

ECE/CS 433 Introduction to Computer Graphics


Class 19
October 25, 2007

Pradeep Sen
Advanced Graphics Lab




Announcements

- Homework 3 is due Sunday


Scalar Invariant Raster Image Representation
Through Topological Encoding

Warren Hunt
Graduate Researcher
Advanced Graphics Lab

Friday, October 25, 2007
ECE 118 at noon



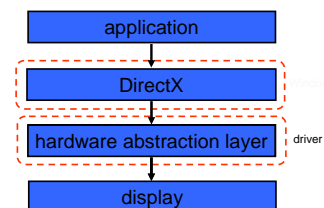
Last time

- Started talking about graphics API's

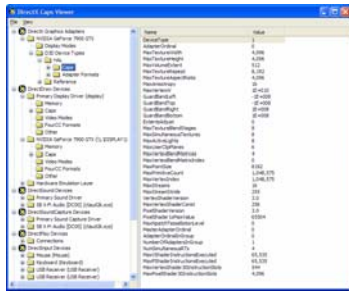
Today

- Graphics API's and vertex/fragment programs

Overview of DirectX



Viewing device capabilities



Taking a look at some code...

- Acquiring a Direct3D interface handle:


```
IDirect3D9 *md3dObject;
md3dObject = Direct3DCreate9(D3D_SDK_VERSION);
```
- Creating the Direct3D object:


```
IDirect3DDevice9 *gd3dDevice = 0;
md3dObject->CreateDevice(
    D3DADAPTER_DEFAULT, // primary adapter
    D3DDEVTYPE_HAL, // device type
    mhMainWnd, // window
    devBehaviorFlags, // vertex processing
    &md3dPP, // present params
    &gd3dDevice); // return created device
```

Taking a look at some code...

- D3DPRESENT_PARAMETERS:


```
typedef struct D3DPRESENT_PARAMETERS {
    UINT BackBufferWidth;
    UINT BackBufferHeight;
    D3DFORMAT BackBufferFormat;
    UINT BackBufferCount;
    D3DMULTISAMPLE_TYPE MultiSampleType;
    DWORD MultiSampleQuality;
    D3DSWAPEFFECT SwapEffect;
    HWND hDeviceWindow;
    BOOL Windowed;
    BOOL EnableAutoDepthStencil;
    D3DFORMAT AutoDepthStencilFormat;
    DWORD Flags;
    UINT FullScreen_RefreshRateInHz;
    UINT PresentationInterval;
} D3DPRESENT_PARAMETERS;
```

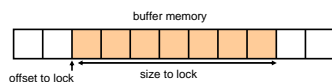
Taking a look at some code...

- Sample settings


```
D3DPRESENT_PARAMETERS d3dpp;
d3dpp.BackBufferWidth = 800;
d3dpp.BackBufferHeight = 600;
d3dpp.BackBufferFormat = D3DFMT_X8R8G8B8;
d3dpp.BackBufferCount = 1;
d3dpp.MultiSampleType = D3DMULTISAMPLE_NONE;
d3dpp.MultiSampleQuality = 0;
d3dpp.SwapEffect = D3DSWAPEFFECT_DISCARD;
d3dpp.hDeviceWindow = hwnd;
d3dpp.Windowed = true;
d3dpp.EnableAutoDepthStencil = true;
d3dpp.AutoDepthStencilFormat = D3DFMT_D24S8;
d3dpp.Flags = 0;
d3dpp.FullScreen_RefreshRateInHz = D3DPRESENT_RATE_DEFAULT;
d3dpp.PresentationInterval = D3DPRESENT_INTERVAL_IMMEDIATE;
```

Vertex and index buffers

- Vertex buffer – a chunk of contiguous memory that contains the vertex data (represented by the IDirect3DVertexBuffer9 interface)
- Index buffer – a chunk of contiguous memory that contains the index data (represented by the IDirect3DIndexBuffer9 interface)
- Access this memory through Lock/Unlock methods



Taking a look at some code...

- Creating a vertex buffer


```
gd3dDevice->CreateVertexBuffer(
    8 * sizeof(VertexPos), // num bytes
    D3DUSAGE_WRITEONLY, // dynamic/static
    0, // vertex format
    D3DPPOOL_MANAGED, // memory pool
    &nVB, // handle to vb
    0); // n/a
```

Simple example

Program to draw two shapes

```
// create two vertex buffers, one for each shape
gd3dDevice->CreateVertexBuffer(sizeof(shape),
    D3DUSAGE_WRITEONLY, 0, D3DPPOOL_MANAGED, &pShape1VB, NULL);
gd3dDevice->CreateVertexBuffer(sizeof(shape),
    D3DUSAGE_WRITEONLY, 0, D3DPPOOL_MANAGED, &pShape2VB, NULL);

// pass down vertex information for both
pShape1VB->Lock(0, sizeof(pData), (void**)&pData, 0); //lock
memcpy(pData, shape1, sizeof(shape)); //copy data to buffer
pShape1VB->Unlock();

pShape2VB->Lock(0, sizeof(pData), (void**)&pData, 0); //lock
memcpy(pData, shape2, sizeof(shape)); //copy data to buffer
pShape2VB->Unlock();
```



ECE/CS 433 Introduction to Computer Graphics
Pradeep Sen

Class 19 – October 25, 2007

Simple example

Now onto the drawing loop

```
gd3dDevice->Clear(0, NULL, D3DCLEAR_TARGET | D3DCLEAR_ZBUFFER,
    D3DCOLOR_XRGB(0,0,0), 1.0f, 0);
gd3dDevice->BeginScene(); // start drawing

// draw 1st shape
gd3dDevice->SetStreamSource(0, pShape1VB, 0, sizeof(D3DVERTEX));
gd3dDevice->DrawPrimitive(D3DPT_TRIANGLELIST, 0, 1);

// draw 2nd shape
gd3dDevice->SetStreamSource(0, pShape2VB, 0, sizeof(D3DVERTEX));
gd3dDevice->DrawPrimitive(D3DPT_TRIANGLELIST, 0, 1);

gd3dDevice->EndScene(); // stop drawing
gd3dDevice->Present(NULL, NULL, NULL, NULL); // swap buffers
```



ECE/CS 433 Introduction to Computer Graphics
Pradeep Sen

Class 19 – October 25, 2007

Some things are easier than OpenGL

Creating and using textures

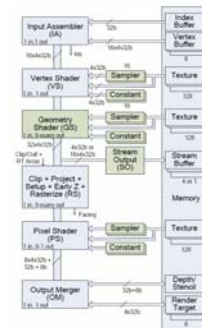
```
D3DXCreateTextureFromFile(g_App.GetDevice(), "texture.png",
    &pMyTexture);
g_App.GetDevice()->SetTexture(0, pMyTexture);
```



ECE/CS 433 Introduction to Computer Graphics
Pradeep Sen

Class 19 – October 25, 2007

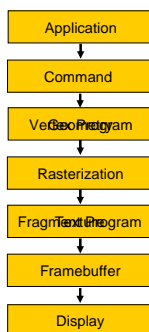
Direct3D 10 rendering pipeline



ECE/CS 433 Introduction to Computer Graphics
Pradeep Sen

Class 19 – October 25, 2007

Programmable graphics pipeline



ECE/CS 433 Introduction to Computer Graphics
Pradeep Sen

Class 19 – October 25, 2007

Vertex program

Extension GL_ARB_vertex_program



ECE/CS 433 Introduction to Computer Graphics
Pradeep Sen

Class 19 – October 25, 2007

Simple vertex program

```
!!ARBvpl.0
ATTRIB v          = vertex.position; # vertex position

# pass in the.mvp matrix
PARAM.mvp[4]     = { state.matrix.mvp };

# define the output
OUTPUT.oPos      = result.position # output vertex

# transform the vertex v by the.mvp matrix
DP4.oPos.x,.mvp[0],v;
DP4.oPos.y,.mvp[1],v;
DP4.oPos.z,.mvp[2],v;
DP4.oPos.w,.mvp[3],v;

END
```

Simple vertex program

```
!!ARBvpl.0
ATTRIB v          = vertex.position; # vertex position

# pass in the.mvp matrix
PARAM.mvp[4]     = { state.matrix.mvp };

# define the output
OUTPUT.oPos      = result.position # output vertex

# temporaries
TEMP.temp;

# transform the vertex v by the.mvp matrix
DP4.temp.x,.mvp[0],v;
DP4.temp.y,.mvp[1],v;
DP4.temp.z,.mvp[2],v;
DP4.temp.w,.mvp[3],v;

MUL.oPos,temp,9999;

END
```

Per-vertex attributes (inputs)

- Some of the per-vertex inputs to the vertex program include:
 - vertex.position
 - vertex.weight
 - vertex.weight[n]
 - vertex.normal
 - vertex.color
 - vertex.fogcoord
 - vertex.texcoord, unit 0
 - vertex.texcoord[n], unit n
 - vertex.attrib[n]

Vertex outputs

- The interpolant outputs of the vertex program (and the inputs to the fragment program) include:
 - result.position
 - result.color
 - result.fogcoord
 - result.texcoord, unit 0
 - result.texcoord[n], unit n

Vertex program instruction set

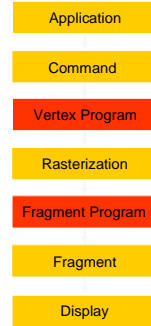
ABS	v	v	absolute value
ADD	v,v	v	add
ARL	s	a	address register load
DP3	v,v	ssss	3-component dot product
DP4	v,v	ssss	4-component dot product
DPH	v,v	ssss	homogeneous dot product
DST	v,v	v	distance vector
EX2	s	ssss	exponential base 2
EXP	s	v	exponential base 2 (approximate)
FLR	v	v	floor
FRC	v	v	fraction
LG2	s	ssss	logarithm base 2
LIT	v	v	compute light coefficients
LOG	s	v	logarithm base 2 (approximate)
MAD	v,v,v	v	multiply and add

Vertex program instruction set

MAX	v,v	v	maximum
MIN	v,v	v	minimum
MOV	v	v	move
MUL	v,v	v	multiply
POW	s,s	ssss	exponentiate
RCP	s	ssss	reciprocal
RSQ	s	ssss	reciprocal square root
SGE	v,v	v	set on greater than or equal
SLT	v,v	v	set on less than
SUB	v,v	v	subtract
SWZ	v	v	extended swizzle
XPD	v,v	v	cross product

Vertex program binding and loading

Programmable graphics pipeline



Fragment program

- Often known as “pixel shaders”
- Extension GL_ARB_fragment_program

Simple fragment program

```
!!ARBfp1.0  
  
MOV result.color, {1, 0, 0, 0};  
  
END
```

Per-fragment inputs

- Also known as interpolants
- Same list as before (output of vertex program)

Fragment program outputs

- Only two outputs:
 - result.color
 - result.depth

Fragment program instruction set

ABS	v	v	absolute value
ADD	v,v	v	add
CMP	v,v,v	v	compare
COS	s	ssss	cosine with reduction to $[-\pi,\pi]$
DP3	v,v	ssss	3-component dot product
DP4	v,v	ssss	4-component dot product
DPH	v,v	ssss	homogeneous dot product
DST	v,v	v	distance vector
EX2	s	ssss	exponential base 2
FLR	v	v	floor
FRC	v	v	fraction
KIL	v	v	kill fragment
LG2	s	ssss	logarithm base 2
LIT	v	v	compute light coefficients
LRP	v,v,v	v	linear interpolation
MAD	v,v,v	v	multiply and add



ECE/CS 433 Introduction to Computer Graphics
Pradeep Sen

Class 19 – October 25, 2007

Fragment program instruction set

MAX	v,v	v	maximum
MIN	v,v	v	minimum
MOV	v	v	move
MUL	v,v	v	multiply
POW	s,s	ssss	exponentiate
RCP	s	ssss	reciprocal
RSQ	s	ssss	reciprocal square root
SCS	s	ss--	sine/cosine without reduction
SGE	v,v	v	set on greater than or equal
SIN	s	ssss	sine with reduction to $[-\pi,\pi]$
SLT	v,v	v	set on less than
SUB	v,v	v	subtract
SWZ	v	v	extended swizzle
TEX	v,u,t	v	texture sample
TXB	v,u,t	v	texture sample with bias
TXP	v,u,t	v	texture sample with projection
XPD	v,v	v	cross product



ECE/CS 433 Introduction to Computer Graphics
Pradeep Sen

Class 19 – October 25, 2007

Reading

- Angel thru Ch 8



ECE/CS 433 Introduction to Computer Graphics
Pradeep Sen

Class 19 – October 25, 2007