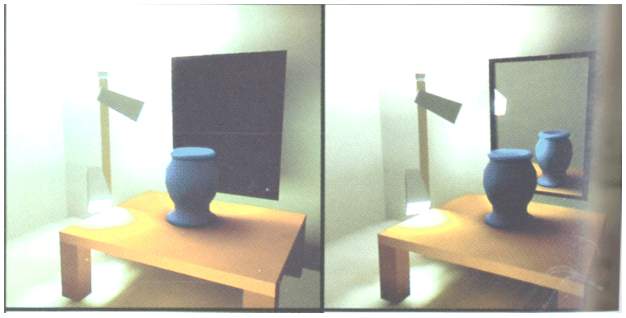
ρ_i is the reflectivity of patch i. (For an incident light, what ratio of energy is reflected)

- Once we have obtained a radiosity for each patch, we can render the scene with a scan-conversion method using the calculated radiosities as the intensities of the patches.

- Note that the radiosities calculated are view-independent. Hence, the radiosity method although can deal with diffuse reflection well, cannot deal with specular reflection. (Specular reflection is view-dependent).

- To be able to deal with both, we may combine the radiosity method with the ray-tracing method.

The prices are the added complexity of the algorithm and the increase in computational time. An example is a 2-pass approach that includes a view-independent radiosity process executed in the first pass, followed by a view-dependent ray-tracing approach in the second pass.

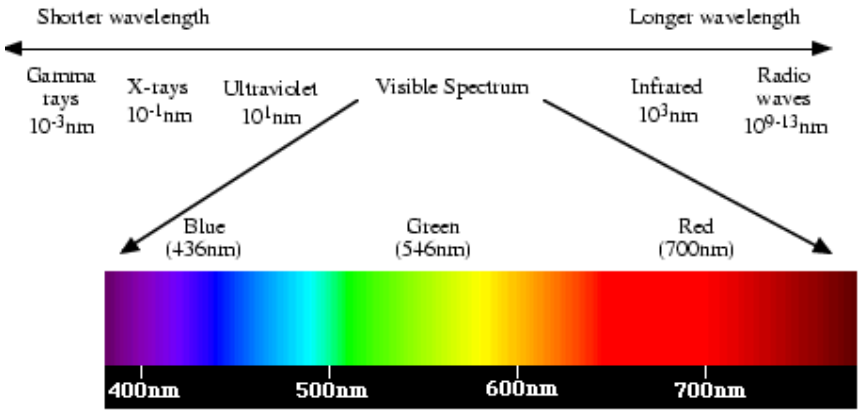


Left: radiosity. Right: Diffuse first pass and ray-traing second pass.

COLOUR IMAGE PROCESSING

The human visual system can distinguish hundreds of thousands of different colour shades and intensities, but only around 100 shades of grey. Therefore, in an image, a great deal of extra information may be contained in the colour, and this extra information can then be used to simplify image analysis, e.g. object identification and extraction based on colour.

Three independent quantities are used to describe any particular colour. The *hue* is determined by the dominant wavelength. Visible colours occur between about 400nm (violet) and 700nm (red) on the electromagnetic spectrum, as shown in the below figure.



The visible spectrum

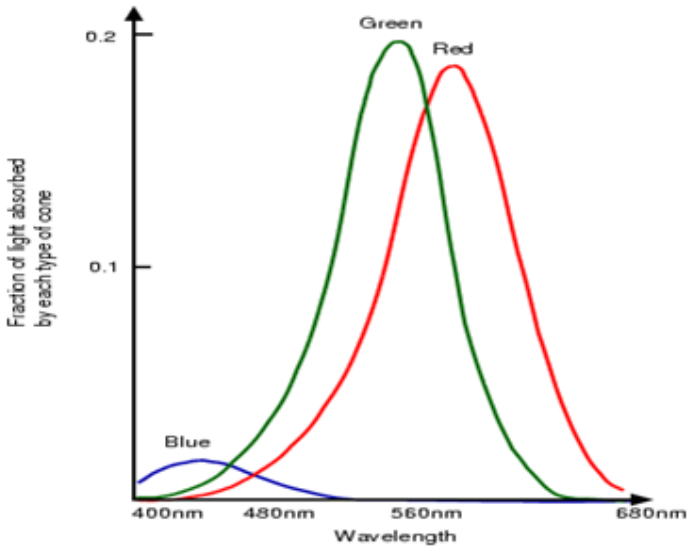
The *saturation* is determined by the excitation purity, and depends on the amount of white light mixed with the hue. A pure hue is fully saturated, i.e. no white light mixed in. Hue and saturation together determine the *chromaticity* for a given colour. Finally, the *intensity* is determined by the actual amount of light, with more light corresponding to more intense colours.

Achromatic light has no colour - its only attribute is quantity or intensity. Greylevel is a measure of intensity. The *intensity* is determined by the energy, and is therefore a physical quantity. On the other hand,*brightness* or *luminance* is determined by the perception of the colour, and is therefore psychological. Given equally intense blue and green, the blue is perceived as much darker than the green. Note also that our perception of intensity is nonlinear, with changes of normalised intensity from 0.1 to 0.11 and from 0.5 to 0.55 being perceived as equal changes in brightness.

Colour depends primarily on the reflectance properties of an object. We see those rays that are reflected, while others are absorbed. However, we also must consider the colour of the light source, and the nature of human visual system. For example, an object that reflects both red and green will appear green when there is green but no red light illuminating it, and conversely it will appear red in the absense of green light. In pure white light, it will appear yellow (= red + green).

Tristimulus theory of colour perception

The human retina has 3 kinds of cones. The response of each type of cone as a function of the wavelength of the incident light is shown in figure 2. The peaks for each curve are at 440nm (blue), 545nm (green) and 580nm (red). Note that the last two actually peak in the yellow part of the spectrum.

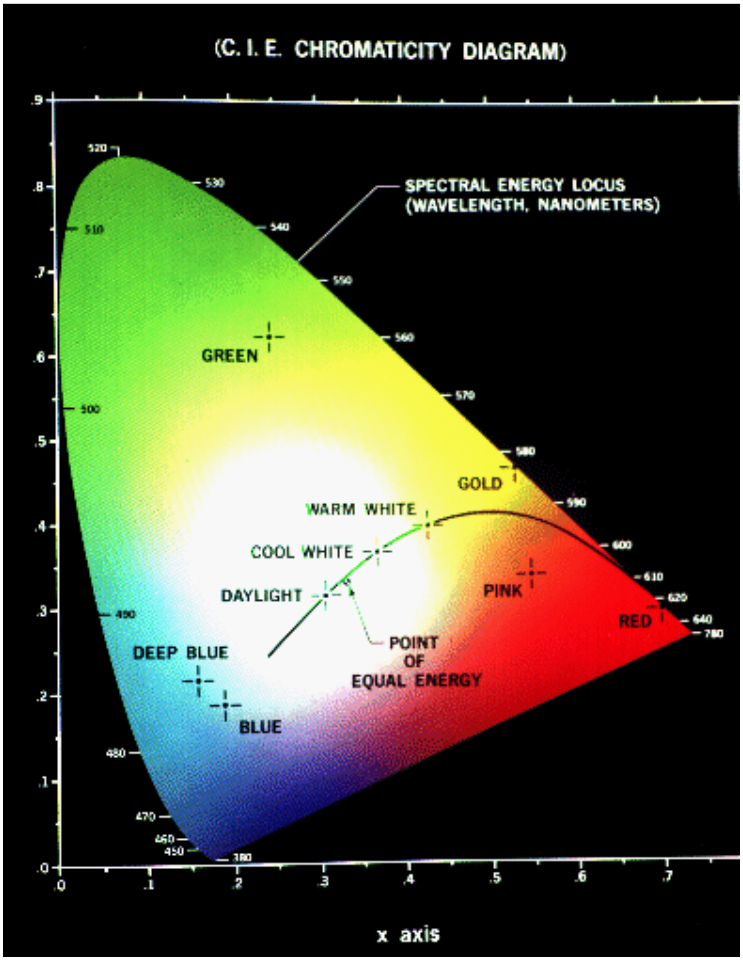


Spectral response curves for each cone type. The peaks for each curve are at 440nm (blue), 545nm (green) and 580nm (red).

CIE primaries

The tristimulus theory of colour perception seems to imply that any colour can be obtained from a mix of the three primaries, red, green and blue, but although nearly all visible colours can be matched in this way, some cannot. However, if one of the primaries is added to one of these unmatchable colours, it can be matched by a mixture of the other two, and so the colour may be considered to have a negative weighting of that particular primary.

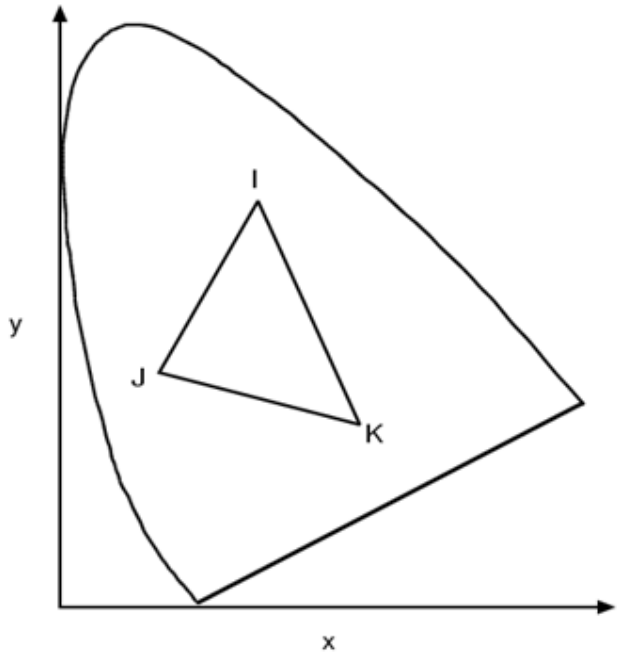
In 1931, the Commission Internationale de l'Éclairage (CIE) defined three standard primaries, called X, Y and Z, that can be added to form all visible colours. The primary Y was chosen so that its colour matching function exactly matches the luminous-efficiency function for the human eye, given by the sum of the three curves in the below figure.



The CIE Chromaticity Diagram showing all visible colours. x and y are the normalised amounts of the X and Y primaries present, and hence z = 1 - x - y gives the amount of the Z primary required.

The CIE Chromaticity Diagram shows all visible colours. The x and y axis give the normalised amounts of the X and Y primaries for a particular colour, and hence z = 1 - x - y gives the amount of the Z primary required. Chromaticity depends on dominant wavelength and saturation, and is independent of luminous energy. Colours with the same chromaticity, but different luminance all map to the same point within this region.

The pure colours of the spectrum lie on the curved part of the boundary, and a standard white light has colour defined to be near (but not at) the point of equal energy x = y = z = 1/3. Complementary colours, i.e. colours that add to give white, lie on the endpoints of a line through this point. As illustrated in below figure, all the colours along any line in the chromaticity diagram may be obtained by mixing the colours on the end points of the line. Furthermore, all colours within a triangle may be formed by mixing the colours at the vertices. This property illustrates graphically the fact that all visible colours cannot be obtained by a mix of R, G and B (or any other three visible) primaries alone, since the diagram is not triangular!



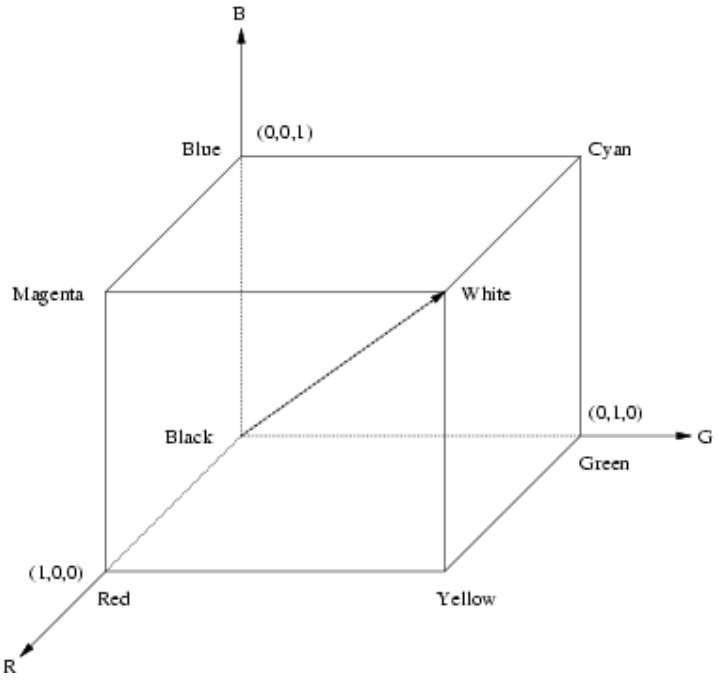
Mixing colours on the chromaticity diagram. All colours on the line IJ can be obtained by mixing colours I and J, and all colours in the triangle IJK can be obtained by mixing colours I, J and K.

COLOR MODELS

Colour models provide a standard way to specify a particular colour, by defining a 3D coordinate system, and a subspace that contains all constructible colours within a particular model. Any colour that can be specified using a model will correspond to a single point within the subspace it defines. Each colour model is oriented towards either specific hardware (RGB,CMY,YIQ), or image processing applications (HSI).

The RGB Model

In the RGB model, an image consists of three independent image planes, one in each of the primary colours: red, green and blue. Specifying a particular colour is by specifying the amount of each of the primary components present. Below figure shows the geometry of the RGB colour model for specifying colours using a Cartesian coordinate system. The greyscale spectrum, i.e. those colours made from equal amounts of each primary, lies on the line joining the black and white vertices.



The RGB colour cube. The greyscale spectrum lies on the line joining the black and white vertices.

This is an additive model, i.e. the colours present in the light add to form new colours, and is appropriate for the mixing of coloured light for example. The image on the left of figure 6 shows the additive mixing of red, green and blue primaries to form the three secondary colours yellow (red + green), cyan (blue + green) and magenta (red + blue), and white ((red + green + blue).

The RGB model is used for colour monitors and most video cameras.

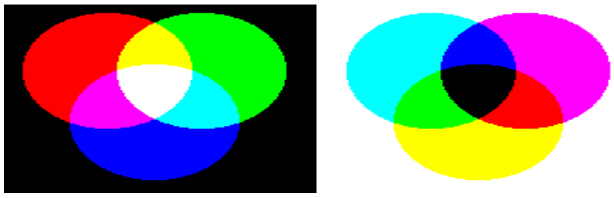
The CMY Model

The CMY (cyan-magenta-yellow) model is a subtractive model appropriate to absorption of colours, for example due to pigments in paints. Whereas the RGB model asks what is added to black to get a particular colour, the CMY model asks what is subtracted from white. In this case, the primaries are cyan, magenta and yellow, with red, green and blue as secondary colours.

When a surface coated with cyan pigment is illuminated by white light, no red light is reflected, and similarly for magenta and green, and yellow and blue. The relationship between the RGB and CMY models is given by:

$$\begin{pmatrix} C \\ M \\ Y \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} - \begin{pmatrix} R \\ G \\ B \end{pmatrix}.$$

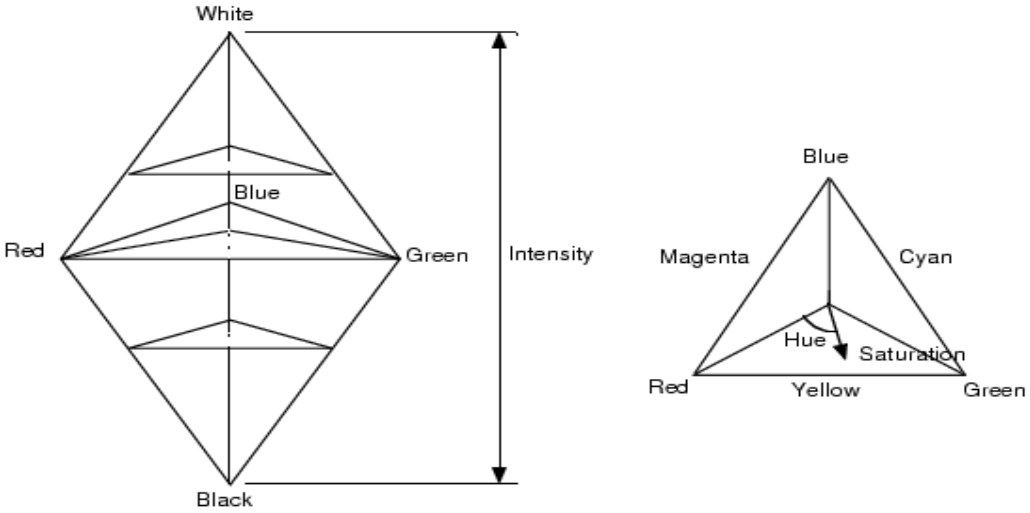
The CMY model is used by printing devices and filters.



The figure on the left shows the additive mixing of red, green and blue primaries to form the three secondary colours yellow (red + green), cyan (blue + green) and magenta (red + blue), and white ((red + green + blue). The figure on the right shows the three subtractive primaries, and their pairwise combinations to form red, green and blue, and finally black by subtracting all three primaries from white.

The HSI Model

As mentioned above, colour may be specified by the three quantities hue, saturation and intensity. This is the HSI model, and the entire space of colours that may be specified in this way is shown in below figure.



The HSI model, showing the HSI solid on the left, and the HSI triangle on the right, formed by taking a horizontal slice through the HSI solid at a particular intensity. Hue is measured from red, and saturation is given by distance from the axis. Colours on the surface of the solid are fully saturated, i.e. pure colours, and the greyscale spectrum is on the axis of the solid. For these colours, hue is undefined.

Conversion between the RGB model and the HSI model is quite complicated. The intensity is given by,

$$I = (R+G+B)/3$$

where the quantities R, G and B are the amounts of the red, green and blue components, normalised to the range [0,1]. The intensity is therefore just the average of the red, green and blue components. The saturation is given by,

$$S = (\min(R,G,B)) / I$$

where the min(R,G,B) term is really just indicating the amount of white present. If any of R, G or B are zero, there is no white and we have a pure colour.

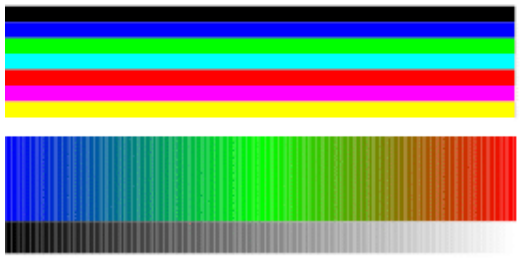
The YIQ Model

The YIQ (luminance-inphase-quadrature) model is a recoding of RGB for colour television, and is a very important model for colour image processing. The conversion from RGB to YIQ is given by:

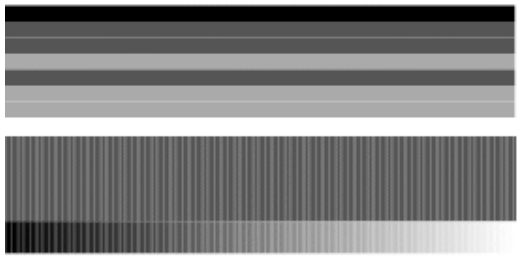
$$\begin{pmatrix} Y \\ I \\ Q \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.523 & 0.311 \end{pmatrix} \cdot \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

The luminance (Y) component contains all the information required for black and white television, and captures our perception of the relative brightness of particular colours. That we perceive green as much lighter than red, and red lighter than blue, is indicated by their respective weights of 0.587, 0.299 and 0.114 in the first row of the conversion matrix above. These weights should be used when converting a colour image to greyscale if you want the perception of brightness to remain the same. This is not the case for the intensity component in an HSI image, as shown in below figure.

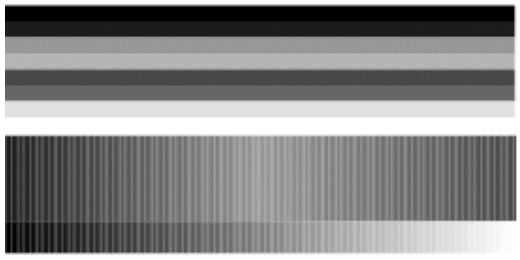
The Y component is the same as the CIE primary Y.



(a) Colour Image



(b) Intensity Image



(c) Luminance Image

Image (a) shows a colour test pattern, consisting of horizontal stripes of black, blue, green, cyan, red, magenta and yellow, a colour ramp with constant intensity, maximal saturation, and hue changing linearly from red through green to blue, and a greyscale ramp from black to white. Image (b) shows the intensity for image (a). Note how much detail is lost. Image (c) shows the luminance. This third image accurately reflects the brightness variations preceived in the original image.

Applying Greyscale Transformations to Colour Images

Given all these different representations of colour, and hence colour images, the question arises as to what is the best way to apply the image processing techniques we have covered so far to these images? One possibility is to apply the transformations to each colour plane in an RGB image, but what exactly does this mean? If we want to increase the contrast in a dark image by histogram equalisation, can we just equalise each colour independently? This will result in quite different colours in our transformed image. In general it is better to apply the transformation to just the intensity component of an HSI image, or the luminance component of a YIQ image, thus leaving the chromaticity unaltered.

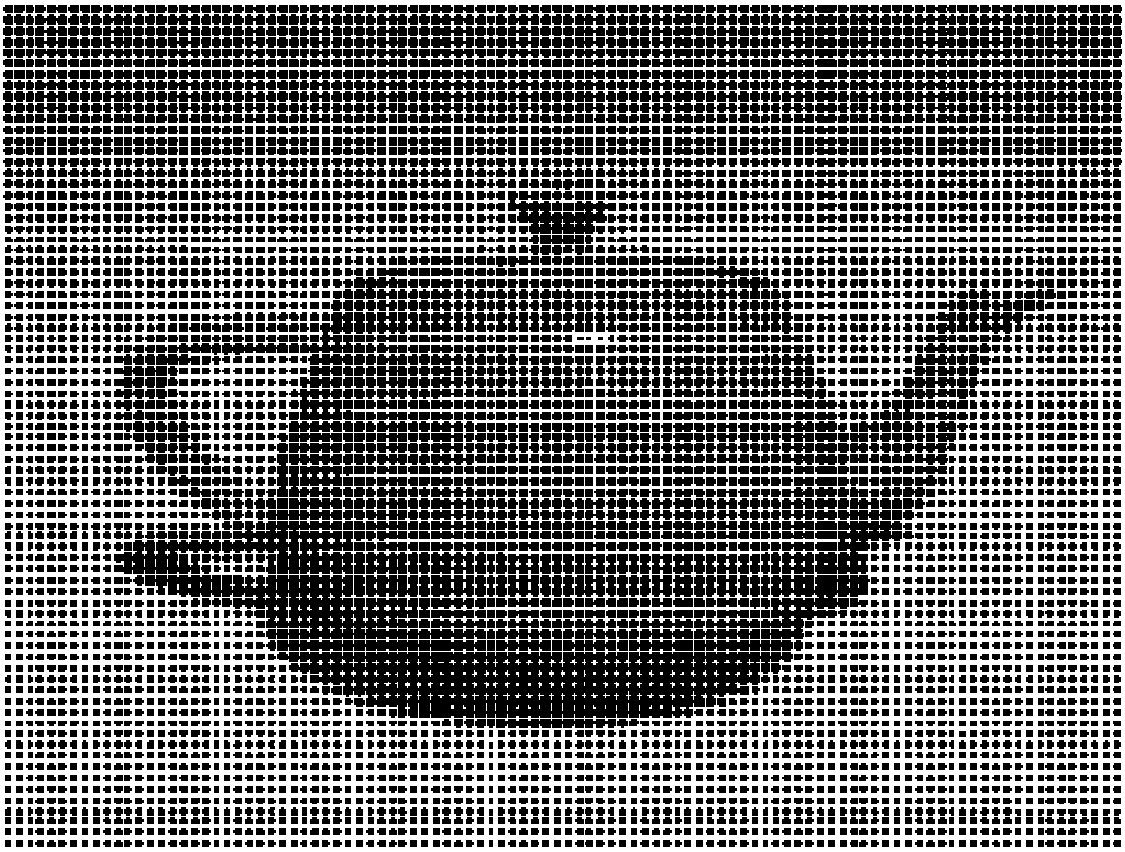
An example is shown in below figure. When histogram equalisation is applied to each colour plane of the RGB image, the final image is lighter, but also quite differently coloured to the original. When histogram equalisation is only applied to the luminance component of the

image in YIQ format, the result is more like a lighter version of the original image, as required.

The top image is a very dark image of a forest scene. The middle image is the result of applying histogram equalisation to each of the red, green and blue components of the original image. The bottom image is the result of converting the image to YIQ format, and applying histogram equalisation to the luminance component only.

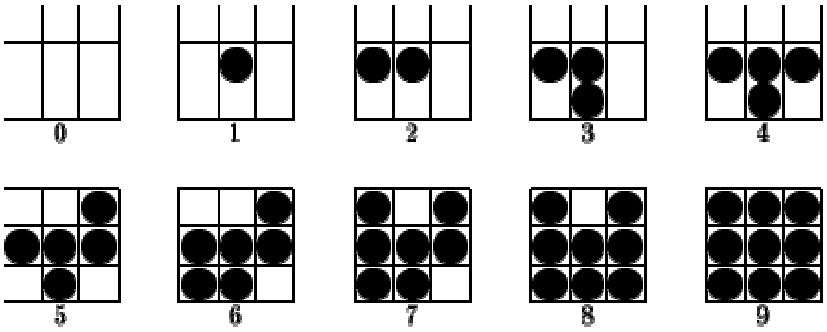
HALFTONE APPROXIMATIONS

If we cannot display all the required intensity levels (e.g. on a printer) we need a trick using the spatial integration that our eyes performs: In normal light the eye can only detect about one arc minute (1/60 degree). This is called VISUAL ACUITY. Thus instead of gray dots a small black disk with radius varying according to the blackness (1-I) is printed. Usually, for newspaper 60-80 and for magazines 150-200 different radiuses are used. This process is called HALFTONING.

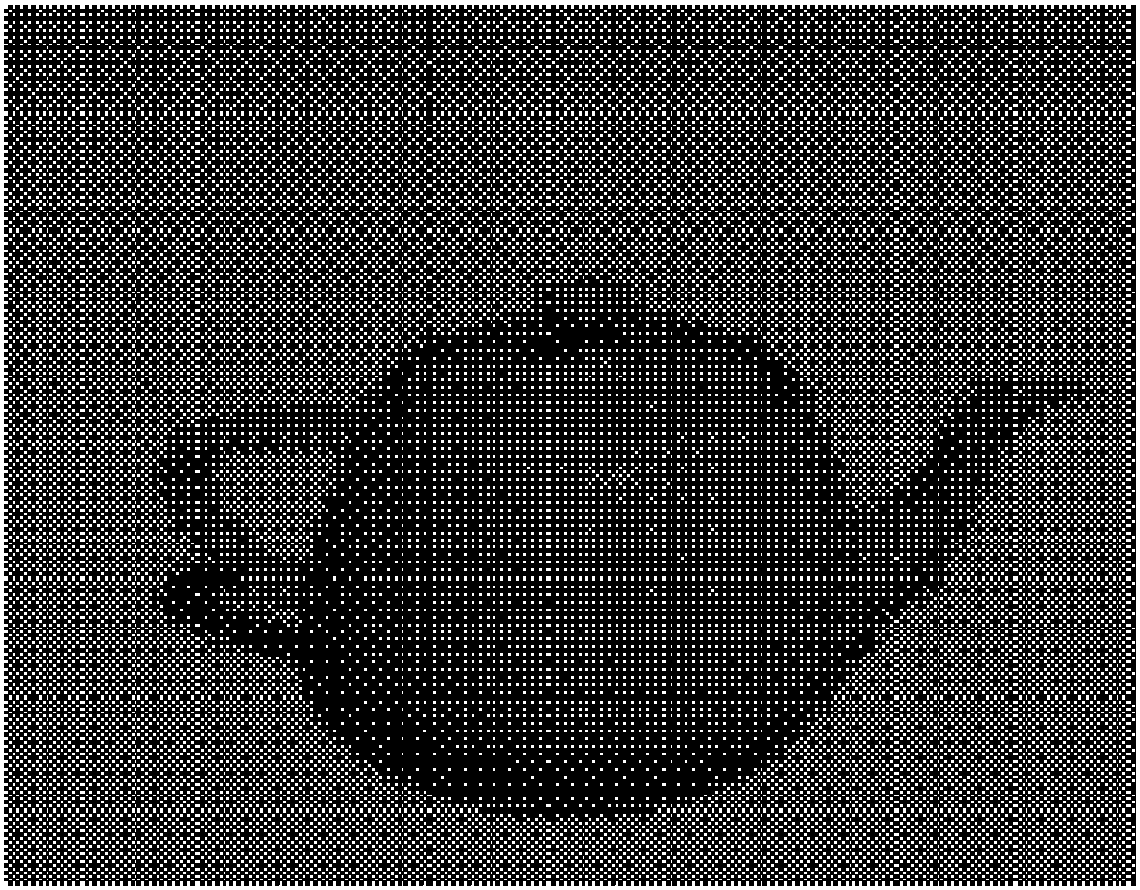


Halftoning

For computers this is implemented as CLUSTERED-DOT ORDERED DITHERING, e.g. the following patterns are used for each pixel of intensity `0...9`:



Clustered-dot ordered dithering



Clustered-dot ordered dithering

This can be described by a dither matrix

$$\begin{pmatrix} 6 & 8 & 4 \\ 1 & 0 & 3 \\ 5 & 2 & 7 \end{pmatrix}$$

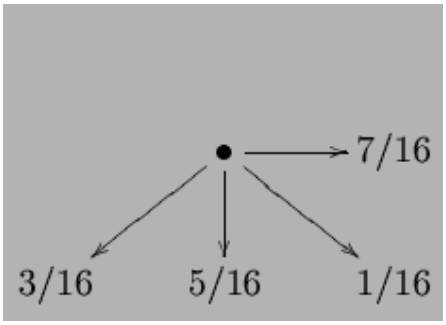
saying that in order to display intensity I one should turn on all those pixels whose value in this matrix is less than I. It is chosen in such a way, that visual artifacts are avoided:

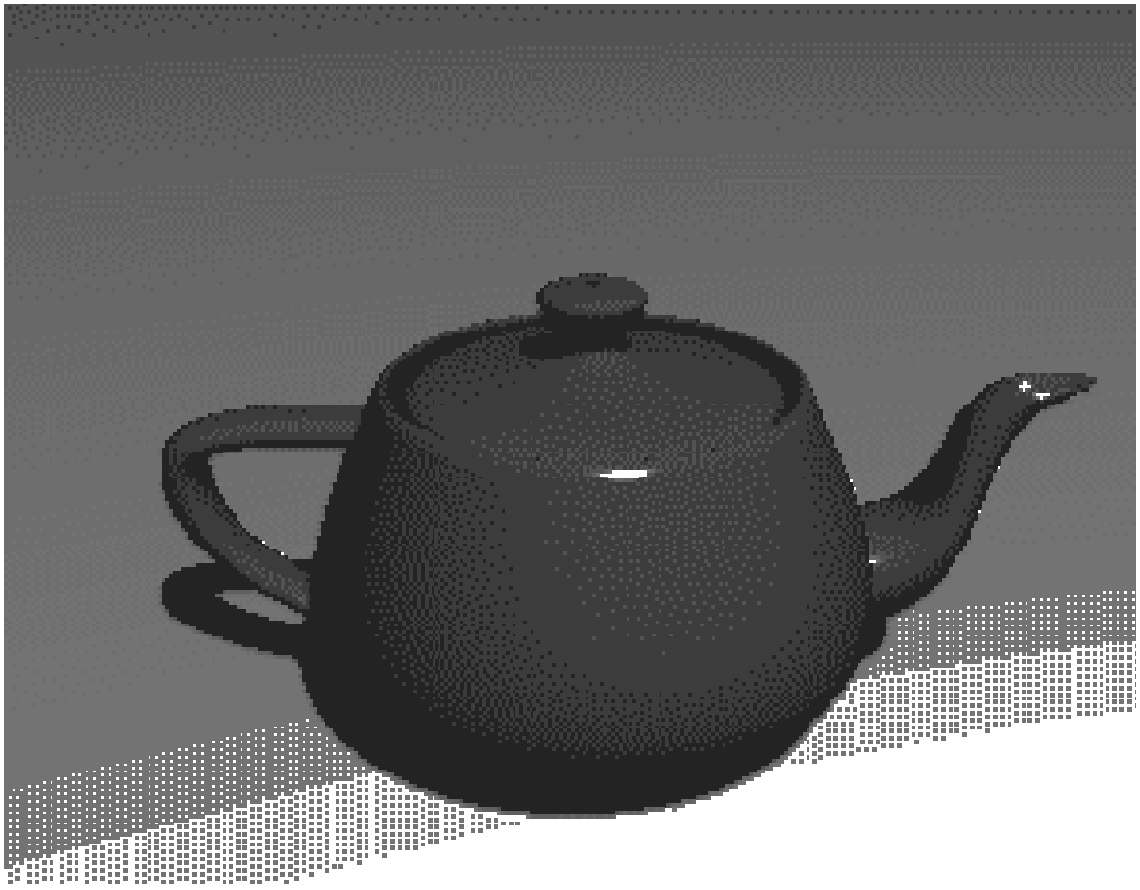
Growth sequence to minimize contour effects, i.e. the pattern are subsets of each other (hence the name ordered). They start in the center and expand towards the boundary.

Keep the one's adjacent to each other (hence the name clustered dot), but this is not necessary for monitors.

For high-quality reproduction we need an 8x8 (64 gray levels) or even a 10x10 matrix and thus quite a highest resolution.

If such a high resolution is not available one can use ERROR DIFFUSION: There we use fewer intensity levels than desired, but we distribute the errors, $(1-I_i)$ to the neighbouring pixels with the following weights:





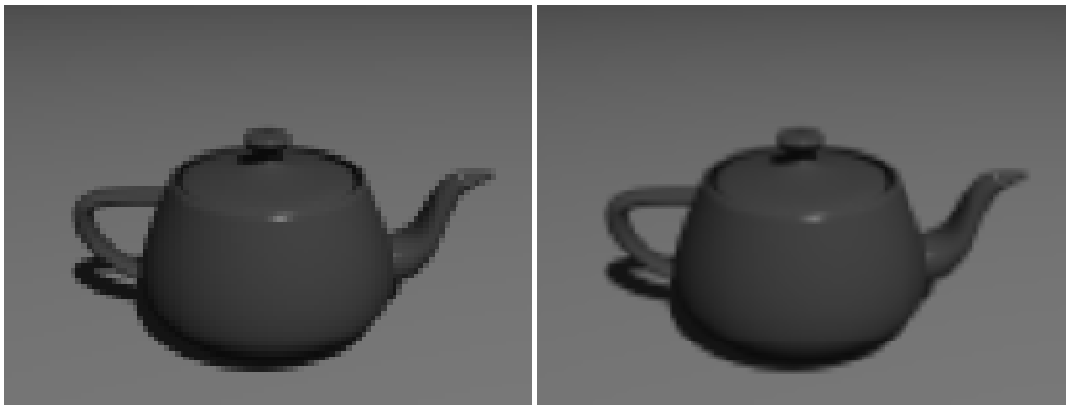
8 colors with Floyd-Steinberg error diffusion

A similar trick can be used for enlarging a picture by taking interpolation values to neighbouring pixels as intermediate values. E.g. for doubling the size of the picture one inserts new rows and columns and takes as new values

$$I'_{2i,2j} := I_{i,j} \qquad I'_{2i+1,2j} := \frac{1}{2}(I_{i,j} + I_{i+1,j})$$

$$I'_{2i,2j+1} := \frac{1}{2}(I_{2i,j} + I_{2i,j+1}) \qquad I'_{2i+1,2j+1} := \frac{1}{4}(I_{i,j} + I_{i,j+1} + I_{i+1,j} + I_{i+1,j+1})$$

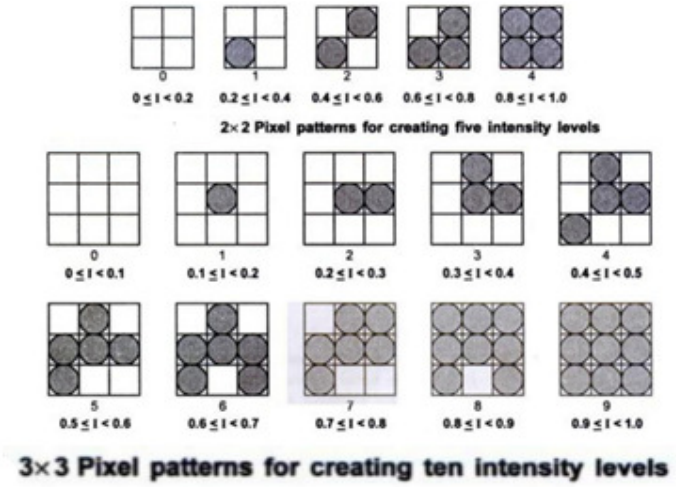
This way one avoids that small square-size pixels can be seen. Disadvantage is that some blurring occurs.



Enlarged versus enlarged with diffusion

Half toning- Main Points

- Many displays and hardcopy devices are bi-level
- They can only produce two intensity levels.
- In such displays or hardcopy devices we can create an apparent increase in the number of available intensity value.
- When we view a very small area from a sufficient large viewing distance, our eyes average fine details within the small area and record only the overall intensity of the area.
- The phenomenon of apparent increase in the number of available intensities by considering combine intensity of multiple pixels is known as half toning.
- The half toning is commonly used in printing black and white photographs in newspapers, magazines and books.
- The pictures produced by half toning process are called halftones.
- In computer graphics, halftone reproductions are approximated using rectangular pixel regions, say 22 pixels or 33 pixels.
- These regions are called halftone patterns or pixel patterns.
- Figure shows the halftone pattern to create number of intensity levels.



DITHERING TECHNIQUES

- Dithering refers to techniques for approximating halftones without reducing resolution, as pixel grid patterns do.
- The term dithering is also applied to halftone approximation method using pixel grid, and something it is used to refer to color halftone approximations only.
- Random values added to pixel intensities to break up contours are often referred as dither noise.
- Number of methods is used to generate intensity variations.
- Ordered dither methods generate intensity variations with a one-to-one mapping of points in a scene to the display pixel.
- To obtain n2 intensity levels, it is necessary to set up an n*n dither matrix Dn whose elements are distinct positive integers in the range of 0 to n2 −1.
- For example it is possible to generate four intensity levels with

$$D_2 = \begin{matrix} 3 & 1 \\ 0 & 2 \end{matrix}$$ and it is possible to generate nine intensity levels with

$$D_3 = \begin{matrix} 7 & 2 & 6 \\ 4 & 0 & 1 \\ 3 & 8 & 5 \end{matrix}$$

- The matrix element for D_2 and D_3 are in the same order as the pixel mask for setting up 22 and 33 pixel grid respectively.
- For bi-level system, we have to determine display intensity values by comparing input intensities to the matrix elements.
- Each input intensity is first scaled to the range 0< I < n2.
- If the intensity I is to be applied to screen position (x, y), we have to calculate numbers for the either matrix as

i=(xmodn)+1 j=(ymodn)+1

- If I>Dn(i,j) the pixel at position (x, y) is turned on; otherwise the pixel is not turned on.
- Typically the number of intensity levels is taken to be a multiple of 2.
- High order dither matrices can be obtained from lower order matrices with the recurrence relation.

$$D_n = \begin{matrix} 4D_{n/2} + D_2(1,1)u_{n/2} & 4D_{n/2} + D_2(1,2)u_{n/2} \\ 4D_{n/2} + D_2(2,1)u_{n/2} & 4D_{n/2} + D_2(2,\ 2)u_{n/2} \end{matrix}$$

- Another method for mapping a picture with mn points to a display area with mn pixels is error diffusion.
- Here, the error between an input intensity value and the displayed pixel intensity level at a given position is dispersed, or diffused to pixel position to the right and below the current pixel position

References

- R.C. Gonzales and R.E. Woods, Digital Image Processing, Addison-Wesley, Reading, 1992.
- J.D. Foley, A. van Dam, S.K. Feiner and J.F. Hughes, Computer Graphics, Principles and Practice, Addison-Wesley, Reading, 1990.
- Newton, Opticks, 4th Edition, Dover, New York, 1704/1952.
- E.B. Goldstein, Sensation and Perception, Brooks/Cole, California, 1989.
- http://www.mat.univie.ac.at/~kriegl/Skripten/CG/node11.html
- http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/OWENS/LECT14/lecture12.html
- http://www.mat.univie.ac.at/~kriegl/Skripten/CG/node6.html
- http://www.ques10.com/p/11466/explain-half-toning-and-dithering-techniques/