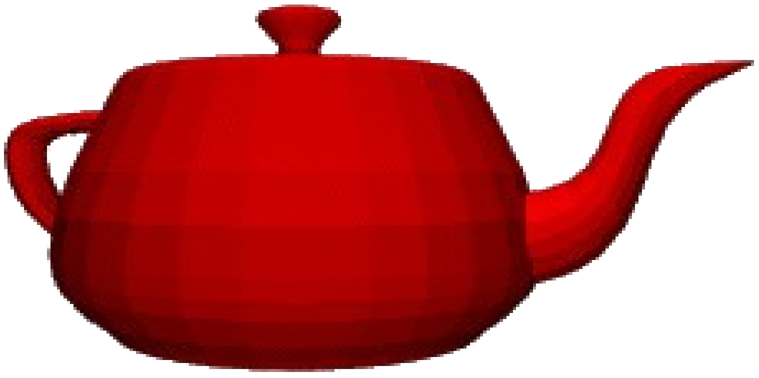
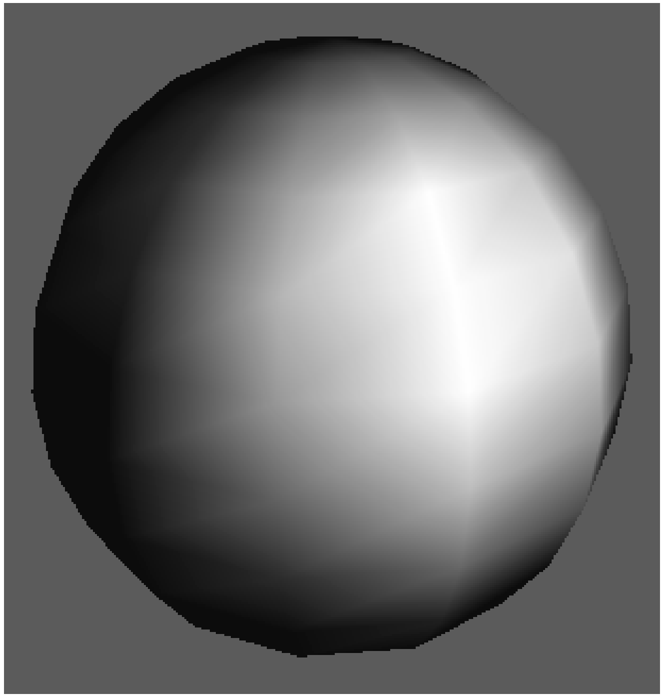
Osvetljevanje in senčenje

(lighting - shading)



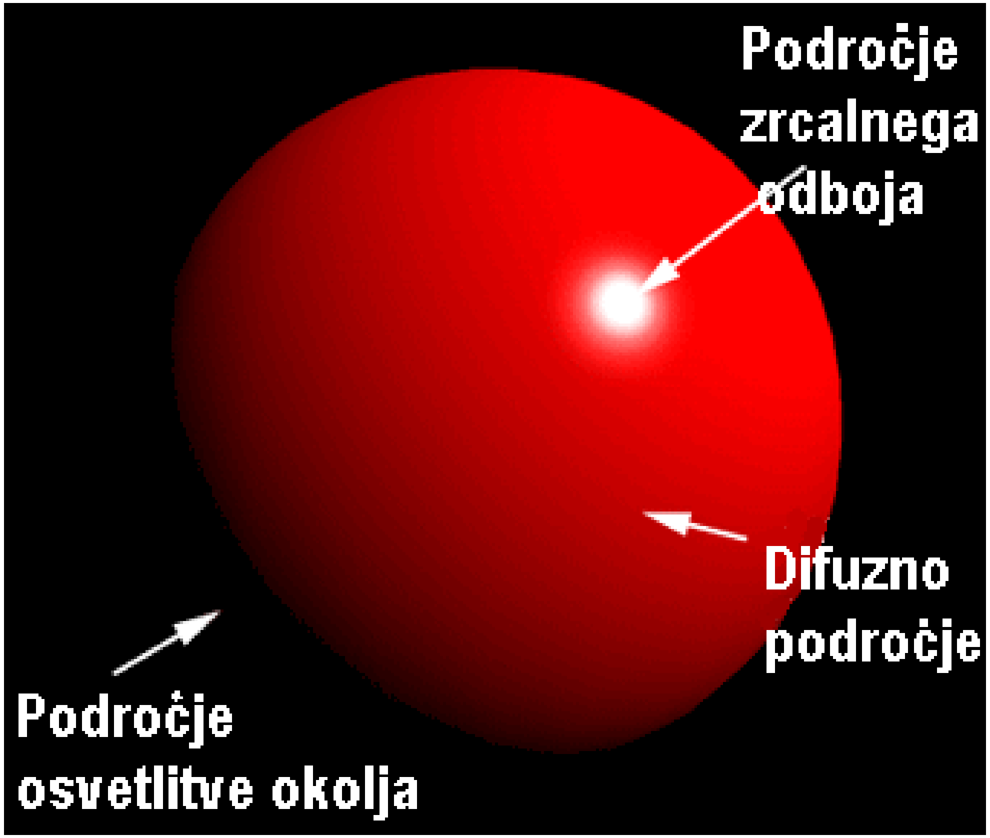
“Osvetljevanje”

- Two components:
  - **Lighting Model** or **Shading Model** - how we calculate the intensity at a point on the surface
  - **Surface Rendering Method** - How we calculate the intensity at each pixel



Lastnosti snovi

- Ambient color
- Diffuse color
- Specular color
- Shininess
- Color of emitted light



Modeli osvetljevanja

- Ambient
  - Normals don’t matter
- Lambert/Diffuse
  - Angle between surface normal and light
- Phong/Specular
  - Surface normal, light, and viewpoint
- Rendering Equation (Kajiya)

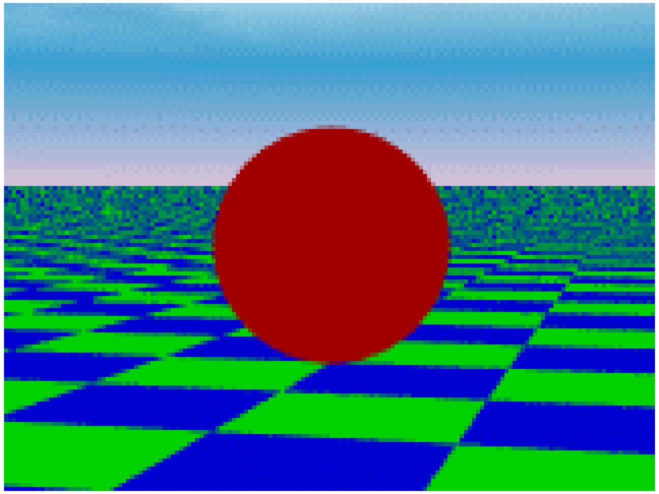
$$I(x,x') = g(x,x')\left[\varepsilon(x,x') + \int_S \rho(x,x',x'')I(x',x'')dx''\right]$$

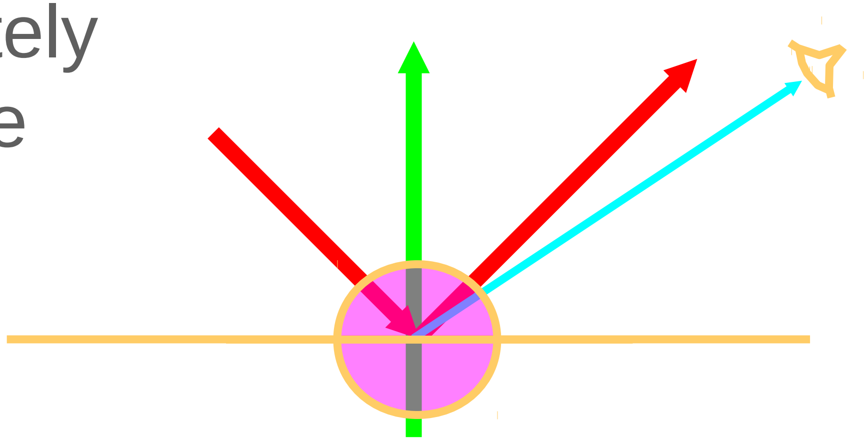
Svetloba okolja (ambient color)

- Affects the overall color of the object
- Is most noticeable where an object receives no direct light
- Total ambient reflectance is affected by global ambient light and ambient light from all light sources

Svetloba okolja (ambient color)

- It represents the approximate contribution of the light to the general scene, regardless of location of light and object
- Indirect reflections that are too complex to completely and accurately compute
- $I_{ambient}$ = color

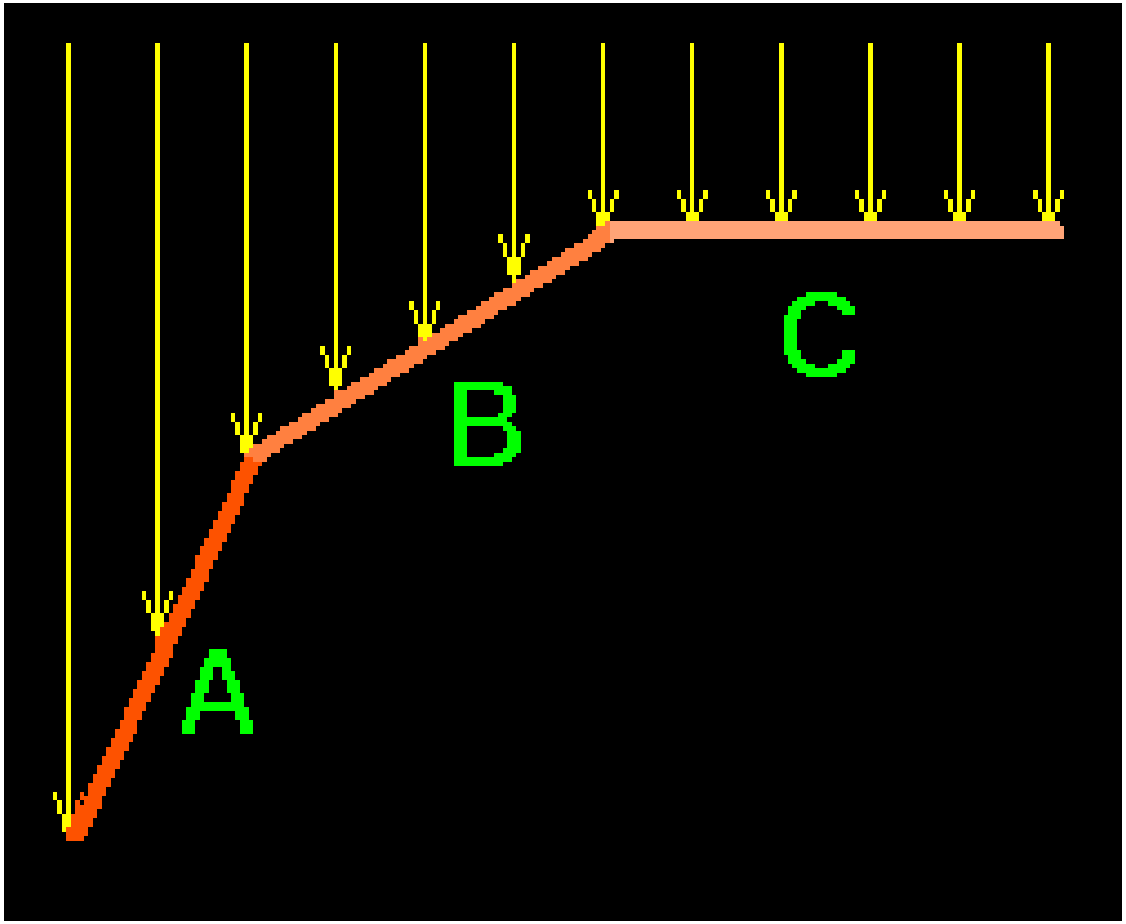




Difuzni odboj

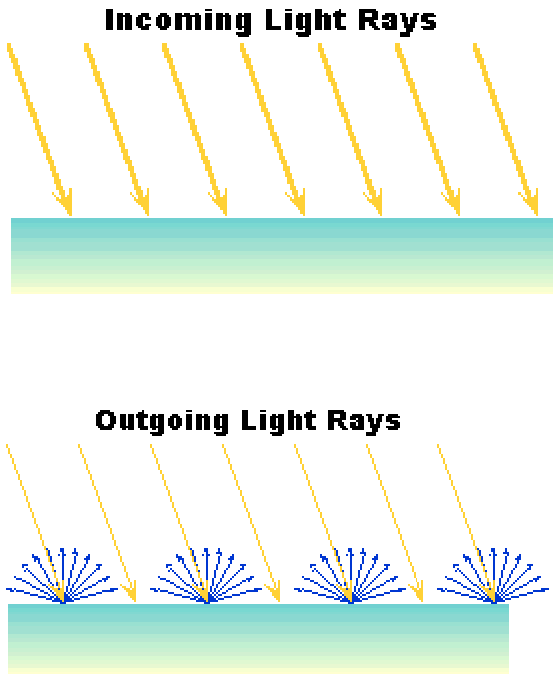
- Plays an important role in the color you perceive for an object
- Is affected by the color of the incident diffuse light and the angle of the incident light relative to the normal direction
- Viewer position doesn’t affect diffuse reflection

Osvetlitev površine – svetlost predmeta



Difuzni odboj

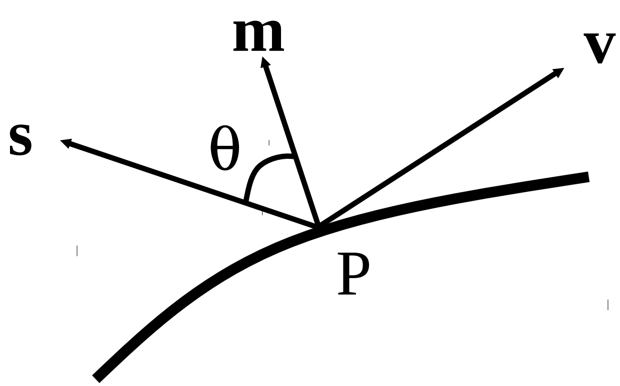
- Contribution that a light has on the surface, ***regardless of viewing direction***.
- Diffuse surfaces, on a microscopic level, are very rough. This means that a ray of light coming in has an equal chance of being reflected in *any* direction.



Model difuzne svetlobe

- Lambert's Law

The intensity of the light reflected from a perfect diffuser, I_d, is proportional to the cosine of the angle between the light source direction and the normal to the surface.



$$I_d = I_s p_d \cos\theta$$

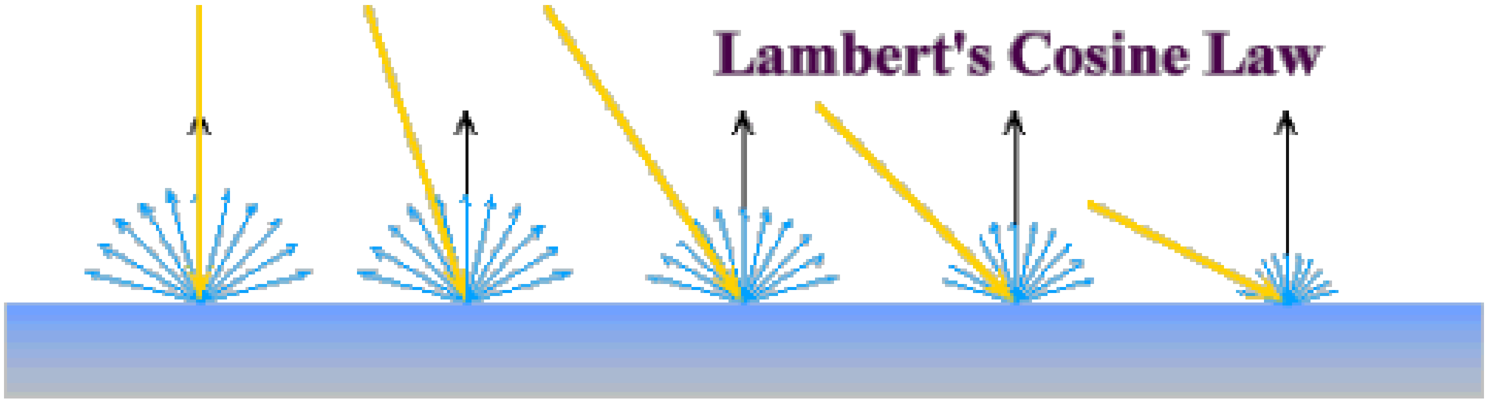
and $$\cos\theta = \frac{s \bullet m}{|s||m|}$$

$$I_d = I_s p_d \max\left(0, \frac{s \bullet m}{|s||m|}\right)$$

p_d : diffuse reflection coefficient

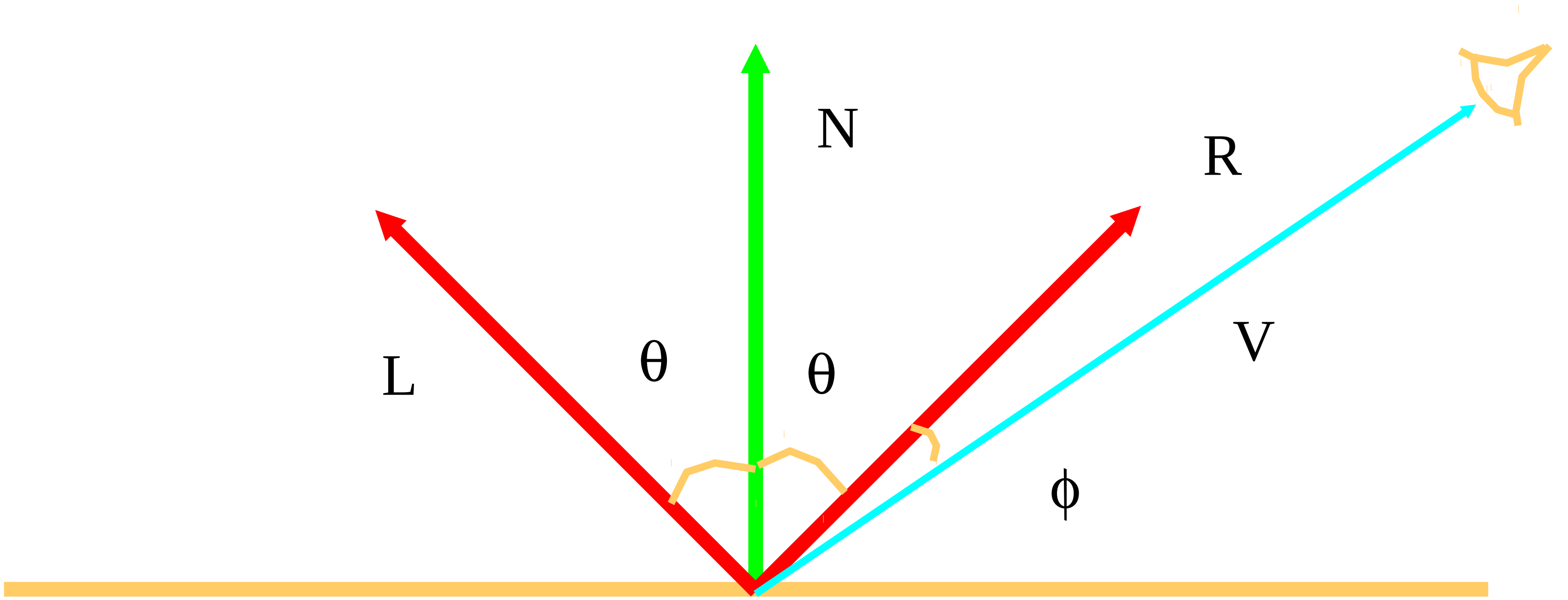
Lambertov kosinusni zakon

- Diffuse surfaces follow Lambert’s Cosine Law
- Lambert’s Cosine Law - reflected energy from a small surface area in a particular direction is proportional to the cosine of the angle between that direction and the surface normal.
- Think about surface area and # of rays

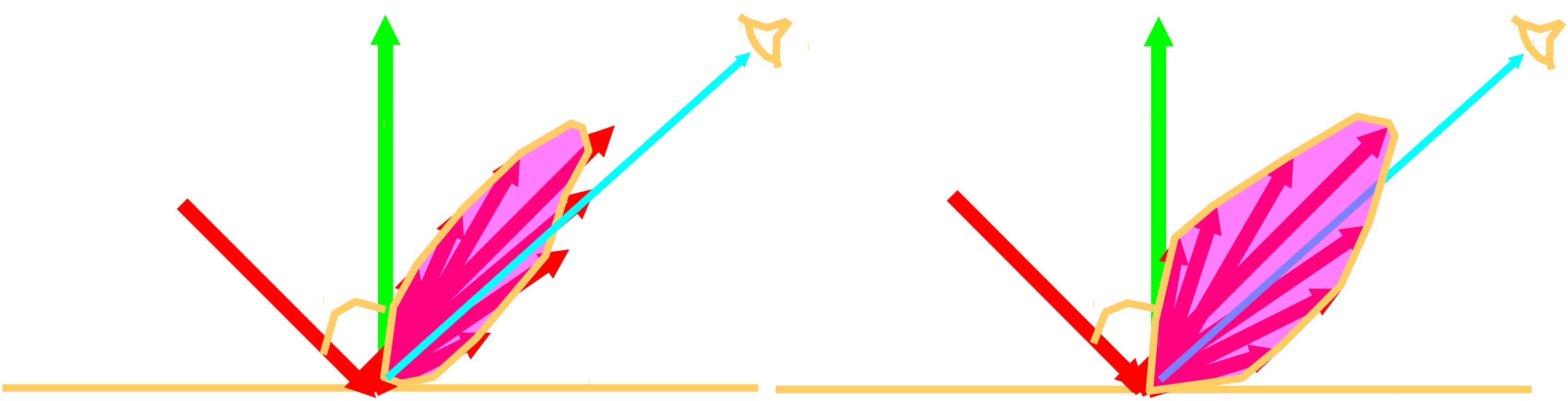


Zrcalni odboj

- Specular contribution can be thought of as the “shiny highlight” of a plastic object.



Bleščeči in motni predmeti

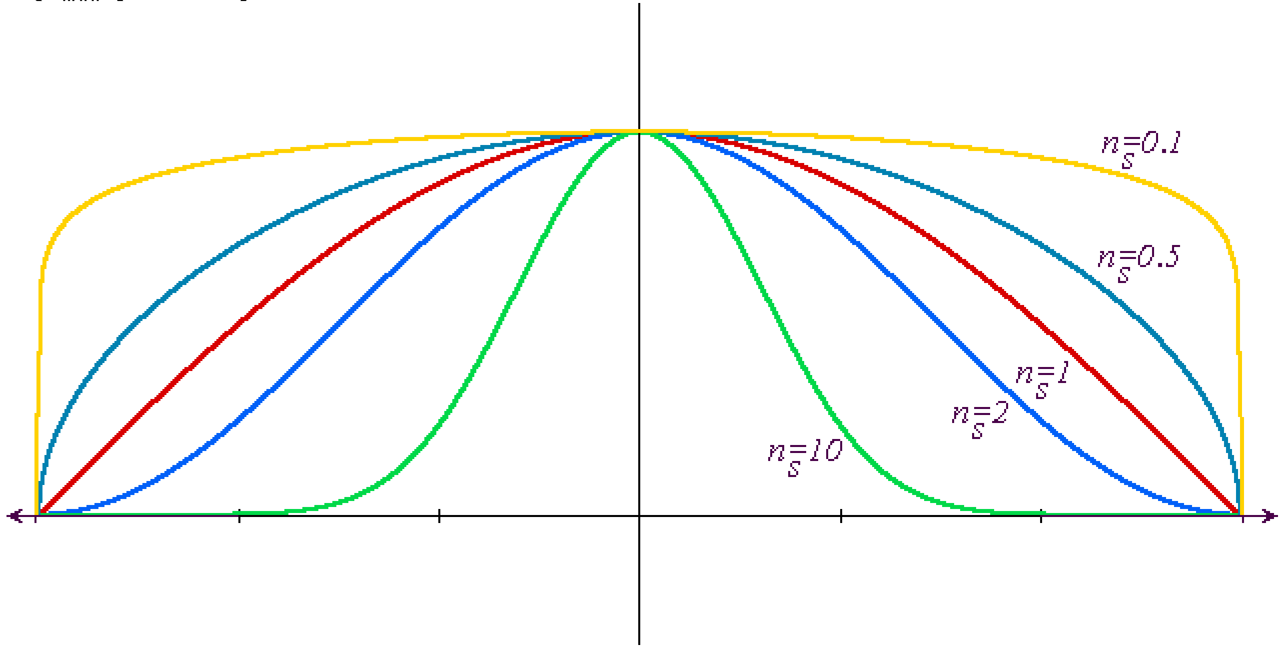


Phongov Model

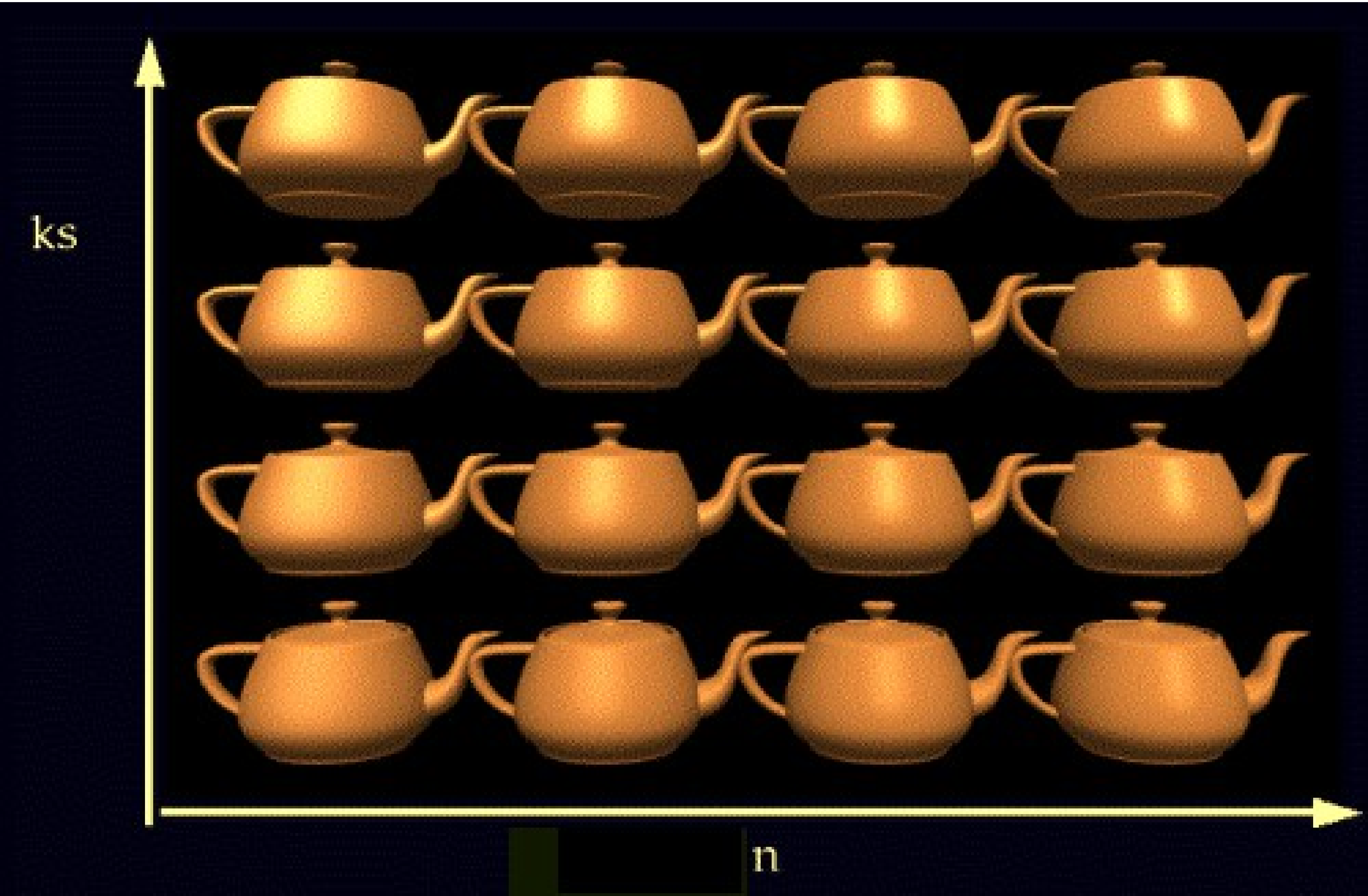
- An approximation is sets the intensity of specular reflection proportional to $(\cos\phi)^{shininess}$

- $I_{specular} = k_s I_{light} (\cos\phi)^{shininess}$

  $= k_s I_{light} (V.R)^{shininess}$



Phongov model senčenja



Phongov model

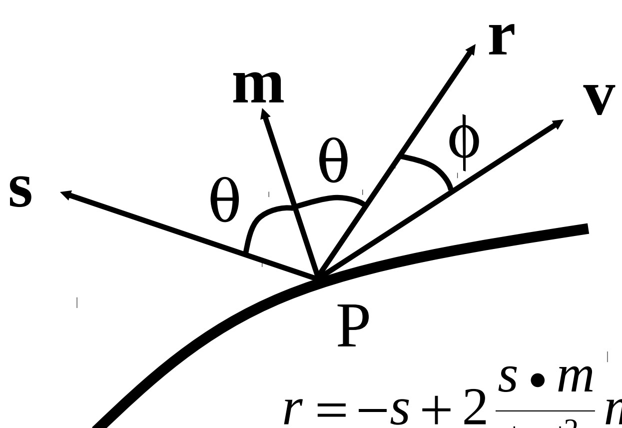
The amount of light reflected is greatest in the direction of perfect mirror reflection. At nearby angle, the amount of reflected light, I_{sp}, varies according to

$\cos^f(\phi)$

$$I_{sp} = I_s p_s \cos^f \phi$$

and

$$\cos^f \phi = \left(\frac{r \bullet v}{|r||v|} \right)^f$$



$$r = -s + 2\frac{s \bullet m}{|m|^2} m$$

$$I_{sp} = I_s p_s \max\left(0, \left(\frac{r \bullet v}{|r||v|} \right)^f \right)$$

p_s : specular reflection coefficient

Kombinacija svetlobe

- Light seen by the viewer =
- Ambient light + Diffuse light + Specular light

$$I = I_s p_a + I_s p_d lambert + I_s p_s phong$$

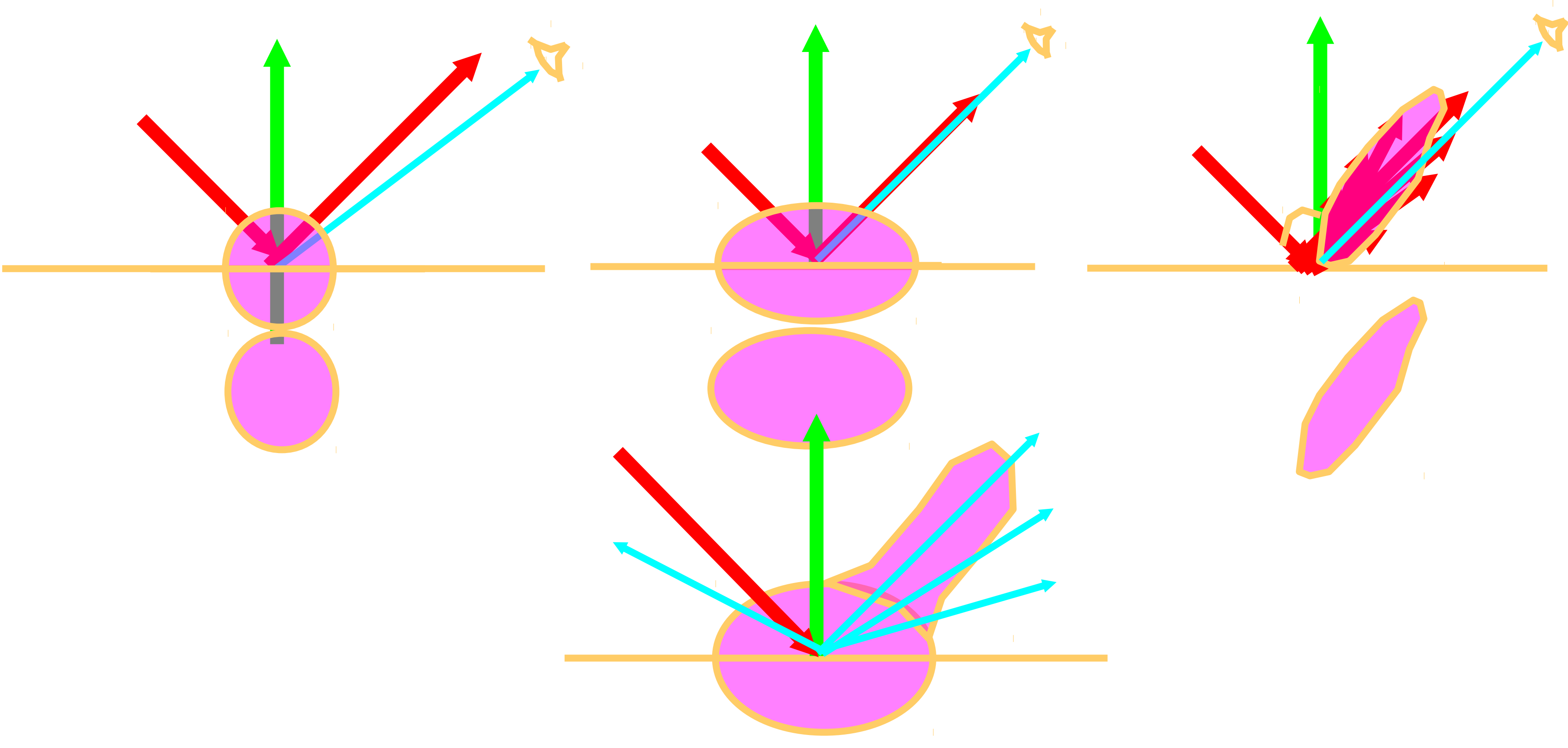
OpenGL allows you to specify different intensities for ambient, difuse and specular light sources

$$I = I_{sa} p_a + I_{sd} p_d lambert + I_{sp} p_s phong$$

I_{sa}, I_{sd}, I_{sp} : set for each light in the environment

p_a, p_d, p_s : set for each material in the environment

Ambientna + difuzna + zrcalna



Žarenje (emission)

- Make an object appear to give off light
- Examples: lamps, street lights, etc.

Slabljenje (attenuation)

- One factor we have yet to take into account is that a light source contributes a higher incident intensity to closer surfaces.
- The energy from a point light source falls off proportional to $1/d^2$.

RGB vrednosti za luči in snov

- RGB values for lights indicate light intensity
- RGB values for materials indicate reflected proportions of those colors

Dodajanje barve

$$I_r = I_{sar} p_a + I_{sdr} p_d lambert + I_{spr} p_s phong$$

$$I_g = I_{sag} p_a + I_{sdg} p_d lambert + I_{spg} p_s phong$$

$$I_b = I_{sab} p_a + I_{sdb} p_d lambert + I_{spb} p_s phong$$

Senčenje (shading)

Shading is how we “color” a triangle.

Senčenje (shading)

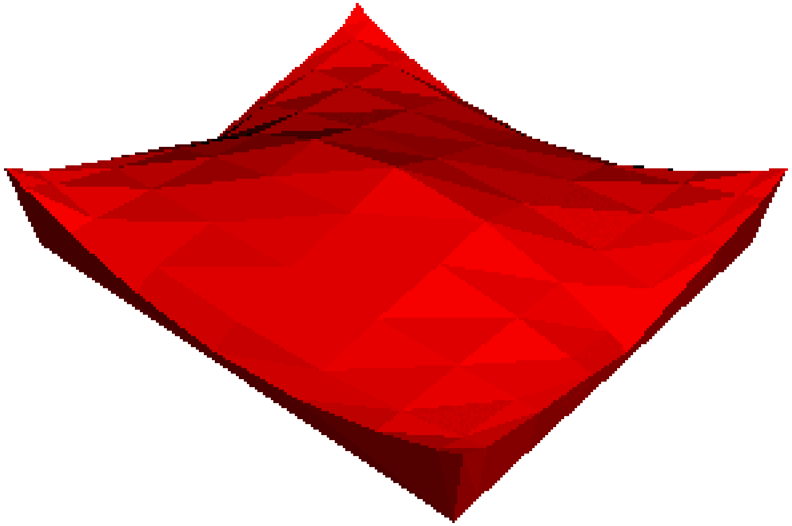
- Flat shading
- Accentuates the individual polygons
- Smooth shading
- Smooths the edges between polygons
  - Phong shading
  - Gourand shading

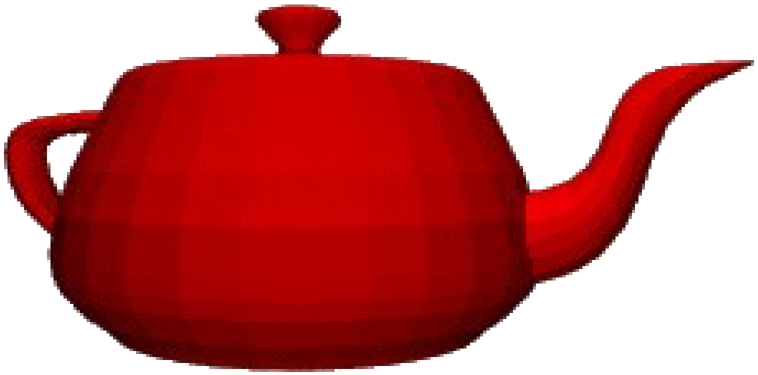
Modeli senčenja (direktna svetloba)

- Flat Shading
  - Compute Phong lighting once for entire polygon
- Gouraud Shading
  - Compute Phong lighting at the vertices and interpolate lighting values across polygon
- Phong Shading
  - Compute averaged vertex normals
  - Interpolate normals across polygon and perform Phong lighting across polygon

Plosko, konstantno senčenje

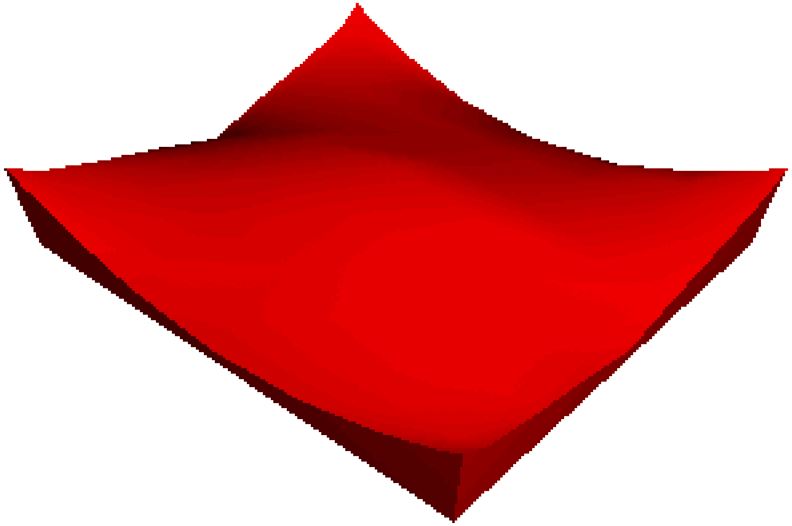
- Constant Intensity or Flat Shading
- One color for the entire triangle. Color is computed for one vertex and assigned to entire polygon
- Fast
- Good for some objects
- What happens if triangles are small?
- Sudden intensity changes at borders
- Specular highlights are rendered poorly





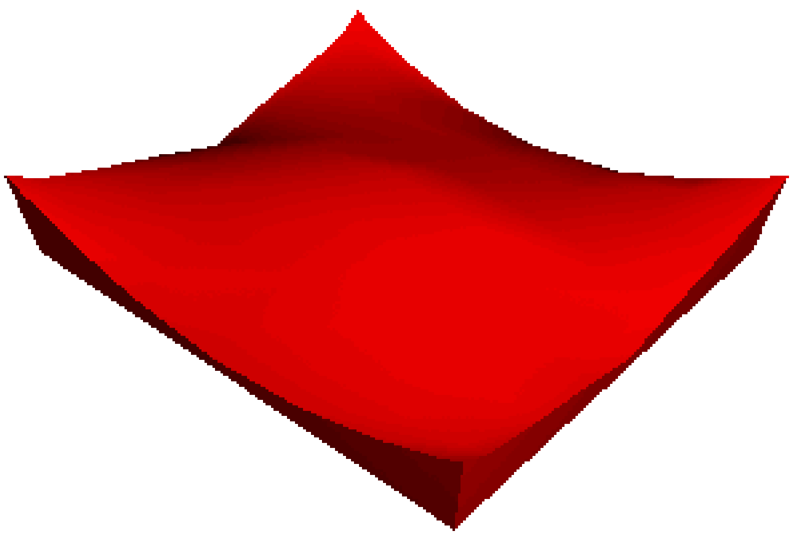
Mehko senčenje (Gouraud)

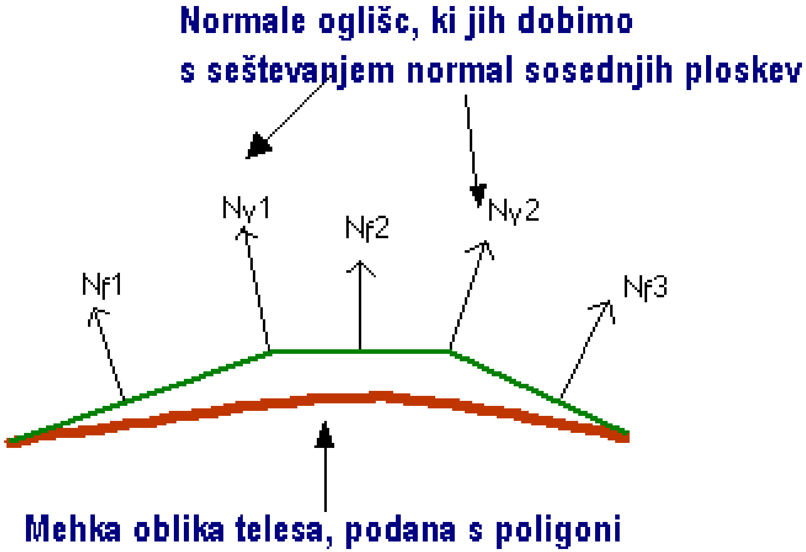
- Color is linearly interpolated from vertex points
- Specular highlights appear interpolated



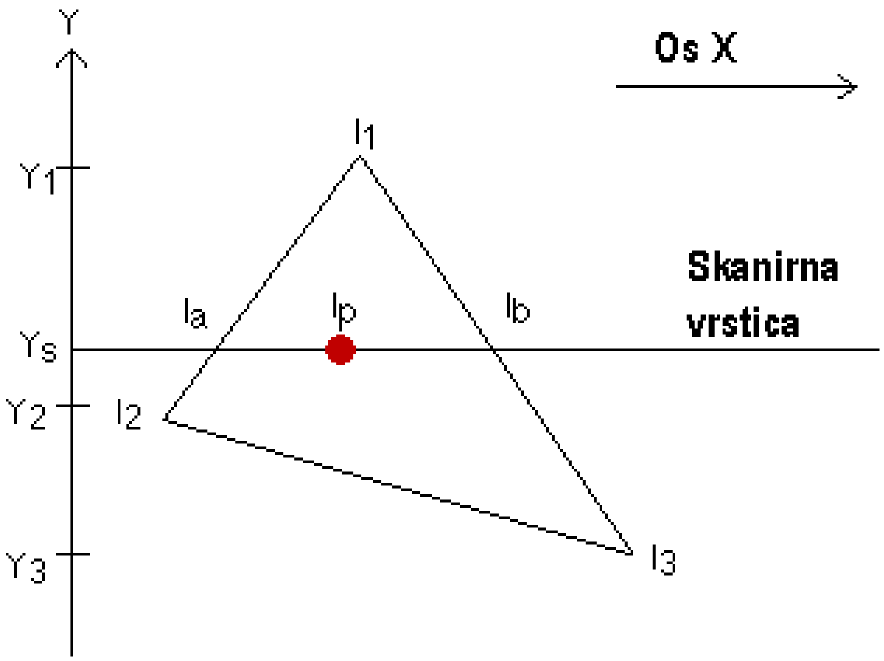
Gouraudovo senčenje

- Intensity Interpolation Shading
- Calculate lighting at the vertices. Then interpolate the colors as you scan convert
- Relatively fast, only do three calculations





Gouraudovo senčenje

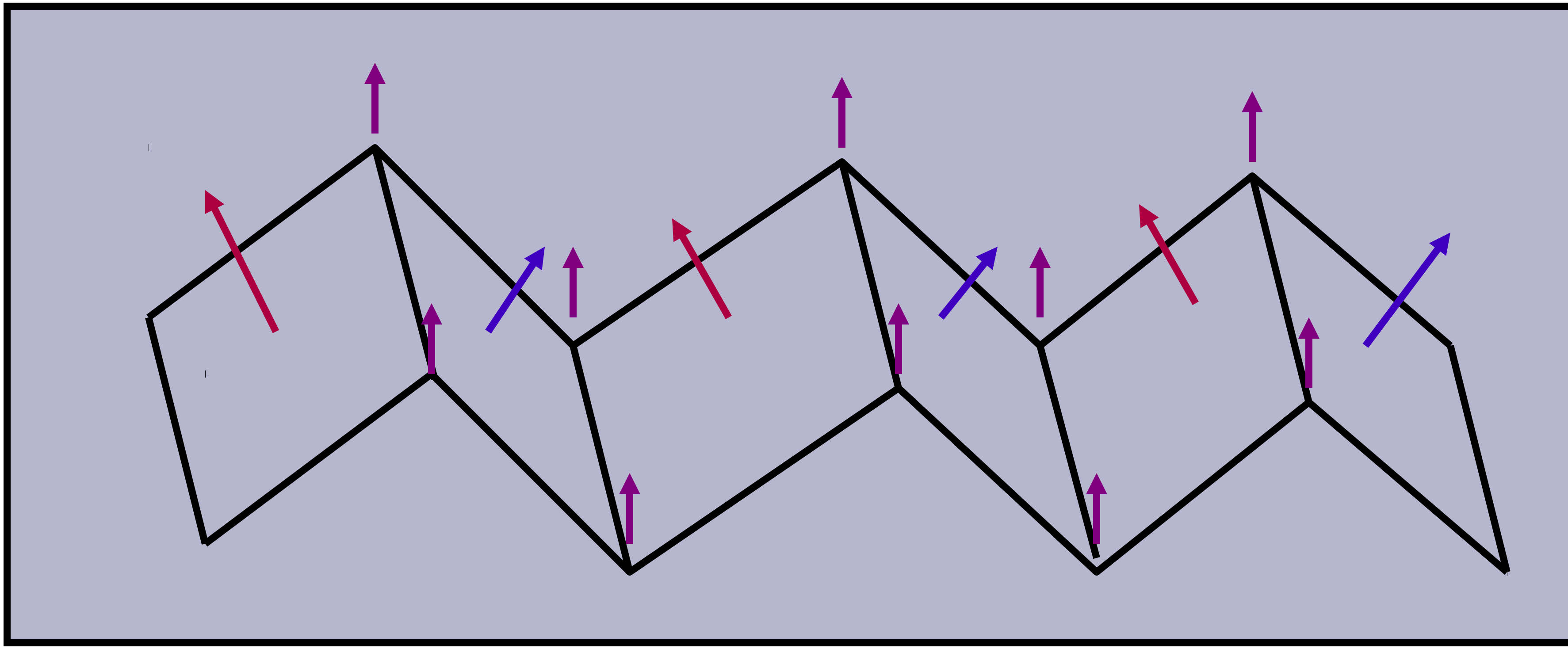


$$I_a = I_1 - (I_1 - I_2)(Y_1 - Y_s)/(Y_1 - Y_2)$$

$$I_b = I_1 - (I_1 - I_3)(Y_1 - Y_s)/(Y_1 - Y_3)$$

$$I_p = I_b - (I_b - I_a)(X_b - X_p)/(X_b - X_a)$$

Slabo povprečenje verteksov

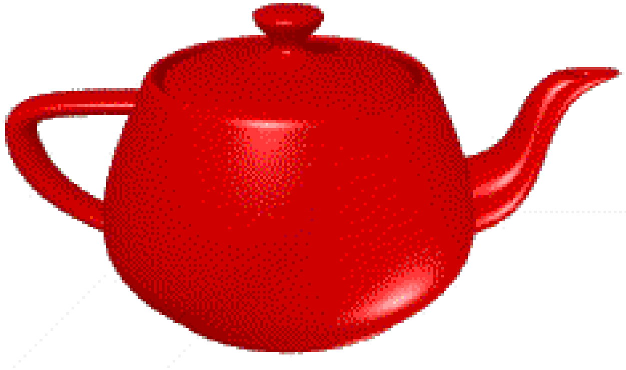


Mehko senčenje (Phong)

- Phong
- Interpolate normal vectors for each point on the face of an object
- Better realism but more computationally expensive

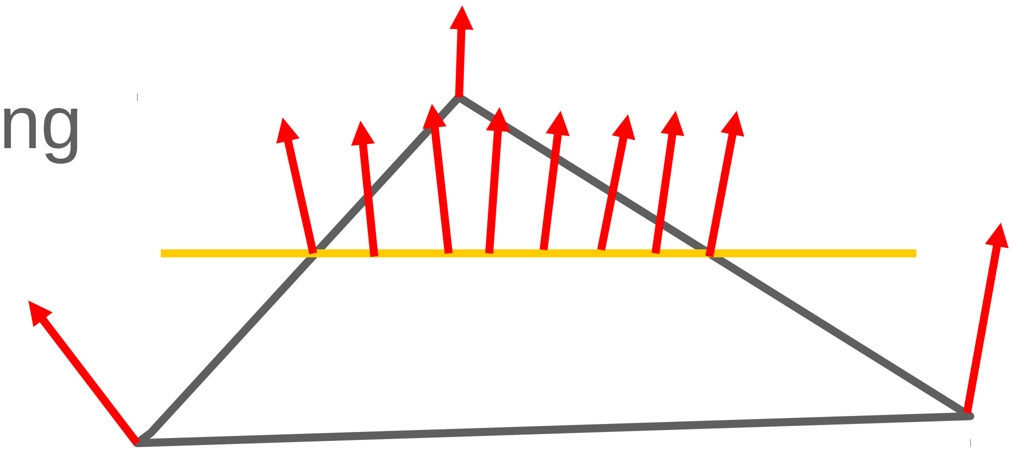
Phongovo senčenje

- *Phong shading* is not the same as Phong lighting, though they are sometimes mixed up
  - Phong lighting: the empirical model we've been discussing to calculate illumination at a point on a surface
  - Phong shading: linearly interpolating the surface normal across the facet, applying the Phong lighting model at every pixel
    - Same input as Gouraud shading
    - Usually very smooth-looking results:
    - But, considerably more expensive

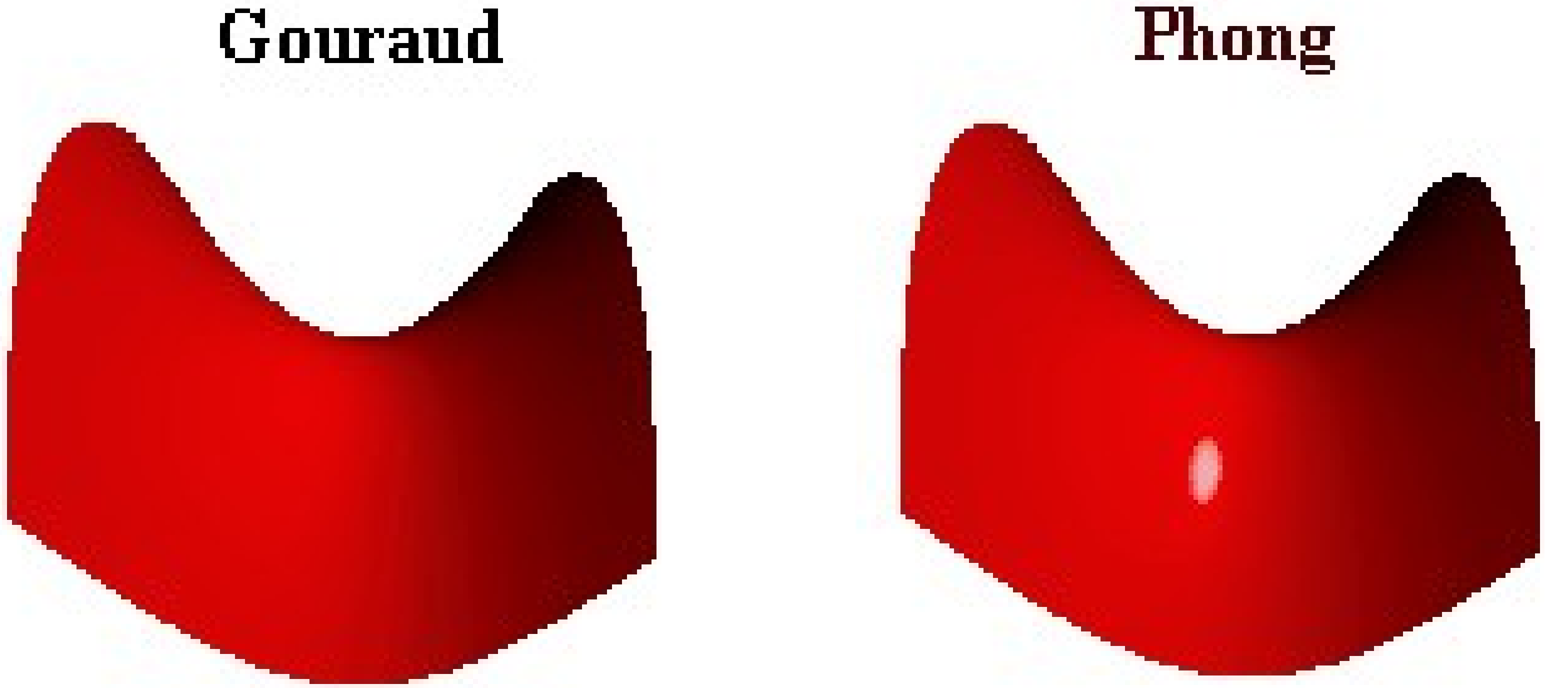


Phongovo senčenje

- Interpolate the normal, since that is the information that represents the "curvature"
- Linearly interpolate the vertex normals. For ***each*** pixel, as you scan convert, *calculate* the lighting *per pixel*.
- True "per pixel" lighting

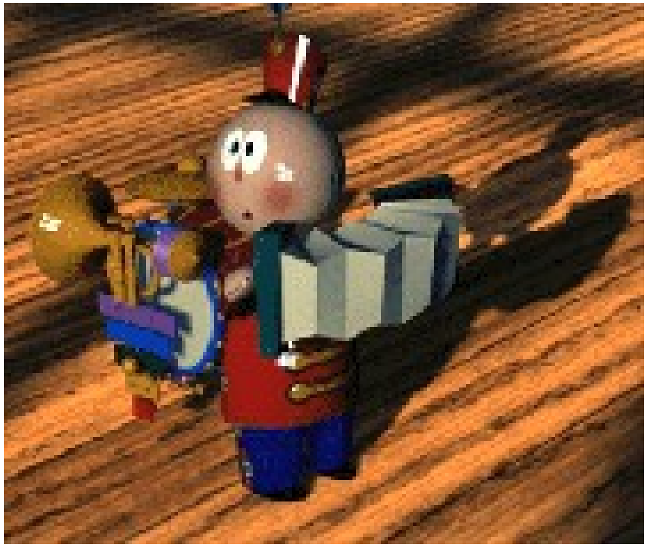


Primerjava Gouraud:Phong



Senčniki (Shaders)

- Local illumination quite complex
  - Reflectance models
  - Procedural texture
  - Solid texture
  - Bump maps
  - Displacement maps
  - Environment maps
- Need ability to collect into a single shading description called a *shader*
- Shaders also describe
  - lights, e.g. spotlights
  - atmosphere, e.g. fog



Senčenje v realnem času

- Moore’s law: CPU power doubles every 18 months
  - Advances in materials
  - Advances in methods
  - Advances in marketing
- Graphics version: GPU power doubles every 6 months
- Supports more sophisticated shading, though in unexpected ways (e.g. strange uses of the texture maps)

Peercy, Olano, Airey , Ungar, "Interactive Multi-Pass Programmable Shading", SIGGRAPH 2000