Universität Bielefeld

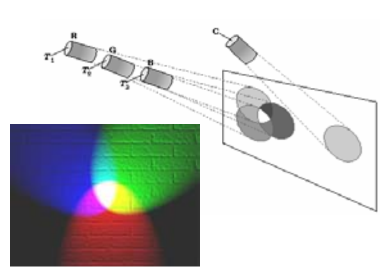
Virtual Reality for Human Computer Interaction

Appearance: Lighting

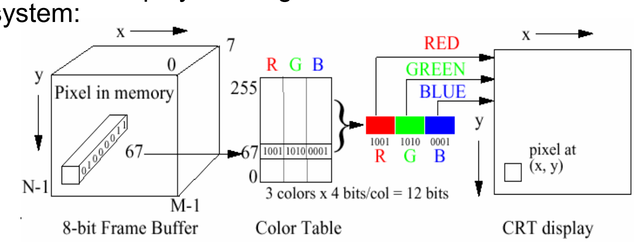
Representation of Light and Color

- Do we need to represent all I_λ to represent a color C(I) ?
- No – we can approximate using a three-color additive system (taking into account the described problems):

$$C = T_1 C_1 + T_2 C_2 + T_3 C_3$$

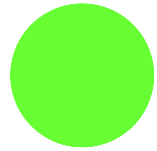


- Frames can be displayed using RGB system:



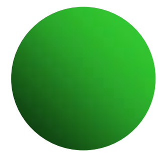
Motivation

- Suppose we build a model of a green sphere using many polygons and just color it.
  We get something like:



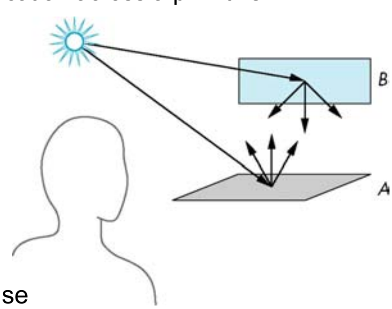
➢ The image of the sphere looks flat!

- But light-material interactions should cause each point to have a different color or shade to generate depth perception.
- Need to consider
  - Light sources
  - Material properties
  - Location of viewer
  - Surface orientation



Principle Lighting Model

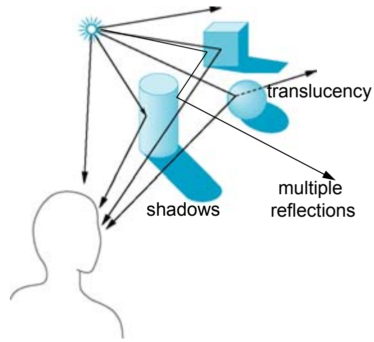
1. **Lighting** (or illumination): Description or model of light-object-eye interaction.
2. **Shading:** (Algorithmical) lighting application across a primitive.



- Physically, surfaces may reflect or emit light or both.
- Color that we see is determined by multiple interactions between light and surfaces.
- Recursive process:
  Light from A is reflected on B is reflected on A is reflected on B…
- Equations could be derived which use principles like conservation of energy to describe this process.
- This results in integral equation which can not be solved analytically…
- …but **global model** lighting approaches like **ray-tracing** and **radiosity** use numerical approximations which are becoming real-time capable (depending on parameterization and HW-support).

Principle Lighting Model

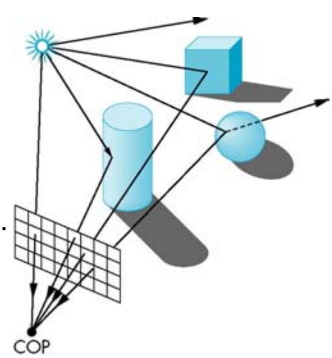
- Correct shading requires a global calculation involving all objects and light sources.
  - Incompatible with pipeline model which shades each polygon independently (local rendering).
  - Numerical solutions are expensive but can in principle be sped up using dedicated hardware.
- For real time computer graphics, approaches are utilized which imitate physically correct light-matter-eye interaction, hence which “look right”.
  - Exist many techniques for approximating global effects



Local Lighting Model

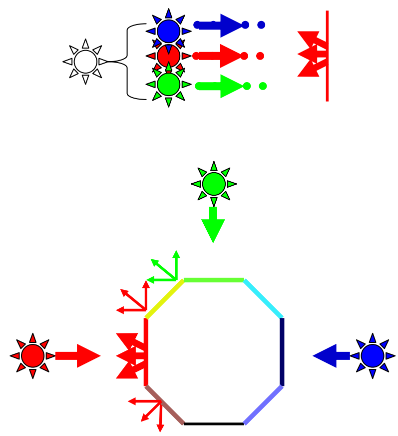
Local model:

- Following rays of light from light emitting surfaces (**light-sources**) instead of looking at a global energy balance.
- Derive a model which describes how these rays interact with reflecting surfaces.
- Will focus on single interaction in contrast to multiple interaction (like used in ray-tracing).
- This approach requires **light sources** and **reflection model**.
- Viewer sees only light which reaches eye.
  - No reflection inbetween:
    Perception of light source‘s color.
  - With surface reflection:
    Perception based on light source‘s color and surface material.
- Viewer‘s eye is exchanged for COP (Center of Projection) and projection plane.



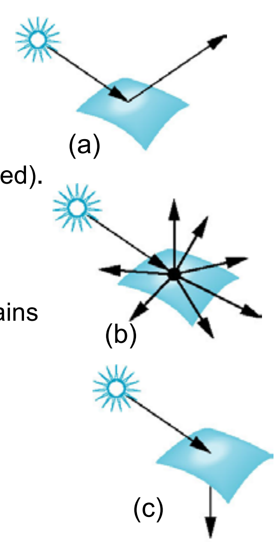
Local Lighting Model

- Light that strikes an object is
  - partially absorbed and
  - partially scattered (reflected).
- The amount reflected determines
  - the color and
  - brightness of the object.
  - ➢ A surface appears red under white light because the red component of the light is reflected and the rest is absorbed
- The reflected light is scattered in a manner that depends on
  - the smoothness and
  - orientation of the surface.



Reflecting Surfaces

- (a) Specular surfaces:
  - Appear shiny because most of reflected light is scattered in a narrow range of abgles close to angle of reflection.
  - Ideal reflectors: Mirrors (parts can be still absorbed).
  - Angle of incidence is equal angle of reflection.
- (b) Diffuse surfaces:
  - Reflected light is scattered in all directions.
  - E.g., walls painted with matte or flat paint or terrains seen from hight.
  - Perfect diffuse surfaces scatters equally in all directions.
- (c) Transluscent surfaces:
  - Allow some light to penetrate the surface and to emerge from another location -> Refraction.
  - Some incident light may be reflected as well.



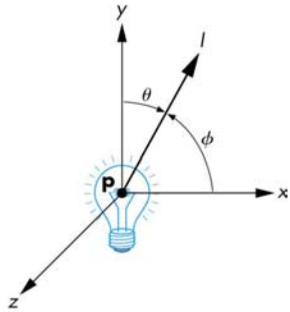
Light Sources

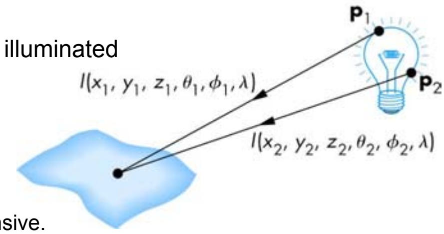
- In general, light sources should integrate light coming from all points on the source.
- Light can leave a surface by
  - self-emission and/or
  - reflection.
- Each point p=(x,y,z) on the surface is characterized by
  - the direction of emission (θ, ϕ) and
  - the intensity of energy at each wavelength λ and hence
  - the illumination function

$$I(x, y, z, \theta, \phi, \lambda)$$

- To calculate the source's contribution to an illuminated surface one has to
  - integrate over the source's surface,
  - account for the emission angles and
  - account for the distance between source and surface.

➢ Integration (analytical or numerical) is expensive.



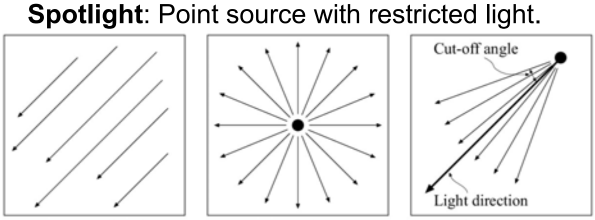


Light Sources

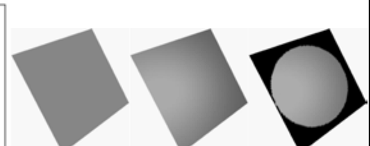
- An approximation to light-material interaction
  - uses 4 different light sources to
  - calculate an **intensity function *I*** →
  - using the three color model of the human visual system.

$$I = \begin{bmatrix} I_r \\ I_g \\ I_b \end{bmatrix}$$

1. **Ambient light**: Same amount of light everywhere, can model contribution of many sources and reflecting surfaces.
2. **Point source**: Model with position and color.
3. **Distant (directional) light**: Point source in infinite distance (parallel rays).
4. **Spotlight**: Point source with restricted light.



Directional Light — Point Light — Spot Light



Realtime 3D Computer Graphics / Virtual Reality – WS 2005/2006 – Marc Erich Latoschik

Ambient Light

- Near uniform lighting created by highly diffused light sources.
- One could model all light sources and interactions or use a concept called “ambient light” which
  - lights all surfaces uniformly.
  - is not viewer location dependent.

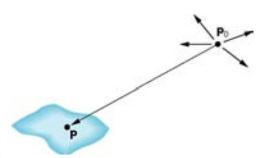
$$\vec{I}_{amb} = \vec{\mathbf{m}}_{amb} \otimes \vec{\mathbf{s}}_{amb} \qquad \vec{I}_{amb} = \begin{bmatrix} I_{ambr} \\ I_{ambg} \\ I_{ambb} \end{bmatrix}$$

- $\mathbf{m}_{amb}$ vector is a material attribute
- $\mathbf{s}_{amb}$ vector is a light source attribute

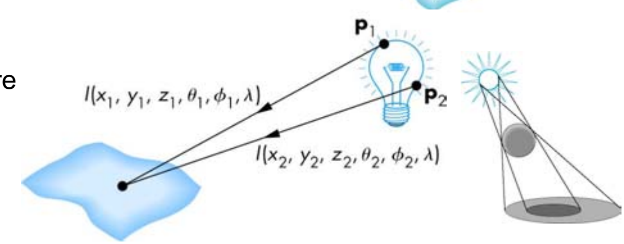
Point Light Sources

- Ideal point emits light in all directions.
- Intensity of illumination is inverse square of distance between source and surface.

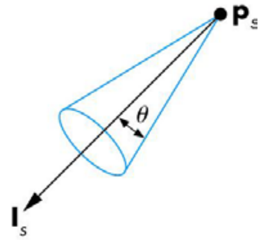
$$\vec{I}(p_0) = \begin{bmatrix} I_r(p_0) \\ I_g(p_0) \\ I_b(p_0) \end{bmatrix} \; and \;\; \vec{I}(p, p_0) = \frac{1}{|p - p_0|^2} \vec{I}(p_0)$$



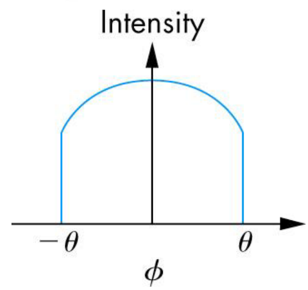
Use of point sources is more a matter of efficiency rather than realism as most sources have a dimension:



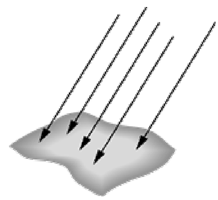
Spotlights and Distant Lights



Spotlights:
- Point source with limited direction.
- Point source Ps in a direction I_s and a width of θ.
- Spotlight attenuation:
  - Greater realism can be obtained by varying the intensity of light across the cone
  - Typical Function: $\cos(\phi) = \mathbf{S} \cdot \mathbf{I}$



Distant lights:
- Light sources that are distant to the surface
- Light is parallel:



Lighting in OpenGL

- Light sources can be turned on/off:

```
glEnable(GL_LIGHTING);
glEnable(GL_LIGHT0);
```

- Support: multiple lights
  - (but performance suffers)
- For each light:
  - Ambient, Diffuse, Specular per RGB
  - Position, Direction
  - Spotlight Exponent and Cutoff Angle
  - Light to Surface Distance Attenuation

Lighting At A Point

- Lighting at a point on an object's surface:

```
For each color in(Red, Green, Blue):
   For each light source:
        For each light type (ambient, diffuse, specular):
                Determine the amount of light reaching the point
                (Typically Ignore Shadowing)
                Determine the amount of light reflected
                (Based on properties of the surface)
```

- I_λ => sum of all light reflection from each light source

Lighting At A Point

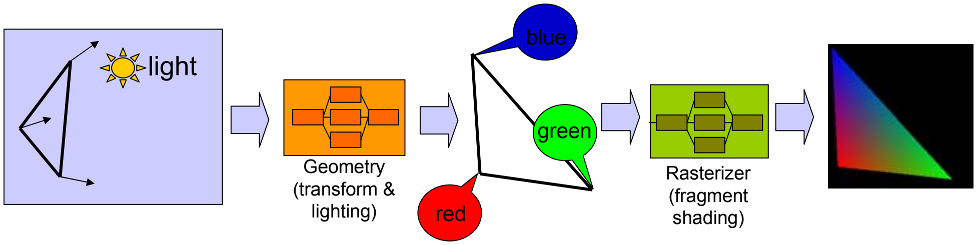
- Illumination, I, at a point is modeled as the sum of several terms:
  - More terms give more plausible results.
  - Fewer terms give more efficient computations.
- Each additive term of I is expressed in primary colors, I_r, I_g and I_b, i.e. I_λ where λ is r, g, or b (typically defined as a range from 0 to 1)
- Each of these colors (I_λ) is computed independently.
- Components (I_λ), can be used to express how much light a source emits and a surface reflects.
- Total illumination: Sum of each light source

$$I_\lambda = I_{\lambda 1} + I_{\lambda 2} + I_{\lambda 3}$$

- Various solutions for dealing with possible overflow (>1), e.g.,
  - clamp to max allowable
  - normalize individual terms:

$$I_\lambda = \frac{I_{\lambda 1}}{(I_{\lambda 1} + I_{\lambda 2} + I_{\lambda 3})} + \frac{I_{\lambda 2}}{(I_{\lambda 1} + I_{\lambda 2} + I_{\lambda 3})} + \frac{I_{\lambda 3}}{(I_{\lambda 1} + I_{\lambda 2} + I_{\lambda 3})}$$

Applying a lighting model



- Calculating lighting using objects defined by their surfaces:
  - In which coordinate system should the lighting be applied?
  - For which points on objects' surfaces should lighting be applied?
    - Sampling into surface may be to coarse
    - or may be to detailed and may produce unnecessary computational overhead
    - and sampling artifacts.
- Idea:
  - Lighting calculation **per vertex** and surface approximation **in screen space**.
  - Supported by **pipeline architecture**.