Illumination Models and Surface Rendering Methods.

In order to obtain better realism of our graphical objects, we should consider the distribution and intensity of the light sources and the optical properties of the objects (opacity, color, reflectance, surface texture, etc.).

**Light Sources:**

Point light: Light source without size  
Extended source light: Light source with size  
Background light: Some fixed ambient lighting value.

We will be dealing with point lights If one point light is far away from the object, we may consider that the rays of light are parallel.

A light source could be characterized by six parameters: x, y, z, its position coordinates, its intensity I, and the direction of emission, and the angles of emission, $\phi$ and $\theta$.

When ray of light hits a surface, several things can happen:

A reflection can happen: All the light hitting the surface is returned by the object  
A refraction can take place: Certain portion of the light hitting the surface penetrates the object.

**Reflection:**  
Diffuse – Reflection that goes in all directions  
Specular – Reflection that goes only in one direction

**Refraction:**  
Diffuse - Light is transmitted in all directions inside the object  
Specular – Light is transmitted in only one direction.

Remark: The equation that we are going to obtain is an abstraction. We must consider an equation for the red wavelength, another for the blue wavelength and another the green one.

**Specular Reflection:** Snell law governs it. The angle of incidence ($\alpha_1$) is equal to the angle of reflection ($\alpha_2$). See figure.
But we are interested in the intensity of the light coming from the incident point into the eye, through the line of vision, whose vector of direction $\vec{V}$ is also called the viewing direction. This line is not necessarily in the plane determined by $\vec{L}$, $\vec{N}$, and $\vec{R}$. If we call $I_p$ the intensity of the light source, we could write the equation

$$I_s = k_s (\vec{R} \cdot \vec{V}) I_p \frac{d}{d^2}$$

Where $k_s$ is the specular reflectivity, $d$ is the distance from the eye to the incident point, plus the distance to the source, $\vec{R}$ and $\vec{V}$ are normalized vectors, such that $\vec{R} \cdot \vec{V} = \cos \phi$.

Unfortunately applying the physics laws that way yield pictures with exaggerated artifacts. Therefore two different equations are used:

i) $$I_s = k_s (\vec{R} \cdot \vec{V}) I_p \frac{d}{d^2} + I_p$$

Where $d_0$ is small.

Or.

ii) $$I_s = k_s (\vec{R} \cdot \vec{V})^n I_p$$

Where $n$ gives the glossiness of the surface.

Diffuse reflection:

Let us denote $k_d$, $0 \leq k_d \leq 1$, the diffuse reflectivity. Then, by the same consideration that we took into account previously, we could write:

$$I_d = k_d (\vec{N} \cdot \vec{L}) \frac{I_p}{d + d_0}$$

Or
\[ I_d = k_d (\vec{N} \cdot \vec{L}) I_p \]

We assume the intensity is isotropic, i.e., the same in all directions. The factor \((\vec{N} \cdot \vec{L}) = \cos \theta\), represents the factor of the light intensity that hits the surface. See figure

Our model

1) All light sources are point sources
2) All other illumination comes from isotropic ambient light
3) All objects on the scene are opaque (no refraction)
4) You may use distance (or not) in your formulas for intensity depending on the model you choose.

Therefore, given a light source, the intensity seen to be arriving to a surface is given by

\[ I_t = k_a I_a + k_s (\vec{R} \cdot \vec{V})^n \frac{I_p}{d + d_0} + k_d (\vec{N} \cdot \vec{L}) \frac{I_p}{d + d_0} \]

With linear addition of intensities, if we consider several light sources.

Note that other models will substitute \(d + d_0\) by \(a + bd + cd^2\), or completely omit it.

In order to compute the illumination, we need, for the surface point corresponding to a pixel, the following vectors:

1) The direction vector to the light source, \(\vec{L}\).
2) The normal vector to the surface, \(\vec{N}\).
3) The viewing vector, \(\vec{V}\).
4) The reflection vector, \(\vec{R}\).

If there are more than one light sources, we must repeat the process for each one.
Computations of vectors

Let us assume that \((x, y, z)\) are the coordinates of the point in the surface of an object, corresponding to the pixel on the screen. Let \((S_x, S_y, S_z)\) the coordinates of the light source, and \((E_x, E_y, E_z)\) the coordinates of the position of the eye. Then

1) \[
\vec{L} = \frac{(S_x - x, S_y - y, S_z - z)}{\sqrt{(S_x - x)^2 + (S_y - y)^2 + (S_z - z)^2}}
\]

2) The normal vectors to each polygon are easy to compute using the counterclockwise description of the vertices.

3) \[
\vec{V} = \frac{(E_x - x, E_y - y, E_z - z)}{\sqrt{(E_x - x)^2 + (E_y - y)^2 + (E_z - z)^2}}
\]

4) The reflection vector \(\vec{R}\) is a little more complicated to compute. We are going to use two different methods to compute it.

a) Geometric method.

b) Analytic method.

a) See next figure.

We assume that the vectors \(\vec{L}, \vec{N}\), and \(\vec{R}\) are normalized (i.e. they have norm 1). Draw \(\vec{A}\) in the opposite direction of \(\vec{L}\), at the point where \(\vec{N}\) ends. We obtain an isosceles triangle, with one side having length one (the one corresponding to \(\vec{N}\)). The other two sides have length \(a\). Using the cosine theorem, we can write

\[a^2 = a^2 + 1 - 2 \cdot a \cdot 1 \cdot \cos \phi.\]
Solving for \( a \), we obtain \( \cos \phi = \frac{1}{2a} \)

On the other hand, using vector addition, we can write

\[
a \vec{R} = \vec{N} + \vec{A}, \quad \text{and consequently} \quad \vec{R} = \frac{\vec{N}}{a} - \vec{L} = \vec{N} 2 \cos \phi - \vec{L}
\]

and since \( \cos \phi = \vec{N} \cdot \vec{L} \) we can write

\[
\vec{R} = 2 \vec{N} (\vec{N} \cdot \vec{L}) - \vec{L}
\]

Note that if \( \vec{N} \cdot \vec{L} < 0 \), then the light hits the surface from the backside of the normal to the surface.

b) Again vectors \( \vec{L}, \vec{N} \) and \( \vec{R} \) are supposed to be normalized to be normalized. We also assume that \( \vec{L} \) and \( \vec{N} \) are not collinear, otherwise \( \vec{R} = \vec{L} \).

Let \( \cos \phi = \vec{N} \cdot \vec{L} \). The cross product of \( \vec{L} \) and \( \vec{N} \), \( \vec{N} \times \vec{L} \), gives us a vector \( \vec{P} \) normal to \( \vec{N} \times \vec{L} \).

\[
\vec{P} = (Ny Lz - Nz Ly, Nz Lx – Nx Lz, Nx Ly – Ny Lx)
\]

Therefore the vector \( \vec{R} = (x, y, z) \) satisfies the following three conditions:

i) \( \vec{P} \cdot \vec{R} = 0 \)

ii) \( \vec{N} \cdot \vec{R} = \cos \phi \).

iii) \( \| \vec{R} \| = 1 \)

Multiplying i) by \( N_z \), ii) by \( P_z \) and subtracting the these two, we obtain

\[
- N_z (xP_x + yP_y) + P_z (xN_x + yN_y) = P_z \cos \phi.
\]

\[
x (P_z N_x - P_x N_z) + y (P_z N_y - P_y N_z) = P_z \cos \phi. \quad (1)
\]

Multiplying i) by \( N_y \), ii) by \( P_y \) and subtracting the these two, we obtain
Solving for \( y \) in (1) and for \( z \) in (2), we can write

\[
y = \frac{P_z \cos \phi - x(P_z N_x - P_x N_z)}{P_z N_y - P_y N_z}
\]

\[
z = \frac{P_y \cos \phi - x(P_y N_x - P_x N_y)}{P_y N_z - P_z N_y}
\]

Placing these values into iii), we have

\[
x^2 + \frac{(P_z \cos \phi - x(P_z N_x - P_x N_z))^2}{(P_z N_y - P_y N_z)^2} + \frac{(P_y \cos \phi - x(P_y N_x - P_x N_y))^2}{(P_y N_z - P_z N_y)^2} = 1
\]

Multiplying both sides by \((P_z N_y - P_y N_z)^2\) we obtain

\[
x^2 (P_z N_y - P_y N_z)^2 + (P_z \cos \phi - x(P_z N_x - P_x N_z))^2 + (P_y \cos \phi - x(P_y N_x - P_x N_y))^2 = (P_z N_y - P_y N_z)^2
\]

This can be simplified in the form:

\[
a_x x^2 - b_x x + c_x = 0,
\]

where, for instance, \(a_x = (P_z N_y - P_y N_z)^2 + (P_z N_x - P_x N_z)^2 + (P_y N_x - P_x N_y)^2\).

Using the same approach for \( y \) and \( z \), we would obtain the quadratic equations for \( y \) and \( z \). It can be seen that \(a_x = a_y = a_z\). On the other hand, the \( b \) terms are different.

Since the vector \( \vec{L} \) is a solution for equations i), ii) and iii), its coordinates are solutions of the respective quadratic equations. So \( L_x \) is a solution of

\[
a_x x^2 - b_x x + c_x = 0, \text{ or } x^2 - \frac{b_x}{a_x} x + \frac{c_x}{a_x} = 0.
\]

Consider now the following product:

\[
(x - \frac{b_x}{a_x} + L_x) (x - L_x) = x^2 - \frac{b_x}{a_x} x + L_x x - L_x x + \frac{b_x}{a_x} L_x - L_x^2 =
\]
\[ x^2 - \frac{b_x}{a_x} x - (L_x^2 - \frac{b_x}{a_x} L_x) = x^2 - \frac{b_x}{a_x} x + \frac{c_x}{a_x} = 0. \]

The last substitution was performed because \( L_x \) is a solution of the quadratic equation.

But then we can conclude that \( x = \frac{b_x}{a_x} - L_x \) is the other solution for the vector \( R \), in the \( x \) coordinate.

If \( \cos \phi = 1 \), \( \vec{L} \) and \( \vec{N} \) are collinear, and \( \vec{R} = \vec{L} \).

Otherwise, we compute

\[
a = (P_z N_y - P_y N_z)^2 + (P_z N_x - P_x N_z)^2 + (P_y N_x - P_x N_y)^2
\]

\[
b_x = 2 \cos \phi \left[ P_y (P_y N_x - P_x N_y) + P_z (P_z N_x - P_x N_z) \right]
\]

\[
b_y = 2 \cos \phi \left[ P_x (P_x N_y - P_y N_x) + P_z (P_z N_y - P_y N_z) \right]
\]

\[
b_z = 2 \cos \phi \left[ P_x (P_x N_z - P_z N_x) + P_y (P_y N_z - P_z N_y) \right]
\]

and

\[
R_x = \frac{b_x}{a} - L_x
\]

\[
R_y = \frac{b_y}{a} - L_y
\]

\[
R_z = \frac{b_z}{a} - L_z
\]

and since \( \vec{L} \) and \( \vec{N} \) are not collinear, \( a \neq 0 \), and the computations can be performed.

**Polygonal Shading**

We do not want to compute the illumination very often, because it is very expensive to compute. There are several techniques to reduce the amount of times that we compute the illumination.
Flat shading or Lambert shading

Here we implicitly assume that: there is a distant viewer, so \( \mathbf{V} \) is constant; the light source is distant, so \( \mathbf{L} \) is constant. Since we are considering a plane, \( \mathbf{N} \) is constant. Then for our visible polygon in our list:

1) Choose a point \( \mathbf{P} \) on the polygon.
2) Find the normal \( \mathbf{N} \) to that polygon
3) Compute the illumination at \( \mathbf{P} \) using the illumination equation
4) Fill the projected polygon with the illumination computed in step 3)

An easy choice for the point \( \mathbf{P} \) consists in selecting a vertex.
A better choice for \( \mathbf{P} \) is to select a convex combination of the vertices, i.e.

\[
\mathbf{P} = \alpha_1 \mathbf{V}_1 + \alpha_2 \mathbf{V}_2 + \ldots + \alpha_n \mathbf{V}_n \quad \alpha_i \geq 0, \forall i, \sum_{i=1}^{n} \alpha_i = 1
\]

Flat shading is simple and fast, but the resulting pictures are very often not realistic.

Gourad shading

The basic idea here is to display each surface of the object with an illumination intensity that varies smoothly across it.

1) At each vertex we compute a fictitious surface normal corresponding to that vertex, as the average of the normal vectors to each of the polygons concurring to that vertex.
2) Compute the illumination intensity at each vertex, using the illumination equation and the normal vector found in step 1).
3) Fill the projected polygon with an intensity obtained by a bilinear interpolation of the intensities at the vertices.

Let us look at the bilinear interpolation using the next figure.
We want to find the illumination at the point R, which lies in the same scan line as P and Q. P and Q are at the intersection of the scan line with the edges AB and BC, respectively.

First we obtain the intensities at P and Q, knowing the intensities at the vertices of the edges, $I_A$, $I_B$, and $I_C$, using a linear interpolation:

$$I_P = s I_A + (1 - s) I_B \quad \text{with} \quad s = \frac{PB}{AB}$$

$$I_Q = t I_C + (1 - t) I_B \quad \text{with} \quad t = \frac{QB}{CB}$$

We don’t need to compute the lengths PB, AC, QB and CB, to obtain these values. The same ratios are obtained using the differences of the y-coordinates (using similar triangles).

$$\frac{PB}{AB} = \frac{y_p - y_B}{y_A - y_B} \quad \text{and} \quad \frac{QB}{CB} = \frac{y_Q - y_B}{y_C - y_B}$$

Note that the denominators can never be zero because the fill in algorithm ignores edges parallel to the scan line.

Once we have the intensities at P and Q, we again use a linear interpolation to find the intensity at R.

$$I_R = s_R I_P + (1 - s_R) I_Q \quad \text{with} \quad s_R = \frac{RQ}{PQ}$$

and using an argument similar to the previous one, $s_R = \frac{RQ}{PQ} = \frac{x_r - x_Q}{x_P - x_Q}$
But we don’t even need to compute $s_R$ because it can be obtained incrementally for a given scan line.

Consider $M$ and $N$ two consecutive pixels on the same scan line:

$$I_M = s_M \cdot I_P + (1 - s_M) \cdot I_Q$$

$$I_N = s_N \cdot I_P + (1 - s_N) \cdot I_Q$$

By subtraction we obtain $I_N = I_M + (s_N - s_M)(I_P - I_Q)$.

The term $(s_N - s_M)(I_P - I_Q)$ needs to be computed only once for every scan line. Therefore from one pixel to the next on the scan line, we only need to make a simple addition.

In fact, this term can be expressed using a difference on the $x$-coordinates of the intersection of the scan line with the edges:

$$(s_N - s_M)(I_P - I_Q) = \frac{x_M + 1 - x_Q}{x_P - x_Q} - \frac{x_M - x_Q}{x_P - x_Q}(I_P - I_Q) = \frac{I_P - I_Q}{x_P - x_Q}$$

There are two problems with Gouraud shading:

1) A surface with corners, like the one on the picture, will look flat.

2) Highlights from specular reflection are poorly represented. They cannot move. They disappear and then appear at a different location.

Phong shading:

It is similar to Gouraud shading in its use of the vertex normal vector. It is different because it interpolates the vertex normal vectors instead of the intensities at the vertices.
1) At each vertex we compute a fictitious surface normal corresponding to that vertex, as the average of the normal vectors to each of the polygons concurring to that vertex.

2) Compute the normal vector at each pixel, using the bilinear interpolation method and the normal vectors found in step 1).

3) Fill the projected polygon with an intensity obtained by using the illumination equation and the normal vectors found in step 2).

For step 2) the procedure is similar to the one used in Gouraud method. See figure

Now if \( N_A \), \( N_B \) and \( N_C \) are the normal vectors at the vertices A, B, and C, then using bilinear interpolation we can write:

\[
N_P = s N_A + (1 - s) N_B \quad \text{with} \quad s = \frac{y_P - y_B}{y_A - y_B}
\]

\[
N_Q = t N_C + (1 - t) N_B \quad \text{with} \quad t = \frac{y_Q - y_B}{y_C - y_B}
\]

\[
N_R = s_R N_P + (1 - s_R) N_Q \quad \text{with} \quad s_R = \frac{x_R - x_Q}{x_P - x_Q}
\]

And using consecutive pixels in the same scan line:

\[
N_N = N_M + (s_N - s_M) (N_P - N_Q) = \frac{N_P - N_Q}{x_P - x_Q}
\]

As soon as we obtain the normal vector \( N_R \), we normalize it. Then we found the illumination for that pixel, assuming that the vectors \( \vec{V} \) and \( \vec{L} \) are constant. Still you may
want to change them, but then you must work in the object space, finding the points in
the 3D polygons that are projected into the pixel, to compute $V$ and $L$.

Phong approach is more expensive than Gouraud, but it yields smoothly shaded objects
that look better than the Gouraud-shaded ones. It does not present the problem with
highlights of the specular reflection.

**Example:** A surface is defined by three points $A(1,0,1)$, $B(1,0,0)$ and $C(1,1,1)$. An
illumination source is located at a large distance from this surface in the direction $\frac{1}{3\sqrt{10}}$
$(1, 5, 8)$. Determine the intensity of the light necessary at this source if the shaded
intensity for the surface is to be set at 2. Assume there is no specular reflection and the
diffuse reflectivity of the surface is 0.3

We know that $I = I_s k_d \cos \phi$, where $\cos \phi = \overrightarrow{N} \cdot \overrightarrow{L}$.

Since $\overrightarrow{N} = (1, 0, 0)$ and $\overrightarrow{L} = \frac{1}{3\sqrt{10}} (1, 5, 8)$, we have $\cos \phi = \frac{1}{3\sqrt{10}}$.

Therefore $2 = I_s 0.3 \frac{1}{3\sqrt{10}}$, and $I_s = 20 \sqrt{10} = 63.246$

**Variation of the example:** Same example as before but the specular reflectivity is 0.1,
the glossiness of the surface is 2, and the viewer is located at the point $(4, -2, -1)$.

Now $I = I_s k_d (\overrightarrow{N} \cdot \overrightarrow{L}) + I_s k_s (\overrightarrow{V} \cdot \overrightarrow{R})^2$

We need to find $\overrightarrow{V}$ and $\overrightarrow{R}$. We choose a point in the surface, as a linear combination of
the 3 vertices $P = \frac{1}{3} (1, 0, 1) + \frac{1}{3} (1, 0, 0) + \frac{1}{3} (1, 1, 1) = (1, \frac{1}{3}, \frac{2}{3})$ and therefore the
direction of

$\overrightarrow{V}$ will be $\overrightarrow{W} = (4, -2, -1) \cdot (1, \frac{1}{3}, \frac{2}{3}) = (3, -\frac{5}{3}, -\frac{1}{3})$. If we find the norm of this

vector $\overrightarrow{W}$, we obtain $\|\overrightarrow{W}\| = \sqrt{9 + \frac{25}{9} + \frac{1}{9}} = \sqrt{\frac{81 + 25 + 1}{3}} = \frac{\sqrt{107}}{3}$. Therefore $\overrightarrow{V} = \frac{3}{\sqrt{107}} (3, \frac{-5}{3}, \frac{-1}{3})$. 

Using the geometric approach we can compute \( \bar{R} \):

\[
\bar{R} = 2 \left( \vec{N} \cdot \vec{L} \right) \vec{N} - \vec{L} = 2 \left( \frac{1}{\sqrt{10}} \right) (1, 0, 0) - \frac{1}{\sqrt{10}} (1, 5, 8) = \frac{1}{\sqrt{10}} (1, -5, -8)
\]

Using the analytical approach for computing \( \bar{R} \):

\[
\cos \phi = \vec{N} \cdot \vec{L} = \frac{1}{\sqrt{10}}
\]

\[
\vec{P} = \vec{N} \times \vec{L} = \frac{1}{\sqrt{10}} (0, -8, 5)
\]

\[
a = 0^2 + \left( \frac{-8}{\sqrt{10}} \right)^2 + \left( \frac{5}{\sqrt{10}} \right)^2 = \frac{89}{90}
\]

\[
b_x = 2 \left( \frac{1}{\sqrt{10}} \right) \left[ \frac{-8}{\sqrt{10}} \left( \frac{-8}{\sqrt{10}} \right) \left( 1 - 0 \right) + \frac{5}{\sqrt{10}} \left( \frac{5}{\sqrt{10}} \right) \left( 1 - 0 \right) \right] = \frac{89}{90}
\]

\[
b_y = 2 \left( \frac{1}{\sqrt{10}} \right) \left[ 0 \left( 0 - \frac{5}{\sqrt{10}} \right) \left( 1 - 0 \right) + 0 \left( \frac{5}{\sqrt{10}} \right) \left( 0 - \frac{5}{\sqrt{10}} \right) \left( 0 - \frac{5}{\sqrt{10}} \right) \right] = 0
\]

\[
b_z = 2 \left( \frac{1}{\sqrt{10}} \right) \left[ 0 \left( 0 * 0 - \frac{5}{\sqrt{10}} \right) \left( 1 \right) + \frac{-8}{\sqrt{10}} \left( \frac{-8}{\sqrt{10}} \right) \left( 0 - \frac{5}{\sqrt{10}} \right) \right] = 0
\]

Therefore

\[
R_x = \frac{b_x}{a} - L_x = \frac{2}{3\sqrt{10}} \cdot \frac{1}{3\sqrt{10}} = \frac{1}{3\sqrt{10}}
\]

\[
R_y = \frac{b_y}{a} - L_y = 0 - \frac{5}{3\sqrt{10}} = -\frac{5}{3\sqrt{10}}
\]

\[
R_z = \frac{b_z}{a} - L_z = 0 - \frac{8}{3\sqrt{10}} = -\frac{8}{3\sqrt{10}}
\]

As we see, we obtain the same result using one method or the other.

\[
(\vec{V} \cdot \bar{R}) = \frac{3}{\sqrt{107}} \left( 3, -\frac{5}{3}, -\frac{1}{3} \right) \cdot \frac{1}{\sqrt{1070}} (1, -5, -8) = \frac{1}{\sqrt{1070}} (3 + \frac{25}{3} + \frac{8}{3}) = \frac{42}{3\sqrt{1070}} = \frac{14}{\sqrt{1070}}
\]
and \( (\bar{V} \cdot \bar{R})^2 = \frac{196}{1070} \)

Consequently

\[
2 = I_s (0.3 \frac{1}{3\sqrt{10}} + 0.1 \frac{196}{1070}). \quad I_s = \frac{2}{\frac{1}{10\sqrt{10}} + \frac{196}{10700}} = 40.048
\]

**Example:** For the rectangular plane defined by points A (0, 0), B (1, 0), C (1, 1) and D (0, 1), find the reflected intensity at point (0.5, 0.5) using the Gourad shading technique, assuming that the average intensities of reflected illumination at the four vertices are:

\( I_A = 8, \quad I_B = 9, \quad I_C = 2, \quad I_D = 4. \)

\[
I_P = \frac{4 + 8}{2} = 6, \quad I_Q = \frac{6 + 5}{2} = 5.5, \quad I_R = \frac{6 + 5.5}{2} = 5.75.
\]

Note that \( P = s (0, 0) + (1 - s) (0, 1) = (0, 0.5) \), which means that \( s = 0.5 \).
Similarly \( Q = t (1, 0) + (1 - t) (1, 1) = (1, 0.5) \) which again it gives us that \( t = 0.5 \) (This is a coincidence because the point chosen is \((0.5, 0.5)\) and we are dealing with a square with sides parallel to the axes. You could work with the point \((\frac{1}{3}, \frac{2}{3})\) on your own, and change the square to be a triangle that contains this point).

\[ R = u(0, 0.5) + (1 - u) (1, 0.5). \]

We again find that \( u = 0.5 \).

**Fast Phong shading:**

Since Phong shading requires a great amount of computations, there is a variant of the Phong method, the Fast Phong, which uses Taylor-series expansions to approximate some of the calculations. It also requires that all the polygons are reduced to triangles.

Since the Phong method interpolates the normal \( \vec{N} \) using the normal at the vertices, we can write that the normal \( \vec{N} \) at the position \((x,y)\) is given by the formula

\[ \vec{N} = \vec{A} x + \vec{B} y + \vec{C}, \]

where \( \vec{A}, \vec{B}, \vec{C} \) are vectors, given as a solution to the equations on the three known vertices, i.e.,
\[ \vec{N}_k = \vec{A}x_k + \vec{B}y_k + \vec{C} \quad \text{for } k = 1, 2, 3. \]

Here \((x_k, y_k)\) are the pixel coordinates of the vertices of the triangle.

Restricting ourselves to the diffuse reflection, we have:

\[ I(x, y) = k_d (\vec{N} \cdot \vec{L}) \, I_p, \]

where \(\vec{N}, \vec{L}\) are valued at the point \((x, y)\).

In our computations \(\vec{N}, \vec{L}\) were always unit vectors. Since we are now computing these values, we cannot always assume that the vectors given by \(\vec{N} = \vec{A}x + \vec{B}y + \vec{C}\) will be unitary. So,

\[ I(x, y) = k_d \left( \frac{\vec{N} \cdot \vec{L}}{|| \vec{N} || \cdot || \vec{L} ||} \right) I_p. \]

Let’s omit the constant factors \(k_d\) and \(I_p\) in the formula.

\[ \frac{\vec{N} \cdot \vec{L}}{|| \vec{N} || \cdot || \vec{L} ||} = \frac{\vec{L} \cdot (\vec{A}x + \vec{B}y + \vec{C})}{|| \vec{L} || \cdot || \vec{A}x + \vec{B}y + \vec{C} ||} = \frac{\vec{L} \cdot \vec{A}x + \vec{L} \cdot \vec{B}y + \vec{L} \cdot \vec{C}}{|| \vec{L} || \cdot || \vec{A}x + \vec{B}y + \vec{C} ||} \]

The last expression can be written in the form

\[ \frac{ax + by + c}{\sqrt{dx^2 + exy + fy^2 + gx + hy + i}}, \]

where the coefficients \(a, b, c, \text{ etc.} \) are obtained through the dot product of \(\vec{L}\) times \(\vec{A}\), \(\vec{L}\) times \(\vec{B}\), etc.. Expanding the denominator \(\sqrt{dx^2 + exy + fy^2 + gx + hy + i}\) as a Taylor series, performing the division, and keeping terms up to second degree in \(x\) and \(y\), we may write

\[ I(x, y) = k_d \, I_p( \, T_0 + T_1x + T_2y + T_3x^2 + T_4xy + T_5y^2 \) \]

where \(T_k\) is given in terms of \(a, b, c, d, \text{ etc.}\).

But we can make some simplifications in the calculations. We compute the starting point for a scan line, i.e. \(y\) will be constant. Therefore at the scan line \(x_j\) we have

\[ I(x_{k+1}, y_j) = I(x_k, y_j) + \Delta x_k, \]

\[ I(x_{k+1}, y_j) = k_d \, I_p( \, T_0 + T_1(x_k + 1) + T_2y_j + T_3(x_k + 1)^2 + T_4(x_k + 1)y_j + T_5y_j^2 \) \]
\[ I(x_k, y_j) = k_d I_p (T_0 + T_1 x_k + T_2 y_j + T_3 x_k^2 + T_4 x_k y_j + T_5 y_j^2) \]

\[ \Delta x_{k,j} = k_d I_p (T_1 + 2T_3 x_k + T_3 + T_4 y_j) \]

Define \( \Delta^{(2)} x_{k,j} = \Delta x_{k+1,j} - \Delta x_{k,j} = k_d I_p 2T_3 \)

We need the values of the illumination at the first 3 pixels at each scan line. Let’s call them \( x_0, x_1, x_2 \).

\[ I_0 = I(x_0, y_j) = k_d I_p (T_0 + T_1 x_0 + T_2 y_j + T_3 x_0^2 + T_4 x_0 y_j + T_5 y_j^2) \]
\[ I_1 = I(x_0 + 1, y_j) = k_d I_p (T_0 + T_1 x_1 + T_2 y_j + T_3 x_1^2 + T_4 x_1 y_j + T_5 y_j^2) \]
\[ I_2 = I(x_0 + 2, y_j) = k_d I_p (T_0 + T_1 x_2 + T_2 y_j + T_3 x_2^2 + T_4 x_2 y_j + T_5 y_j^2) \]

\[ \Delta x_{0,j} = I_1 - I_0 = k_d I_p (T_1 + 2T_3 x_0 + T_3 + T_4 y_j) \]
\[ \Delta x_{1,j} = I_2 - I_1 = k_d I_p (T_1 + 2T_3 x_1 + T_3 + T_4 y_j) \]

\[ \Delta^{(2)} x_{0,j} = \Delta x_{1,j} - \Delta x_{0,j} = k_d I_p 2T_3 \] Note that \( \Delta^{(2)} x_{0,j} \) is the same for all \( x_k \).

Looking backwards, we can compute.

\[ I(x_0 + 3, y_j) = I_3 \]

In fact, \( k_d I_p 2T_3 = \Delta^{(2)} x_{1,j} = \Delta x_{2,j} - \Delta x_{1,j} = (I_3 - I_2) - (I_2 - I_1) \).

So \( I_3 = I_2 + \Delta x_{1,j} + \Delta^{(2)} x_{1,j} = I_2 + \Delta x_{2,j} \)

Here \( \Delta^{(2)} x_{1,j} \) is a constant, and \( \Delta x_{1,j} \) can be kept from the previous calculations. So, we need a couple of addition to obtain the rest of the values of the illumination for the same scan line. The following table would be constructed:

<table>
<thead>
<tr>
<th>( t )</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>( I_0 ) (Given) ↓</td>
<td>( I_1 ) (Given) ↓</td>
<td>( I_2 ) (Given) ↓</td>
<td>Left + left down</td>
<td></td>
</tr>
<tr>
<td>( \Delta )</td>
<td>( I_1 - I_0 ) ↓</td>
<td>( I_2 - I_1 )</td>
<td>Left + left down</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( \Delta^{(2)} )</td>
<td>( I_2 - 2I_1 + I_0 ) →</td>
<td>( I_3 - 2I_2 + I_1 )</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Example:**

Assume that the values at the first 3 pixels of the scan line are: 1, 7, 23.

Let’s construct the table one value at a time
<table>
<thead>
<tr>
<th>t</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>1</td>
<td>7</td>
<td>23</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Δ</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Δ^{(2)})</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>t</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>1</td>
<td>7</td>
<td>23</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Δ</td>
<td>6</td>
<td>16</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Δ^{(2)})</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>t</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>1</td>
<td>7</td>
<td>23</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Δ</td>
<td>6</td>
<td>16</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Δ^{(2)})</td>
<td>10</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>t</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>1</td>
<td>7</td>
<td>23</td>
<td>49</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Δ</td>
<td>6</td>
<td>16</td>
<td>26</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Δ^{(2)})</td>
<td>10</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
</tr>
<tr>
<td>I</td>
<td>1</td>
<td>7</td>
<td>23</td>
<td>49</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\Delta$</td>
<td>6</td>
<td>16</td>
<td>26</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\Delta^{(2)}$</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>t</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>1</td>
<td>7</td>
<td>23</td>
<td>49</td>
<td>85</td>
<td></td>
</tr>
<tr>
<td>$\Delta$</td>
<td>6</td>
<td>16</td>
<td>26</td>
<td>36</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\Delta^{(2)}$</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>t</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>1</td>
<td>7</td>
<td>23</td>
<td>49</td>
<td>85</td>
<td></td>
</tr>
<tr>
<td>$\Delta$</td>
<td>6</td>
<td>16</td>
<td>26</td>
<td>36</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\Delta^{(2)}$</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td></td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>t</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>1</td>
<td>7</td>
<td>23</td>
<td>49</td>
<td>85</td>
<td></td>
</tr>
<tr>
<td>$\Delta$</td>
<td>6</td>
<td>16</td>
<td>26</td>
<td>36</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\Delta^{(2)}$</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Etc..