Laboratorium grafiki komputerowej i animacji

Ćwiczenie VI - Biblioteka OpenGL - teksturowanie

Przygotowanie do ćwiczenia:

1. Zapoznać się z zasadami teksturowania obiektów w OpenGL.
2. Zapoznać się z zestawem komend OpenGL umożliwiającym definiowanie kwadryk (biblioteka GLU).

Przebieg ćwiczenia:

1. Założenia:
   a. Celem prac na zajęciach laboratoryjnych jest uzupełnienie trójwymiarowego modelu robota Puma o teksturę z nazwą robota (rysunek 1.1)
   b. Wynikiem prac na dzisiejszych zajęciach ma być program zbliżony w działaniu do programu „**puma_tekstury.exe”** dołączonego do materiałów laboratoryjnych.
   c. Realizacja ćwiczenia polega na uzupełnieniu kodu programu „**gl_template**” modyfikowanego na ostatnich zajęciach.

Rys 1.1 Teksturowany model manipulatora Puma

2. Przebieg ćwiczenia:
   a. Należy pobrać i rozpakować archiwum zawierające przykładowy program **glTemplate_tex.c** oraz przykładowe tekstury.
   b. Program **glTemplate_tex.c** ma taką samą strukturę jak aplikacja **glTemplate**, ale wprowadzono w nim pewne dodatkowe modyfikacje ułatwiające bardziej realistyczne wyświetlanie sceny i teksturowanie:
      - Funkcję **ChangeSize()** zmodyfikowano tak, że scena jest odwzorowywana w rzucie perspektywicznym, funkcję **RenderScene()** uzupełniono wywołaniem polecenia OpenGL:
        ```
        glTranslated(0,0,-200);
        ```
        Polecenie „oddala” kamerę o 200 jednostek od środka sceny.
      - Wprowadzono nowe zmienne globalne:
        ```
        BITMAPINFOHEADER  bitmapInfoHeader; // nagłówek obrazu
        unsigned char*    bitmapData;       // dane tekstury
        ```

```
unsigned int      texture[1];       // obiekt tekstury
```

- Wprowadzono nową definicję, zastosowaną później w sprawdzeniu, czy otwarto plik typu bitmapa:
```
#define BITMAP_ID 0x4D42
```
- Wprowadzono nową funkcję odczytującą plik typu *.bmp i zwracającą wskazanie na tablicę pikseli dostoswaną do konwersji na teksturę OpenGL (szczegóły we wprowadzeniu):
```
unsigned char *LoadBitmapFile(char *filename, BITMAPINFOHEADER
*bitmapInfoHeader;)
```
- Zmodyfikowano obsługę komunikatu WM_CREATE o zestaw wywołań funkcji, które tworzą teksturę OpenGL (szczegóły we wprowadzeniu).
- Wprowadzono nowe funkcje:
```
void skrzynka(void);
void walec01(void);
void kula(void);
```

c. Proszę uruchomić program znajdujący się w projekcie **glTemplate_tex**, aktywując kolejno w funkcji **RenderScene()** funkcje: skrzynka(), walec01(), kula(). W wymienionych funkcjach pokazano kolejno: sposób rozpinania tekstury na wielokącie, zastosowanie kwadryk, pokrywanie kwadryk teksturami.

d. Proszę zmodyfikować kod źródłowy własnej aplikacji w taki sposób, aby możliwe było w niej teksturowanie.

e. Proszę rozpiąć na jednym z ramion robota teksturę z napisem „PUMA" dostarczoną w materiałach do ćwiczenia (plik "napis.bmp") w podobny sposób.
UWAGA: Aby napis był odpowiednio rozłożony na trapezie, trzeba odpowiednio dobrać współrzędne tekstury.

f. Zadnie dodatkowe:
Należy skonstruować prostą scenę z wykorzystaniem kwadryk, podobną do sceny w programie „**lody.exe**" (rysunek 1.2)

Rys 1.2 Scena wykorzystująca kwadryki OpenGL

Uwagi do postawionego problemu:
Efekt przezroczystości osiąga się postępując według następujących zasad:

- „Przezroczyste" elementy sceny muszą być rysowane na końcu
- Funkcję SetupRC() należy uzupełnić o wywołanie komendy: glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);

- Elementy „przezroczyste” umieszcza się na scenie w następujący sposób (wartość A_val decyduje o przezroczystości obiektu):

```
glEnable(GL_BLEND);
glColor4f(R_val, G_val, B_val, A_val);

gluCylinder(obj,
                    0.0,
                    30.0,
                    0.0,
                    20.0,
                    3.0
            );
glDisable(GL_BLEND);
```

- Dla zapewnienia odpowiedniego mieszania kolorów elementów teksturowanych z elementami przezroczystymi przed wywołaniem obiektu zateksturowanego należy wywołać komendę: glTexEnvi(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_BLEND);