3D Game Programming 07

afewhee@gmail.com

- 스텐실
- DX Font
  - 2D Font
  - 3D Font
  - BackBuffer
- Texture Font
  - 2D Font
  - 3D Font
- 실습

- 스텐실 버퍼 (Stencil Buffer)
  - 깊이 버퍼처럼 후면 버퍼의 Pixel 갱신 여부를 위한 버퍼로 비교에 대한 Masking 할 수 있는 값을 가짐

- 스텐실 테스트 (Stencil Test)
  - 후면 버퍼의 색상을 갱신여부를 위해 마스킹을 이용해 스텐실 버퍼의 값을 비교하는 것. TRUE, or FALSE 임

  - Stencil Test = (Application Ref Value & Application Stencil Mask) StencilFunc (Stencil Buffer Value & Application Stencil Mask)

Ex)

```
pDevice->SetRenderState( D3DRS_STENCILENABLE, TRUE);
pDevice->SetRenderState( D3DRS_STENCILREF, 0x1 );
pDevice->SetRenderState( D3DRS_STENCILMASK, 0x0000FFFF);
pDevice->SetRenderState( D3DRS_STENCILFUNC, D3DCMP_NOTEQUAL);
```

➔ Test = (0x1 & 0x0000FFFF) != (Stencil Buffer Value & 0x0000FFFF)

  - D3DCMP_...: NEVER-항상 실패, ALWAYS-항상 성공, LESS - '<', EQUAL-'=', GREATER-'>', LESSEQUAL, GREATEREQUAL

- 테스트 후 스텐실 버퍼 갱신
  - pDevice->SetRenderState(D3DRS_STENCIL..., Stencil_Operation)
  - FAIL: 테스트 실패 시 해당 버퍼를 Stencil Operation 값으로 설정
  - ZFAIL: 스텐실 테스트는 성공, 깊이 테스트 실패 시 적용
  - PASS: 스텐실, 깊이 테스트 모두 성공

- Stencil Operation
  - D3DSTENCILOP_...
  - KEEP: 현재의 스텐실 값 유지
  - ZERO: 0으로 설정
  - REPLACE: Application Ref value로 대치
  - INVERT: 반전
  - INCRSAT: 값을 증가. 최대 값보다 크면 최대값. 2^n-1 ← n-Bit
  - DECRSAT: 값을 감소, 0보다 작으면 0

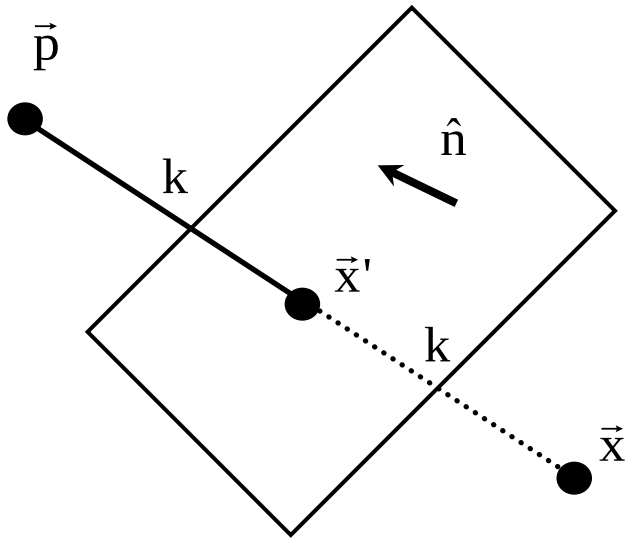
- 유용한 Rendering State Type
  - D3DRS_ZENABLE
    - z(Depth)-Buffering 활성화
    - 상태를 FALSE로 놓으면 depth test는 거치지 않음
  - D3DRS_ZWRITEENABLE
    - z(Depth)-Buffer 갱신
    - D3DRS_ZFUNC 에 의존
  - D3DRS_ZFUNC
    - 깊이 값 비교 형식
    - default: D3DCMP_LESSEQUAL ← '=<'
  - Depth Bias
    - z-Fighting을 줄이기 위해 z 값을 이동
    - Offset = max * D3DRS_SLOPESCALEDEPTHBIAS + D3DRS_DEPTHBIAS

    Ex)

```
FLOATfSlope=0.f;
FLOATfBias = 1.f;
pDevice->SetRenderState( D3DRS_SLOPESCALEDEPTHBIAS, *((DWORD*)&fSlope) );
pDevice->SetRenderState( D3DRS_DEPTHBIAS, *((DWORD*)&fBias) );
```

  - D3DRS_COLORWRITEENABLE
    - 색상 버퍼 갱신 :default - 0xF
    - pDevice->SetRenderState(D3DRS_COLORWRITEENABLE, (D3DCOLORWRITEENABLE_ALPHA | D3DCOLORWRITEENABLE_BLUE| D3DCOLORWRITEENABLE_GREEN | D3DCOLORWRITEENABLE_RED);

- Reflection Matrix
  - D3DXMatrixReflect()



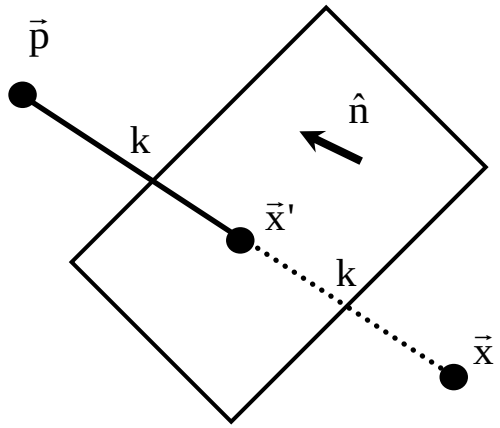
$$\vec{x} = \vec{p} + k\vec{L}$$
$$\hat{n}\cdot\vec{x}' + d = 0$$
$$\hat{n}\cdot\vec{p} = \hat{n}\cdot(\vec{x}' + k\hat{n})$$
$$= -d + k$$
$$k = \hat{n}\cdot\vec{p} + d$$

$$\vec{x} = \vec{p} - 2k\hat{n}$$
$$= \vec{p} - 2(\hat{n}\cdot\vec{p} + d)\hat{n}$$
$$= \vec{p}\cdot(\overleftrightarrow{I} - 2\hat{n}\hat{n}) - 2d\hat{n}$$

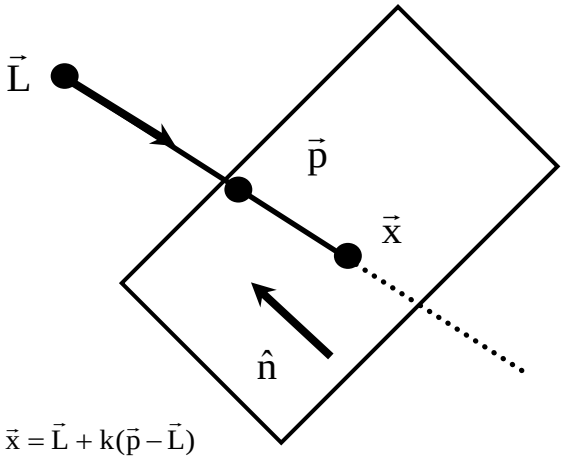
$$\vec{x} = \vec{p}\cdot\overleftrightarrow{M}_{Reflect}$$

$$\therefore \overleftrightarrow{M}_{Reflect} = \begin{pmatrix} 1-2n_xn_x & -n_xn_y & -n_xn_z & 0 \\ -2n_yn_x & 1-2n_yn_y & -n_yn_z & 0 \\ -2n_zn_x & -n_zn_y & 1-2n_zn_z & 0 \\ -2dn_x & -2dn_y & -2dL_z & 1 \end{pmatrix}$$

- Shadow Matrix
  - D3DXMatrixShadow()



$$\vec{x} = \vec{p} + k\vec{L}$$
$$\hat{n}\cdot\vec{x} + d = 0$$
$$\hat{n}\cdot\vec{p} + k(\hat{n}\cdot\vec{L}) + d = 0$$
$$\therefore \vec{x} = \vec{p} - \frac{d + \hat{n}\cdot\vec{p}}{\hat{n}\cdot\vec{L}}\vec{L}$$
$$= \frac{\vec{p}\cdot((\hat{n}\cdot\vec{L})\vec{I} - \hat{n}\vec{L}) - d\vec{L}}{\hat{n}\cdot\vec{L}}$$
$$\vec{x} = \vec{p}\cdot\vec{M}_{Shadow}$$



$$\vec{x} = \vec{L} + k(\vec{p} - \vec{L})$$
$$\hat{n}\cdot\vec{x} + d = 0$$
$$\hat{n}\cdot\vec{L} + k(\hat{n}\cdot\vec{p} - \hat{n}\cdot\vec{L}) + d = 0$$
$$\therefore \vec{x} = \vec{L} + \frac{d + \hat{n}\cdot\vec{L}}{-\hat{n}\cdot(\vec{p} - \vec{L})}(\vec{p} - \vec{L})$$
$$= \frac{-\vec{L}(\hat{n}\cdot\vec{p}) + \vec{L}(\hat{n}\cdot\vec{L}) + (\hat{n}\cdot\vec{L})\vec{p} + d\vec{p} - \vec{L}(\hat{n}\cdot\vec{L}) - d\vec{L}}{-\hat{n}\cdot\vec{p} + \hat{n}\cdot\vec{L}}$$
$$= \frac{\vec{p}\cdot((d + \hat{n}\cdot\vec{L})\vec{I} - \hat{n}\vec{L}) - d\vec{L}}{-\hat{n}\cdot\vec{p} + \hat{n}\cdot\vec{L}}$$
$$\vec{x} = \vec{p}\cdot\vec{M}_{Shadow}$$

$$\vec{P}(a,b,c,d) = (n_x, n_y, n_z, d)$$
$$\vec{L} = (L_x, L_y, L_z, L_w)$$
$$d + \hat{n}\cdot\vec{L} = \vec{P}\cdot\vec{L} = D$$
$$\hat{n}\cdot\vec{L} = D - P_d L_w$$

$$\vec{M}_{Shadow} = \begin{pmatrix} D - P_a L_x & -P_a L_y & -P_a L_z & -P_a L_w \\ -P_b L_x & D - P_b L_y & -P_b L_z & -P_b L_w \\ -P_c L_x & -P_c L_y & D - P_c L_z & -P_c L_w \\ -P_d L_x & -P_d L_y & -P_d L_z & D - P_d L_w \end{pmatrix}$$

or $\vec{M}_{Shadow} = D\vec{I} - \vec{P}\vec{L}$

$$\therefore \vec{M}_{Parallel} = \begin{pmatrix} \hat{n}\cdot\vec{L} - n_x L_x & -n_x L_y & -n_x L_z & 0 \\ -n_y L_x & \hat{n}\cdot\vec{L} - n_y L_y & -n_y L_z & 0 \\ -n_z L_x & -n_z L_y & \hat{n}\cdot\vec{L} - n_z L_z & 0 \\ -dL_x & -dL_y & -dL_z & \hat{n}\cdot\vec{L} \end{pmatrix}$$

$$\therefore \vec{M}_{Spot} = \begin{pmatrix} d + \hat{n}\cdot\vec{L} - n_x L_x & -n_x L_y & -n_x L_z & -n_x \\ -n_y L_x & d + \hat{n}\cdot\vec{L} - n_y L_y & -n_y L_z & -n_y \\ -n_z L_x & -n_z L_y & d + \hat{n}\cdot\vec{L} - n_z L_z & -n_z \\ -dL_x & -dL_y & -dL_z & \hat{n}\cdot\vec{L} \end{pmatrix}$$

- 2D Font
  - ID3DXFont
  - D3DXCreateFont(...);
  - D3DXCreateFontIndirect(...) ← Need FONT_DESC structure 변수

- 3D Font
  - Unicode 문자열만 허용
  - GDI폰트를 이용함
  - 문자열을 메쉬(ID3DXMesh)로 생성: D3DXCreateText(...);

```
Ex)
ID3DXMesh*      m_StrMesh;

// Setup LogFont
LOGFONT lf;
...
// Create the text mesh based on the selected font in the HDC.
D3DXCreateText(GDEVICE, hdc, _T("안녕하세요Hello world"), 0.5f, 1.f, &m_StrMesh, 0, 0);

// Rendering
GDEVICE->SetMaterial(&WHITE_MTRL);
m_StrMesh->DrawSubset(0);

...
```


- DX Font는 하나의 폰트 객체를 이용해서 여러 문자열을 표현하므로 문자열을 변경할 때 속도 저하 발생
  - Backbuffer에 출력, Texture에 출력

- BackBuffer
  - Backbuffer의 DC에 직접 문자열을 출력
  - Backbuffer 포멧은 알파가 없는 포멧.
    - D3DFMT_R5G6B5, D3DFMT_X1R5G5B5, D3DFMT_R8G8B8, D3DFMT_X8R8G8B8
  - 디바이스를 생성할 때 Present Parameter의 Flags 값이 D3DPRESENTFLAG_LOCKABLE_BACKBUFFER 이어야 함
    - m_d3dpp.Flags |= D3DPRESENTFLAG_LOCKABLE_BACKBUFFER;

  - 문자열 출력 Ex)

```
LPDIRECT3DSURFACE9  m_pBackBuffer;
// 후면 버퍼의 서피스를 가져온다
m_pd3dDevice->GetBackBuffer( 0, 0, D3DBACKBUFFER_TYPE_MONO, &m_pBackBuffer );
m_pBackBuffer->GetDC(&m_hDC);

// 서피스의 DC를 가져온다
if(m_hDC)
{
	TCHAR sMsg[256] = "";

	sprintf( sMsg, "이 문장은 DC를 이용한 것이다.  %s %s", m_strDeviceStats, m_strFrameStats );

	// DC에 GDI방식으로 문자열을 출력한다.
	SetBkMode(m_hDC, TRANSPARENT);
	SetTextColor(m_hDC, RGB(128, 255, 255));
	TextOut(m_hDC, 10, 10, sMsg, strlen(sMsg));

	// DC 해제
	m_pBackBuffer ->ReleaseDC(m_hDC);
}

// 서피스 해제
m_pBackBuffer >Release();
```


- 후면 버퍼에 출력을 하면 속도의 잇점이 있으나 윈도우의 겹침등을 표현 할 때 제약이 많음 → Texture에 출력

- Texture에 출력
  - Direct3D9은 후면 버퍼와 같이 알파가 없는 텍스처는 GetDC() 함수를 호출 할 수 있음

- 장점
  - 문자열이 변화가 없는 경우 속도에 이득텍스처를
  - 문자열 테두리를 쉽게 구현
  - 폰트를 폴리곤에 붙일 수 있음

- 방법
  - 불투명 임시 텍스처를 실시간으로 생성: D3DXCreateTexture(..., D3DFMT_X8R8G8B8,..., &pTempTex)
  - 불투명 텍스처에서 서피스 가져옴: pTempTex->GetSurfaceLevel(0, &pSfTxtS);
  - 불투명 텍스처의 서피스의 DC를 가져옴: pSfTxtS->GetDC(&hDC);
  - DC에 문자열 출력
  - 렌더링 할 반 투명 텍스처를 생성하고 이 반투명 텍스처의 서피스에 불 투명 텍스처 서피스를 D3DXLoadSurfaceFromSurface () 함수를 이용해서 복사
    - D3DXCreateTexture(..., D3DFMT_A8R8G8B8, ...&m_pTxD);
    - D3DXLoadSurfaceFromSurface(pSfTxtD, ..., pSfTxtS, ...);
  - 알파가 있는 반투명 텍스처를 화면에 렌더링

- Diffuse Map, Detail Map을 적용한 지형에 색상을 적용해 보시오.

- 태양계의 행성운동을 Tree 자료 구조를 이용해서 표현해 보시오.