Lenguaje de Programación: C++
GLUT
Iluminación

José Luis Alonzo Velázquez

Universidad de Guanajuato

Noviembre 2010

Iluminación

Mediante la iluminación es como se consigue un mayor efecto de realismo. El modelo de iluminación de OpenGL es bastante sencillo, los algoritmos son de la década de los 70-80, pero a diferencia de aquella época ahora se realizan en tiempo real a una velocidad de 60-100 frames por segundo, estos avances han sido gracias a la evolución del hardware. La calidad de las imágenes de OpenGL no es comparable a una película de animación por ordenador donde el cálculo de un frame puede llevar horas al realizarse con algoritmos de trazado de rayos mucho más sofisticados.

Modos de Renderización

Cuando renderizamos una primitiva disponemos de dos métodos de rellenado. Para cambiar entre uno u otro se hace con la función void glShadeModel(GLenum mode), como parámetro le pasamos uno de los siguientes modos:

- GL_FLAT : Renderiza la primitiva con un solo color, elegido de uno de los vértices dependiendo de la primitiva.
- GL_SMOOTH : Renderiza la primitiva interpolando el color de los vértices.

Esto en cuanto a la manera de renderizar la primitiva, el color de cada uno de los vértices hasta ahora lo especificábamos con la función glColor. En el modo de iluminación activado el color de cada uno de los vértices es calculado a partir del material del vértice y las luces.

Modelo de iluminación

El modelo de iluminación como hemos mencionado anteriormente es bastante básico y pasamos a describirlo a continuación describiendo cada una de sus partes.
El cálculo del color se realiza en cada vértice según las luces que haya encendidas y las propiedades del material de dicho vértice, no tiene en cuenta posibles oclusiones de terceros objetos que producirían sombras, reflejos de la luz que permitiría tener espejos o refracciones de la luz para ver a través de materiales como el agua.

Luces

Para habilitar el modo de iluminación se realiza con la función glEnable(GL_LIGHTING), cuando renderizamos en este modo la instrucción glColor no tienen ninguna influencia ya que el color resultante de cada vértice es calculado a partir de los materiales del vértice y las luces que haya encendidas.
Disponemos de ocho luces que podemos encender o apagar con la función glEnable(GL_LIGHTi) donde i =[0...7]. Cada luz la podemos configurar estableciendo los siguientes parámetros con la función glLight{if}v :

Luces

Parámetro	Valor por defecto	Significado
GL_AMBIENT	(0.0, 0.0, 0.0, 1.0)	intensidad ambiente de la luz
GL_DIFFUSE	(1.0, 1.0, 1.0, 1.0) GL_LIGTH0 (0.0, 0.0, 0.0, 1.0) GL_LIGTH1-GL_LIGHT7	intensidad difusa de la luz
GL_SPECULAR	(1.0, 1.0, 1.0, 1.0) GL_LIGTH0 (0.0, 0.0, 0.0, 1.0) GL_LIGTH1-GL_LIGHT7	intensidad especular de la luz
GL_POSITION	(0.0, 0.0, 0.0, 1.0)	(x, y, z, w) posición de la luz
GL_SPOT_DIRECTION	(0.0, 0.0, -1.0)	(x, y, z) dirección de la luz
GL_SPOT_EXPONENT	0.0	exponente del foco
GL_SPOT_CUTOFF	180.0	ángulo de abertura del foco
GL_CONSTANT_ATTENUATION	1.0	factor de atenuación constante
GL_LINEAR_ATTENUATION	0.0	factor de atenuación lineal
GL_QUADRATIC_ATTENUATION	0.0	factor de atenuación cuadrático

Componentes de la luz

La luz tiene tres componentes: ambiente, difusa y especular:

- Ambiente : Es la luz que se emite en todas direcciones y rebota en todas direcciones.
- Difusa : Luz emitida desde el punto de luz que rebota en todas las direcciones.
- Especular : Luz emitida en una dirección que rebota según la normal del polígono, provoca el brillo puntual del objeto, p.e. el brillo blanco de la bola de billar negra.

Tipo de fuente de luz

Además de las componentes de luz es necesario saber que tipo de fuente de luz se trata, en OpenGL disponemos de dos tipos:

- Direccional : Todos los rayos son paralelos e inciden en la dirección, el valor w de GL_POSITION es cero, la dirección se especifica mediante (x, y, z) de GL_POSITION. Un ejemplo de luz direccional es el sol.
- Posicional : La luz esta colocada en el punto (x, y, z) el valor w de GL_POSITION es distinto de cero.

Si se trata de una luz posicional, un foco, podemos alterar una serie de parámetros:

- GL_SPOT_DIRECTION : La dirección en que apunta.
- GL_SPOT_EXPONENT : Concentración de la luz en en centro del foco.
- GL_SPOT_CUTOFF : Apertura del foco en grados, el ángulo se multiplica por dos, 180 significa que la luz sale en todas direcciones

Más luces

Además de las ocho luces tenemos una luz ambiente global que podemos modificar mediante la instrucción:
glLightModel{if}v(GLenum param, TYPE* param) : Nos permite cambiar la luz ambiente global y otros parámetros de configuración del modelo de iluminación.

Parámetro	Valor por defecto	Significado
GL_LIGHT_MODEL_AMBIENT	(0.2, 0.2, 0.2, 1.0)	intensidad ambiente de la luz global
GL_LIGHT_MODEL_LOCAL_VIEWER	0.0 o FALSE	como se calculan los ángulos para la reflexión de la luz especular
GL_LIGHT_MOEL_TWO_SIDE	0.0 o FALSE	permite elegir entre iluminación a una o dos caras
GL_LIGHT_MODEL_CONTROL_COLOR	GL_SINGLE_COLOR	la componente especular se calcula separado de la ambiente y difusa

Materiales

En el modelo de renderización sin iluminación definíamos el color de cada vértice, ahora tenemos que definir los materiales. Se definen cuatro componentes de material para cada vértice.

- Ambiente : Es la componente del material que refleja la luz independientemente de la dirección que venga.
- Difusa : Es la componente del material que refleja según la dirección de procedencia de la luz.
- Especular : Componente que refleja la luz especular, provoca el efecto brillo.
- Emisiva : Componente de color que se añade independientemente de la luz recibida.

Materiales

Para establecer las componentes del material lo hacemos con la función glMaterial

Parámetro	Valor por defecto	Significado
GL_AMBIENT	(0.2, 0.2, 0.2, 1.0)	componente ambiente
GL_DIFFUSE	(0.8, 0.8, 0.8, 1.0)	componente difusa
GL_SPECULAR	(0.0, 0.0, 0.0, 1.0)	componente espcular
GL_SHININESS	0.0	exponente de brillo
GL_EMISSION	(0.0, 0.0, 0.0, 1.0)	componente emisiva

Tenemos la posibilidad de dotar de funcionalidad a la función glColor habilitandola con la instrucción glEnable(GL_COLOR_MATERIAL). Con la opcion color-material activado la función glColor cambia el material que indicamos en la función glColorMaterial : glColorMaterial (GLenum face, GLenum mode) : Especifica que materiales (mode) cambia y de que caras (face) al usar la función glColor . Face toma valores entre GL_FRONT, GL_BACK, GL_FRONT_AND_BACK (por defecto), mode entre GL_AMBIENT, GL_DIFFUSE, GL_SPECULAR, GL_AMBIENT_AND_DIFFUSE (por defecto).

Normales

Una de las características que se puede establecer de un vértice es el color (o el material si estamos en modo iluminacion), pero podemos establecer otras características como la coordenada de la textura o la normal que sirve para realizar los cálculos de la iluminación resultante.
No importa el orden en que se definan las características pero siempre se hará antes de definir el vértice.

- glColor3f(1.0, 0.3, 0.4); // Color del vértice
- glNormal3f(1.0, 0.0, 0.0); // Normal del vértice
- glTexCoord2f(0.0, 1.0); // Coordenada de la textura
- glVertex3f(1.0, 3.2, 3.6); // Vértice

El vector normal se utiliza para calcular la incidencia de la luz sobre el vértice, si el vector normal no es unitario el vértice estará sobreiluminado (mayor que 1) o infrailuminado (menor que 1).
Es muy importante que para cada vértice se defina la normal, ya que OpenGL no asigna la normal a partir de la orientación de la primitiva. De lo contrario utilizará como normal la ultima normal establecida.


Programming Principles and Practice Using C++, Bjarne Stroustrup.


http://www.codeblocks.org


http://www.wxwidgets.org


(O'Reilly) Practical C Programming (3rd Edition)


http://www.cplusplus.com


http://es.wikipedia.org/wiki/GLUT