Computer Graphics: 3-Local Illumination Models

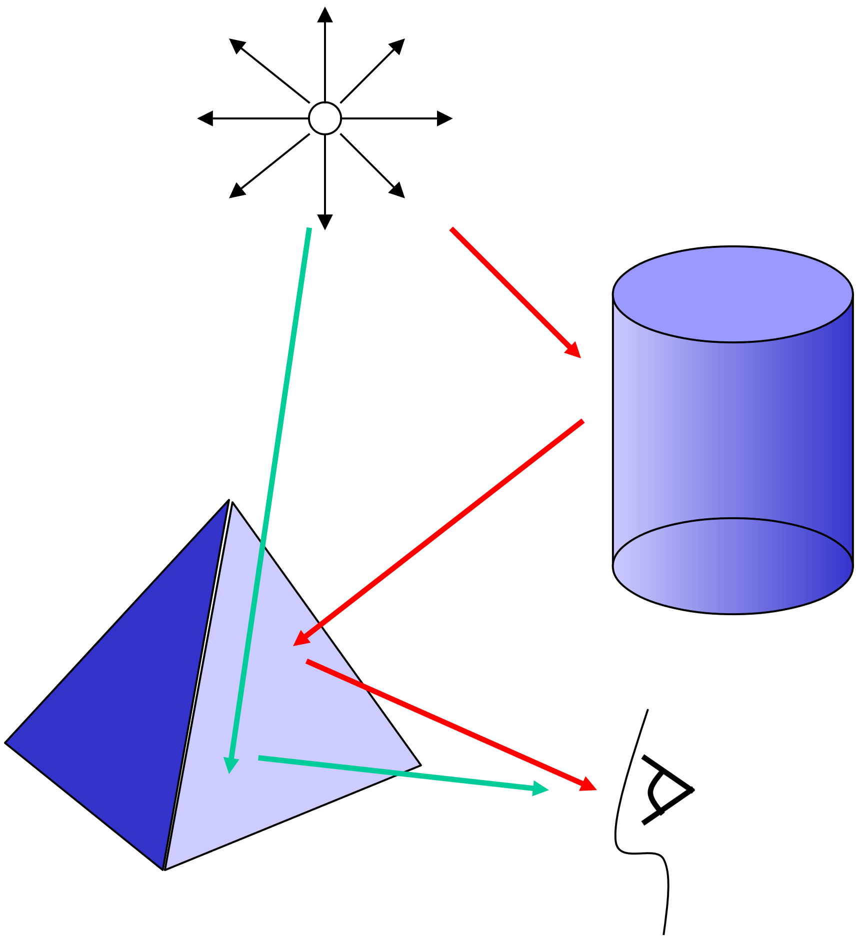
Prof. Dr. Charles A. Wüthrich,
Fakultät Medien, Medieninformatik
Bauhaus-Universität Weimar
caw AT medien.uni-weimar.de

Introduction

- After having illustrated how to draw polygons onto the screen, it is time to deal with computing the exact color that should be used.
- In Computer Graphics, we mostly have to deal with surfaces
- The aim of producing photorealistic images is to simulate the interaction between light and the surfaces in a 3D environment
- We will first develop the models for achromatic light, and successively see how they can be extended to deal with colour
-
-

Illumination models

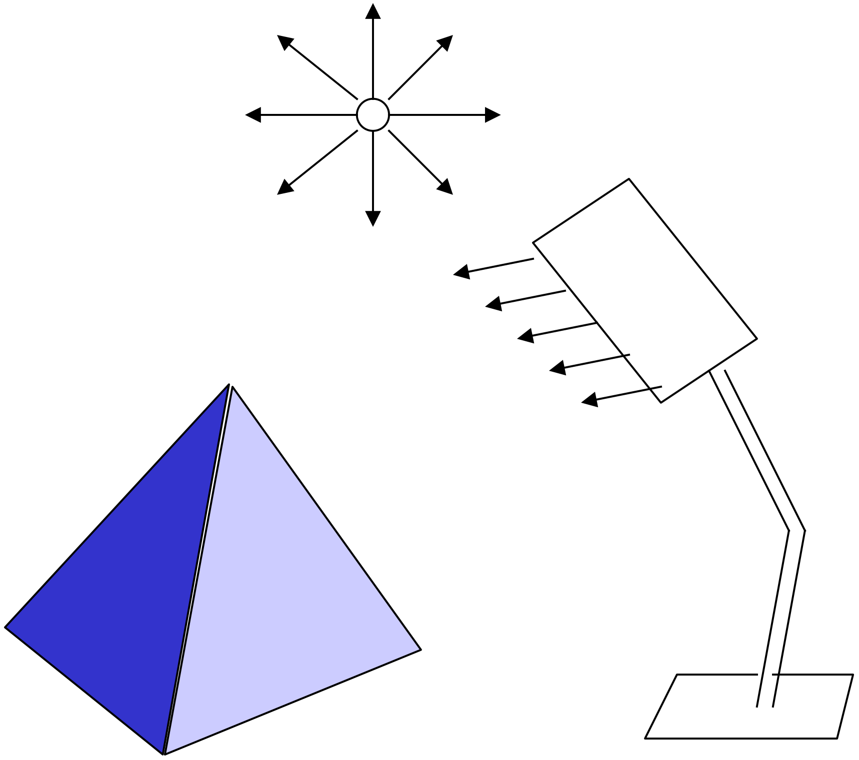
- There are two types of illumination models
  - Local illumination models:
    Light reflected by a surface (and therefore its colour) is dependent only on the surface itself and the direct light sources
  - Global illumination models:
    Light reflected by a surface is dependent only on the surface itself, the direct light sources, and light which is reflected by the other surfaces on the environment towards the current surface



- We will do global models later in the course

Lights

- To illuminate a surface, light is needed
- Two major models for light sources
  - Point light sources
    - By putting point light source at infinity one can simulate solar light
    - Diffusion cone can be restricted to simulate spotlights (use additional filter function for dimming light)
  - Area light sources (distributed)



Illumination models

- For any object in the environment, a shading function describing how its surface reacts to light is necessary
- For the enviroment, an illumination model is used, which determines what parts of the shading functions are used while rendering the scene
- This illumination model is expressed through an illumination equation
- Given a surface and an illumination model, through the illumination equation the color of its projected pixels on the screen can be computed
- For local illumination, the Illumination Equation is composed of different terms, each adding realism to the scene

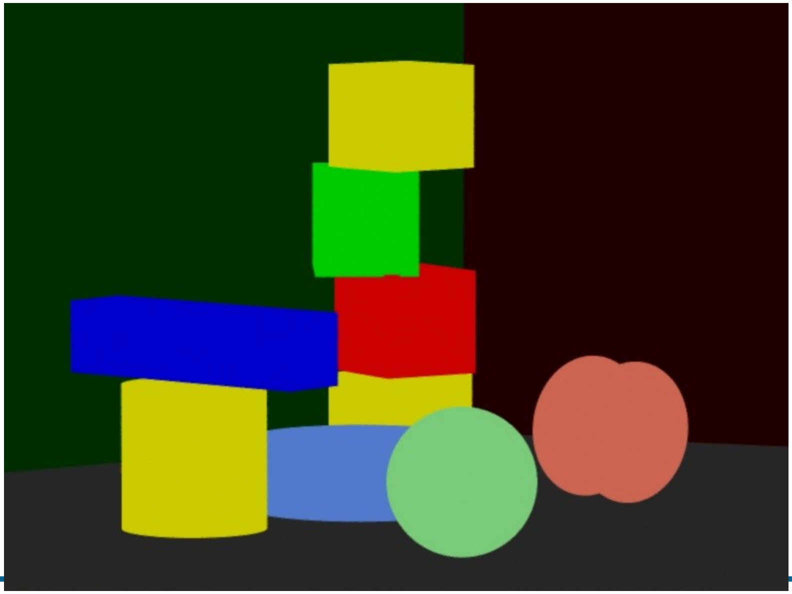
Ambient light

- In local illuminaton models, ambient light is used to model the light that does not come directly from a light source
  - i.e. under the table
- This is the basic colour of an object, to which the other components will be added

- Illumination equation:

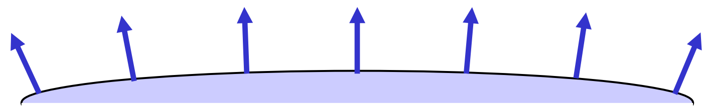
  $$I=k_a I_a$$

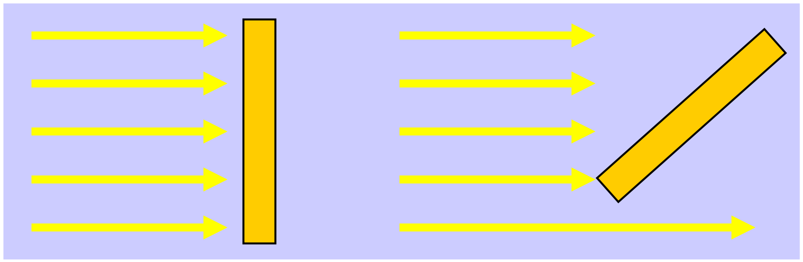
  where
  - k_a: says how much of ambient light is reflected by the object (ϵ [0,1])
  - I_a: Instensity of ambient light, equal for whole environment



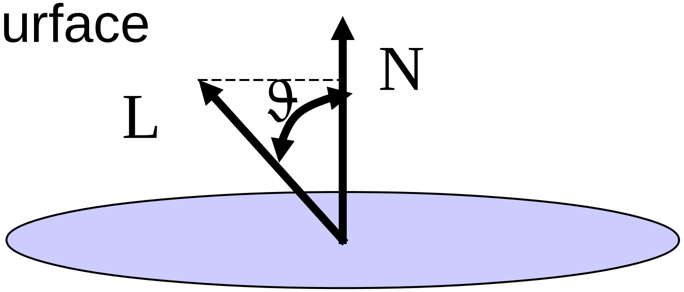
Diffuse reflection: Lambert Law

- Suppose one has a directed light source (point or sunlight)
- Some materials reflect the light equally in all directions
  - Example: chalk
- Lambert observed that the more the incident angle to a surface parts from the surface normal, the darker the colour of the surface
- He also noticed, that objects show the same intensity even if the viewer is moving around
- The reason is that viewers percieve the same amount of light per angle on the retina, no matter what their viewing angle





- Reflected light intensity must therefore be dependent from the projection of the light vector onto the normal to the surface



Diffuse reflection: Lambert Law

- Thus, the total reflected light is given by

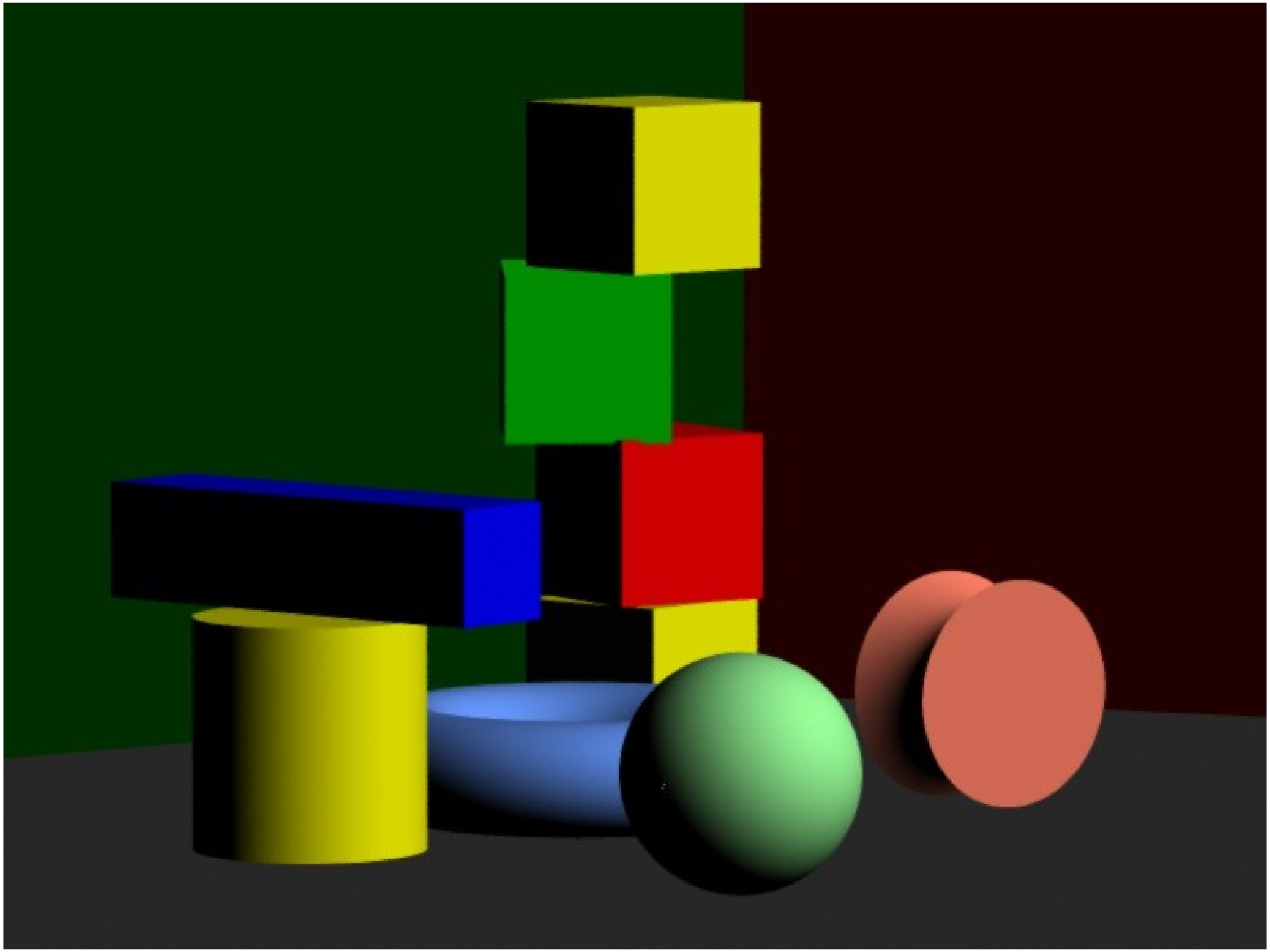
$$I_{diff,Lam}=I_P k_d \cos\vartheta$$

where

  - I_P: Intensity of incident light at surface
  - k_d: diffuse reflection coefficient of the material Î [0,1]
  - ϑ: angle between normal to surface and direction of incident light
- Or, by using normalized vectors N and L

$$I_{diff,Lam}=I_P k_d (N \cdot L)$$

Diffuse reflection: Lambert Law



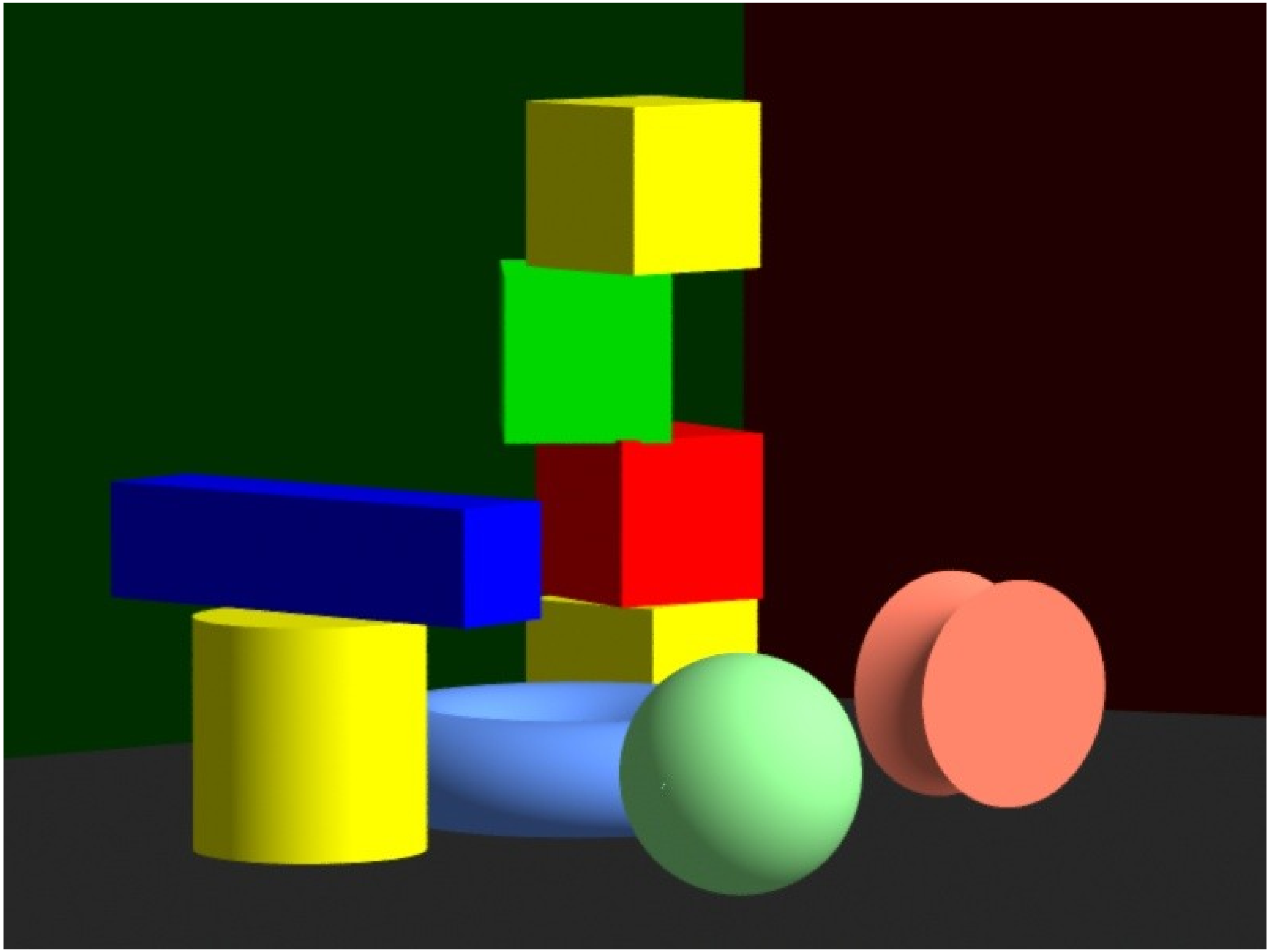
Ambient + Diffuse Illumination

- The two lighting models presented above can be combined

$$I_{diff} = I_a k_a + I_P k_d \cos\vartheta$$

to obtain an illumination equation that encompasses both illumination methods presented

Ambient + Diffuse Illumination



Light source attenuation

- In fact, light does not travel through space keeping its illuminating power at the same level.
- Farther objects get less light than closer ones, because it is partially absorbed by particles in the air
- If constant light intensity is used, then one would get a kind of illumination which is similar to the sunlight
- To solve this problem, an attenuation factor is added to the illumination equation, decreasing light intensity with distance from the light source

- The resulting illumination equation is

$$I = I_a k_a + f_{att} I_p k_d \left(\bar{N} \cdot \bar{L} \right)$$

where the attenuation factor is:

  – $$f_{att} = \frac{1}{d_L^2}$$

  which gives a too hard decay of light

  – or $$f_{att} = \frac{1}{Max\left(\left| c_1 d_L^2 + c_2 d_L + c_3 \right|, 1 \right)}$$

  where the coefficients are chosen ad hoc

Adding colour

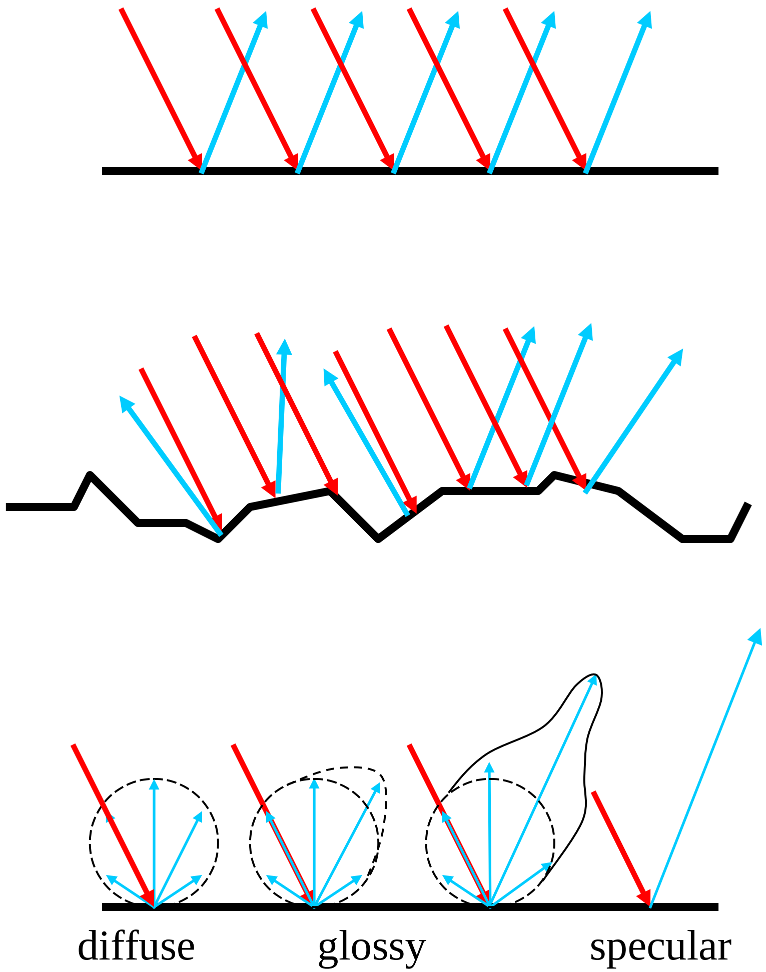
- In order to add colour, the computation of the illumination function is repeated three times, one for each of the colour components RGB.
- The illumination equation is therefore repeated for each one of the three components, and colur components are computed separately
- In general, for a wavelength λ in the visible spectrum, we have that

$$I_{\lambda} = I_{a\lambda} k_a O_{d\lambda} + f_{att} I_{p\lambda} k_d O_{d\lambda} \left(\overline{N} \cdot \overline{L} \right)$$

- In the usual case, one computes one of these each for R,G,B

Specular highlights

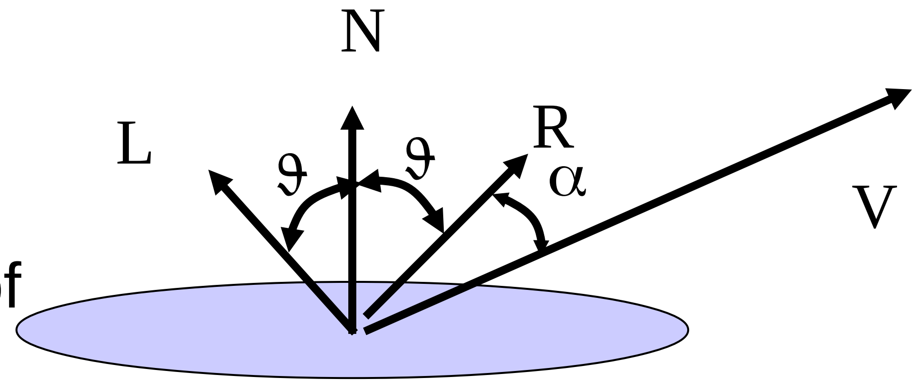
- In real life, most surfaces are glossy, and not matte
  - Think for ex. at an apple
- Gloss is due to the non-plain-ness at the microscopic level of surfaces (microfacet theory) , and do micro reflecting surfaces of the material
- Since their distribution is approximately gaussian, then most mirror like reflected light is reflected in direction of the specular reflection, and some is scattered in other directions
-

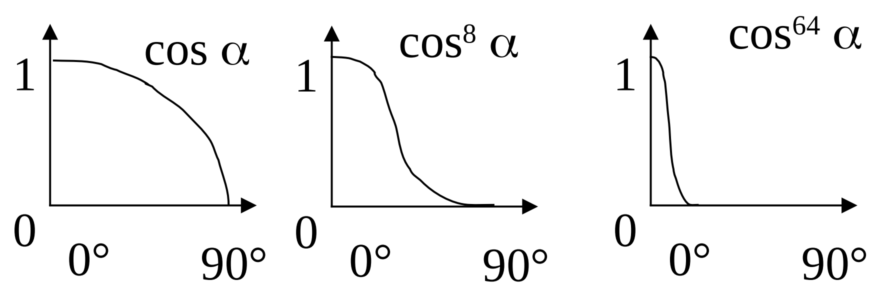


Specular highlights

- Around the direction of the viewer, reflection rays are scattered and generate a highlight

Empirically, scattering decay from the direction of the viewer can be seen to behave similarly as the power of the cosinus of the angle α





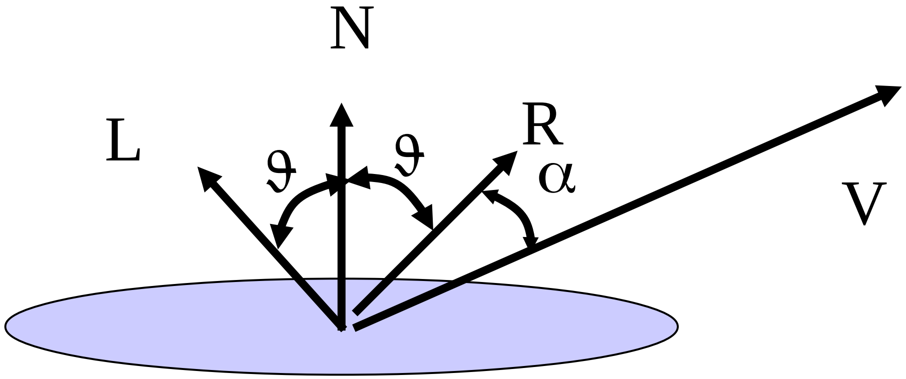
Specular light

- Heuristic model proposed by Phong Bui-Tuong
- Add term for specular highlight to equation

$$I_\lambda = I_{a\lambda} k_a O_{d\lambda} + f_{att_i} I d\lambda_i [k_d O_{d\lambda} (\overrightarrow{N} \cdot \overrightarrow{L}) + k_s O_{s\lambda} (\overrightarrow{R} \cdot \overrightarrow{V})^n]$$

where

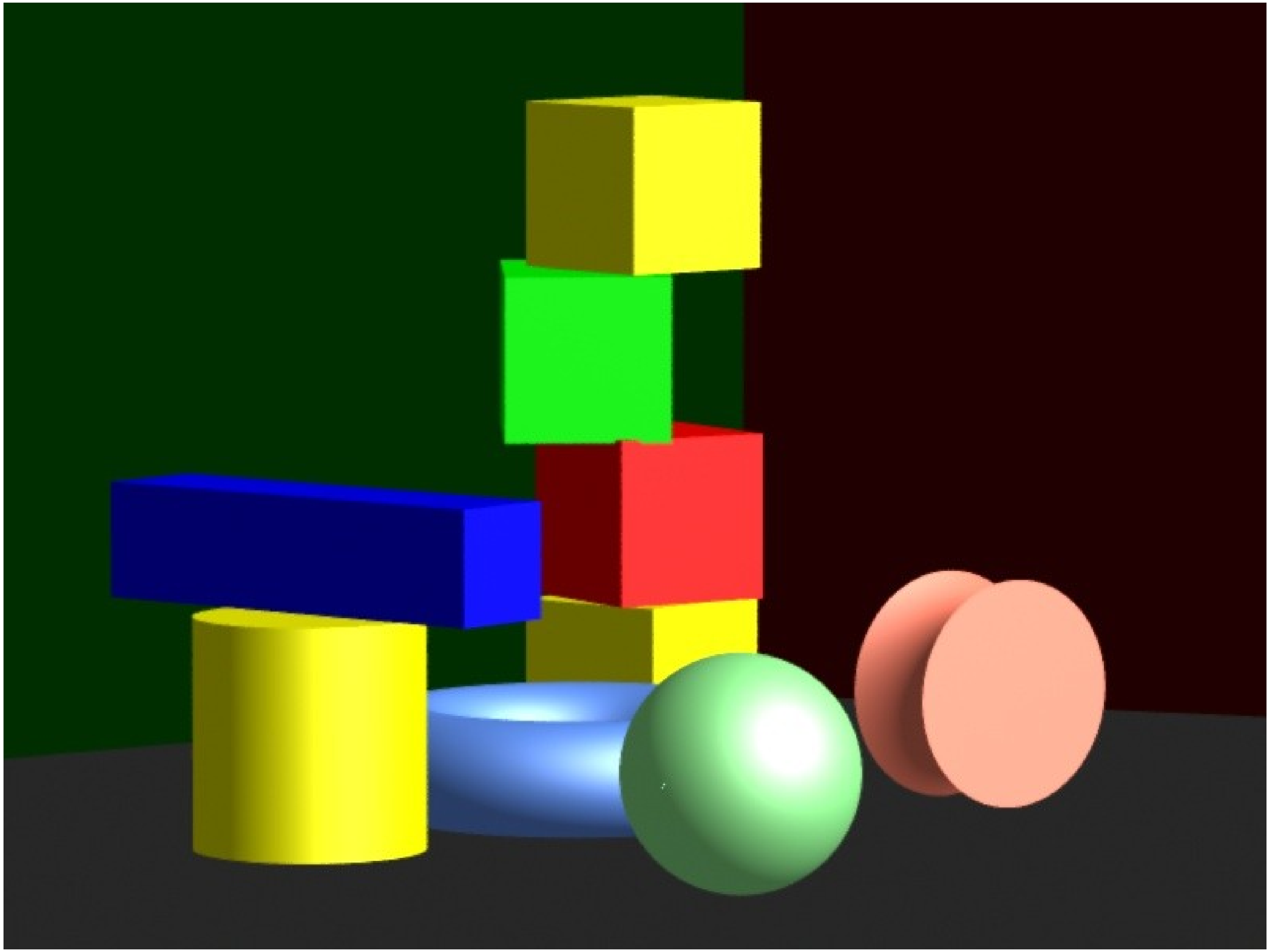
- k_s= specular reflection constant of surface
- $(R \cdot V)^n = \cos^n \alpha$



Specular highlights

- Applying Phong illumination allows to obtain quite convincing images
- Note: these illumination models are implemented in hardware nowadays

Specular highlights



Multiple lights

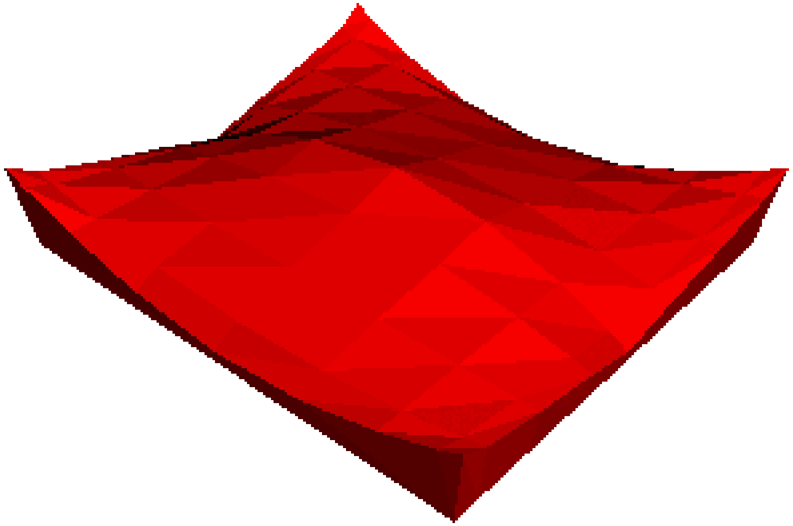
- Adding multiple light sources to the illumination model in an environment is simple:
  - Just add contributions due to the single lights
  - This, of course, doubles the comptations in the case of two lights

Shading polygonal models

- In theory, given the Illumination equations, every surface is representable.
- For each surface pont, just compute illumination at that point, and map it to the screen
- However, this is mostly very computationally intensive
- In the last step before rendering, surfaces are therefore transformed into polygonal models by discretizing their surface
- The problem by polygonal models is that the elements are flat

Flat shading

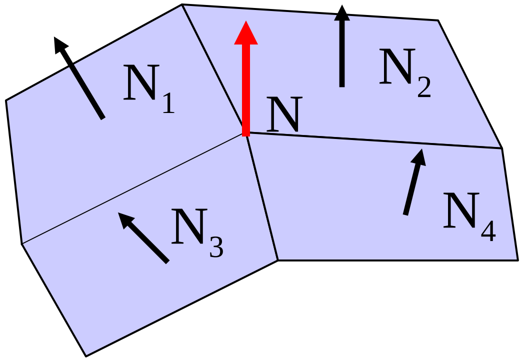
- For each polygon in the mesh, compute illumination equation
- Render the whole polygon with the obtained colour (using the scanline algorithm for example)
- Results are already good for giving an idea of the shape
- However, all polygonal facets are seeable, and this is mostly unwanted

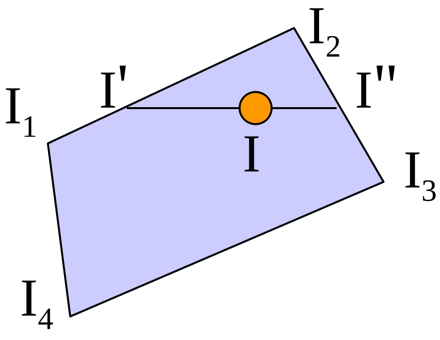


Courtesy Stephen Chenney,
University of Wiscounsin at Madison

Gouraud shading

- To avoid this problem, Gouraud in 1971 proposed a method to smooth polygonal surface rendering
- The idea is to
  - compute normal vectors at vertices of each polygon (average adjacent polygon normals $N=N_1+N_2+N_3+N_4$)
  - compute illumination at vertices
  - linearly interpolate colour values along the edges of the polygons
  - linearly interpolate colour values between edges to find out colour at a given point
- Bilinear interpolation

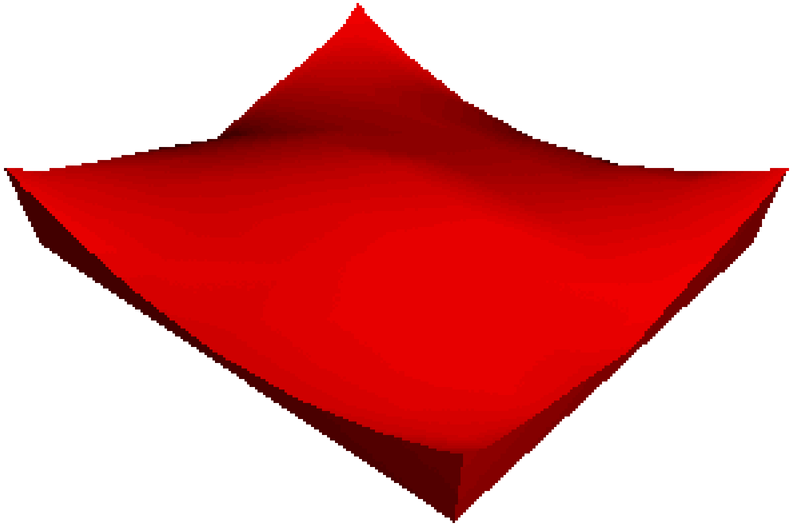




$$I' = tI_2 + (1-t)I_1$$
$$I'' = tI_2 + (1-t)I_3$$
$$I = tI' + (1-t)I''$$

Gouraud shading

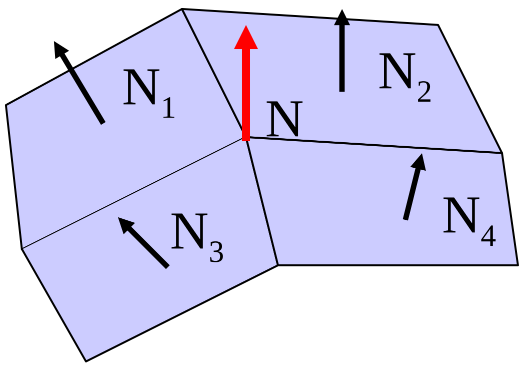
Courtesy Stephen Chenney,
University of Wisconsin at Madison

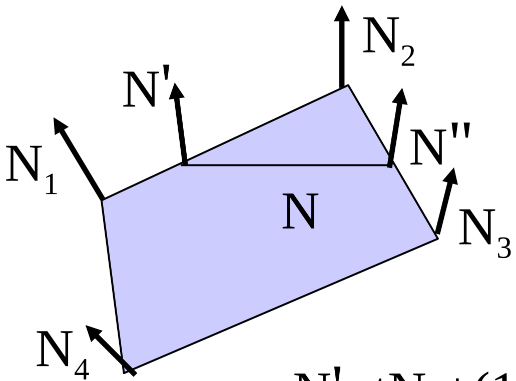


- Gouraud shading gives smooth surfaces
- Sometimes highlights are missed

Phong shading

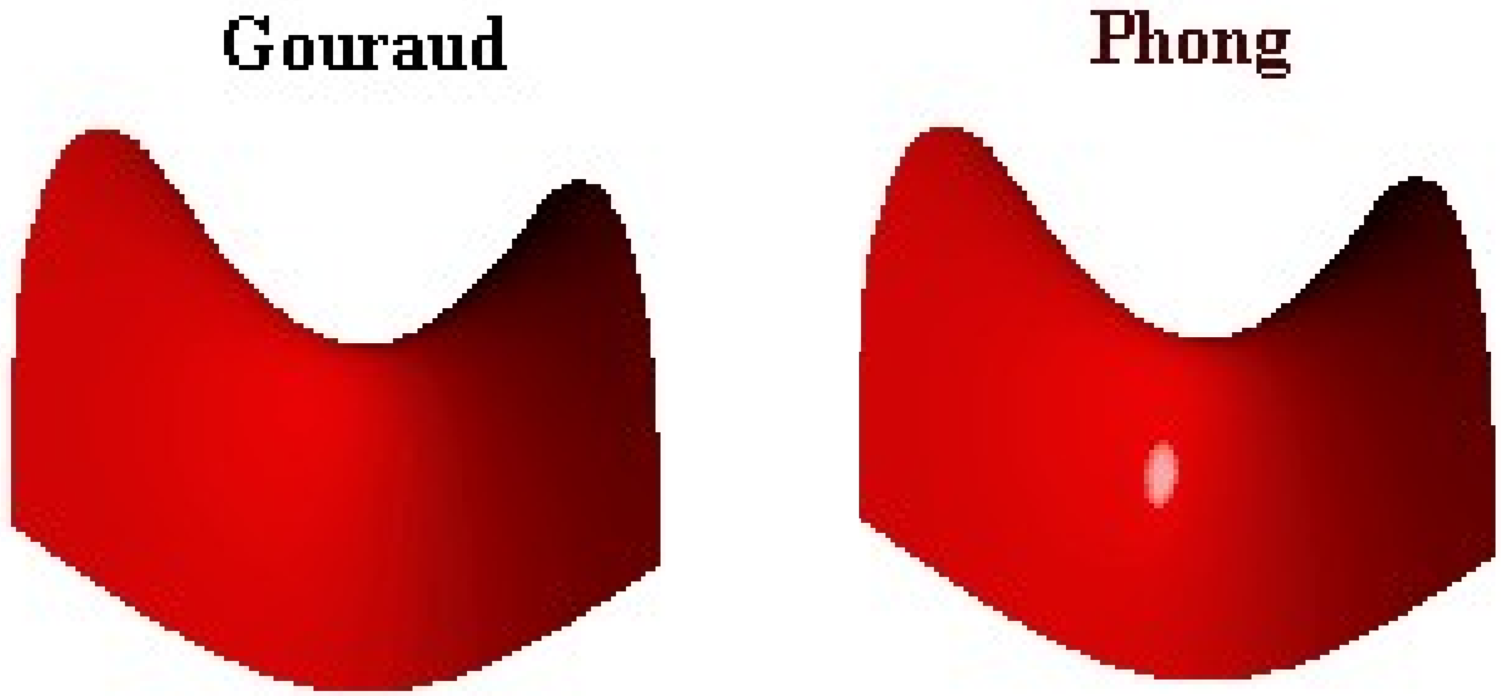
- Phong proposed an improvement to Gouraud‘s idea
- Phong interpolates between normals and not between colour to find the normal at a point inside the polygon
- Note that both interpolations (Phong, Gouraud) have three components to compute
- Only once known the interpolation normal at the point, illumination is computed at each point
- Much more computationally intensive than Gouraud





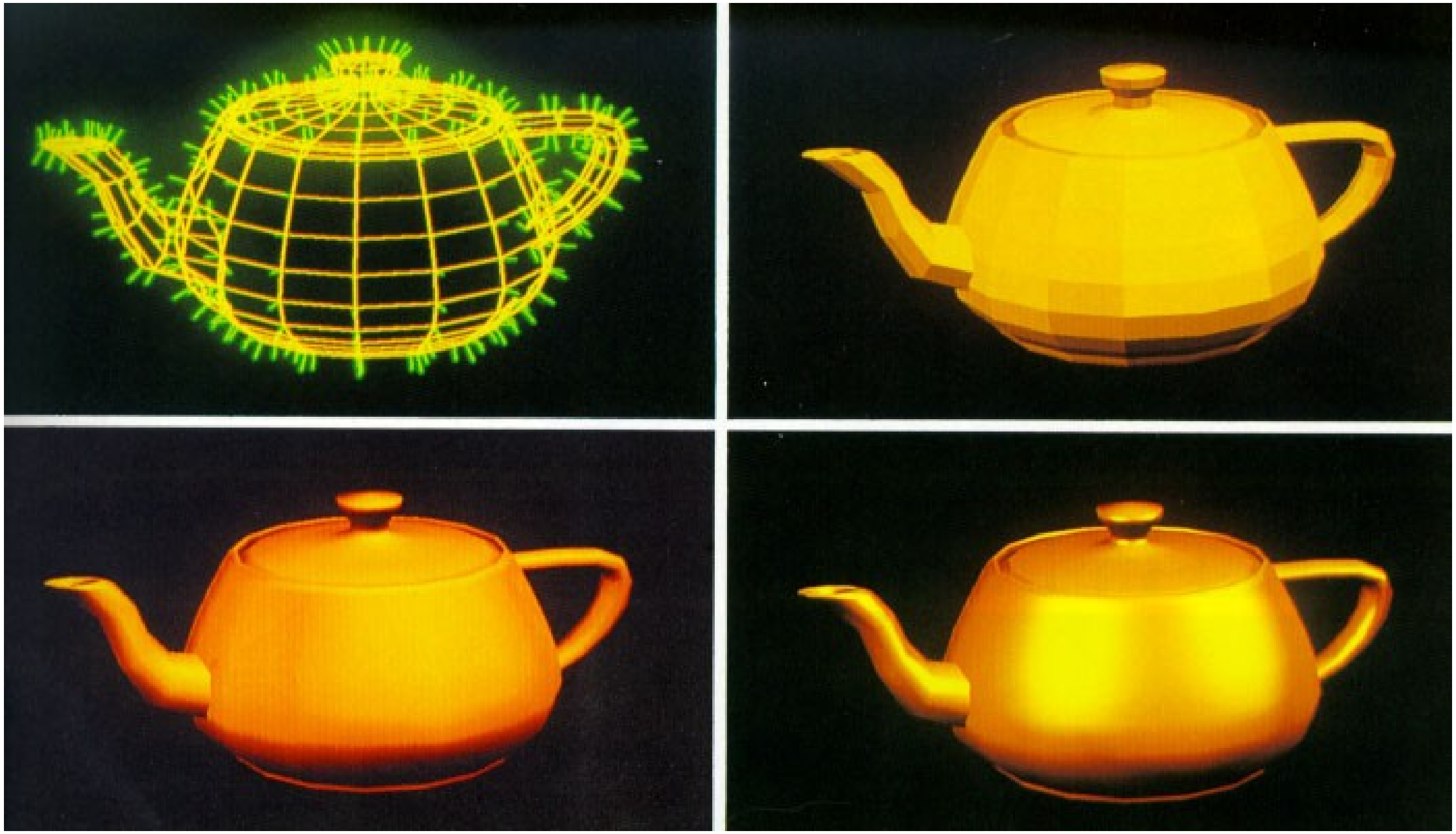
$$N' = tN_2 + (1-t)N_1$$
$$N'' = tN_2 + (1-t)N_3$$
$$N = tN' + (1-t)N''$$

Comparing shading methods



Courtesy Stephen Chenney,
University of Wisconsin at Madison

Comparing shading methods



End

+++ Ende - The end - Finis - Fin - Fine +++ Ende - The end - Finis - Fin - Fine +++