Chapter 14: Illumination and Shading

Illumination models, also called lighting models or shading models, are used to shade surfaces based on the position, orientation, and characteristics of the surfaces and the light sources illuminating them.

- **Self-Illuminating Objects**

  Some object's body may emit light. The resulting intensity, I, perceived from the camera is:

  $$I = k_i$$

  where the coefficient k_i is the intrinsic intensity of the object.

- **Ambient Light**

  - An ambient light is a diffuse, non-directional source of light, typically the result of multiple reflections of light from many surfaces in the environment.
  - It is a uniform constant at any point in the environment.

Illumination Euqation of ambient Light:

$$I = I_a * k_a$$

where

- I_a is the intensity of ambient light (global for all objects)
- k_a is the ambient-reflection coefficient of an object's surface that is determined by the material property of the surface.

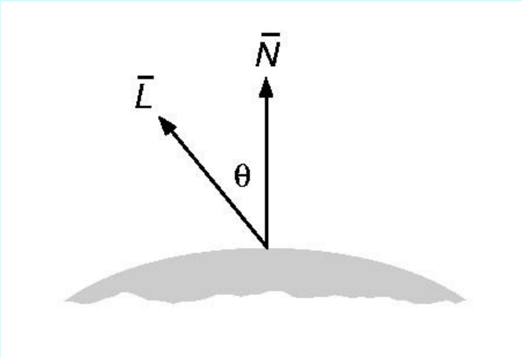
Ambient light will generate a uniform intensity across the surface of an object.

- **Diffuse Reflection**

Point Light Source - whose rays emanate uniformly in all directions from a single point.

Lambertian Reflection (associated with dull, matte surfaces)

- The surface reflects light with equal intensity in all direction.
- As a result, the surface appears equally bright from all viewing angles.

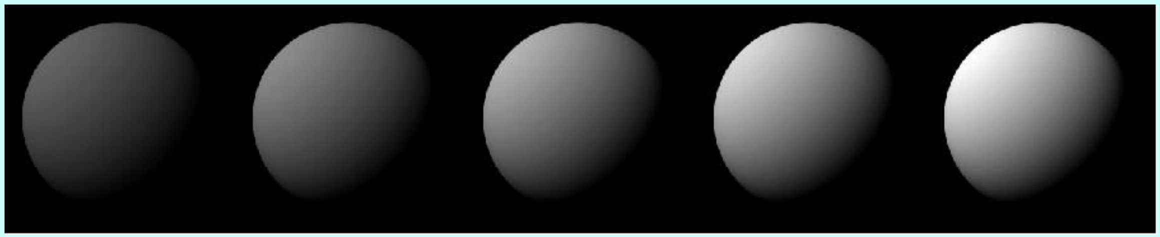


Diffuse Illumination Equation:

$$I = I_p k_d \cos\theta = I_p k_d (\overline{N} \bullet \overline{L})$$

where

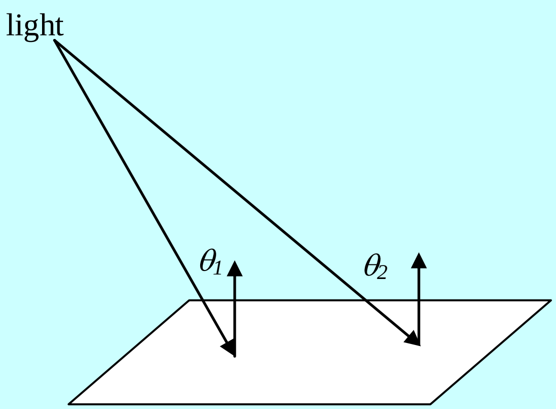
- I_p is the point light source's intensity;
- k_d, for $0 <= k_d <= 1$, is the material's diffuse reflection coefficient;
- θ, for $0 <= \theta <= 90^0$, is the angle between the surface normal and light direction. (Assume that *N* and *L* are unit length vectors)

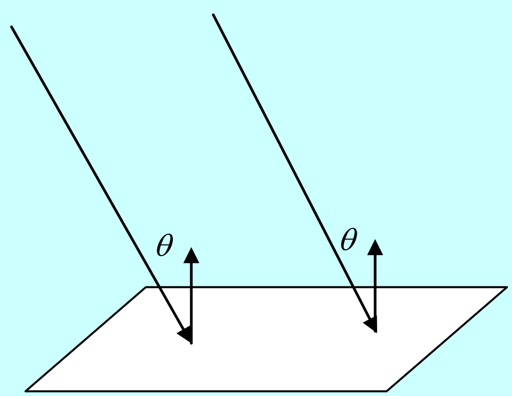


Directional Light Source - if a point light source is sufficiently distant from the object being illuminated, then light rays are essentially parallel.

In this case, L is a constant for all points on an object.

Compare the difference between the point light source and the directional light source:





Combination of Ambient and Point Light Source:

$$I = I_a k_a + I_p k_d (\overline{N} \bullet \overline{L})$$

- **Light Source Attenuation**

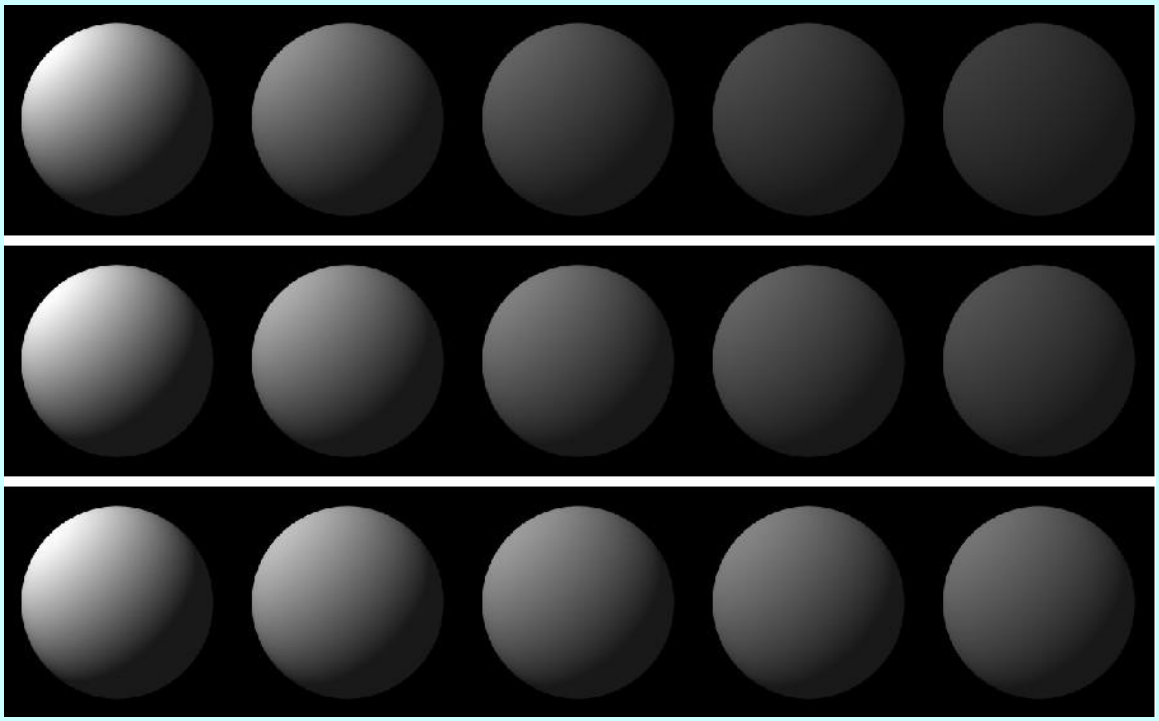
The distance of the object is considered - the energy from a point light source that reaches a given part of a surface falls off as the inverse square of d_L, the distance the light travels from the point light source to the surface.

$$I = I_a k_a + f_{att} I_p k_d (\overline{N} \bullet \overline{L})$$

where f_{att} is the light-source attenuation factor.

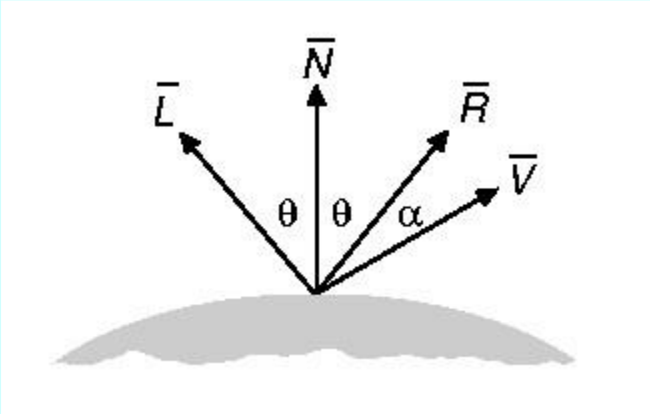
A simple choice for fatt is:

$$f_{att} = 1/d_l^2$$



- **Specular Reflection** (associated with shining surfaces)

On a perfect specular surface (such as mirror), the light is reflected ONLY in a single direction:



Phong Illumination Model (for non-perfect specular reflection)

$$I = I_a k_a + f_{att} I_p [k_d \cos\theta + W(\vartheta) \cos^n \alpha]$$

or

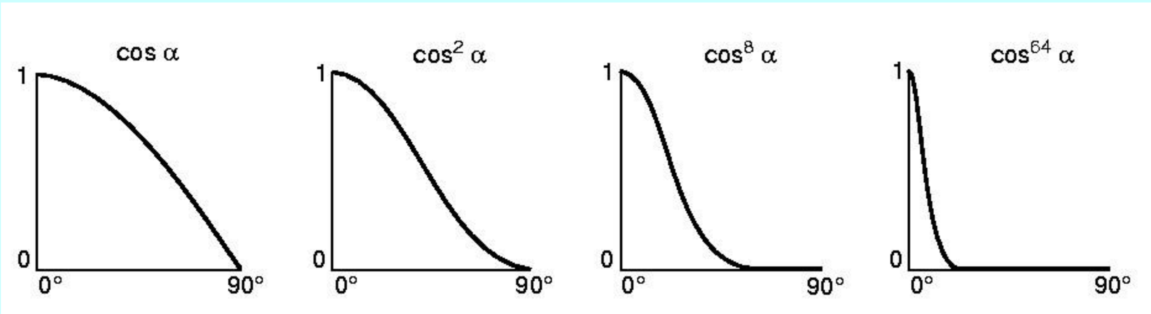
$$I = I_a k_a + f_{att} I_p [k_d (\bar{N} \bullet \bar{L}) + k_s (\bar{R} \bullet \bar{V})^n]$$

where

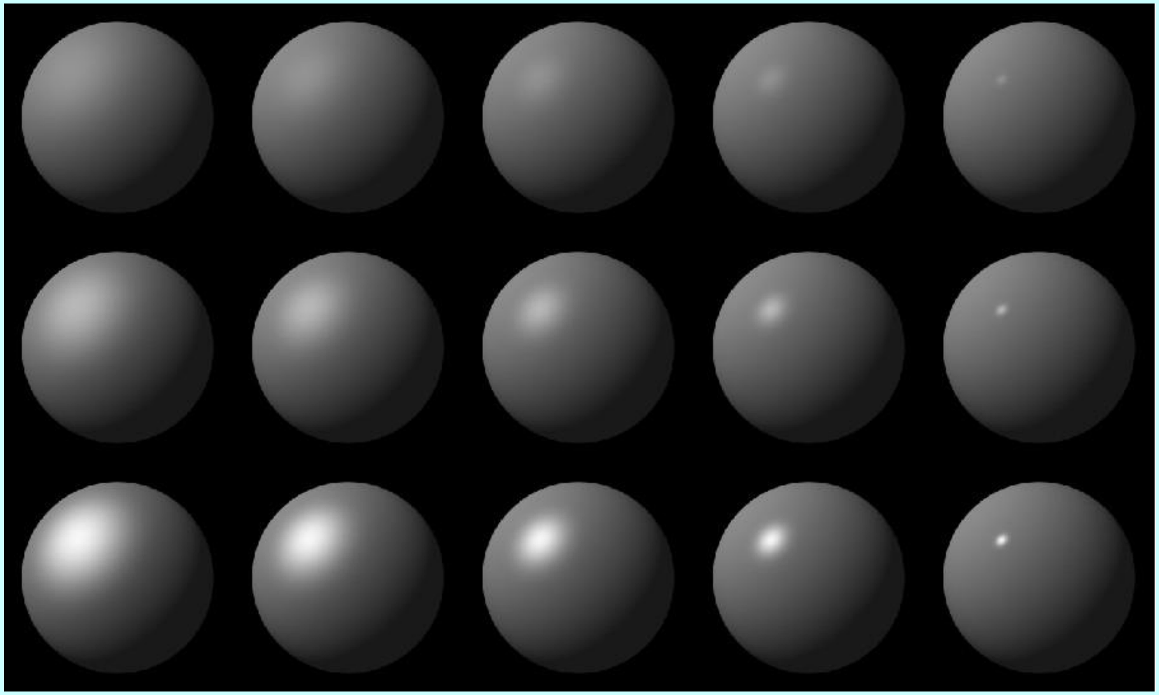
- α is the angle between the viewing direction V and the reflection direction R.
- $W(\theta)$ is the fraction of specularly reflected light.

- $W(\theta)$ is typically set to a constant k_s, the simplified material's specular-reflection coeffcient, and $0<=k_s<=1$.
- The value of k_s is selected experimentally.

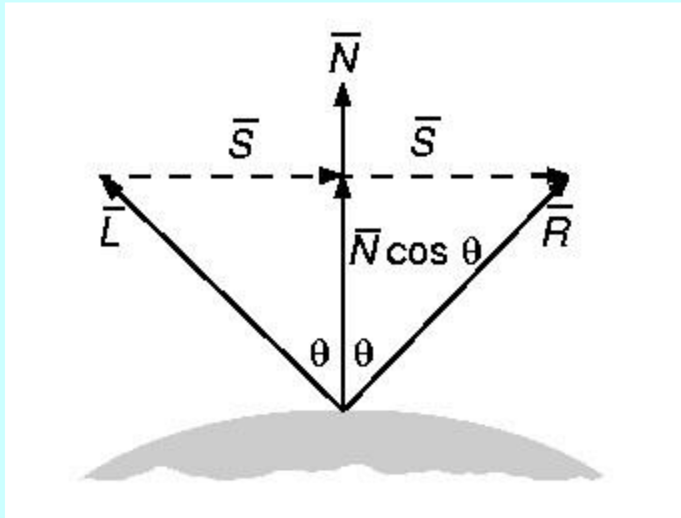
The effect of power n:



As n increses, the specular reflection is reduced to a very small range, giving a perception of that the surface is more shining.



How to calculate the reflection vector R?



Assume that N and L are unit length vectors.

$$\bar{R} = \bar{N}\cos\theta + \bar{S}$$

where $|\bar{S}| = \sin\theta$, and

$$\bar{S} = \bar{N}\cos\theta - \bar{L} \, .$$

Thus,

$$\bar{R} = 2\bar{N}\cos\theta - \bar{L} \, .$$

We have also $\cos\theta = \overline{N} \bullet \overline{L}$. We obtain:

$$\overline{R} = 2\overline{N}(\overline{N} \bullet \overline{L}) - \overline{L}$$

and

$$\overline{R} \bullet \overline{V} = (2\overline{N}(\overline{N} \bullet \overline{L}) - \overline{L}) \bullet \overline{V}$$

- **Shading Interpolation Models for Polygons**

Constant Shading (or flat shading, or faceted shading)

- This approach applies an illumination model once to determine a single intensity value that is applied to shade an entire polygon.

This approach is reasonable is several assumptions are true:

- Directional light, i.e. light rays are in parallel (or in other words, the light source is at infinity);
- The viewer is at infinity.

Interpolated Shading

- The shading information is linearly interpolated across a polygon from the values determined at its vertices.

Gauraud Interpolated Shading (also called intensity interpolated shading, or color interpolated shading):

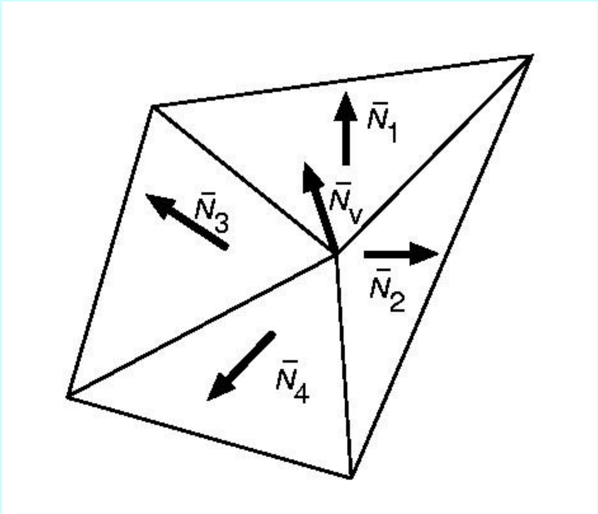
- A shading value is calculated at each vertex first;
- The shading value at any interior pixel of a polygon is calculated by a linear interpolation from the shading values at vertices.

Features:

- It generates a smooth variation of shading across a polygon surface;
- It also eliminates intensity discontinuities between the neighboring polygons on a polygon mesh. (WHY?)

Question: *How to you calculate the shading value at a vertex?*

You would need the surface normal vector at a vertex in order to apply the above illumination models.



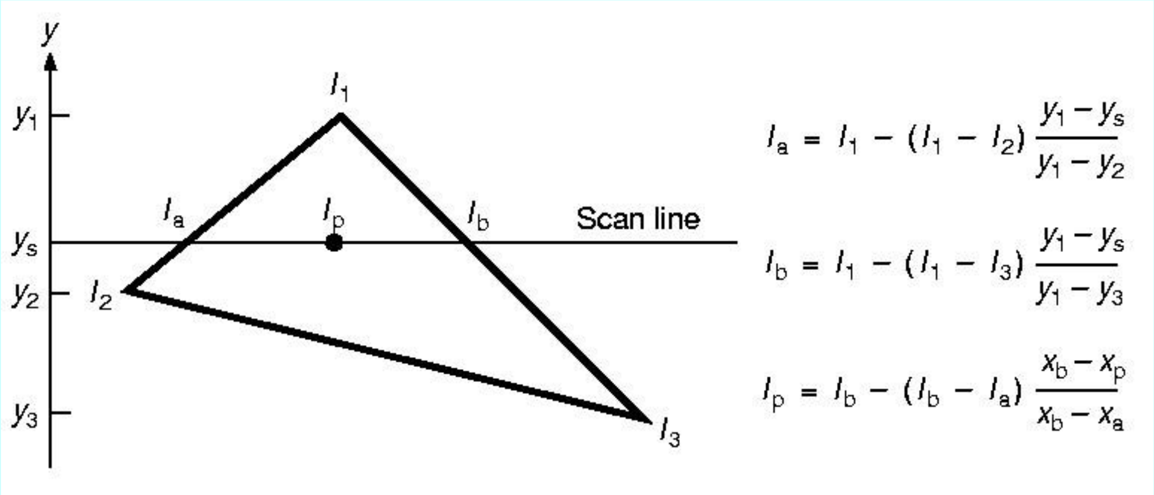
Consider the vertex N_v, that is shared by the four polygons.

- Mathematically, there is no definition for the normal vector at this vertex.
- Practically, we have two ways to obtain a normal vector at this vertex.
  1) The polygon mesh is an approxiamation of a curved surface. In the mesh generation process, we can save the accurate normal vector for each vertex derived from the original curved surface.
  2) We can define an approximate normal vector by taking the average of the normal vectors of neighboring polygons that share this vertex:

$$\overline{N}_v = \frac{\sum_{1 \le i \le n} \overline{N}_i}{\left| \sum_{1 \le i \le n} \overline{N}_i \right|}$$

Once the normal vector is available at each vertex, you can apply one of the illumination models given above to calculate the intensity at each vertex.

The interpolation can be integrated into your 2D polygon rendering routine:



- In each edge node, you may add another member, called I, and dI.

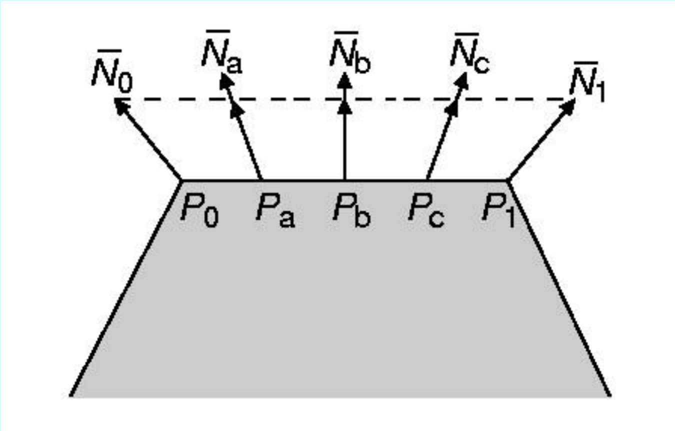
- I and dI are handled in exactly the same manner as that for z and dz.
- When we move from one scanline to the next, I is updated by dI.
- On each scanline, we have a pair of I's for a span.
- The I is linearly interpolated between the two I's across the span.

The result of Gouraud Shading is shown in the leftmost picture below:

- In the above example, a highlight is assumed at the lower-left vertex.
- By Gouraud Shading, this high intensity of specular reflection will spread linear to the interior of the polygon which is not realistic.
- An improvement can be made by Phong Interpolated Shading below.

Phong Interpolated Shading (also known as normal-vector interpolated shading)

- Phone Interpolated Shading interpolates the normal vector for each interior pixel across a polygon;
- An illumination model is applied at each pixel to calculate the shading value.



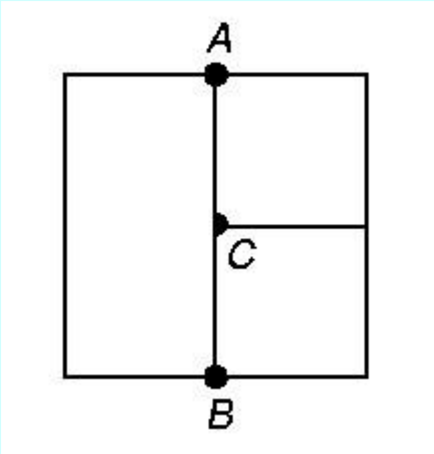
Features:

It produces handles specular reflection much better:

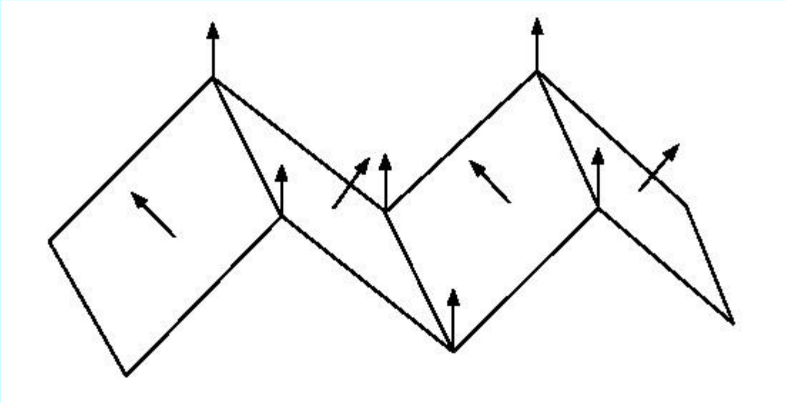
- It will not spread a high intensity specular reflection at a corner into the polygon widely (the 2nd picture from the left above).
- It will not miss a high intensity specular reflection inside (the rightmost one above), while Gouraud Shading would (the 3rd picture from the left above). (WHY?)

Problems with Interpolated Shading Methods:

- Polygonal silhouette
- Perspective distortion
- Orientation dependence
- Problems at shared vertices:



- Unrepresentative Vertex Normals



- **Shadows**

  - **Shadow algorithms** determine which (part of) surfaces can be "seen" from the light source (i.e. the parts that are not in shadow).
  - Therefore, shadow algorithms and visible-surface algorithms are essentially the same.

Point Light Sources

Assume there are m point light sources in the environment, the illumination model becomes:

$$I = I_a k_a + \sum_{1 \le i \le m} S_i f_{att} I_p [k_d (\bar{N} \bullet \bar{L}) + k_s (\bar{R} \bullet \bar{V})^n]$$

where

s_i = 0, if the light i is blocked at a point on the surface

1, if the light i is not blocked at the point on the surface.

How to determine s_i?

The Two-Pass Z-buffer Shadow Algorithm

1. Take the *ith* light source position as the view point, and calculate and store just the z-buffer (normally called light-buffer);
2. Take the camera position as the view point, calculate both image and Z-buffer.
   (a) for a surface point, with a 3D coordinates (x, y, z) in the camera coordinate system, that is visible from the camera position,
      transform (x, y, z) into a coordinates (x', y', z') in the light's coordinate system;
   (b) project (x', y', z') onto the projection plane in the light coordinate system, yielding (i, j) and z';

(1) compare z’ with light’s z_L = Z_buffer[i][j];
(2) if z’> z_L, z’ is farther from the the light, thus the point (x, y, z) is in shadow. So, set $s_i = 0$;
Otherwise, the point (x, y, z) is not in shadow. Set $s_i = 1$.

- In the above calculation, the 3D coordinates (x, y, z) in the camera coordinate system, correspinding to a pixel (i, j) on the image plane, is to be calculated.
- Recall the Z-buffer hidden-surface algorithm we discussed in Chapter 13. We have presented a mathod (based on ray-tracing idea) to find the (x, y, z).

------------------------------ END ------------------------------------