Bit - Sekcja Grafiki

Wprowadzanie do Direct3D

Dawid Fatyga
fatyga@student.agh.edu.pl

Akademia Górniczo Hutnicza

Grudzień 3, 2009

O tym co dzisiaj

O tym co dzisiaj

Powtórka z matematyki

1. Trochę o wektorach
2. Macierze, wyznacznik i działania na macierzach
3. Transformacje macierzowe

Definicja

Definition (Wektor)

Obiekt geometryczny w matematyce elementarnej, mający moduł (zwany też długością), kierunek i zwrot określający orientację wzdłuż danego kierunku.

Reprezetancja

Definition

W trójwymiarowej przestrzeni euklidesowej (lub $\mathbb{R}^3$) wektory reprezentowane są jako trójki liczb odpowiadającym współrzędnym kartezjańskim punktu końcowego, co można zapisać:

$$\overrightarrow{v} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}$$

Definition (Długość wektora)

$$|\overrightarrow{v}| = \sqrt{(v_x^2 + v_y^2 + v_z^2)}$$

Reprezetancja

Definition

W trójwymiarowej przestrzeni euklidesowej (lub $\mathbb{R}^3$) wektory reprezentowane są jako trójki liczb odpowiadającym współrzędnym kartezjańskim punktu końcowego, co można zapisać:

$$\overrightarrow{v} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}$$

Definition (Długość wektora)

$$|\overrightarrow{v}| = \sqrt{(v_x^2 + v_y^2 + v_z^2)}$$

Działania na wektorach

Definition (Iloczyn skalarny)

Jest to pewne działanie przyporządkowujące parze wektorów pewną wartość rzeczywistą (skalarną).

$$\overrightarrow{v} \cdot \overrightarrow{u} = v_x * u_x + v_y * u_y + v_z * u_z$$

Definition (Interpretacja geometryczna iloczynu skalarnego)

Wartość iloczynu wektorowego jest równa iloczynowi długości tych wektorów i kosinusa konta między nimi:

$$\overrightarrow{v} \cdot \overrightarrow{u} = |\overrightarrow{v}||\overrightarrow{u}| \cos(\overrightarrow{v}, \overrightarrow{u})$$

Działania na wektorach

Definition (Iloczyn skalarny)

Jest to pewne działanie przyporządkowujące parze wektorów pewną wartość rzeczywistą (skalarną).

$$\overrightarrow{v} \cdot \overrightarrow{u} = v_x * u_x + v_y * u_y + v_z * u_z$$

Definition (Interpretacja geometryczna iloczynu skalarnego)

Wartość iloczynu wektorowego jest równa iloczynowi długości tych wektorów i kosinusa konta między nimi:

$$\overrightarrow{v} \cdot \overrightarrow{u} = |\overrightarrow{v}||\overrightarrow{u}| \cos(\overrightarrow{v}, \overrightarrow{u})$$

Iloczyn wektorowy

Definition

Jest to pewne działanie przyporządkowujące parze wektorów $\overrightarrow{v}$ i $\overrightarrow{u}$ pewien wektor $\overrightarrow{w}$:

1. Jeśli $\overrightarrow{v}$ i $\overrightarrow{u}$ są liniowo zależne wynikiem jest wektor zerowy.
2. W przeciwnym wypadku:
   - $\overrightarrow{w} \perp \overrightarrow{v} \wedge \overrightarrow{w} \perp \overrightarrow{u}$
   - $|\overrightarrow{w}| = |\overrightarrow{v}||\overrightarrow{u}| \sin(\overrightarrow{v}, \overrightarrow{u})$

O tym jak go wyznaczyć za chwilę...

Definicja

Definition (Macierz)

Układ zapisanych w postaci prostokątnej tablicy danych nazywanych elementami bądź współczynnikami. Dla nas szczególnie interesującę bedą macierze kwadratowe o wymiarach 4x4, dla przykładu:

$$\mathbf{M} = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix}$$

Wyznacznik macierzy

Definition

Operacja przyporządkowująca każdej macierzy prostokątnej pewną wartość rzeczywistą.

$$\det M = \sum_{k=0}^{N} (-1)^{1+k} a_{ik} \det M_{i,k}$$

Wyznacznik macierzy a iloczyn wektorowy

Definition

$$\overrightarrow{v} \times \overrightarrow{u} = \det \begin{bmatrix} \hat{i} & \hat{j} & \hat{k} \\ v_x & v_y & v_z \\ v_x & u_y & u_z \end{bmatrix} = \hat{i}(v_y u_z - v_z u_y) + \hat{j}(v_x u_z - v_z u_x) +$$

$$\hat{k}(v_x u_y - v_y u_x) = \begin{bmatrix} v_y u_z - v_z u_y \\ v_x u_z - v_z u_x \\ v_x u_y - v_y u_x \end{bmatrix} = \overrightarrow{w}$$

Działania na macierzach

1. Transpozycja
2. Dodawanie
3. Mnożenie

Działania na macierzach

1. Transpozycja
2. Dodawanie
3. Mnożenie

Działania na macierzach

1. Transpozycja
2. Dodawanie
3. Mnożenie

Transformacje macierzowe

1. Translacja
2. Skalowanie
3. Obrót

Transformacje macierzowe

1. Translacja
2. Skalowanie
3. Obrót

Transformacje macierzowe

1. Translacja
2. Skalowanie
3. Obrót

Wprowadzenie do Direct3D

1. Pierwszy program
2. Gdzie, jak i co mam rysować?
3. Trzeci wymiar
4. Przekształcenia
5. Teksturowanie

Nagłówki

```
#ifdef DEBUG
#define D3D_DEBUG_INFO
  #pragma comment (lib, "d3dx9d.lib")
#else
  #pragma comment (lib, "d3dx9.lib")
#endif // DEBUG
#pragma comment (lib, "d3d9.lib")
#include <d3dx9.h>
```

Tworzenie urządzenia

```
IDirect3D9* d3d = Direct3DCreate9(D3D_SDK_VERSION);
// ... parametry
IDirect3DDevice9* device;
HRESULT result = d3d->CreateDevice(
                        D3DADAPTER_DEFAULT,
                        D3DDEVTYPE_HAL,
                        app.window_handle(),
    D3DCREATE_SOFTWARE_VERTEXPROCESSING, // ;(
                        &parameters,
                        &device);
if(FAILED(result)) return 1; // Smutno :(
```

Parametry urządzenia

```
// ... parametry
D3DPRESENT_PARAMETERS parameters;
ZeroMemory(&parameters, sizeof(parameters));
parameters.Windowed = true;
```

- `Windowed` – okno czy na pełny ekran?
- `SwapEffect` – zalecane `D3DSWAPEFFECT_DISCARD`
- `BackBufferCount` – ilość dodatkowych buforów
- `BackBufferFormat` – `D3DFMT_X8R8G8B8`
- `EnableAutoDepthStencil` – użyć bufora głębi?
- `BackBufferFormat` – `D3DFMT_D16` albo `D3DFMT_D24S8`

Główna pętla i sprzątanie

```
while(app.running()){
    device->Clear(0, NULL,
        D3DCLEAR_TARGET,
        D3DCOLOR_XRGB(0, 0, 0),
        1.0f, 0);
    device->BeginScene();
    // ...
    device->EndScene();
    device->Present(NULL, NULL, NULL, NULL);
}
device->Release();
d3d->Release();
```

Wierzchołki

```
#define Vertex_Format \
  (D3DFVF_XYZRHW | D3DFVF_DIFFUSE)
struct Vertex {
  float x, y, z, rhw;
  DWORD color;
};
```

- `D3DFVF_XYZRHW` – współrzędne do rysowania od razu na ekranie
- `D3DFVF_XYZ` – współrzędne do transformacji
- `D3DFVF_DIFFUSE` – kolor wierzchołka
- `D3DFVF_TEX1` – współrzędne tekstury

Tablice wierzchołków

```
Vertex triangle[] = {
// X     Y     Z  W  Color
   {400, 100, 1, 1, D3DCOLOR_XRGB(0, 0, 255)},
   {600, 500, 1, 1, D3DCOLOR_XRGB(0, 255, 0)},
   {200, 500, 1, 1, D3DCOLOR_XRGB(255, 0, 0)}
};
```

Rysujemy

```
device->SetFVF(Vertex_Format);
device->Clear(0, NULL, D3DCLEAR_TARGET,
        D3DCOLOR_XRGB(0, 0, 0), 1.0f, 0);

device->BeginScene();
device->DrawPrimitiveUP(D3DPT_TRIANGLESLIST, 1,
                        triangle, sizeof(Vertex));
device->EndScene();
device->Present(NULL, NULL, NULL, NULL);
```

Bufor wierzchołków

Jest jedynym słuszym sposobem rysowania czegokolwiek.

```
IDirect3DVertexBuffer9* buffer;
device->CreateVertexBuffer(3 * sizeof(Vertex), 0,
    Vertex_Format,
    D3DPOOL_MANAGED,
    &buffer,
    NULL);
// a po wywolaniu device->BeginScene() ...
device->SetStreamSource(0, buffer,
    0, sizeof(Vertex));
device->DrawPrimitive(D3DPT_TRIANGLELIST, 0, 1);
```

Rodzaje rysowania trójkątów

- `D3DPT_TRIANGLELIST` – Lista trójkątów
- `D3DPT_TRIANGLESTRIP` – Nowy wierzchołek tworzy trójkąt z dwoma poprzednimi
- `D3DPT_TRIANGLESFAN` – Nowy wierzchołek tworzy trójkąt z poprzednim i pierwszym wierzchołkiem (pierwszy wierchołek jest centrum "wachlarza")

Trzeci wymiar – czego potrzebujemy?

- Innych wierzchołków – `D3DFVF_XYZ`
- Przekształcenia projekcji – perspektywiczne / ortogonalne.
- Przekształcenia widoku, czyli oka kamery.
- Modelu do rysowania – przykładowo pudełka.

Trzeci wymiar – czego potrzebujemy?

- Innych wierzchołków – `D3DFVF_XYZ`
- Przekształcenia projekcji – perspektywiczne / ortogonalne.
- Przekształcenia widoku, czyli oka kamery.
- Modelu do rysowania – przykładowo pudełka.

Trzeci wymiar – czego potrzebujemy?

- Innych wierzchołków – `D3DFVF_XYZ`
- Przekształcenia projekcji – perspektywiczne / ortogonalne.
- Przekształcenia widoku, czyli oka kamery.
- Modelu do rysowania – przykładowo pudełka.

Trzeci wymiar – czego potrzebujemy?

- Innych wierzchołków – `D3DFVF_XYZ`
- Przekształcenia projekcji – perspektywiczne / ortogonalne.
- Przekształcenia widoku, czyli oka kamery.
- Modelu do rysowania – przykładowo pudełka.

Kiedy kolory nam się nudzą...

- Nowa flaga w formacie wierzchołków – `D3DFVF_TEX1`
- Współrzędne tekstury.
- Obiekty tekstur.

Kiedy kolory nam się nudzą...

- Nowa flaga w formacie wierzchołków – `D3DFVF_TEX1`
- Współrzędne tekstury.
- Obiekty tekstur.

Kiedy kolory nam się nudzą...

- Nowa flaga w formacie wierzchołków – `D3DFVF_TEX1`
- Współrzędne tekstury.
- Obiekty tekstur.

Tekstury w pigułce

- Najprościej:

```
LPDIRECT3DTEXTURE9 texture;
D3DXCreateTextureFromFile(device,
  "images\\texture.jpg",
  &texture);
```

- `D3DXCreateTextureFromFileEx()` daje większe możliwości.
- Włączamy teksturę za pomocą:

```
device->SetTexture(0, texture)
```

- Usuwamy standardowo za pomocą `texture->Release()`

Tekstury w pigułce

- Najprościej:

```
LPDIRECT3DTEXTURE9 texture;
D3DXCreateTextureFromFile(device,
  "images\\texture.jpg",
  &texture);
```

- `D3DXCreateTextureFromFileEx()` daje większe możliwości.
- Włączamy teksturę za pomocą:

```
device->SetTexture(0, texture)
```

- Usuwamy standardowo za pomocą `texture->Release()`

Tekstury w pigułce

- Najprościej:

```
LPDIRECT3DTEXTURE9 texture;
D3DXCreateTextureFromFile(device,
  "images\\texture.jpg",
  &texture);
```

- `D3DXCreateTextureFromFileEx()` daje większe możliwości.
- Włączamy teksturę za pomocą:

```
device->SetTexture(0, texture)
```

- Usuwamy standardowo za pomocą `texture->Release()`

Tekstury w pigułce

- Najprościej:

```
LPDIRECT3DTEXTURE9 texture;
D3DXCreateTextureFromFile(device,
  "images\\texture.jpg",
  &texture);
```

- `D3DXCreateTextureFromFileEx()` daje większe możliwości.
- Włączamy teksturę za pomocą:

```
device->SetTexture(0, texture)
```

- Usuwamy standardowo za pomocą `texture->Release()`

Odnośniki

- www.chadvernon.com/blog/tutorials/directx9/
- www.two-kings.de/tutorials/dxgraphics/
- www.danielloran.com/study/directx/Default.aspx
- www.codesampler.com/dx9src.htm