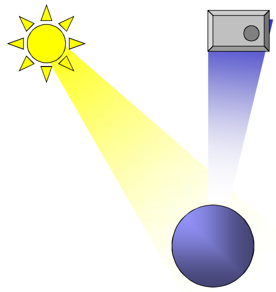
Illumination and shading

Adding “reality” to images
Light sources and lighting models
Surface properties
Shading algorithms
- Flat
- Gouraud
- Phong

Rendering: setting up the scene

- Given
  - Object surfaces
  - Light sources
  - Camera
- Compute
  - Colour of each pixel on the screen
  - This is colour that bounces off the surface point and goes in the direction of the camera (viewer)



Light

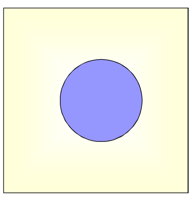
- Sources
  - Ambient
  - Directional: diffuse
  - Directional: point source
  - Divergence
- Location
  - w.r.t object
  - w.r.t. camera
- Colour √

Inputs to computation

- Light sources (emitters)
  - Colour (emission spectrum)
  - Geometry (position and direction)
  - Directional attenuation
- Surfaces (reflectors)
  - Colour (reflectance and absorption spectrum of the material)
  - Geometry (position, orientation of each surface patch)
  - Micro-structure

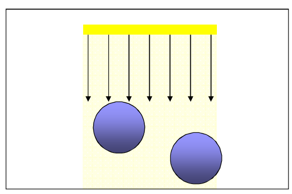
Light sources

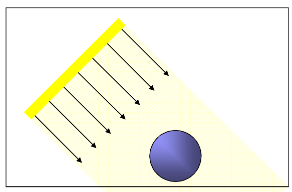
- Ambient light
  - Global, background light
  - No spatial of directional characteristics - seems to come from all directions
  - Makes objects visible
  - Does not depend on the orientation or position of a surface
  - Does not depend on the orientation or position of a camera
  - Does not have diffuse or specular reflection components



Light sources

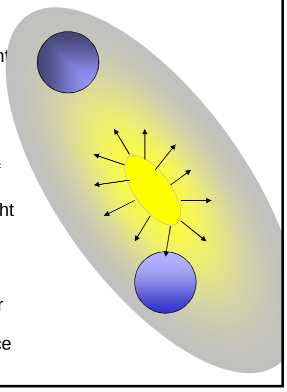
- Diffuse Light – parallel
  - Has parallel light rays that travel in one direction along the specified vector (e.g. sunlight)
  - Contributes to diffuse and specular reflections, which depend on the orientation of an object's surface (with respect to light direction vector) but not its position





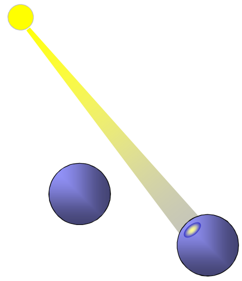
Light sources

- Diffuse Light – point source
  - Light source located at a fixed point in space (e.g. light bulb)
  - Radiates light equally in all directions away from the light source
  - Light is attenuated as a function of distance, i.e. its brightness decreases as distance from the light source increases
  - The rate of decrease is defined by an attenuation factor
  - Contributes to diffuse and specular reflections, which depend on the orientation and position of a surface



Light sources

- Spot Light
  - Light source located at a fixed point in space (e.g. light bulb)
  - Light radiating from the source forms a cone defined by a vector in a particular direction and by the angle determining its spread
  - Light is attenuated as a function of distance, i.e. its brightness decreases as distance from the light source increases (defined by an attenuation factor)
  - Contributes to diffuse and specular reflections, which depend on the orientation and position of a surface



Surfaces

Reminder

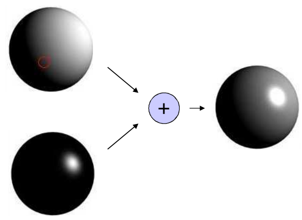
- Properties
  - Geometry (position, orientation of each surface patch)
  - Colour (reflectance and absorption spectrum of the material)
  - Micro-structure

Surfaces

Reminder

Micro-structure
- Defines reflectance properties

Reflectance
- Diffuse: Matte surfaces
- Specular: Shiny surfaces



Computing reflectance: shading model

- Requires
  - Surface geometry, microstructure and colour
  - Positions and type of light sources
  - Position of the viewer (camera)
- Combines the three contributions:
  - Ambient light
  - Diffuse reflectance
  - Specular reflectance

Pixel colour: Ambient + Diffuse + Specular

Shading model

Ambient term: colour

- Colour / intensity of ambient light: I_a
  - Colour is normally assumed white, i.e. I_a=[1.0, 1.0, 1.0];
- Object colour: ambient coefficient K_a
  - Represents the reflectivity of ambient light
  - $0 < K_a < 1$; 0: no reflectivity; 1: full reflectivity
- Complete ambient term: $\boxed{A = K_a I_a}$

Shading model

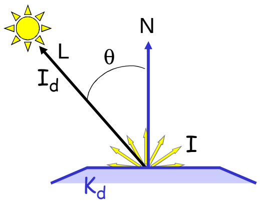
Ambient term: reflectance

- No physical interpretation
- Does not depend on light position
- Looks the same seen from anywhere

Increasing K_a

Shading model

Diffuse term: Reflectance



- Object surface: Lambertian (matte)
  - Light reflected equally in every direction
  - The amount of light reflected depends on the angle between the direction of light and a surface normal at each point
  - Defined by the "Cosine Law"

$$I = I_d \cos(\theta)$$

$$-\pi/2 < \theta < \pi/2$$

Shading model

Diffuse term: colour and reflectance

- Colour / intensity of diffuse light: I_d
  - Colour must be defined, $I_d = [r_d, g_d, b_d]$;
- Object colour: material diffuse reflectivity coefficient K_d
- Complete diffuse term: $\boxed{D = K_d I_d \cos \theta_d}$

Shading model

Diffuse term

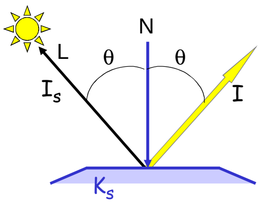
- Shading varies along surface
- A given point looks the same irrespective of the position of the viewer

Increasing K_d

Shading model

Specular term: Reflectance

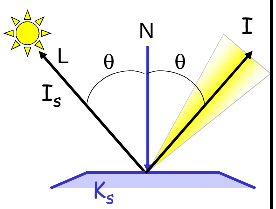
- Object surface: Mirror (ideal case)
  - Light reflected in accordance with the Snell's Law:
    The angle of reflection equals to the angle of incidence (with respect to the surface normal)
  - The amount of light reflected towards the viewer / camera varies, depending on the relative position of the light source, position of the viewer and surface orientation



Shading model

Specular term: Reflectance

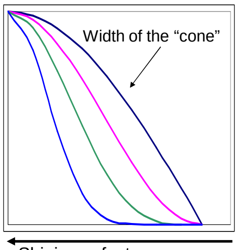
- Object surface: glossy
  - Extends the ideal (mirror) case
  - The reflected light forms a "cone" around the ideal (Snell-law) reflectance vector



Shading model

Specular term: Reflectance

- The amount of light reflected within the cone decreases from the centre outwards (as a function of exponential decay)
- The rate of decay n is a "shininess factor"
- Object colour: material specular reflectivity coefficient K_s
- Colour / intensity of specular light: I_s
  - Colour must be defined, $I_s = [r_s, g_s, b_s]$;
  - Common assumption: $I_s = I_d$

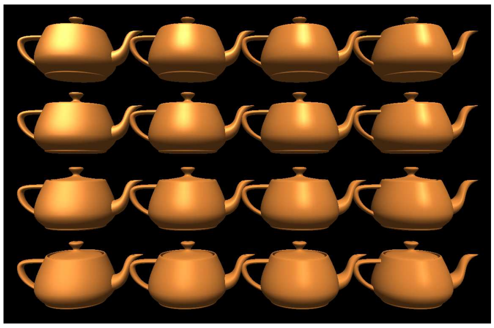


Complete specular term: $\boxed{S = K_s\, I_s\, (\cos\theta_s)^n}$

Shading model

Specular term

Increasing K_s



Increasing n

A complete shading model

Combines all the terms:

Pixel colour: Ambient + Diffuse + Specular

$$I = A + D + S$$

$$\boxed{I = K_a\, I_a + K_d\, I_d \cos\theta_d + K_s\, I_s\, (\cos\theta_s)^n}$$

where $\cos\theta_{s/d} = \underline{N \cdot L_{s/d}}$

dot product

Algorithms for shading of surfaces

- Shading model so far showed how to compute reflectance for individual points on a surface
- Shading varies across surfaces
- Point-by-point computation very expensive
- Three approaches for computing shading for polygonal surfaces
  - Flat shading
  - Gouraud shading
  - Phong shading

Flat shading

- One reflectance value per polygon surface
- Advantages
  - Computationally simple
- Drawbacks
  - Not very realistic for curved surfaces
  - Polygon structure visibly obvious

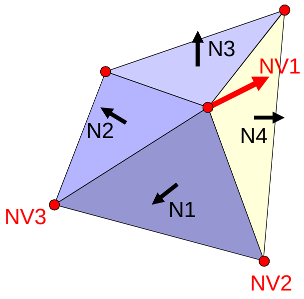
Gouraud shading

- Interpolates the reflectance (colour) across the surface of each polygon from a subset of points for which reflectance has been computed from the model
- Advantages
  - Visually more realistic than flat shading
- Drawbacks
  - Some artifacts may still be visible
  - Computationally more expensive than flat shading

Gouraud shading

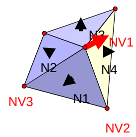
- Input
  - Surface normals
- Algorithm

1. For each surface calculate vertex normals by interpolating surface normals



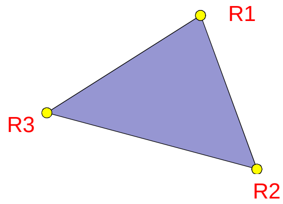
Gouraud shading

- Algorithm



2. Compute reflectance at all the vertices

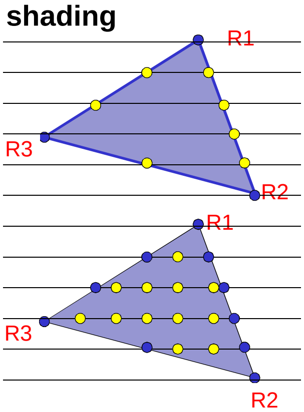
$$I = K_a I_a + K_d I_d \cos \theta_d$$
$$+ K_s I_s (\cos \theta_s)^n$$



Gouraud shading

- Algorithm

3. Compute reflectance for points on the edges for every scan-line by interpolating vertex intensities
4. Compute reflectances for every point within the patch of surface by interpolating along each scan-line between edge reflectances



Phong shading

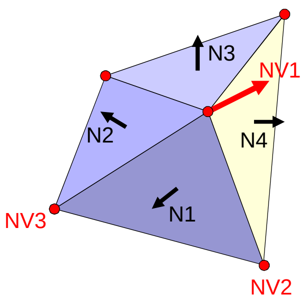
- Interpolates the surface normals across the surface of each polygon from a subset of points for which the normal has been computed from the model

- Advantages
  - Visually more realistic than Gouraud shading
- Drawbacks
  - Computationally more expensive than Gouraud shading

Phong shading

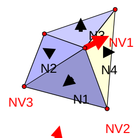
- Input
  - Surface normals
- Algorithm

1. For each surface calculate vertex normals by interpolating surface normals

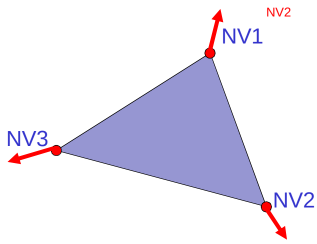


Phong shading

- Algorithm



1. For each surface calculate vertex normals by interpolating surface normals

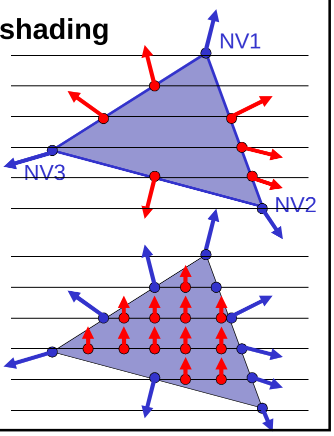


Phong shading

- Algorithm

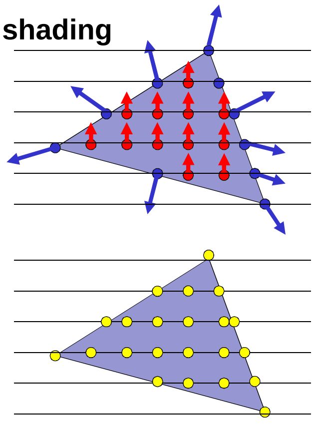
3. Compute normal vectors for points on the edges for every scan-line by interpolating vertex normals

4. Compute normal vectors for every point within the patch of surface by interpolating along each scan-line between edge normal vectors



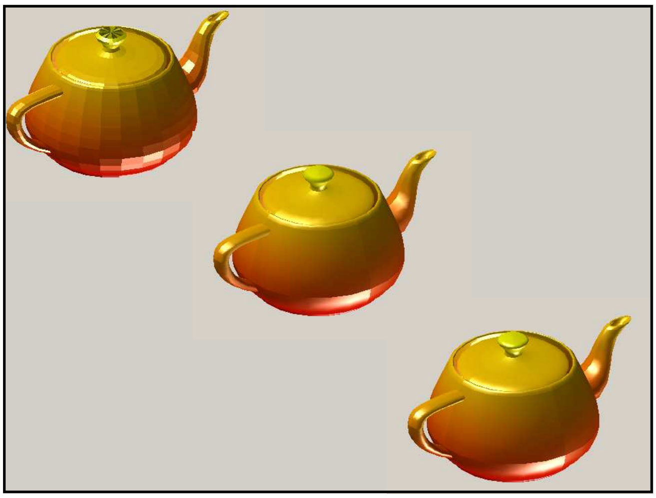
Phong shading

- Algorithm



5. Compute surface reflectance directly from normal vectors for every point within the patch of surface

$$I = K_a I_a + K_d I_d \cos \theta_d$$
$$+ K_s I_s (\cos \theta_s)^n$$



Technical issues

- Normal vectors must be normalised (i.e. length always set to 1) (WHY?)
- After interpolation vectors must be re-normalised
- Transforming normal vectors (e.g. after change of the view or after changing position of an object) is not straighforward
- see e.g. http://groups.csail.mit.edu/graphics/classes/6.837/F01/Lecture15/lecture15.ppt

Credits

- This presentation has used slides from various web sources, including:
  - www.classes.cec.wustl.edu/~cse452/lectures/lect11_Illumination_2pp.pdf
  - http://www1.cs.columbia.edu/~cs4160/slides/lecture15.ppt#767,2,Rendering: 1960s (visibility)
  - groups.csail.mit.edu/graphics/classes/6.837/F01/Lecture15/lecture15.ppt
  - http://artis.imag.fr/~Nicolas.Holzschuch/cours/class9.pdf

Homework

- A surface is of a uniform red colour.

  Given two vertices at
  - V1 = [-80 00 58]
  - V2 = [-65 -47 58]

  their vertex normals
  - N1 = [-0.80 -0.04 0.60]
  - N2 = [-0.65 -0.50 0.60]

  a vector specifying the direction of light
  - [-0.30 -2.20 2.80]

  and light colour vector
  - [1 0.5 0.5]

  compute the colours (RGB vectors) of the 10 points lying on the line joining the two vertices V1 and V2