Contents

- Texture mapping
- Texture mapping on Direct3D
Texture Mapping

Kang, Seong-tae
Computer Graphics, 2008 Spring
Texture

- In real world, only few surfaces have uniform color
- How to describe non-uniform surface?
  - Tessellation
    - Too complicated!

![Diagram showing tessellation process](image-url)
Texture

- Bitmap describes detailed surface information
- Wrapping wallpaper
Texture Coordinate

- Independent of the position coordinate
  - [0,1] ranging
  - UV for 2D texture
  - UVW for 3D texture
Texture Coordinate

- Texture coordinates indicate mappings between vertices and a texture
3D Texture

(0.5,0,0) (0.5,1,0)
(0.5,0,1) (0.5,1,1)
(0,0,0) (1,1,1)
Texture Filtering

- Nearest-point sampling
  - Pick the nearest grid value
  - Jaggy effect

- Linear texture filtering
  - Bilinear filtering (2D)
    \[ F = dF_0 + cF_1 \]
    \[ = d(bF_{00} + aF_{01}) + c(bF_{10} + aF_{11}) \]
    \[ = caF_{11} + cbF_{10} + daF_{01} + dbF_{00} \]
  - Trilinear filtering (3D)
    - Get a plane by linear interpolation, and bilinear interpolate in the plane
Addressing Convention

- Behavior for out-of-range sampling
  - Border Color
  - Wrap
  - Mirror
  - Clamp

- Diagrams showing different border color and wrap/mirror behaviors.
Mipmap texture

- Mipmap
  - pre-calculated, optimized collections of bitmap images that accompany a main texture

- Mipmap filtering
  - A high-resolution mipmap image is used for objects that are close to the user
  - A lower-resolution images are used as the object appears farther away

- Pros
  - Quality improvement

- Cons
  - More memory consumption
Anisotropic Texture Filtering

- Use adaptive mipmaps on per-pixel level
  - Pros
    - Degrades blurring artifacts of extreme-angle surfaces
  - Cons
    - Expensive operation
- All recent hardwares support this method

Bilinear filtering  Anisotropic filtering
Texture Mapping on Direct3D

Kang, Seong-tae
Computer Graphics, 2008 Spring
D3D Texture Mapping Overview

SetTextureStageState

TextureStage Slot 0
TextureStage Slot 1
TextureStage Slot 2

TextureStage Slot 7

VertexBuffer

Position
Diffuse
TexCoord0
TexCoord1
TexCoord2

SetStreamSource

SetTexture

Sampler

Sampler

Sampler

SetSamplerState

Stream Source

Blender

Output
Setting Texture Coordinates

- Texture coordinate representation
  - Float or D3DXVECTOR

```c
struct CUSTOMVERTEX
{
    D3DXVECTOR3 position; // The position
    D3DCOLOR color; // The color
    FLOAT tu, tv; // The texture coordinates
};
```
Setting Texture Coordinates

- **D3DFVF**
  - D3DFVF_TEX\(n\) : \(n\) texture coordinates for a vertex
  - D3DFVF_TEXCOORDSIZE\(m(k)\) : \(k\)-th texture coordinate is \(m\)-dimension

<table>
<thead>
<tr>
<th>D3DFVF_XYZ</th>
<th>D3DFVF_TEX2</th>
</tr>
</thead>
<tbody>
<tr>
<td>3D position and two 2D texture coordinates</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>D3DFVF_XYZ</th>
<th>D3DFVF_TEX1</th>
<th>D3DFVF_TEXCOORDSIZE3(0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3D position with one 3D texture coordinate</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Creating a Texture

2D Texture

HRESULT IDirect3DDevice9::CreateTexture(UINT Width, UINT Height, UINT Levels, DWORD Usage, D3DFORMAT Format, D3DPOOL Pool, IDirect3DTexture9** ppTexture, HANDLE* pSharedHandle);

- **Width, Height**: Size of the texture
- **Levels**: Number of levels in the texture. 0 for no mipmap
- **Usage**: Usage of the resource. See D3DUSAGE
- **Format**: Format of the texture
- **Pool**: Description of the memory class that holds the buffer. See D3DPOOL
- **ppTexture**: Pointer of the texture object
- **pSharedHandle**: Not used
Creating a Texture

3D Texture

HRESULT IDirect3DDevice9::CreateTexture
    (UINT Width, UINT Height, UINT Depth, UINT Levels,
    DWORD Usage, D3DFORMAT Format, D3DPOOL Pool,
    IDirect3DTexture9** ppTexture, HANDLE* pSharedHandle);
Locking & Unlocking

- Similar to vertex buffer locking/unlocking

2D locking

- D3DLOCKED_RECT & LockRect

```c
typedef struct D3DLOCKED_RECT {
    INT Pitch; // # of bytes in one row of the surface
    void * pBits; // Pointer to the locked bits
} D3DLOCKED_RECT, *LPD3DLOCKED_RECT;

HRESULT IDirect3DTexture9::LockRect(UINT Level, D3DLOCKED_RECT * pLockedRect, 
    CONST RECT * pRect, DWORD Flags);

- Level : The level of surfaces of the texture to lock
- pLockedRect : Pointer to a D3DLOCKED_RECT structure
- pRect : Pointer to a RECT structure that specifies the region to lock. NULL means the
  whole area.
- Flags : Locking flags
Creating a Texture

- 3D locking
  - D3DLOCKED_BOX & LockBox

```c
typedef struct D3DLOCKED_BOX {
    int RowPitch; // # of bytes in one row
    int SlicePitch; // # of bytes in one slice
    void * pBits; // Pointer to the locked bits
} D3DLOCKED_BOX, *LPD3DLOCKED_BOX;

HRESULT IDirect3DVolumeTexture9::LockBox(UINT Level,
                                      D3DLOCKED_BOX * pLockedBox,
                                      CONST D3DBOX * pBox, DWORD Flags);
```

- Level: The level of surfaces of the texture to lock
- pLockedBox: Pointer to a D3DLOCKED_BOX structure
- pBox: Pointer to a D3DBOX struct that specifies the region to lock. NULL means the whole area.
- Flags: Locking flags
Creating a Texture

- **Indirect creation**
  - Create a texture object suitable for the source
  - Fill the texture object with source data
  - Build mipmap sublevel textures

- **Source**
  - File
    - BMP, DDS, DIB, HDR, JPG, PFM, PNG, PPM, TGA
  - Memory
  - Resource

```cpp
If(FAILED(D3DXCreateTextureFromFile(g_pd3dDevice, "Banana.bmp", &g_pTexture )))
    return E_FAIL;
```
Setting Textures

- Attach the texture to the device
  - TextureStage
    - View in which the device looks the texture resource
    - Up to 8 stages

- Setting texture stage options
  - SetTextureStageState
    - Blending operations
    - Index of the texture coordinate of a vertex to refer

```c
// Attach the texture to the device
void AttachTexture()
{
    g_pd3dDevice->SetTexture( 0, g_pTexture );
    g_pd3dDevice->SetTextureStageState( 0, D3DTSS_TEXCOORDINDEX, 1 );
    g_pd3dDevice->SetTextureStageState( 0, D3DTSS_COLOROP, D3DTOP_MODULATE );
    g_pd3dDevice->SetTextureStageState( 0, D3DTSS_COLORARG1, D3DTA_TEXTURE );
    g_pd3dDevice->SetTextureStageState( 0, D3DTSS_COLORARG2, D3DTA_DIFFUSE );
    g_pd3dDevice->SetTextureStageState( 0, D3DTSS_ALPHAOP, D3DTOP_DISABLE );
}
```
Setting Textures

- Setting sampler options (filtering methods)
  - SetSamplerState
    - Boundary conditions
    - Filtering methods

```cpp
g_pd3dDevice->SetSamplerState( 0, D3DSAMP_MAGFILTER, D3DTEXF_LINEAR);
g_pd3dDevice->SetSamplerState( 0, D3DSAMP_MINFILTER, D3DTEXF_ANISOTROPIC);
g_pd3dDevice->SetSamplerState( 0, D3DSAMP_MIPFILTER, D3DTEXF_POINT);
g_pd3dDevice->SetSamplerState( 0, D3DSAMP_ADDRESSU, D3DTADDRESS_WRAP);
g_pd3dDevice->SetSamplerState( 0, D3DSAMP_ADDRESSV, D3DTADDRESS_CLAMP);
```
Result of Tutorial 5

Banana.bmp