SkyBox

스카이박스(Sky Box)란?

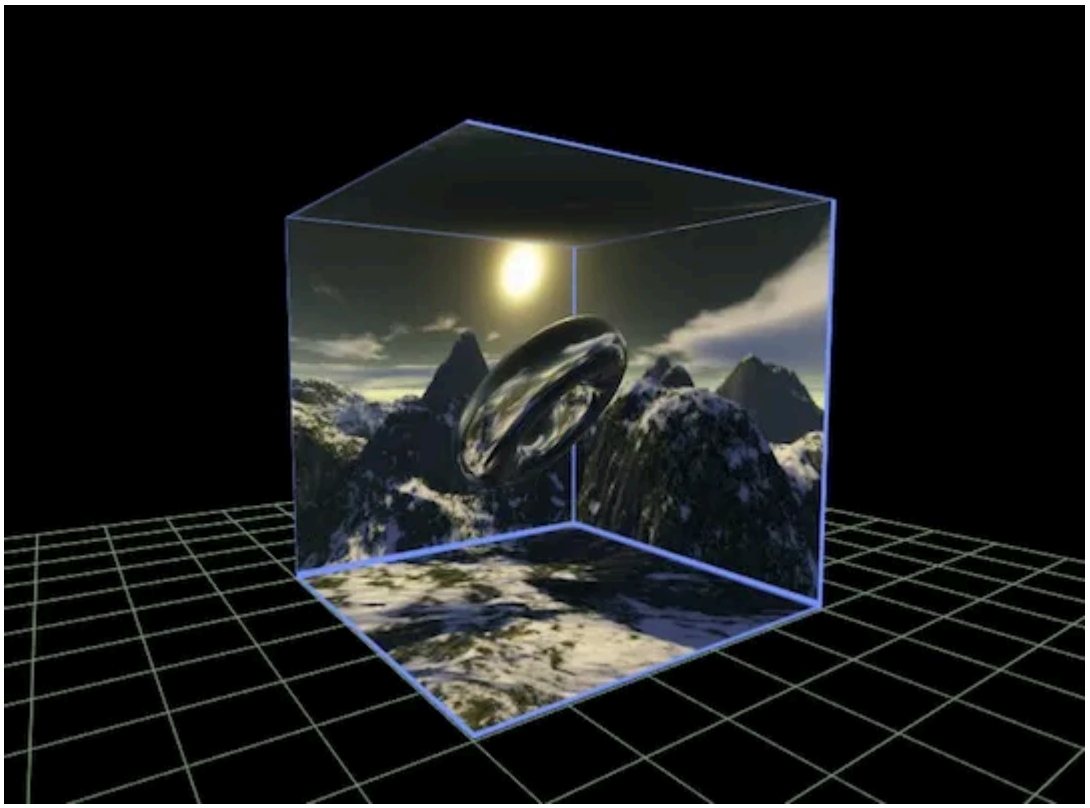
컴퓨터 그래픽스, 특히 3D 게임이나 시뮬레이션에서 3차원 장면의 배경을 표현하는 데 사용되는 기술이다.

육면체의 상자(큐브) 안에 가상 세계를 가두고, 그 상자의 각 면에 미리 렌더링된 배경 이미지를 매핑하여

원근감을 조성한다.

육면체의 상자는 사용자의 시점과 함께 항상 움직이므로, 사용자는 마치 넓은 공간에 있는 것처럼 느끼게됨.

(카메라의 위치를 중심으로 큐브도 같이 움직임)



스카이박스의 텍스처는 DDS확장자의 큐브 이미지를 텍스처로 사용한다.

그렇기 때문에 정점버퍼에서 float Tex 좌표를 전달하는게 아니라 float3 방향벡터를 전달해야 한다.

또한 정점버퍼에서 dir를 전달하지 않아도 vertexShader에서 방향벡터를 계산해서 넣을 수 있다.

VertextShader.hlsl 코드

정점 좌표를 이용하여 방향벡터를 계산한다

상수버퍼에서 카메라의 이동행렬 좌표를 제거한 값을 전달해준다

```
SkyConstant scb{};
XMStoreFloat4×4(&scb.world, XMMatrixTranspose(XMMatrixIdentity()));
XMMATRIX viewNoTrans = m_View;
viewNoTrans.r[3] = XMVectorSet(0.0f, 0.0f, 0.0f, 1.0f);  ← 이동행렬을 제거해주는 부분
XMStoreFloat4×4(&scb.view, XMMatrixTranspose(viewNoTrans));
XMStoreFloat4×4(&scb.projection, XMMatrixTranspose(m_Proj));
D3D11_MAPPED_SUBRESOURCE mapped{};
```

```
HR_T(m_pDeviceContext→Map(S_ConstantBuffer.Get(), 0,
D3D11_MAP_WRITE_DISCARD, 0, &mapped));
memcpy(mapped.pData, &scb, sizeof(scb));
m_pDeviceContext→Unmap(S_ConstantBuffer.Get(), 0);
```

그리고 이값을 버텍스 버퍼에서

float4 pW = mul(float4(vIn.pos, 1.0f), world);

float4 pV = mul(pW, view);

float4 pH = mul(pV, projection);

VerTOut.pos = pH.xyww; //xyww → float4(pH.xy, pH.w, pH.w

아래와 같이 변환하여 전달해준다

→ z값을 w값과 같이 1로 고정 시킴 (깊이 값을 항상 1.0로 고정하여, 스카이박스가 모든 다른 물체(깊이 값이 0.0에서 1.0 사이)보다 항상 가장 멀리 있는 것처럼 만든다.)

여기서 설정한 pos값을 픽셀 셰이더에서 전달받아 텍스쳐를 보여준다.

```
float4 main(VShaderOut pIn) : SV_TARGET
{ return float4(txDiffuse.Sample(samLinear, normalize(pIn.dir)).rgb, 1.0); }
```

스카이박스가 다른 물체를 가리면 안되기 때문에 더 가까이 있는 물체의 경우 (=Z버퍼에 작은 값이 있는 픽셀) Skybox를 자동으로 탈락시키기 위해 깊이 테스트를 TRUE로 해주고 깊이 값을 덮어쓰지 않게 하기 위해 WRITE 값은 OFF로 해주면 된다

```
descDepth.DepthEnable = TRUE;  //깊이 테스트 ON
descDepth.DepthWriteMask = D3D11_DEPTH_WRITE_MASK_ZERO; // 쓰기 OFF
descDepth.DepthFunc = D3D11_COMPARISON_LESS_EQUAL; // 비교 규칙(테스트 방식)
```

스카이박스는 카메라가 박스 안에 존재하기 때문에 Index도 반시계 방향으로 그리도록 해줘야한다

(*기본적으로 시계 방향으로 정점을 정의한 면을 앞면(Front-Face)**으로 간주)

```
UINT indices[] = {
// Y+
1,0,3, 1,3,2,
// Y-
```

```
4,5,6, 4,6,7,
// X-
9,8,11, 9,11,10,
// X+
12,13,14, 12,14,15,
// Z-
17,16,19, 17,19,18,
// Z+
20,21,22, 20,22,23
};
```

스카이박스의 레스터라이저 속성을 아래와 같이 설정해주면 된다

```
D3D11_RASTERIZER_DESC rd = {};
rd.FillMode = D3D11_FILL_SOLID;
rd.CullMode = D3D11_CULL_FRONT;
rd.DepthClipEnable = TRUE;
HR_T(m_pDevice->CreateRasterizerState(&rd,
S_RasterizerState.GetAddressOf()));
```

카메라는 스카이 박스 내부에 존재하기 때문에 FRONT 옵션을 사용해 외부를 제거한다.

스카이박스는 가장 마지막에 랜더링하는 것이 좋다.

1. 깊이 테스트로 배경을 깔끔하게 제한할 수 있다.
    - 앞에 있는 물체는 자동으로 스카이박스가 안그려지기 때문에 물체 실루엣 경계가 깨끗하고, Skybox가 물체를 가리는 일이 없어진다.
2. 오버드로우를 거의 없앨 수 있다.
    - 화면 전부를 칠하는게 아니라 물체들을 먼저 그리고 빈 공간에 스카이 박스를 그리면 되기 때문에

      픽셀 셰이딩 비용이 최소로 든다.
3. Z버퍼 보존(Depth Write OFF)
    - 스카이박스는 깊이 버퍼를 쓰지 않기 때문에 이미 채워진 씬 깊이를 그대로 보존해서, 반투명

      렌더/SSAO/DOF/포스트프로세스 등이 정확한 Z를 사용할 수 있다.

깊이 버퍼를 사용하는 포스트 프로세싱 효과들

- **SSAO (Screen Space Ambient Occlusion)**: 주변광을 차단하는 효과로, 깊이 버퍼를 이용해 각 픽셀 주변의 기하학적 형태를 분석하여 음영을 계산합니다.
- **DOF (Depth of Field)**: 카메라의 초점 효과를 흉내 내는 것으로, 깊이 버퍼를 이용해 카메라로부터의 거리를 판단하고, 초점 거리 밖의 물체는 흐리게 만듭니다.
- **반투명 렌더링**: 반투명 물체를 렌더링할 때, 그 뒤에 있는 불투명 물체의 깊이 정보를 알아야 정확한 투명 효과를 낼 수 있습니다.

위와 같은 효과들이 사용할 Z버퍼값이 오염되지않고 사용할 수 있다.

마지막으로 스카이박스의 랜더링이 끝나면
m_pDeviceContext→OMSetDepthStencilState(nullptr, 0);
와 같이 깊이버퍼를 기본상태로 돌려놔야 한다. 그렇지 않으면 이전 스카이박스 렌더링에 사용된 특수 상태

(깊이 쓰기 비활성화 등)를 그대로 상속받아 큐브들이 안그려지기 때문이다.