Normais e Iluminação

COMPUTAÇÃO GRÁFICA

Normais de planos

A normal de um plano é um vetor de comprimento 1 que é perpendicular ao plano.

A normal de um triângulo é um vetor de comprimento 1 que é perpendicular ao triângulo. É facilmente calculado tomando o produto vetorial de duas de suas arestas (o produto vetorial de a e b produz um vetor que é perpendicular tanto a **a** quanto a **b**), e normalizado: seu comprimento é trazido de volta a 1. Em pseudo-código :

```
triangle ( v1, v2, v3 )
edge1 = v2-v1
edge2 = v3-v1
triangle.normal = cross(edge1, edge2).normalize()
```

Normais de vértices

Por extensão, chamamos a normal de um vértice a combinação das normais dos triângulos circundantes. Isso é útil porque nos shaders de vértices, lidamos com vértices, não com triângulos, então é melhor ter informações sobre o vértice. E de qualquer forma, não podemos ter informações sobre triângulos no OpenGL. Em pseudo-código:

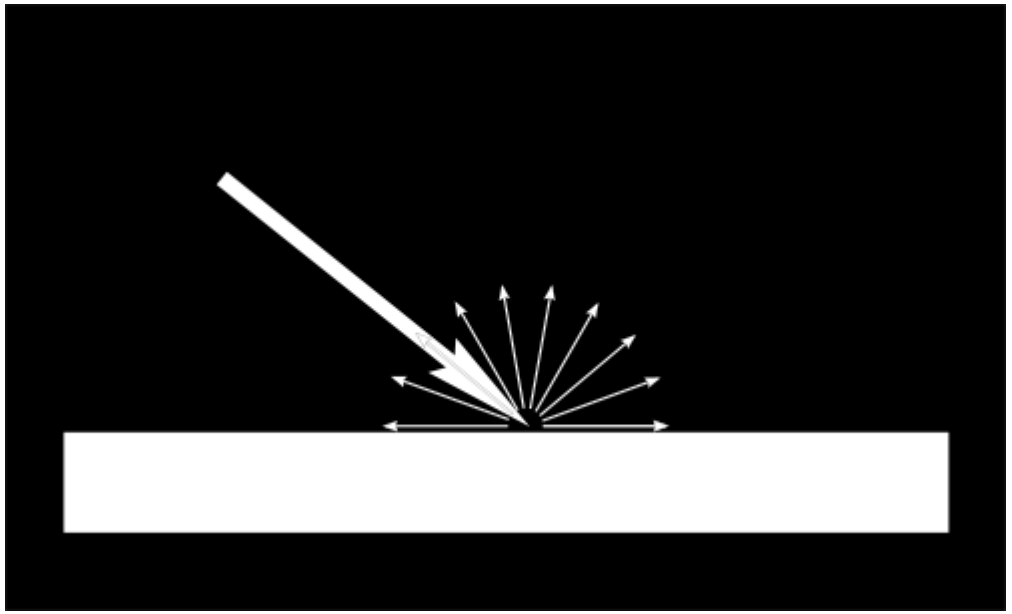
```
vertex v1, v2, v3, ....

triangle tr1, tr2, tr3 // todos compartilham v1

v1.normal = normalize( tr1.normal + tr2.normal +
tr3.normal )
```

Normais no OpenGL

Para usar normais no OpenGL, é simples. Uma normal é um atributo de um vértice, assim como sua posição, sua cor, suas coordenadas UV.

Componente Difuso

Quando a luz atinge um objeto, uma fração importante dela é refletida em todas as direções. Este é o “componente difuso”.

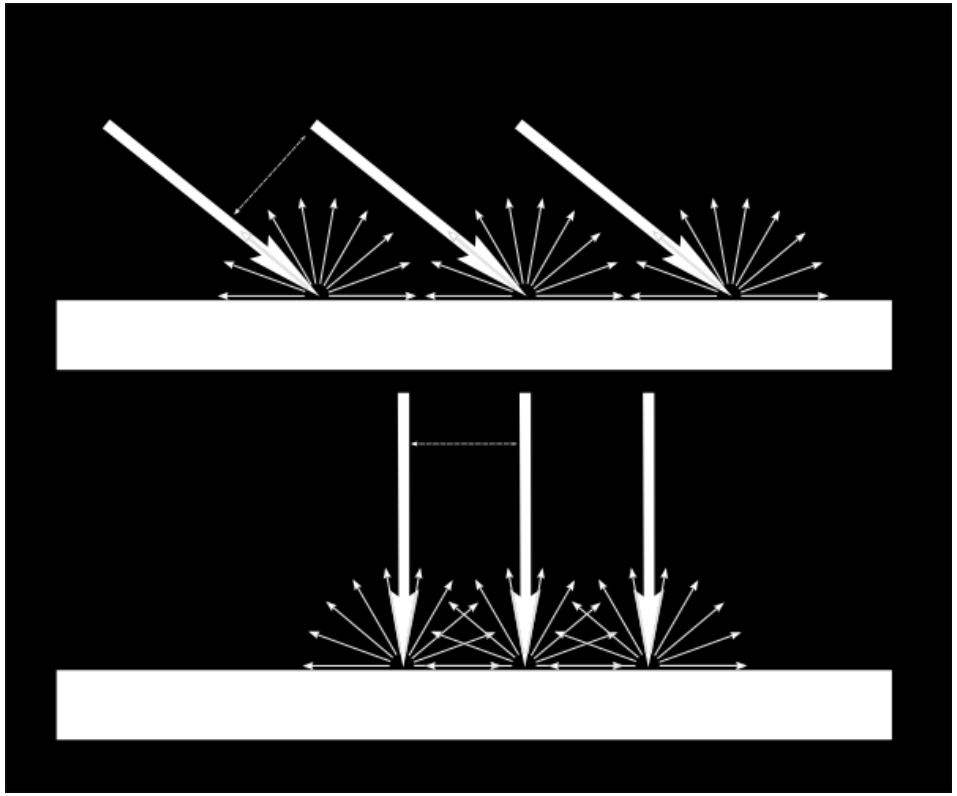


Componente Difuso

Quando um certo fluxo de luz chega à superfície, esta superfície é iluminada de forma diferente de acordo com o ângulo em que a luz chega.

Se a luz é perpendicular à superfície, ela está concentrada em uma pequena superfície. Se chegar em um ângulo mais agudo em relação à superfície, a mesma quantidade de luz se espalha em uma superfície maior:

Componente Difuso



Componente Difuso

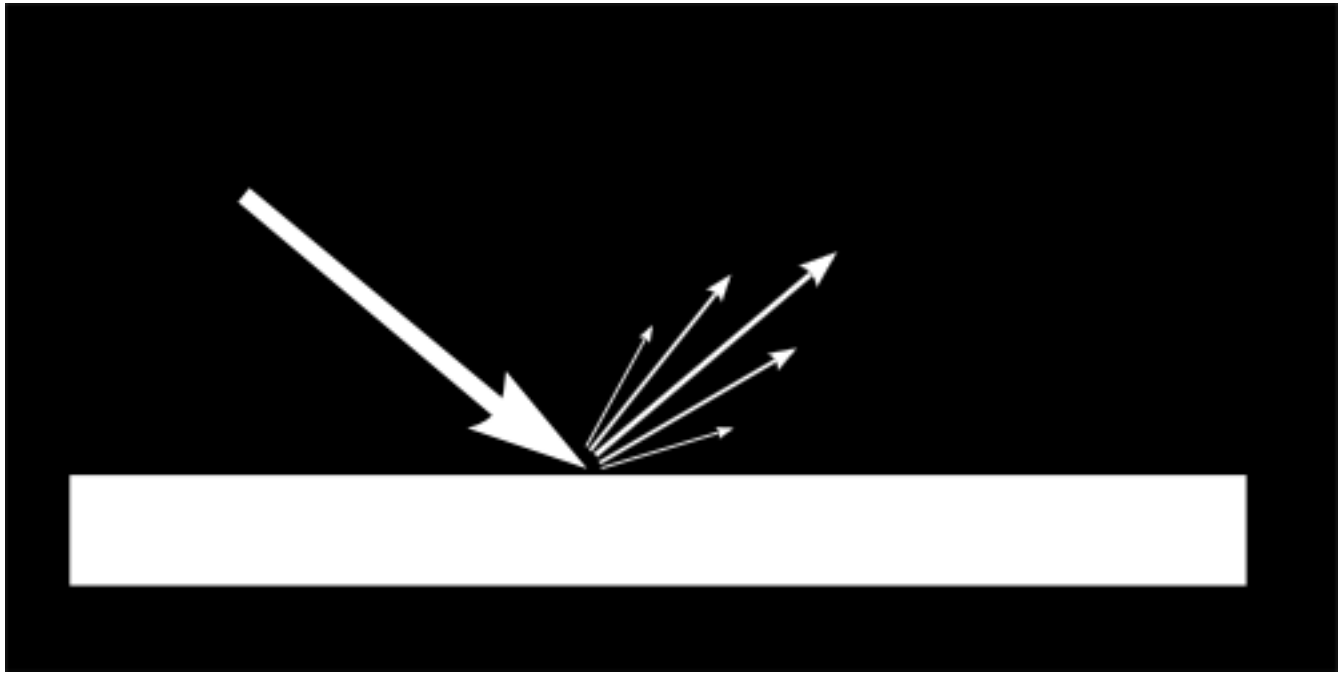
Isso significa que cada ponto da superfície ficará mais escuro com luz em um ângulo maior (mas lembre-se, mais pontos serão iluminados, então a quantidade total de luz permanecerá a mesma)

Isso significa que, quando computamos a cor de um pixel, o ângulo entre a luz incidente e a normal da superfície é importante. Temos assim:

Componente Especular

A outra parte da luz que é refletida é refletida principalmente na direção que é o reflexo da luz na superfície. Este é o componente especular.

Componente Especular



Componentes da Luz

Código

O primeiro trecho de código determina o modelo de iluminação e também ativa tanto a iluminação como o ponto de luz 0.

```
glLightModeli(GL_LIGHT_MODEL_LOCAL_VIEWER, GL_TRUE);
glEnable(GL_LIGHTING);
glEnable(GL_LIGHT0);
```

Código

Aqui são determinadas as cores de cada uma das principais formas de iluminação, difusa, ambiente e especular.

```
GLfloat luz_ambiente[] = { 0.2, 0.2, 0.2, 1.0 };
GLfloat luz_difusa[] = { 0.8, 0.8, 0.8, 1.0 };
GLfloat luz_especular[] = { 1.0, 1.0, 1.0, 1.0 };
glLightfv(GL_LIGHT0, GL_AMBIENT, luz_ambiente);
glLightfv(GL_LIGHT0, GL_DIFFUSE, luz_difusa);
glLightfv(GL_LIGHT0, GL_SPECULAR, luz_especular);
```

Código

Aqui é determinada a posição da luz GL_LIGHT0.

```
GLfloat posicao_luz[] = { .5, .5, 0.0, 1.0 };
glLightfv(GL_LIGHT0, GL_POSITION, posicao_luz);
```

Código

Além da luz, é necessário determinar o material dos objetos desenhados, os componentes deles são calculados juntos com a luz para determinar a iluminação final.

```
GLfloat cor_verde[] = { 0.0, 1.0, 0.0, 1.0 };
GLfloat cor_branco[] = { 1.0, 1.0, 1.0, 1.0 };
glMaterialfv(GL_FRONT, GL_DIFFUSE, cor_verde);
glMaterialfv(GL_FRONT, GL_SPECULAR, cor_branco);
glMaterialf(GL_FRONT, GL_SHININESS, 60);
```

Código

Da mesma forma que determinamos um vértice pelo glVertex3f, podemos determinar a normal antes de chamar um vértice pelo glNormal3f

```
glBegin(GL_QUADS);
glNormal3f(0.0, 0.0, 1.0);
```