Illumination and Shading

Light reflected from nonluminous objects depends on:

- surface:
  - reflectance
  - transparency, opacity, translucency
  - orientation
- illumination:
  - location
  - intensity
  - wavelength
  - point-source, diffuse source
- can confuse the two
  - (see figure)

Shading model - calculates reflected and transmitted light for each small area of image

uses an illumination model

Illumination Models

E = resulting image intensity

a) Luminous objects

$E = k_i$

k_i is the object's intrinsic intensity

Compute just once for each object

b) Diffuse Illumination

diffuse, nondirectional light source with intensity I_a

$E_a = I_a k_a$, E_a is the illumination energy from diffuse light

k_a is the ambient-reflection coefficient for the surface
range of 0 to 1

uniformly illuminated - surface orientation not important

doesn't look natural (but can be natural!)

3D objects look like 2D

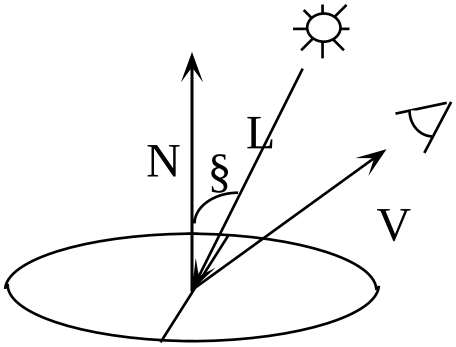
Natural conditions that give diffuse lighting?

c) Point Light Source

illumination varies with surface orientation
I_p is the illumination from a point light source

i) Lambertian Surface

reflect light with equal intensities in all directions



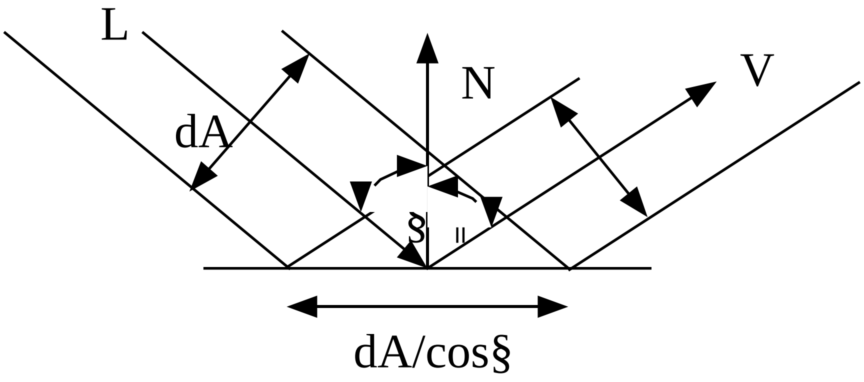
N is the surface normal
L is the direction of the light source
§ is angle between N and L
V is the viewing angle

For Lambertian surface:

$E_p = I_p k_d \cos §$
k_d is the lambertian reflection coefficient

snow, chalk, human skin, etc. - very matt surfaces

Why is Intensity just a function of § for Lambertian surface and
point light source?



For small area dA of light beam,
subtends area dA/cos§ on surface

Lambert's Law
Light reflected by Lambertian surface towards the viewer
is directly proportional to cos¶

Amount of surface seen is inversely proportional to cos¶

(Last two cancel each other)

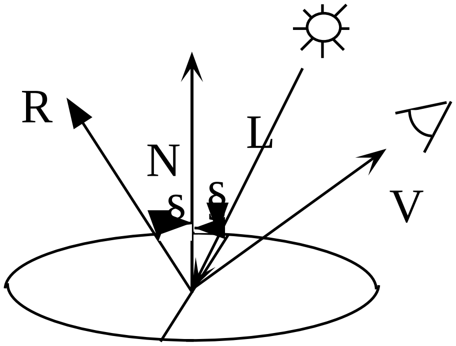
Possible range of §?

no self occlusion

Since angles are important in illumination and shading models,
must only use transformations on scene models which preserve
angles
therefore don't sheer or scale differently in x, y or z

c) Point Light Source, continued

ii) Perfect Reflector



R is the ray of the reflected light
Angle between L and N equals
angle between N and R

Must look in direction R to
see anything

Examples of perfect reflectors?

Most surfaces lie between perfect reflectors and perfect Lambertian

Example illumination with point source Lambertian surface
(see figure)

Add diffuse light source to it
(see figure)

Light source attenuation

Light energy falls off with inverse square of distance from source

$f_{att} = 1/(d^{2})$, where d is distance to light source

$f_{att} I_p$ is the attenuated iluminant

While correct, gives small variations if distance to light is large
compared to interscene distances

Often use other approximations

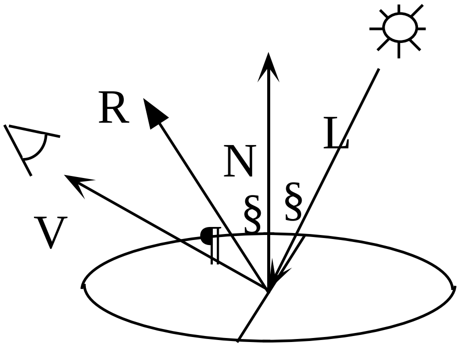
c) Point Light Source continued

iii) Specular reflection

gloss surfaces give varying degrees of specular reflection

highlights - color of illuminant

Beck's vases
(see figure)



Phong's model:

energy falls off by

$\cos^{n} \P$

n is the specular-reflection
exponent

n = infinity for perfect reflector
n = 1 is the miniumum

$E_p = W(\S) \cos^{n} \P \, I_p$

W(§) is the fraction of specularly reflected light (0 to 1)

Often set W(§) to a constant, k_s, the specular reflection coefficient

Not a function of color of object, but of the light source

Torrance-Sparrow model is more exact, but is more complex
to compute

Transparency

Illumination from behind for a transparent object

$E_t = T_p E_b$

E_b is the light coming from behind the object

T_p is the transmission coefficient of the object (0 to 1)

Total energy

$E = E_d + \cdot E_p + E_t$

Thus for each pixel in the image, compute all the energies and sum them to get the total reflected and transmitted energy

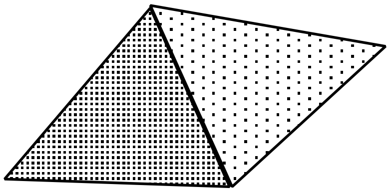
Very computationally expensive

Also must do it for each wavelength to account for color!

Shading Polygons

1) Constant Shading

Compute just one intensity for an entire polygon



When is this a valid shading model?

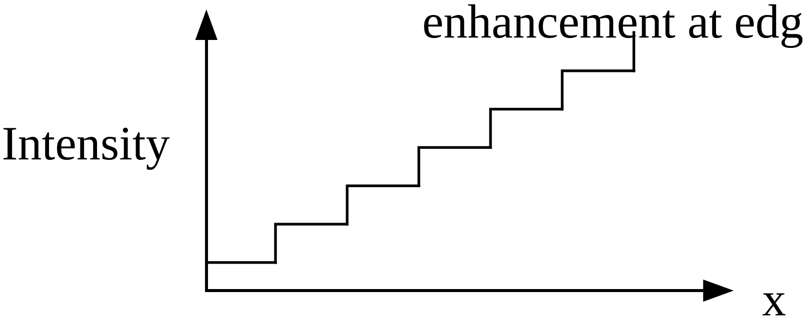
a) Light source is at infinity, thus angle between
N and L is a constant
b) Viewer is at infinity, thus angle between
N and V is a constant
c) Polygon represents actual surface
(not an approximation to a curved surface, etc)

Represent object with a polygon mesh

if use constant shading, it doesn't look constant

see Mach bands

lateral inhibition of eye's sensors does contrast enhancement at edges



Perceived intensity

(see figure)

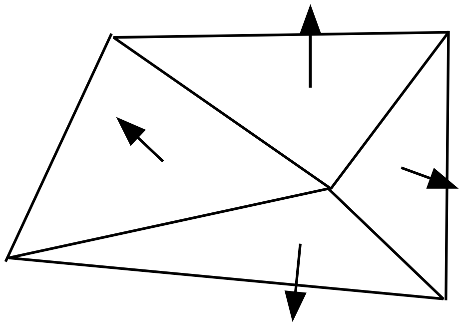
2) Gouraud Shading

Compute correct intensity at each vertex of polygon

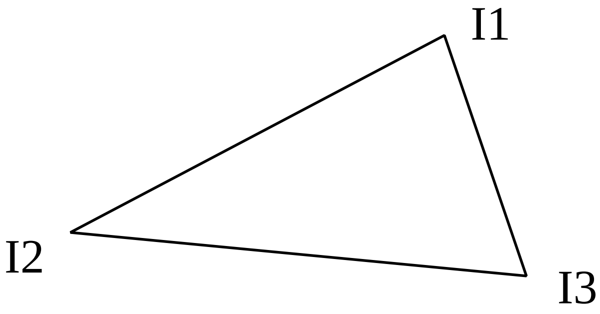
Need vertex normals

Get directly from analytical description

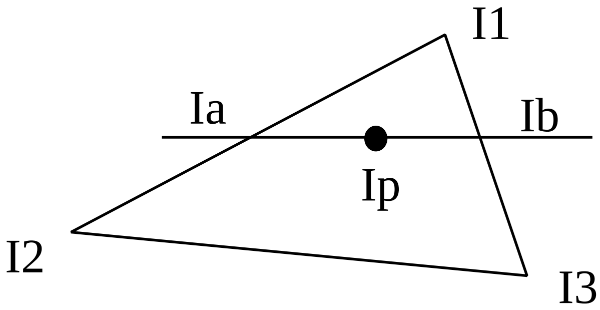
or, average the surface normals of the associated faces



Use vertex normals to compute surface intensity at each vertex



Interpolate intensities along each edge of polygon



Ia = I1-(I1-I2)(y1-yp)/(y1-y2)

Ib = I1-(I1-I3)(y1-yp)/(y1-y3)

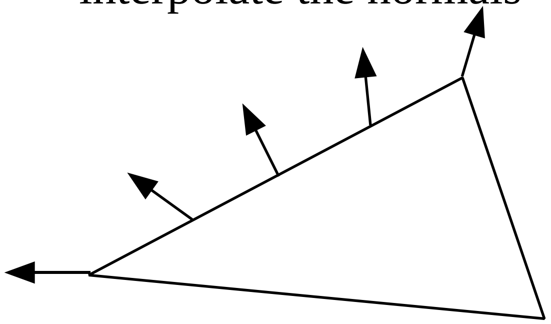
Interpolate along a scan-line

Ip = Ib-(Ib-Ia)(xb-xp)/(xb-xa)

(see plate)

3) Phong Shading

Instead of interpolating the intensities,
interpolate the normals



Interpolate normals along edges

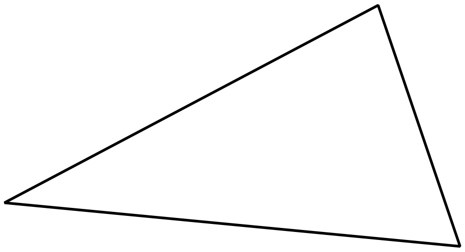
Interpolate normals along scan lines

Compute intensity for each point

Thus must do shading computation per point,
rather than per vertex

Phong better, especially for specular reflection

(see plate)



What happens if specular reflection should be in center

Gouraud?
If not seen at vertices, won't see it at all

Phong?
(see figure)

Problems with Interpolated Shading

a) Polygon silhouette

can't get rid of straight edges

(see plate)

partial solution: finer mesh

b) Perspective distortion

Must do interpolation in 3D screen coordinates
Thus perspective foreshortening already done
Can effect the shading

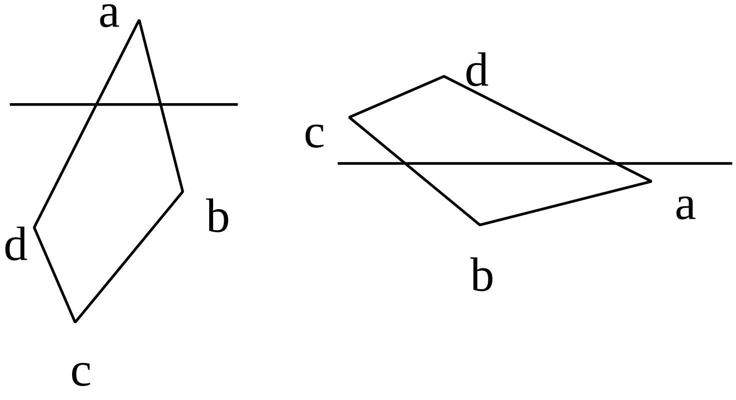
partial solution: finer mesh

c) Rotation dependent

Shading can change as rotate object, since done along a scan-line

Problem in animation

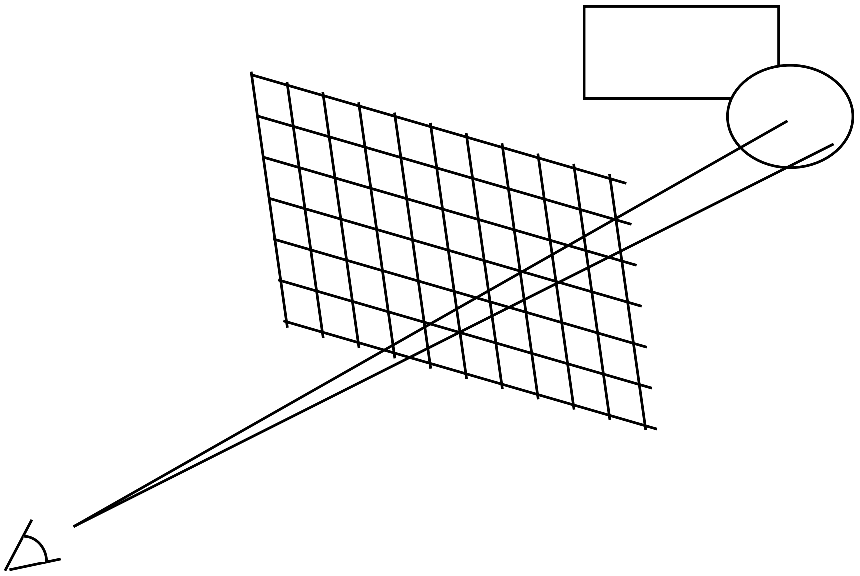
solution: use only triangular polygons



Ray Tracing

Image precision algorithm

For each pixel, trace ray from eye through pixel to scene



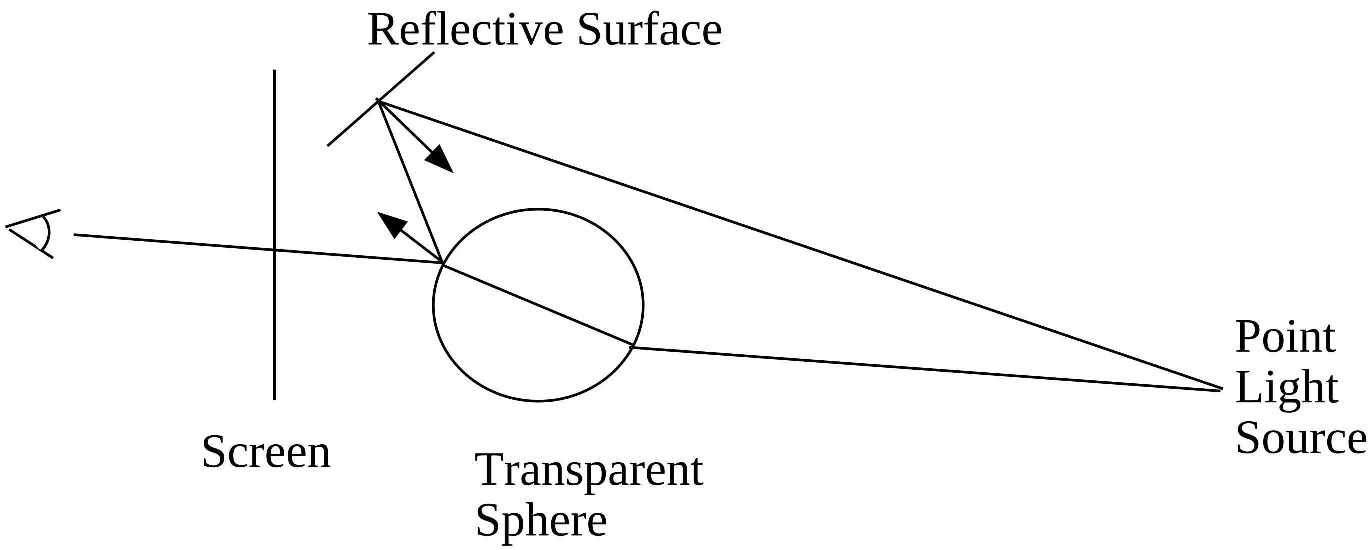
Then compute rays from each light source to the point on the object's surface

Figure which are shadow rays (intersect another surface before reaching object)

Compute the total light energy reflected back to the eye from all light sources

Uses a directionless ambient lighting term (difussed illumination)

Why not trace all rays from the light sources instead?



For sphere, get some reflection and some transmission

Need to add together all the energy in each color

Very time consuming!!

Ray Tracing

Point source light
Use ambient light to approximate Lambertian reflection

Radiosity

Better model of the physics
Includes Lambertian reflection between surfaces
Allows surfaces to emit light (luminous objects)

Break surfaces of image into small patches
Compute radiosity for each patch
(all light energy interaction for fixed illumination)
Use these radiosities when rendering as are viewer independent

Good for animation, as only compute radiosities once per scene