3D Game Programming 23
- Height Field: Splatting

afewhee@gmail.com

3. Alpha Splatting

- Texture Splatting: 텍스처를 혼합하는 기술
  - 최종 색상 = Σ $Texture_i$ * $Weight_i$
- Weight를 지정하는 방법
  - 알파 텍스처
    - 실시간으로 Weight 를 설정할 수 있는 텍스처를 만들어 사용
  - 정점의 Diffuse 또는 Specular 값
    - 정점의 색상 값을 계속 변경 여러 번 렌더링
- 렌더링 방법
  - Multi-Texturing이 지원 불가능한 경우
    - 여러 번 렌더링
  - Multi-Texturing이 지원:
    - 한 번(One Pass)에 렌더링
  - 쉐이더를 사용할 경우
    - Pixel Shader 이용
- 픽셀 쉐이더
  - 고정 파이프 라인에서보다 좀 더 많은 텍스처 스플레팅이 가능

- 알파 텍스처를 사용한 스플레팅
  - ①, ② 그림에서처럼 각 텍스처에 알파 텍스처를 혼합
  - 이를 다시 전부 더해 ③과 같이 최종 색상을 정함

(①, ②에서 알파텍스처의 알파 값은 검은색은 1, 흰색은 0)

프로그램 구현방법

```
//①, ②에 대해서 다음과 같이 텍스처 스테지를 설정
// pDev->SetTextureStageState( 0, D3DTSS_COLORARG1, D3DTA_TEXTURE );
// pDev->SetTextureStageState( 0, D3DTSS_COLOROP,   D3DTOP_SELECTARG1 );

pDev->SetTextureStageState( 0, D3DTSS_ALPHAARG1, D3DTA_DIFFUSE);
pDev->SetTextureStageState( 0, D3DTSS_ALPHAOP,   D3DTOP_SELECTARG1 );

pDev->SetTextureStageState( 1, D3DTSS_COLORARG1, D3DTA_CURRENT );
pDev->SetTextureStageState( 1, D3DTSS_COLOROP,   D3DTOP_SELECTARG1);

pDev->SetTextureStageState( 1, D3DTSS_ALPHAARG1, D3DTA_TEXTURE);
pDev->SetTextureStageState( 1, D3DTSS_ALPHAOP,   D3DTOP_SELECTARG1 );
```

```
//③알파 블렌딩
pDev->SetRenderState( D3DRS_ALPHATESTENABLE,   FALSE);
pDev->SetRenderState( D3DRS_ALPHABLENDENABLE,   TRUE );
pDev->SetRenderState( D3DRS_SRCBLEND,  D3DBLEND_SRCALPHA );
pDev->SetRenderState( D3DRS_DESTBLEND, D3DBLEND_INVSRCALPHA  );

for(i=0; i<m_nTex; ++i)
{
   pDev->SetTexture(0, m_pTxB[i]);   // 디퓨즈 텍스처
   pDev->SetTexture(1, m_pTxA[i]);   // 알파 텍스처
   pDev->DrawPrimitiveUP(D3DPT_TRIANGLEFAN, 2, m_pVtx[i], sizeof(VtxDUV1));
}
```

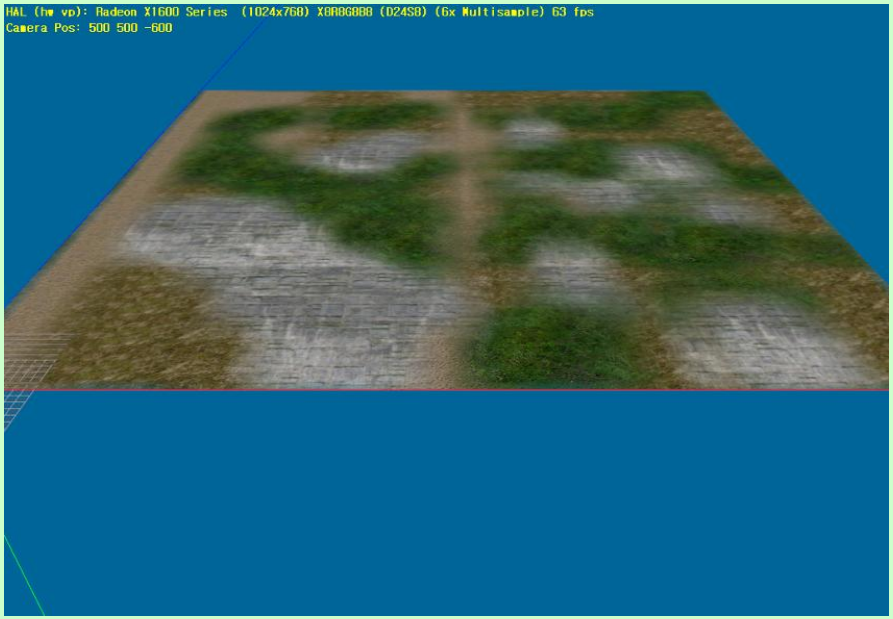
- 지형 위의 Alpha Splatting
  - Run Time Alpha 텍스처 생성:
    D3DXCreateTexture(m_pDev, nTile, nTile, 1, 0, D3DFMT_A8R8G8B8, D3DPOOL_MANAGED, ...);

```
void CLcSplt::CalculateMapTile(int nTx, PDTX& xpTxA)
{
   ...
   for (z=0; z<(int)sf.Height; ++z)
   {
      for (x=0; x<(int)sf.Width; ++x)
      {
         fAlpha = 0.0f;

         for(m=nZBgn; m<=nZEnd; ++m)
         {
            for(n=nXBgn; n<=nXEnd; ++n)
            {
               if(m_IdxLA[z+m][x+n] ==nTx)
                  fAlpha +=1.f;
            }
         }

         ...
         fAlpha *=4.f;
         fAlpha /= fN;
         ...
         pPxl[sf.Width*z + x] = D3DXCOLOR(1,1,1, fAlpha);
      }// for
   }// for
   ....
```


- 지형 위의 Alpha Splatting

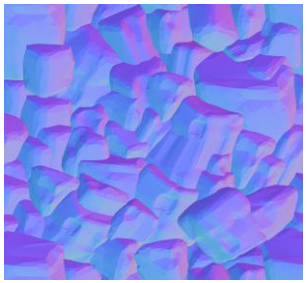
< fAlpha *=1000.0f;>

< fAlpha *=2.0f;>



● 고정 파이프라인의 범프 효과

T Factor ⊗ Dot3


Normal Map

*


Diffuse Map

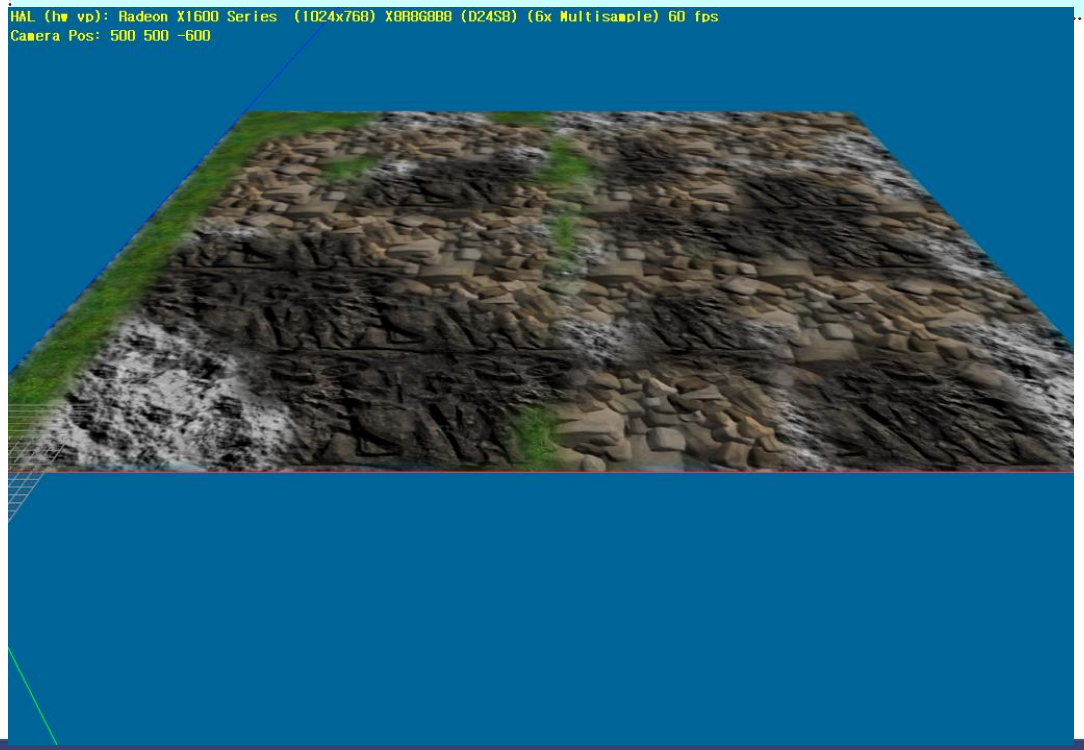
*


Alpha Map

```
// T-Factor 계산
DWORD dwR = (DWORD)(100.f * m_vcLgt.x + 155.f);
DWORD dwG = (DWORD)(100.f * m_vcLgt.y + 155.f);
DWORD dwB = (DWORD)(100.f * m_vcLgt.z + 155.f);

m_dTFt = (DWORD)(0xff000000 + (dwR << 16) + (dwG << 8) + dwB);
```

T-Factor를 통한 범프 계산

```
m_pDev->SetRenderState(D3DRS_TEXTUREFACTOR, m_dTFt);
m_pDev->SetTextureStageState(0, D3DTSS_COLORARG1, D3DTA_TEXTURE) ;
m_pDev->SetTextureStageState(0, D3DTSS_COLORARG2, D3DTA_TFACTOR);
m_pDev->SetTextureStageState(0, D3DTSS_COLOROP,  D3DTOP_DOTPRODUCT3);
...
m_pDev->SetTexture(0, m_pTex[i].pTxN);          // Normal Texture
m_pDev->SetTexture(1, m_pTex[i].pTxB);          // Diffuse Texture
m_pDev->SetTexture(2, m_pTex[i].pTxA);          // Alpha Texture
```

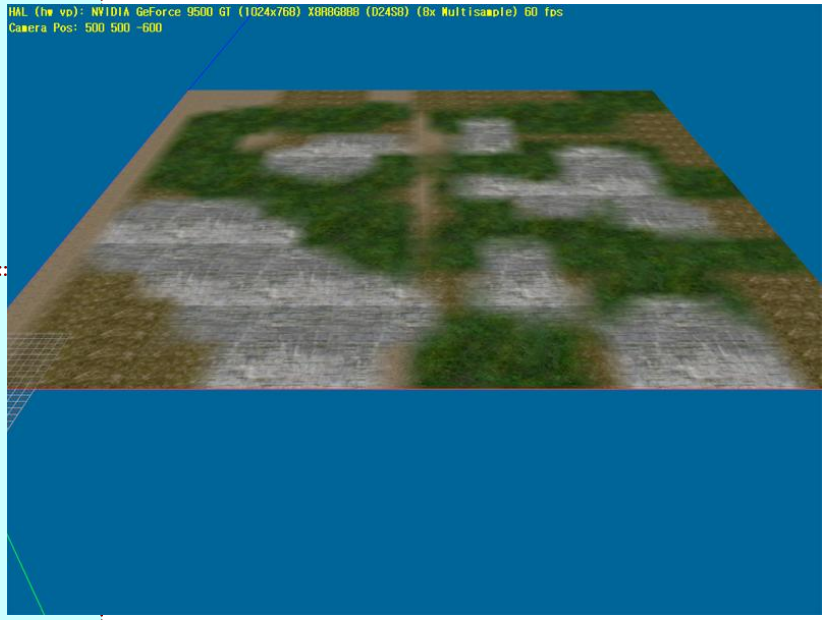


- Diffuse Splatting:
  - 정점의 Diffuse 또는 Specular 값에 Splatting Weight를 기록. 이를 분해해서 렌더링

```
void CLcSplt::CalculateMap()
{
   ...
   for(i=0; i<m_nTex; ++i)
      CalculateMapTile(i, m_pTex[i].pTwgt);
}

void CLcSplt::CalculateMapTile(int nTx, DWORD*  &xpTxA)
{
   ...
   for (z=0; z<m_iTile; ++z)
   {
      for (x=0; x<m_iTile; ++x)
      {
         fAlpha = 0.0f;
         ...
         fN = (abs(nXEnd-nXBgn) + abs(nZEnd-nZBgn)) * 2.f;

         fAlpha *=4.f;
         fAlpha /= fN;

         //Splatting Weight를 기록
         xpTxA[m_iTile*z + x] = D3DXCOLOR(1,1,1, fAlpha);
      }// for
   }// for
}
```


- 렌더링:
  - 텍스처 스플레팅의 Weight를 정점의 Diffuse 값에 설정한 후 렌더링
  - 마지막으로 정점의 Diffuse 값을 한 번 더 렌더링

```
// Splatting Weight를 저장하기 위한 구조체
struct TexWgt
{
   PDTX    pTexB;  // Texture
   DWORD*   pTwgt;  // Layer Weight constructed by Alpha
};
```




```
// 렌더링
for(i=0; i<m_nTex;++i)
{
   m_pDev->SetTexture(0, m_pTex[i].pTexB);

   for(m=0; m<m_iTile; ++m)
   {
      for(n=0; n<m_iTile; ++n)
      {
         // Splatting Weight 값을 정점 Diffuse에 복사
         m_pVtx[m*m_iTile+n].d = m_pTex[i].pTwgt[m*m_iTile+n];
      }
   }

   m_pDev->DrawIndexedPrimitiveUP(...);
}
```

```
//정점의 색상을 출력하기 위해 한번 더 렌더링
for(m=0; m<m_iTile; ++m)
   for(n=0; n<m_iTile; ++n)
      m_pVtx[m*m_iTile+n].d = 0xFFFFFFFF;
```

- Diffuse Splatting +Bump: Alpha Splatting과 동일

```
pDev->SetRenderState(D3DRS_TEXTUREFACTOR, m_dTFt);
pDev->SetTextureStageState(0, D3DTSS_COLORARG1,    D3DTA_TEXTURE);
pDev->SetTextureStageState(0, D3DTSS_COLORARG2,    D3DTA_TFACTOR);
pDev->SetTextureStageState(0, D3DTSS_COLOROP,      D3DTOP_DOTPRODUCT3);
pDev->SetTextureStageState(0, D3DTSS_ALPHAOP,      D3DTOP_DISABLE);

pDev->SetTextureStageState(1, D3DTSS_TEXCOORDINDEX,     0);
pDev->SetTextureStageState(1, D3DTSS_COLORARG1, D3DTA_CURRENT );
pDev->SetTextureStageState(1, D3DTSS_COLORARG2, D3DTA_TEXTURE );
pDev->SetTextureStageState(1, D3DTSS_COLOROP,     D3DTOP_MODULATE2X);

pDev->SetTextureStageState(1, D3DTSS_ALPHAARG1, D3DTA_DIFFUSE);
pDev->SetTextureStageState(1, D3DTSS_ALPHAOP,     D3DTOP_SELECTARG1);
```