Graphics and Visualization

Holger Kenn

International University Bremen

Spring Semester 2006

Shading

- Diffuse Component
- Specular Component
- Ambient Component
- Light Sources

Recap

- Hierarchical Modeling
- Perspective vs Parallel Projection
- Representing solid objects

Shading

- Displaying Wireframe models is easy from a computational viewpoint
- But it creates lots of ambiguities that even perspective projection cannot remove
- If we model objects as solids, we would like them to look “normal”. One way to produce such a normal view is to simulate the physical processes that influence their appearance (Ray Tracing). This is computationally very expensive.
- We need a cheaper way that gives us some realism but is easy to compute. This is shading.

Types of shading

- Remove hidden lines in wireframe models
- Flat Shading
- Smooth Shading
- Adding specular light
- Adding shadows
- Adding texture

Hidden Line Removal

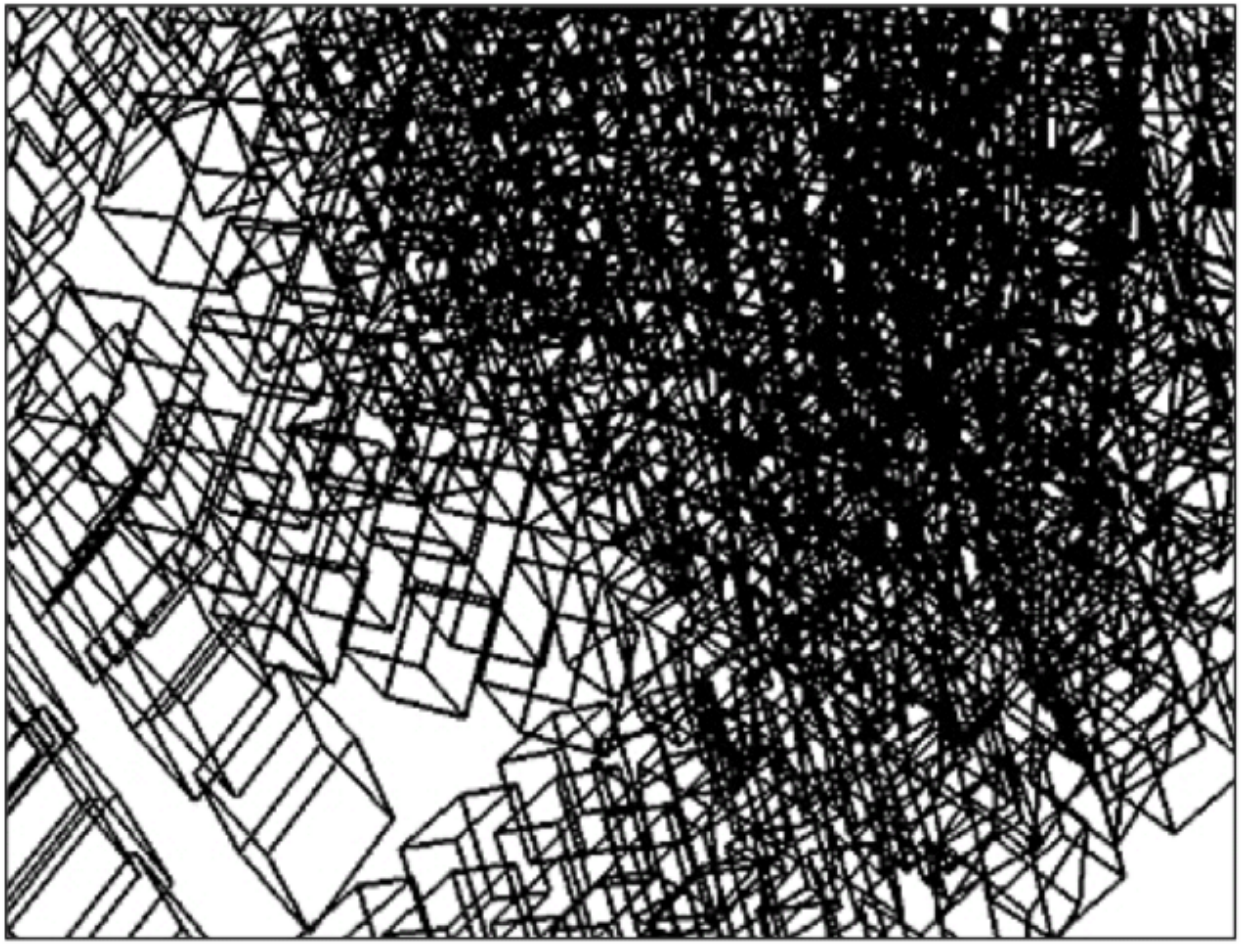
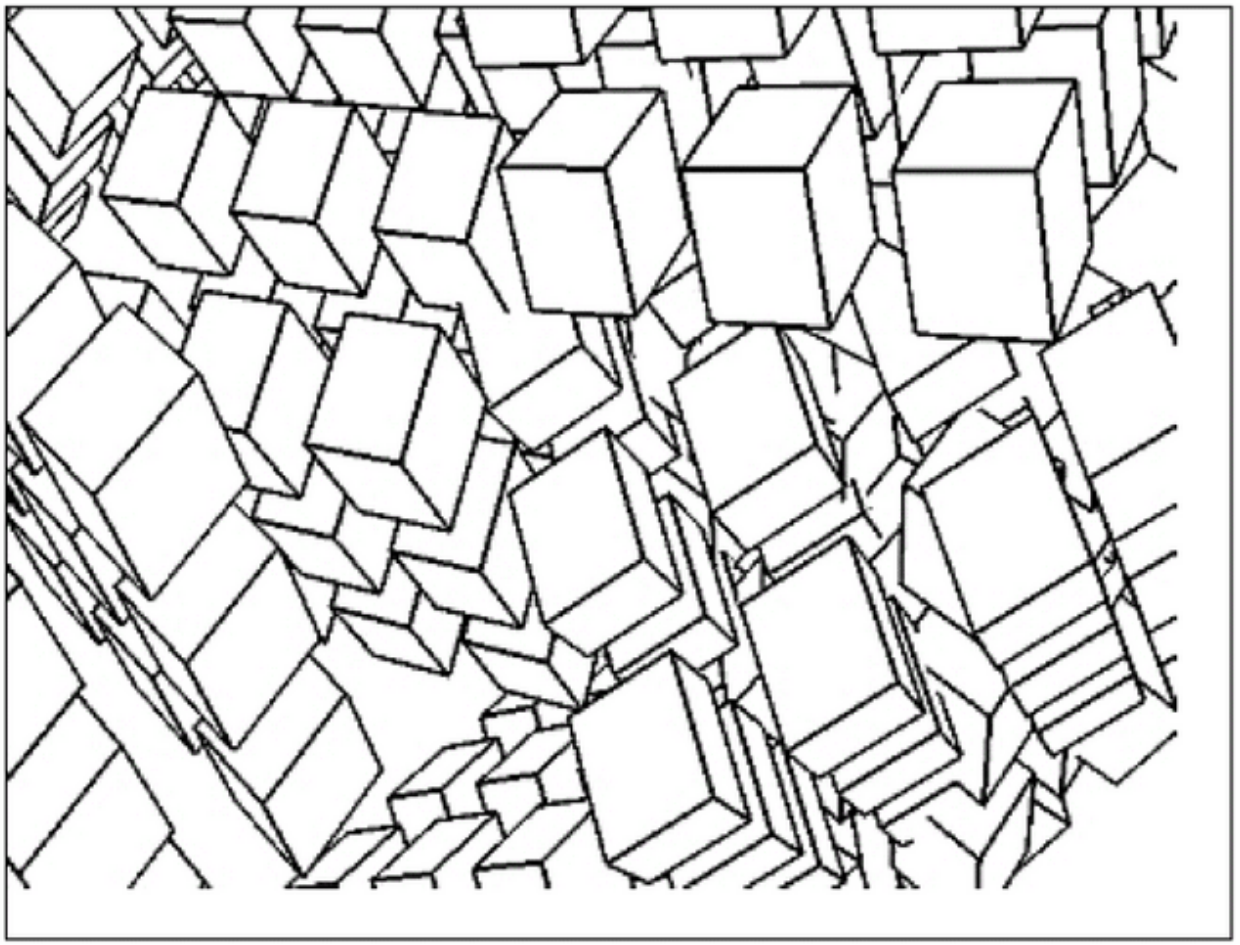


Image from Hill, Chap. 8

Hidden Line Removal



Flat Shading

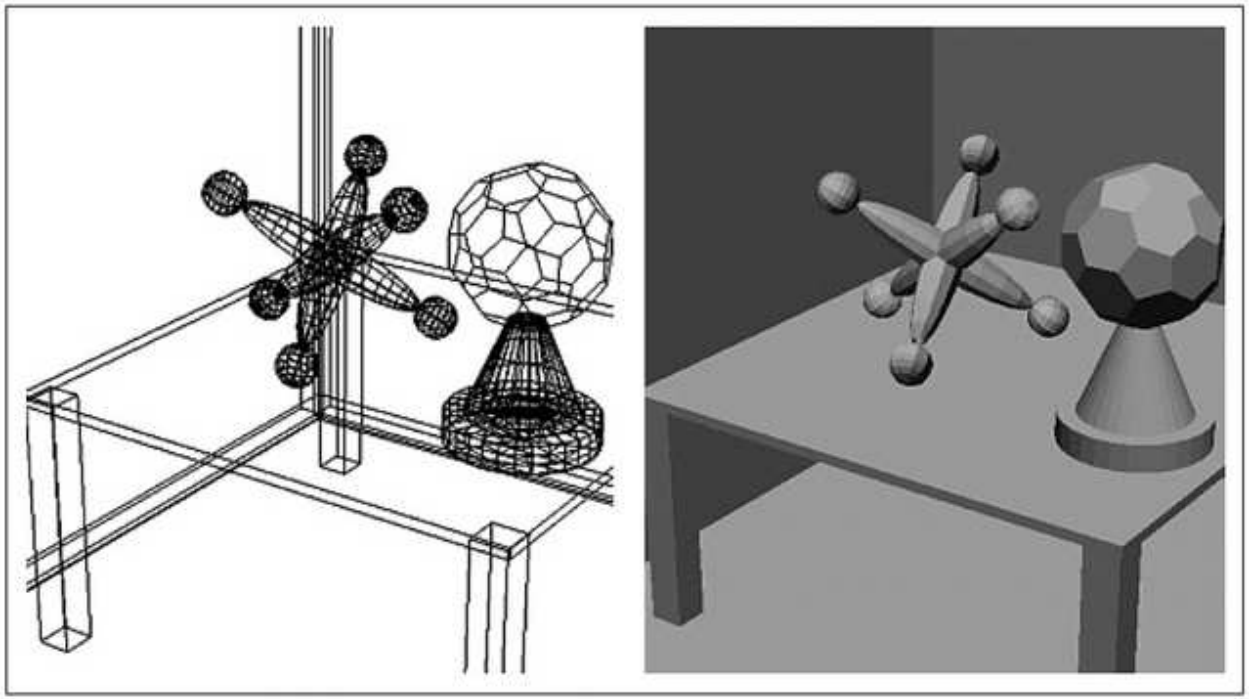


Image from Hill, Chap. 8

- Wireframe vs. flat shading display

Smooth Shading

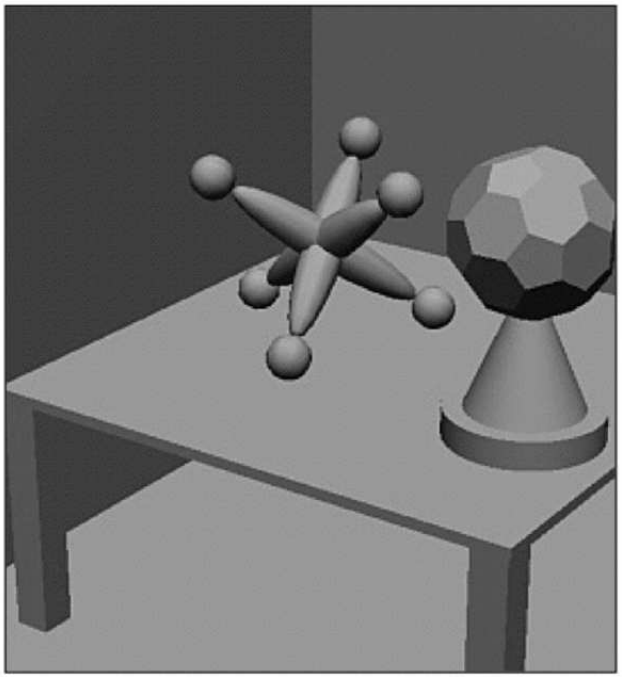


Image from Hill, Chap. 8

- Smooth shading creates “round” objects

Specular Lights

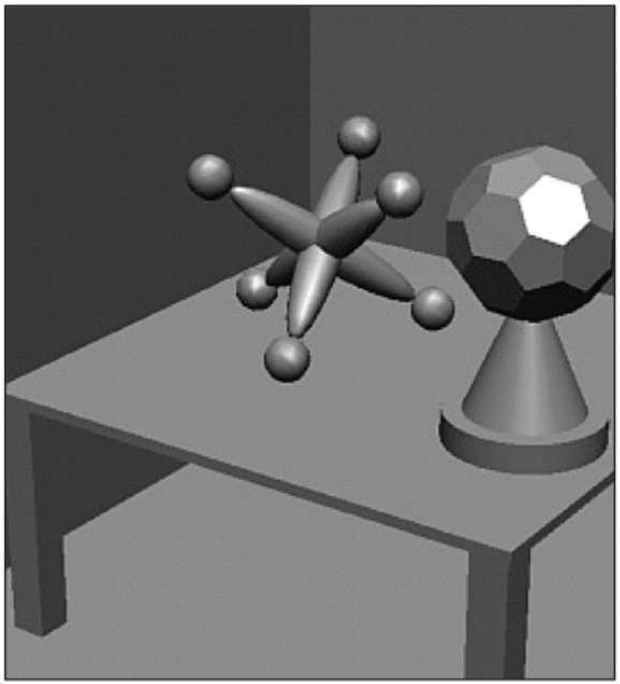


Image from Hill, Chap. 8

- Specular lights create “reflections”

Shadows

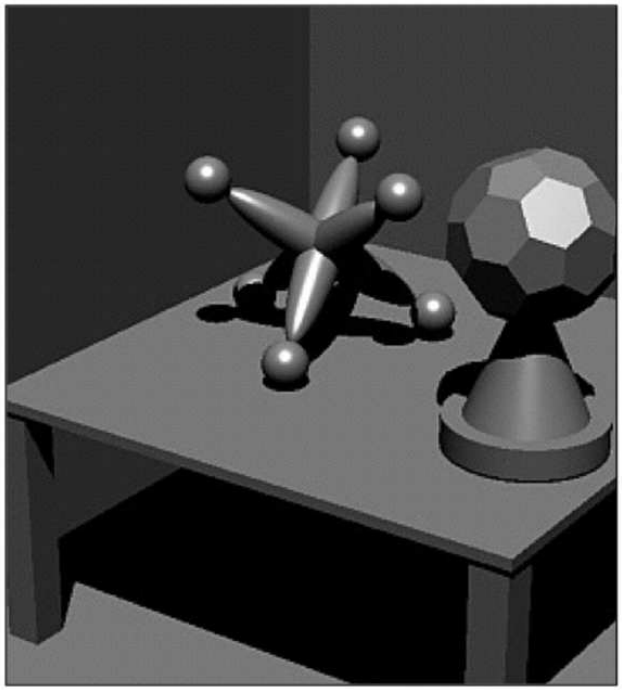


Image from Hill, Chap. 8

- Shadows give more “depth”

Texture

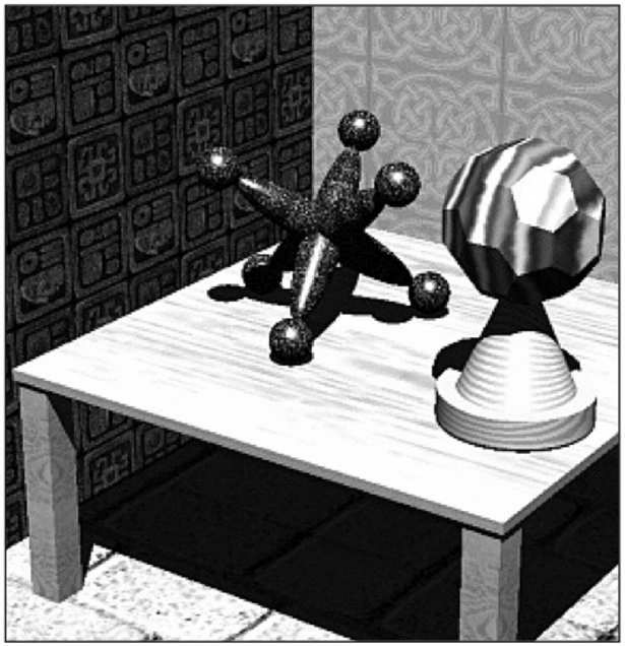


Image from Hill, Chap. 8

- Texture simulates different materials

Toy-Physics for CG

- There are two types of light sources: ambient light and point light sources.
- If all incident light is absorbed by a body, it only radiates with the so-called blackbody radiation that is only dependent of its temperature. We’re dealing with cold bodys here, so blackbody radiation is ignored.

Toy-Physics for CG: Scattering

- Diffiuse Scattering occurs if light penetrates the surface of a body and is then re-radiated uniformily in all directions. Scattered lights interact strongly with the surface, so it is usually colored.
- Specular reflections occur in metal- or plastic-like surfaces. These are mirrorlike and highly directional. They interact differently, so they may have a different color, typically more the color of the light source.
- A typical surface displays a combination of both effects.

Vectors for shading

- The normal vector $\vec{m}$ to the surface P.
- The vector $\vec{v}$ from P to the viewer's eye.
- The vector $\vec{s}$ from P to the light source.

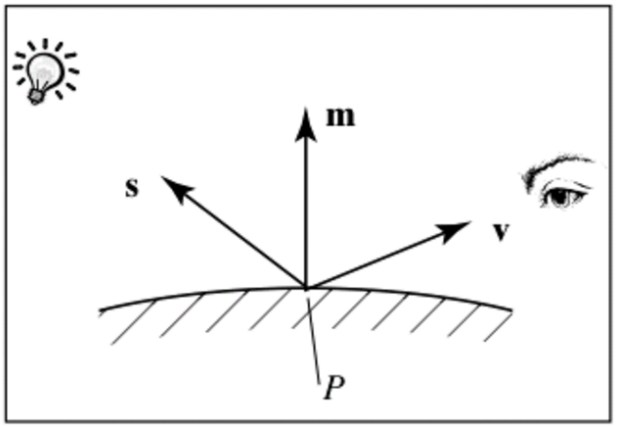


Image from Hill, Chap. 8

Faces have two sides

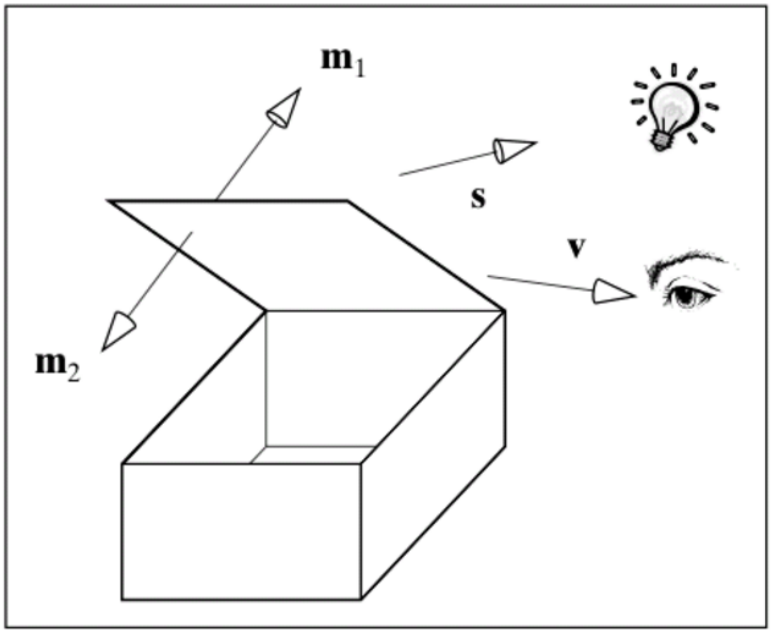


Image from Hill, Chap. 8

- Light calculations may have to be done for both sides of a face
- A face is visible if $\vec{v} \cdot \vec{m} > 0$

Calculating the diffuse component I_d

- Diffuse scattering is uniform, so forget $\vec{v}$
- It depends on $\vec{s}$ and $\vec{m}$.

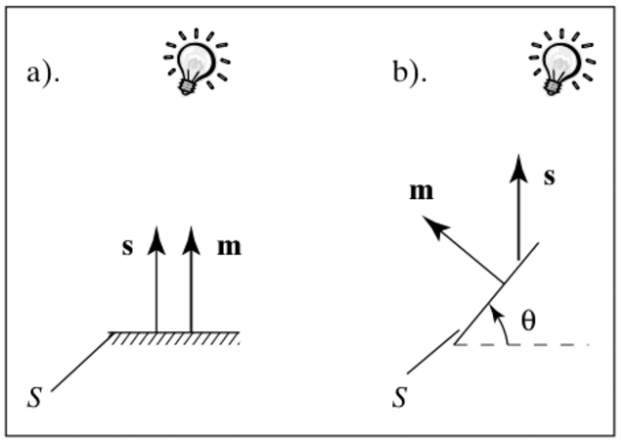


Image from Hill, Chap. 8

Lambert's Law

- Lambert's Law: A surface receives the illumination from a light source that is proportional to the cosine of the angle between the normal of the surface and the direction to the light source.
- $I_d = I_s \rho_d \frac{\vec{s} \cdot \vec{m}}{|\vec{s}||\vec{m}|}$
- I_s is the intensity of the light source, ρ_d is the diffuse reflection coefficient.
- We do not want negative intensities, so we set negative values of the cosine term to zero.

Reflection Coefficient

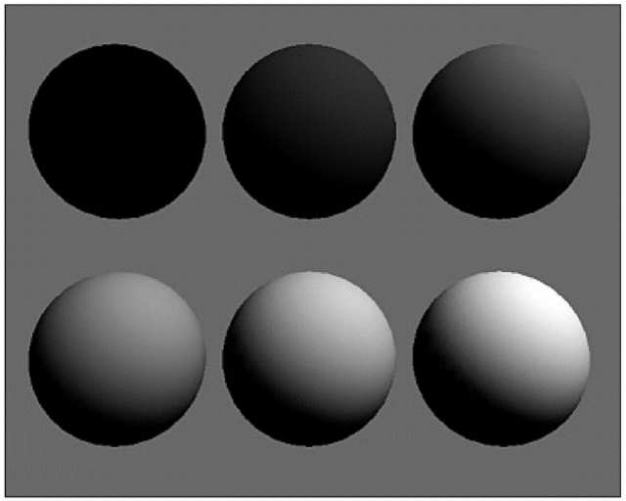


Image from Hill, Chap. 8

- Various reflection coefficient in diffuse light

Specular Reflection

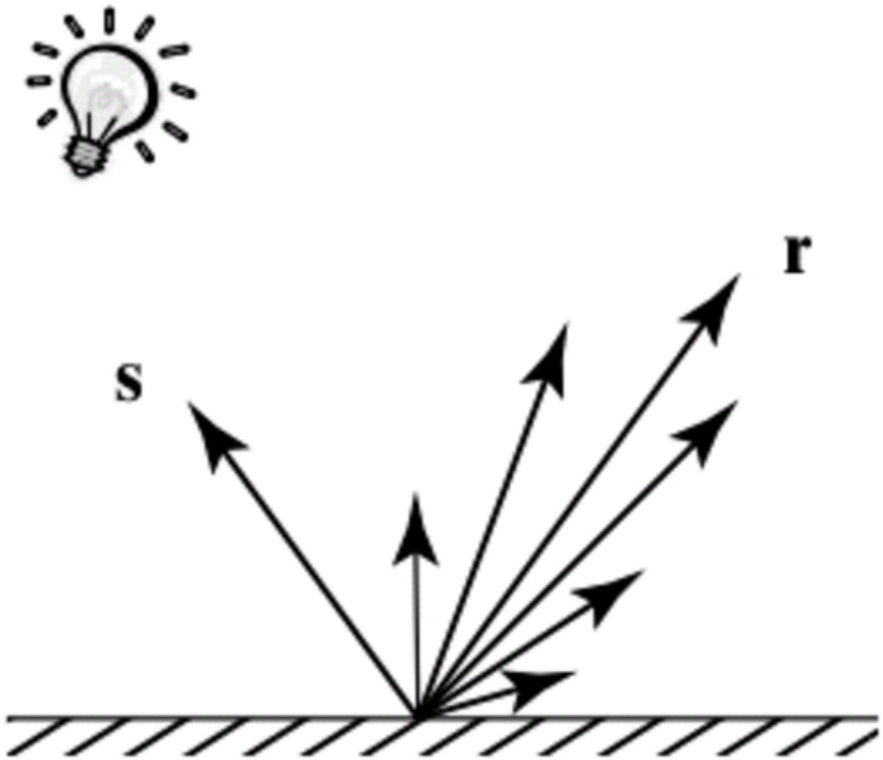


Image from Hill, Chap. 8

- Specular reflection is non-uniform. It depends on $\vec{s}$,$\vec{m}$ and $\vec{r}$

Specular reflection

- Several models have been developed for modeling specular reflection, the one OpenGL uses is closely related to the model by Phong (1975, Communications of the ACM 18: Illumination for Computer Generated Images)
- Phong: The light reflected in the direct mirror direction is the strongest. The ammound of light reflected in other directions diminishes rapidly.

Phong Specular Reflection

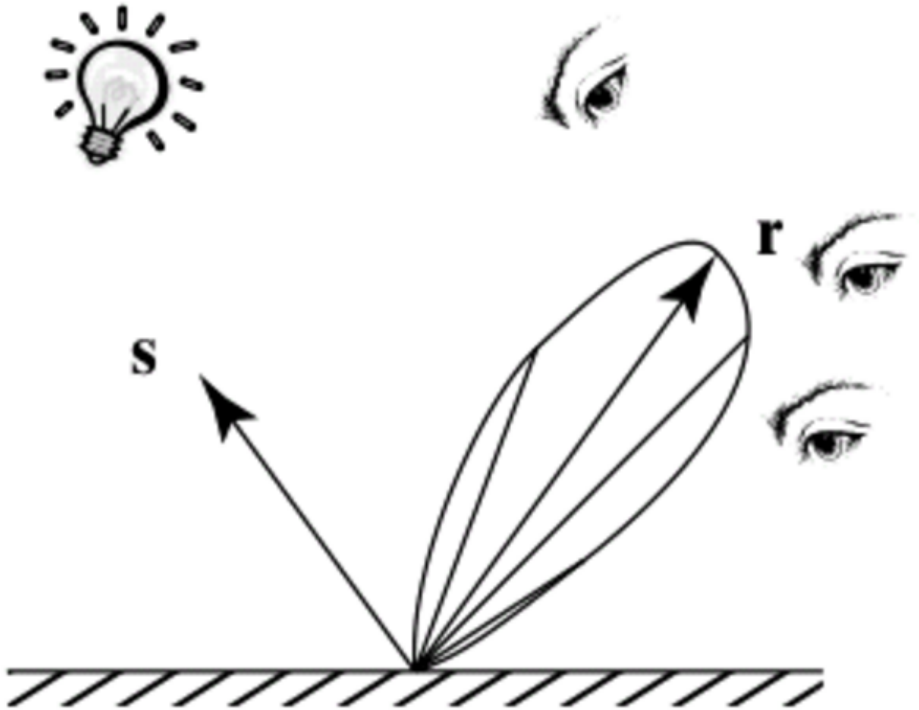


Image from Hill, Chap. 8

- Phong shading intensity pattern

Specular reflection

- Phong: The light reflected in the direct mirror direction is the strongest. The ammound of light reflected in other directions depends on the angle between that direction and the mirror direction and is stronger for small angles.

Phong Specular Reflection

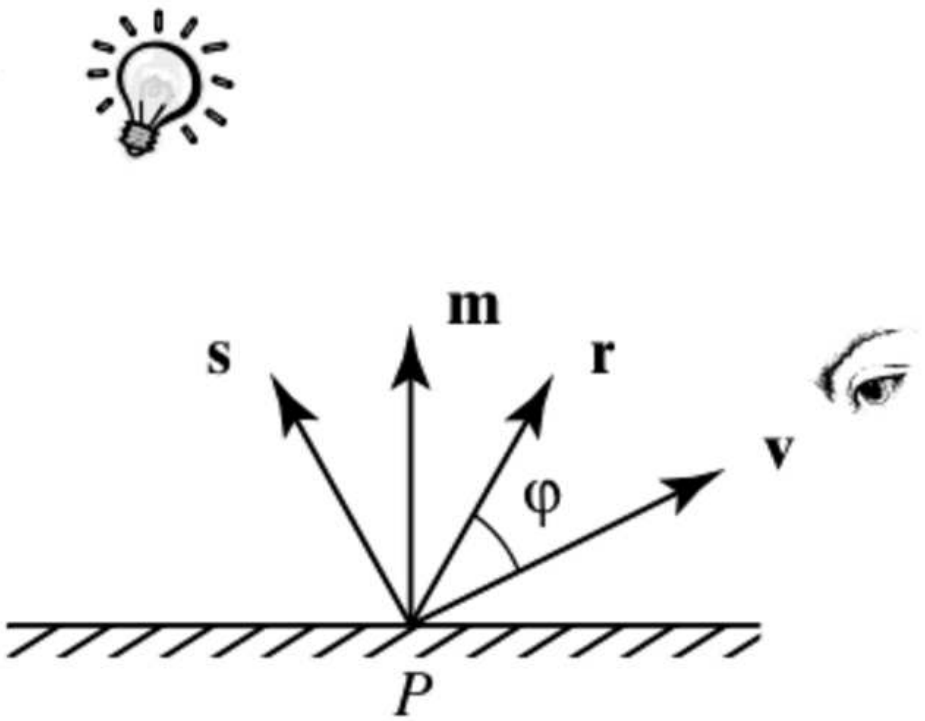


Image from Hill, Chap. 8

- Phong shading intensity depends on the angle to the mirror direction

Phong Specular reflection

- Phong (more precise) : The light reflected in the direct mirror direction is the strongest. Light reflected in other directions is proportional to the f th power of the cosine to the mirror direction.

Specular reflection (2)

- The mirror direction r can be found like this: $\vec{r} = -\vec{s} + 2\frac{(\vec{s}\cdot\vec{m})}{|\vec{m}|^2}\vec{m}$
- $I_{sp} = I_s \rho_s \left(\frac{\vec{r}}{|\vec{r}|} \cdot \frac{\vec{v}}{|\vec{v}|}\right)^f$
- Again, I_s is the intensity of the light source, ρ_s is the specular reflection coefficient. f is determined experimentally and lies between 1 and 200.
- Finding $\vec{r}$ is computationally expensive.

Different values of *f*

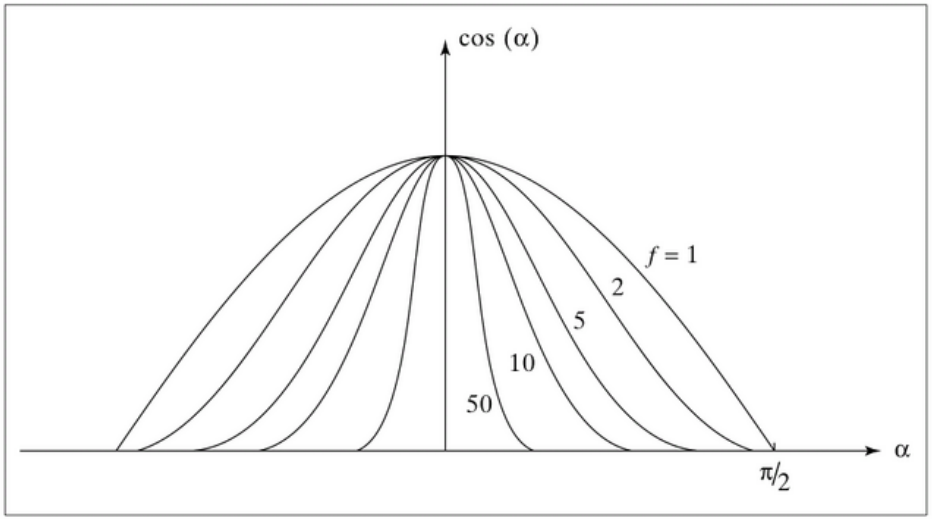


Image from Hill, Chap. 8

- *f* determines the falloff.

Avoid finding $\vec{r}$

- Instead of finding the correct $\vec{r}$, compute the *halfway vector* between $\vec{s}$ and $\vec{v}$: $\vec{h} = \vec{s} + \vec{v}$.
- $\vec{h}$ gives the direction in which the brightest light is to be expected if all vectors are in the same plane.

The halfway vector

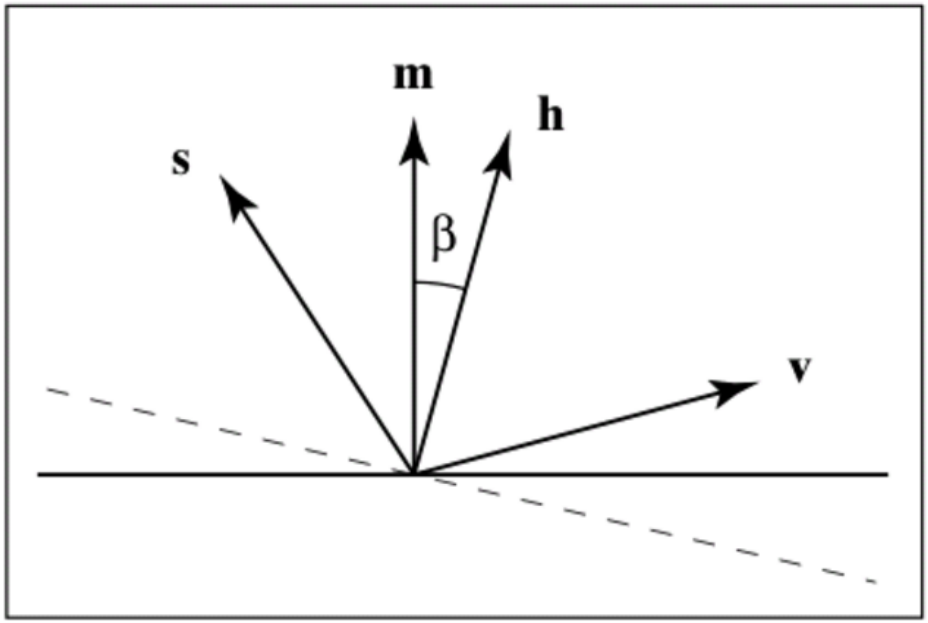


Image from Hill, Chap. 8

- An illustration of the halfway vector

Avoid finding $\vec{r}$

- Instead of finding the correct $\vec{r}$, compute the *halfway vector* between $\vec{s}$ and $\vec{v}$: $\vec{h} = \vec{s} + \vec{v}$.
- $\vec{h}$ gives the direction in which the brightest light is to be expected if all vectors are in the same plane.
- $I_{sp} = I_s \rho_s \left(\frac{\vec{h}}{|\vec{h}|} \cdot \frac{\vec{m}}{|\vec{m}|} \right)^f$
- The falloff of the cosine function is now a different one. But this can be compensated by chosing a different f.
- Of course all these models are not very realistic, but easy to compute.

Specular reflection

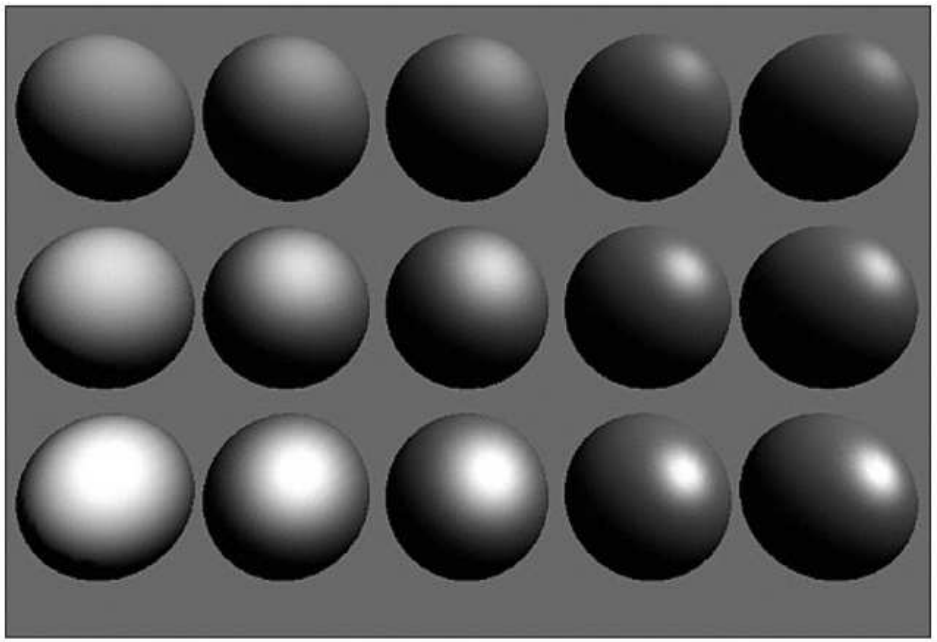


Image from Hill, Chap. 8

- f varies from 3 to 200, ρ_S varies from 0.25 to 0.75

Ambient Light

- Ambient light is a uniform background light that exists everywhere in the scene. It models the light that is usually reflected from surfaces.
- Its source has an intensity I_a. Every surface has an ambient reflection coefficient ρ_a (often equal to ρ_d).
- All light contributions combined:
  $I = I_a \rho_a + I_d \rho_d \times \text{lambert} + I_{sp} \rho_s \times \text{phong}^f$

Ambient Light

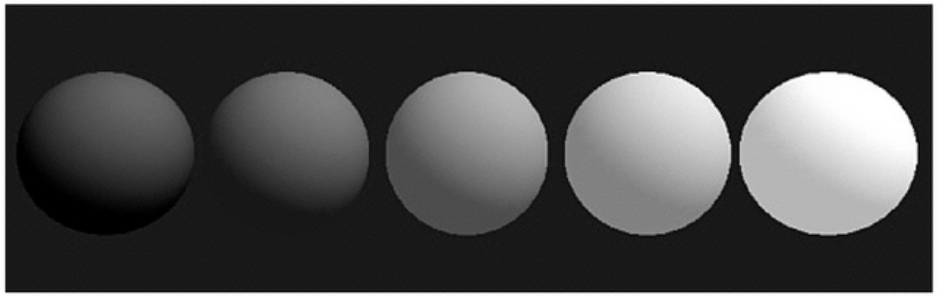


Image from Hill, Chap. 8

- Ambient light coefficients from 0 to 0.7

Color Light

- It's easy to extend this model to colored light: Simply treat the three color components separately:
- $I_r = I_{ar}\rho_{ar} + I_{dr}\rho_{dr} \times \text{lambert} + I_{spr}\rho_{sr} \times \text{phong}^f$
  $I_g = I_{ag}\rho_{ag} + I_{dg}\rho_{dg} \times \text{lambert} + I_{spg}\rho_{sg} \times \text{phong}^f$
  $I_b = I_{ab}\rho_{ab} + I_{db}\rho_{db} \times \text{lambert} + I_{spb}\rho_{sb} \times \text{phong}^f$

In OpenGL

- Creating a light source:

```
GLfloat myLightPosition[]={3.0,6.0,5.0,1.0};
glLightfv(GL_LIGHT0,GL_POSITION,
      myLightPosition);
glEnable(GL_LIGHTING);
glEnable(GL_LIGHT0);
```

- OpenGL handles up to 8 light sources `LIGHT0` to `LIGHT7`.
- Giving a vector instead of a position creates a light source of infinite distance. This type of light source is called *directional* instead of *positional*.

Local and directional sources

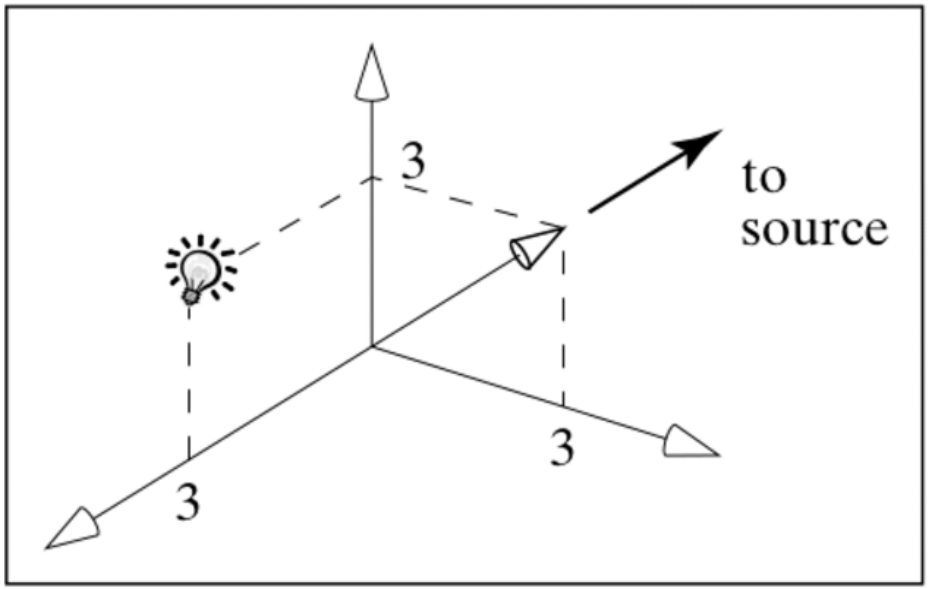


Image from Hill, Chap. 8

- Sources at (0,3,3,1) and (3,3,0,0)

Colored Light

- Creating a light source:

```
GLfloat amb0[]={0.2,0.4,0.6,1.0};
GLfloat diff0[]={0.8,0.9,0.5,1.0};
GLfloat spec0[]={1.0,0.8,1.0,1.0};
glLightfv(GL_LIGHT0,GL_AMBIENT,amb0);
glLightfv(GL_LIGHT0,GL_DIFFUSE,diff0);
glLightfv(GL_LIGHT0,GL_SPECULAR,spec0);
```

- Colors are specified in the RGBA model. A stands for *alpha*. For the moment, we set alpha to 1.0.

Spot Lights

- By default, OpenGL uses point light sources.
- Creating a spot light source:

```
glLightf(GL_LIGHT0,GL_SPOT_CUTOFF,45.0);
glLightfv(GL_LIGHT0,GL_SPOT_EXPONENT,4.0);
GLfloat dir[]={2.0,1.0,-4.0};
glLightfv(GL_LIGHT0,GL_SPOT_DIRECTION, dir);
```

Spot Light model

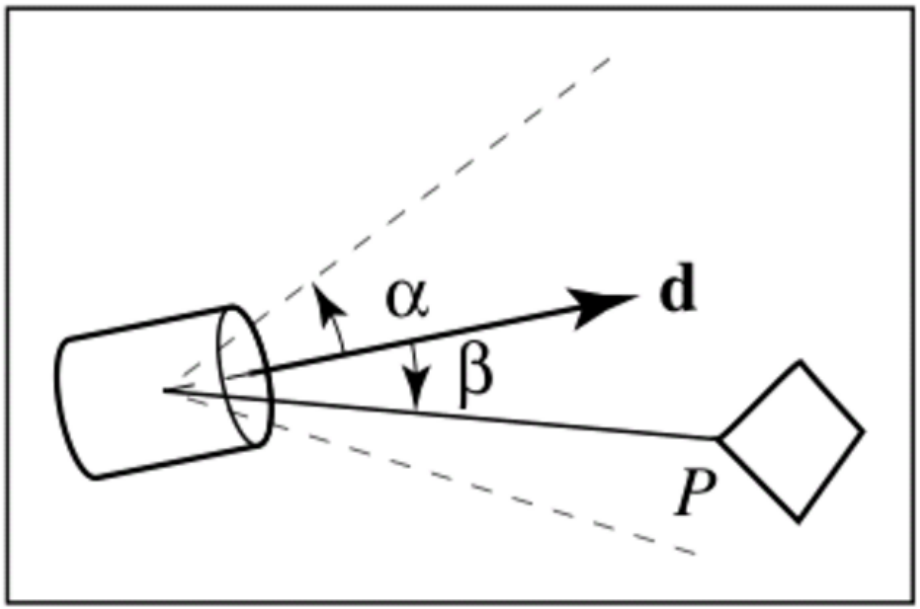


Image from Hill, Chap. 8

- A spotlight with cutof angle α. The intensity varies with $\cos^{\epsilon}(\beta)$

Other light properties

- Light attenuation:

```
glLightf(GL_LIGHT0,
        GL_CONSTANT_ATTENUATION, 2.0);
glLightf(GL_LIGHT0,
        GL_LINEAR_ATTENUATION, 0.2);
glLightf(GL_LIGHT0,
        GL_QUADRATIC_ATTENUATION, 0.1);
```

- Ambient Light:

```
GLfloat amb[]={0.2,0.3,0.1,1.0};
glLightModelfv(
        GL_LIGHT_MODEL_AMBIENT, amb);
```

Other light properties

- Recompute $\vec{v}$ for every point

```
glLightModeli(
        GL_LIGHT_MODEL_LOCAL_VIEWER,
        GL_TRUE);
```

- Faces are two-sided:

```
glLightModeli(
        GL_LIGHT_MODEL_TWO_SIDE,
        GL_TRUE);
```

Material properties

- Set the diffuse component for a surface:

```
GLfloat myDiffuse[]={0.8,0.2,0.0,1.0};
glMaterialfv(GL_FRONT,GL_DIFFUSE,myDiffuse);
```

- The first parameter choses the face: `GL_FRONT`, `GL_BACK`, `GL_FRONT_AND_BACK`
- The second parameter choses the coefficients: `GL_AMBIENT`, `GL_DIFFUSE`, `GL_SPECULAR`,`GL_AMBIENT_AND_DIFFUSE`,`GL_EMISSION`

Summary

- Shading
- Diffuse Component
- Specular Component
- Ambient Component
- Light Sources