

OpenGL – Światło (cieniowanie)

1. Oświetlenie włączanie/wyłączanie

glEnable(GL_LIGHTING); - włączenie mechanizmu oświetlenia

glDisable(GL_LIGHTING); - wyłączenie mechanizmu oświetlenia

glEnable(GL_LIGHT0); - włączenie światła pierwszego. OpenGL standardowo posiada osiem źródeł światła. Zmienne OpenGL: *GL_LIGHT0*, *GL_LIGHT1*, *GL_LIGHT2*, *GL_LIGHT3*, *GL_LIGHT4*, *GL_LIGHT5*, *GL_LIGHT6*, *GL_LIGHT7* określają, którego światła ma dotyczyć operacja.

glDisable(GL_LIGHTx) – wyłączenie światła ($x = 0, \dots, 7$)

2. Wybór cieniowania

Do pokazania efektu światła używane jest cieniowanie (zaciemnianie powierzchni). W OpenGL można ustawić cieniowanie płaskie (*GL_FLAT*) oraz gładkie (*GL_SMOOTH*);

glShadeModel(parametr) – ustawienie rodzaju cieniowania. Parametr może przyjąć wartości: *GL_FLAT*, *GL_SMOOTH*.

3. Normalne

Przy obliczaniu natężenia światła rozproszonego (diffuse) i odbitego (specular) OpenGL korzysta z normalnych. Normalna to w OpenGL wektor wyznaczający kąt odbicia promieni światła. W świecie rzeczywistym normalna jest zawsze prostopadła do powierzchni (wektor długości jeden i prostopadły do powierzchni). W OpenGL normalną można ustawić oddzielnie dla każdego wierzchołka i nie musi być ona prostopadła do powierzchni wyznaczonej przez zbiór wierzchołków. Uśredniając wartości normalnej w danym wierzchołku dla przylegających powierzchni można otrzymać efekt odbicia światła od zaokrąglonej powierzchni (przy cieniowaniu gładkim).

glNormal3f(x, y, z) – ustawia normalną dla danego wierzchołka. Od tego momentu aż do następnej zmiany wartość wektora normalnego wynosi x, y, z .

4. Zmiana parametrów światła

glLightfv(GL_LIGHTx, parametr, wartość) – zmiana parametru światła $x = (0, \dots, 7)$

Wartość **parametr** dotyczy cechy światła jaka ma być zmieniona. Parametr może przybierać wartości:

- *GL_AMBIENT* – przyjmuje parametr typu wektor [**r,g,b**], lub [**r,g,b,a**] w którym określa się natężenie poszczególnych składowych koloru, dla światła bezwzględnego (otoczenia).
- *GL_DIFFUSE* – parametry j.w., dla światła rozproszonego.
- *GL_SPECULAR* – parametry j.w., dla światła odbitego.
- *GL_POSITION* – określa pozycję światła w przestrzeni. Przyjmuje parametr typu wektor [**x,y,z,rodzaj**] gdzie **rodzaj** określa rodzaj światła. Dla wartości 0 jest to światło odległe – promienie światła są równoległe i określone są wektorem [**x, y, z**]. Natomiast dla wartości **rodzaj** równej 1 światło jest punktowe a współrzędne **x, y, z** określają położenie światła w przestrzeni. Położenie to przemnażane jest przez macierz przekształcenia. Dzięki temu w prosty sposób można uzyskać przemieszczanie się światła.

5. Włączenie koloru powierzchni

Cieniowanie zastępuje kolor powierzchni (zdefiniowany funkcją *glColor...*). Można użyć jednocześnie koloru i cieniowania używając komendy:
glEnable(GL_COLOR_MATERIAL);

OpenGL – TEKSTUROWANIE

1. Przygotowywanie tekstur:

Tekstury dla akceleratorów graficznych muszą mieć rozmiary będące wielokrotnością liczby 2. (2^n). Rozmiary w pionie i poziomie są niezależne.

glGenTextures(1, &numer_tekstury); - komenda generuje teksturę o numerze **numer_tekstury**; **numer_tekstury** jest typu Gluint;

glTexParameteri(GL_TEXTURE_2D, typ, parametr); - ustawianie parametrów tekstury.

Opis parametrów:

GL_TEXTURE_2D – zmiana parametru dotyczy tekstur dwuwymiarowych

typ – typ własności, którą chcemy zmienić, np.:

GL_TEXTURE_WRAP_S (GL_TEXTURE_WRAP_T) – parametry określające, której

współrzędnej tekstury będą modyfikowane właściwości (pozioma, pionowa). Dla tego typu własności można ustawić **parametr**:

- *GL_REPEAT* – parametr oznacza zapętlenie tekstury
- *GL_CLAMP_TO_EDGE* – tekstura nie będzie zapętlona

typ: *GL_TEXTURE_MAG_FILTER* (*GL_TEXTURE_MIN_FILTER*) – parametry określające, w jaki sposób rozmywać piksele przy powiększaniu tekstury (pomniejszaniu tekstury). Dla tego typu własności można ustawić **parametr**:

- *GL_NEAREST* – bez filtrowania
- *GL_LINEAR* – przybliżenie liniowe

Dla włączonych Mipmap:

- *GL_NEAREST_MIPMAP_NEAREST* – bez filtrowania
- *GL_NEAREST_MIPMAP_LINEAR* – tekstury bez filtrowania, przybliżenie mipmap liniowe
- *GL_LINEAR_MIPMAP_NEAREST* – filtrowanie dwuliniowe
- *GL_LINEAR_MIPMAP_LINEAR* – filtrowanie trójliniowe

2. Transfer zawartości obrazu do tekstury:

Tekstura standardowa:

```
glTexImage2D(GL_TEXTURE_2D,0,GL_RGBA, w, h,0,GL_RGBA, GL_UNSIGNED_BYTE, *wskaźnik_na_bitmapę);
```

Tekstura z mipmapami:

```
gluBuild2DMipmaps(GL_TEXTURE_2D, GL_RGBA, w, h, GL_RGBA, GL_UNSIGNED_BYTE, image->pixels);
```

3. Włączanie tekstur:

```
glEnable(GL_TEXTURE_2D); - włączanie trybu teksturowania (można go wyłączyć instrukcją glDisable(GL_TEXTURE_2D);)
```

4. Wybieranie tekstury:

```
glBindTexture(GL_TEXTURE_2D, numer_tekstury); - ustawienie aktywnej tekstury
```

5. Współrzędne tekstury:

```
glTexCoord2f(x,y); - ustawienie współrzędnych tekstury w danym wierzchołku. Przy teksturowaniu instrukcja ta powinna być wywołana przed narysowaniem każdego wierzchołka.
```

OpenGL – MULTITEKSTUROWANIE

1. Inicjalizacja multiteksturowania:

Multiteksturowanie (wieloteksturowanie) polega nakładaniu kilku tekstur naraz na jedną powierzchnię. Obecne karty graficzne posiadają co najmniej cztery potoki teksturowania (czasem więcej), dzięki czemu każdy potok generuje inną teksturę dla renderowanego piksela. Obsługa multiteksturowania realizowana jest przez rozszerzenie: *GL_ARB_multitexture*. Aby z niego skorzystać należy upewnić się czy jest ono dostępne w systemie, po czym zainicjalizować.

Deklaracja zmiennych potrzebnych do multiteksturowania:

```
PFNGLMULTITEXCOORD2FARBPROC glMultiTexCoord2fARB = NULL;  
PFNGLACTIVETEXTUREARBPROC glActiveTextureARB = NULL;  
PFNGLCLIENTACTIVETEXTUREARBPROC glClientActiveTextureARB = NULL;  
int maxTextureUnits = 0;
```

Pobranie nazw dostępnych rozszerzeń do zmiennej znakowej **extensions**:

```
char *extensions = (char*)glGetString(GL_EXTENSIONS);
```

Sprawdzenie czy w otrzymanych nazwach rozszerzeń znajduje się *GL_ARB_multitexture*:

```
if(strstr(extensions, "GL_ARB_multitexture"))
```

jeśli tak inicjalizujemy multiteksturowanie:

```
{  
pobranie ilości sprzętowych potoków teksturowania do zmiennej maxTextureUnits:  
glGetIntegerv(GL_MAX_TEXTURE_UNITS_ARB, &maxTextureUnits);  
glActiveTextureARB=(PFNGLACTIVETEXTUREARBPROC)wglGetProcAddress("glActiveTextureARB");  
glMultiTexCoord2fARB=(PFNGLMULTITEXCOORD2FARBPROC)wglGetProcAddress("glMultiTexCoord2fARB");  
glClientActiveTextureARB=(PFNGLCLIENTACTIVETEXTUREARBPROC)wglGetProcAddress("glClientActiveTextureARB");  
}
```

2. Ustalenie na której teksturze mają być wykonywane operacje:

Operacje na teksturach przy multiteksturowaniu są identyczne jak przy teksturowaniu¹, należy tylko określić, którego potoku się tyczą.

glActiveTextureARB(GL_TEXTURE0_ARB); - zmiana aktywnego potoku teksturowania
argument: *GL_TEXTUREx_ARB* – gdzie *x* jest numerem potoku, zaczynając od zera.

3. Ustawienie tekstur na obiekcie:

glMultiTexCoord2fARB(GL_TEXTURE0_ARB, 0.0f, 0.0f); - jest to odpowiednik instrukcji *glTexCoord2f*, z tym że pobiera także parametr, który określa dla tekstury z jakiego potoku ustawiane są współrzędne tekstury: *u*, *v*.

4. Łączenie potoków:

glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_COMBINE); - dodanie do opcji środowiska tekstur parametru: *GL_COMBINE*

glTexEnvf(GL_TEXTURE_ENV, GL_COMBINE, parametr);

- **parametr** może przyjąć:

- *GL_REPLACE* - $w = p$
- *GL_MODULATE* - $w = p * o$
- *GL_ADD* - $w = p + o$
- *GL_ADD_SIGNED* - $w = p + o - 0,5$
- *GL_INTERPOLATE* - $w = (p * \alpha) + (o * (1 - \alpha))$
- *GL_SUBTRACT* - $w = p - o$
- *GL_DOT3_RGB* - $w = (pred - 0,5) * (ored - 0,5) + (pgreen - 0,5) * (ogreen - 0,5) + (pblue - 0,5) * (oblue - 0,5)$

w – kolor wynikowy

p – kolor obecnie wygenerowanej tekstury

o – kolor poprzedniej wygenerowanej tekstury

OpenGL – PRZEZROCZYŚĆ

1. Funkcja mieszania kolorów

glBlendFunc() – ustawianie parametrów łączenia kolorów dla przezroczystości. Aby otrzymać przezroczystość sterowaną kanałem alfa należy ustawić funkcję z parametrami: *glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA)*;

2. Włączanie/wyłączanie przezroczystości

glEnable(GL_BLEND) – Włączanie przezroczystości
glDisable(GL_BLEND) – Wyłączanie przezroczystości

OpenGL – MGŁA

1. Ustawienia

glFogi(GL_FOG_MODE, GL_LINEAR); - Ustawianie typu
glFogfv(GL_FOG_COLOR, fogColor); - Ustawianie koloru

glFogf(GL_FOG_START, 10.0f); - Odległość początkowa
glFogf(GL_FOG_END, 30.0f); - Odległość końcowa

2. Włączanie/wyłączanie

glEnable(GL_FOG) – Włączanie mgły
glDisable(GL_FOG) – Wyłączanie mgły