helion.pl
KSIĘGARNIA INTERNETOWA · WYDAWNICTWO

IDŹ DO
PRZYKŁADOWY ROZDZIAŁ
SPIS TREŚCI

KATALOG KSIĄŻEK
KATALOG ONLINE
ZAMÓW DRUKOWANY KATALOG

TWÓJ KOSZYK
DODAJ DO KOSZYKA

CENNIK I INFORMACJE
ZAMÓW INFORMACJE O NOWOŚCIACH
ZAMÓW CENNIK

CZYTELNIA
FRAGMENTY KSIĄŻEK ONLINE

OpenGL. Ćwiczenia



Autor: Waldemar Pokuta
ISBN: 83-7361-172-X
Format: B5, stron: 144

Biblioteka OpenGL to zestaw procedur graficznych, których możemy używać w rozmaitych językach programowania dla różnych systemów operacyjnych. Jest to ogólnie uznany standard – jego znajomość to podstawa do zajmowania się grafiką komputerową.

Książka, którą trzymasz w ręku, przedstawia jedynie fragment możliwości OpenGL, wystarczający jednak, by rozpocząć przygodę z programowaniem grafiki trójwymiarowej. W odróżnieniu od innych pozycji poświęconych tej bibliotece, przykłady zostały napisane w języku ObjectPascal (Delphi), co umożliwi także programistom nie posługującym się językami C/C++ rozpoczęcie przygody z grafiką 3D.

W książce omówiono:

- Podstawy OpenGL: punkty, linie, trójkąty, wielokąty, kolorowanie
- Rysowanie złożonych obiektów za pomocą kwadryk
- Posługiwanie się perspektywą i kamerą
- Materiały: kolor, połysk, emisję
- Zagadnienia związane ze światłem
- Tworzenie dynamicznych scen
- Składanie przekształceń
- Operacje na macierzach
- Tworzenie brył
- Tekstury i ich właściwości
- Napisy w OpenGL
- Tworzenie prostej gry

Wydawnictwo Helion
ul. Chopina 6
44-100 Gliwice
tel. (32)230-98-63
e-mail: helion@helion.pl

Spis treści

Wstęp **7**

Język programowania 7

Struktura modułu 8

Wymagania 9

Przykłady 9

Rozdział 1. Podstawy **11**

Na początku 11

Punkty 12

Rozmiar punktów 16

Linie 17

Typ linii 17

Trójkąty 18

Głębia 19

Widoczność 19

Czworokąty 20

Typ wielokątów 20

Paski trójkątów 21

Kolorowanie płaskie 22

Wierzchołki trójkątów 22

Kolorowanie płynne 23

Rozdział 2. Kwadryki **25**

Kwadryki z punktów 25

Linie 26

Cieniowanie kwadryk 27

Rozdział 3. Perspektywa i kamera **31**

Perspektywa 31

Kamera 32

Rozdział 4. Materiał ... 35
Kolor materiału ... 35
Połysk ... 36
Emisja ... 37
Przeźroczystość ... 38
Mgła ... 38

Rozdział 5. Światło ... 41
Włączenie oświetlenia ... 41
Światło tła ... 42
Światło rozproszone ... 43
Rozbłyski ... 44
Pozycja ... 45
Reflektor ... 45
Rozmycie ... 47

Rozdział 6. Dynamika sceny ... 49
Przesuw ... 49
Obrót ... 50
Skalowanie ... 51
Kolor ... 52
Światło ... 52
Reflektor ... 53
Ruch kamery ... 54
Rozmycie ... 54

Rozdział 7. Składanie przekształceń ... 57
Pierwszy obiekt ... 57
Obrót wokół własnej osi ... 59
Orbita ... 59
Podorbita ... 62
Ruch po elipsie ... 64

Rozdział 8. Macierze ... 69
Pobranie macierzy ... 69
Załadowanie macierzy ... 70
Macierz tożsamościowa ... 71
Mnożenie macierzy ... 72

Rozdział 9. Tworzenie brył ... 77
Sześcian ... 77
Czworościan ... 79
Ośmiościan ... 80
Czternastościan ... 81
Dwunastościan ... 83
Dwudziestościan ... 86
Cząsteczka metanu ... 88
Maczuga elegancka ... 91
Maczuga profesjonalna ... 92
Maczuga współczesna ... 93

Rozdział 10. Tekstury ... 95
Tekstura 1D ... 95
Teksturowanie kwadryk ... 96
Nakładanie tekstury po kawałku ... 98
Lustro — przeźroczystość tekstur ... 100
Zdjęcie — bitmapa z pliku ... 101

Rozdział 11. Napisy ... 105
Czcionki bitmapowe ... 105
Kontur ... 107
Czcionka 3D ... 108
Czcionka teksturowana ... 109
Metalowy połysk ... 110

Rozdział 12. Prosta gra 3D ... 113
Sześcian ... 113
Oświetlenie ... 115
Figura ... 117
Obrót sceny ... 118
Osiem figur ... 121
Przesuw ... 122
Płynny przesuw ... 123
Obrót ... 124
Płynny obrót ... 126
Skrzynka ... 127
Kolizja ... 129
Napisy ... 131

Dodatek A Własny komponent GlBox ... 137

Rozdział 4.

Materiał

W tym rozdziale dodamy właściwości materiału rysowanych obiektów. Kolor, połysk, emisja światła, przeźroczystość i efekt mgły mogą bardzo ożywić rysowaną scenę.

Otwórz w Delphi projekt *...\Cwiczenia\R04_Material\C04_00_Poczatek\Material.dpr*.

Scena zawiera dwanaście obracających się sześcianów. Naszym zadaniem będzie nadać im różne właściwości.

Kolor materiału

Nowa procedura:

- `glMaterialfv(face, pname, params)` — ustawienie materiału obiektów. Pierwszy argument określa stronę, z której chcemy zmieniać właściwości materiału (przód, tył czy obydwie). Drugi parametr określa cechę, którą chcemy ustawić (`GL_AMBIENT_AND_DIFFUSE` to odbijanie światła tła i światła rozproszonego). Ostatni parametr to wskaźnik do tablicy składowych koloru (`red, green, blue, alpha`).

Ćwiczenie 4.1.

Niech pierwsze dwa sześciany mają kolor materiału zbliżony do żółtego, a pozostałe do niebieskiego.

W procedurze `GLRender` dopisz:

```
procedure TForm1.GLRender(Sender: TObject);
var
  i: integer;
```

```
const
  kat: real = 0;
  amb_dif1: TGLArrayf4 = (0.8, 0.8, 0.2, 1.0);
  amb_dif2: TGLArrayf4 = (0.2, 0.2, 0.9, 1.0);
begin
  glClear (GL_COLOR_BUFFER_BIT or GL_DEPTH_BUFFER_BIT);
  // Załadowanie przekształcenia tożsamościowego
  glLoadIdentity;
  // Oświetlenie
  UstawOswietlenie;
  glTranslatef(0, 0,-150);
  // Rysowanie figur
  glRotatef(-60, 1, 0, 0);
  glRotatef(-kat, 0, 0, 1);
  for i := 0 to 11 do begin
    if i<2 then glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT_AND_DIFFUSE, @amb_dif1)
    else glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT_AND_DIFFUSE, @amb_dif2);
    glPushMatrix;
    glRotatef(i*30, 0, 0, 1);
    glTranslatef(0,-50, 0);
    glRotatef(3*kat+i*30, 0.7, 1, 0.45);
    Szescian(20);
    glPopMatrix;
  end;
  kat := kat+1;
end;
```

Połysk

Aby uzyskać efekt rozbłysków, należy użyć procedury `glMaterialfv` z drugim parametrem równym `GL_SPECULAR`.

Nowa procedura:

- `glMaterialf(face, pname, param)` — procedura podobna do poprzedniej. Jeżeli drugi parametr jest równy `GL_SHININESS`, to trzeci zmienia wielkość plamy rozbłysków na obiektach.

Ćwiczenie 4.2.

Dodaj połysk do wszystkich — oprócz pierwszych czterech — sześcianów.

W procedurze `GLRender` dopisz:

```
...
  amb_dif1: TGLArrayf4 = (0.8, 0.8, 0.2, 1.0);
  amb_dif2: TGLArrayf4 = (0.2, 0.2, 0.9, 1.0);
  spec1   : TGLArrayf4 = (0.0, 0.0, 0.0, 1.0);
  spec2   : TGLArrayf4 = (0.6, 0.6, 0.6, 1.0);
begin
...
    if i<2 then glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT_AND_DIFFUSE, @amb_dif1)
    else glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT_AND_DIFFUSE, @amb_dif2);
```

```
    if i<4 then glMaterialfv(GL_FRONT_AND_BACK, GL_SPECULAR, @spec1)
    else glMaterialfv(GL_FRONT_AND_BACK, GL_SPECULAR, @spec2);
    glPushMatrix;
    glRotatef(i*30, 0, 0, 1);
...
```

Ćwiczenie 4.3.

Ustaw wielkość plamy rozbłysku na wartość 80 w sześcianach 6 do 12.

W procedurze `GLRender` dopisz:

```
...
    if i<4 then glMaterialfv(GL_FRONT_AND_BACK, GL_SPECULAR, @spec1)
    else glMaterialfv(GL_FRONT_AND_BACK, GL_SPECULAR, @spec2);
    if i<6 then glMaterialf (GL_FRONT_AND_BACK, GL_SHININESS, 0)
    else glMaterialf (GL_FRONT_AND_BACK, GL_SHININESS, 80);
    glPushMatrix;
    glRotatef(i*30, 0, 0, 1);
...
```

Emisja

Aby sprawić, że obiekty same będą emitować światło, należy użyć procedury `glMaterialfv` z drugim parametrem równym `GL_EMISSION`.

Ćwiczenie 4.4.

Niech sześciany 8 do 12 zaczną emitować światło zbliżone do czerwieni.

W procedurze `GLRender` dopisz:

```
...
  spec1   : TGLArrayf4 = (0.0, 0.0, 0.0, 1.0);
  spec2   : TGLArrayf4 = (0.6, 0.6, 0.6, 1.0);
  emis1   : TGLArrayf4 = (0.0, 0.0, 0.0, 1.0);
  emis2   : TGLArrayf4 = (0.5, 0.1, 0.1, 1.0);
begin
...
    if i<6 then glMaterialf (GL_FRONT_AND_BACK, GL_SHININESS, 0)
    else glMaterialf (GL_FRONT_AND_BACK, GL_SHININESS, 80);
    if i<8 then glMaterialfv(GL_FRONT_AND_BACK, GL_EMISSION, @emis1)
    else glMaterialfv(GL_FRONT_AND_BACK, GL_EMISSION, @emis2);
    glPushMatrix;
    glRotatef(i*30, 0, 0, 1);
...
```

Przeźroczystość

Aby sprawić, że obiekty staną się przeźroczyste, należy zmienić parametr `alpha` (stopień przeźroczystości) w tablicy określającej kolor materiału, użyć procedury `glBlendFunc` z odpowiednimi argumentami oraz włączyć przeźroczystość procedurą `glEnable`.

Nowa procedura:

- `glBlendFunc(sfactor, dfactor)` — procedura określa sposób, w jaki ma być obliczana przeźroczystość obiektów. Domyślne użycie procedury to: `glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA)`.

Ćwiczenie 4.5.

Niech sześciany 10 do 12 będą ze szkła.

W procedeurze `GLRender` dopisz:

```
...
const
  kat: real = 0;
  amb_dif1: TGLArrayf4 = (0.8, 0.8, 0.2, 1.0);
  amb_dif2: TGLArrayf4 = (0.2, 0.2, 0.9, 0.5);
  spec1   : TGLArrayf4 = (0.0, 0.0, 0.0, 1.0);
...
    if i<8 then glMaterialfv(GL_FRONT_AND_BACK, GL_EMISSION, @emis1)
    else glMaterialfv(GL_FRONT_AND_BACK, GL_EMISSION, @emis2);
    if i<10 then glDisable(GL_BLEND)
    else begin
      glEnable(GL_BLEND);
      glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);
    end;
    glPushMatrix;
    glRotatef(i*30,0,0,1);
...
```

Mgła

Aby uzyskać efekt mgły, należy procedurą `glEnable` włączyć opcję `GL_FOG`, a procedurami `glFogi`, `glFogf` i `glFogfv` trzeba ustawić jej właściwości.

Nowe procedury:

- `glFogi(pname, param), glFogf(pname, param)` — ustalenie właściwości mgły. Jeżeli pierwszym argumentem jest `GL_FOG_MODE`, to drugi argument ustala sposób, w jaki jest obliczana przeźroczystość mgły. Jeżeli pierwszym argumentem jest `GL_FOG_START` lub `GL_FOG_END`, to jest ustalany odpowiednio początek i koniec obszaru mgły;

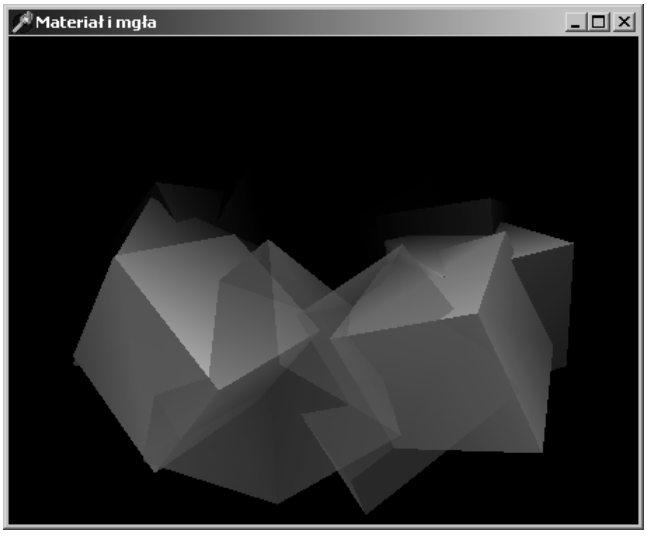
❖ `glFogfv(pname, params)` — procedura podobna do poprzedniej. Jeżeli pierwszym argumentem jest `GL_FOG_COLOR`, to drugi wskazuje na tablicę, która określa kolor mgły.

Ćwiczenie 4.6.

Dodaj do sceny efekt czarnej mgły.

W procedurze `GLRender` dopisz:

```
...
  emis1   : TGLArrayf4 = (0.0, 0.0, 0.0, 1.0);
  emis2   : TGLArrayf4 = (0.5, 0.1, 0.1, 1.0);
  fogCol  : TGLArrayf4 = (0.0, 0.0, 0.0, 1.0);
begin
...
  UstawOswietlenie;
  glTranslatef(0, 0, -150);
  // Mgła
  glFogi(  GL_FOG_MODE, GL_LINEAR);
  glFogfv(GL_FOG_COLOR,   @fogCol);
  glFogf( GL_FOG_START,       100);
  glFogf(   GL_FOG_END,       160);
  glEnable(GL_FOG);
  // Rysowanie figur
  glRotatef(-60, 1, 0, 0);
  glRotatef(-kat, 0, 0, 1);
...
```



Rysunek 4.1. *Efekt końcowy działania programu*