2020182031 이서연 졸업작품 7주차 진행

1. 코드 정리

평소에 쓰던 코드에서 변수들의 이름을 가독성이 있도록 직설적으로 바꾸어 주었다.

(Ex: m\_ppd3dRenderTargetBuffers -> m\_RenderTargetBuffers)

(Ex: m\_pd3dDsvDescriptorHeap -> m\_DSVDescriptorHeap)

프레임워크의 생성 부분에서 Swap Chain이 생성되기도 전에 Render Target View가 생성되게 코드가 짜여 있어서 순서를 바꾸어 주었다.

텍스트이(가) 표시된 사진

자동 생성된 설명

이것이 프레임워크의 생성 순서이다.

1. 새로운 궁금증
2. 프레임워크 생성 코드에서 풀스크린 상태일때만 Render Target View를 생성하는 이유가 뭘 까?

텍스트이(가) 표시된 사진

자동 생성된 설명

1. Deferred Lighting 구현

먼저 Deferred Lighting을 구현하기 위하여 1PASS의 GBuffer로 어떤 정보를 보내야 할지 정해야 한다.

가장 먼저 떠오르는 것은 기본적인 Position, Normal, Texture(Color), Object ID, Depth였다.

이 중 Depth는 Depth/Stencil Buffer값을 복사해 줄 것이기 때문에 Multiple Render Target의 출력 값으로 4가지를 넣어주었다.

텍스트이(가) 표시된 사진

자동 생성된 설명

현재 Object Shader가 사용하고 있는 Pixel Shader는 Vertex Shader로부터 위치, Normal을 받아와서 빛계산을 해준 후 색상을 출력해주는 Forward Lighting의 형태를 띄고 있다.

텍스트이(가) 표시된 사진

자동 생성된 설명

이것이 현재 사용중인 Pixel Shader

텍스트이(가) 표시된 사진

자동 생성된 설명

그리고 이것이 새로 사용하게 될 Pixel Shader이고 아직 제작 중에 있기 때문에 적용은 하지 않았다.

Position값은 Vertex Shader에서 받아온 World Position 그대로 넣어준다.

Normal값은 받아온 World Normal을 Normalize하여 넣어주었고 마지막 남는 a의 자리에는 공간 활용을 위하여 Object ID를 넣어주었다.

Texture는 나중에 넣기 위해 일단 아무 값으로나 채워 넣었다.

그리고 Texture Resource를 Shader 코드에 넘겨주기 위하여 루트 파라미터를 추가했다. 텍스트이(가) 표시된 사진

자동 생성된 설명

T0는 실제 육면체 Texture의 정보를 저장할 장소이다.

T1은 MRT를 저장할 Texture의 배열이다.

Texture를 추가함에 따라서 입력해주어야 하는 정보에 UV도 추가되었다.

그래서 Cube Mesh를 구성하는 Vertex에 Float2인 UV를 넣어주었고, 직접 UV값을 넣어주었다.

텍스트이(가) 표시된 사진

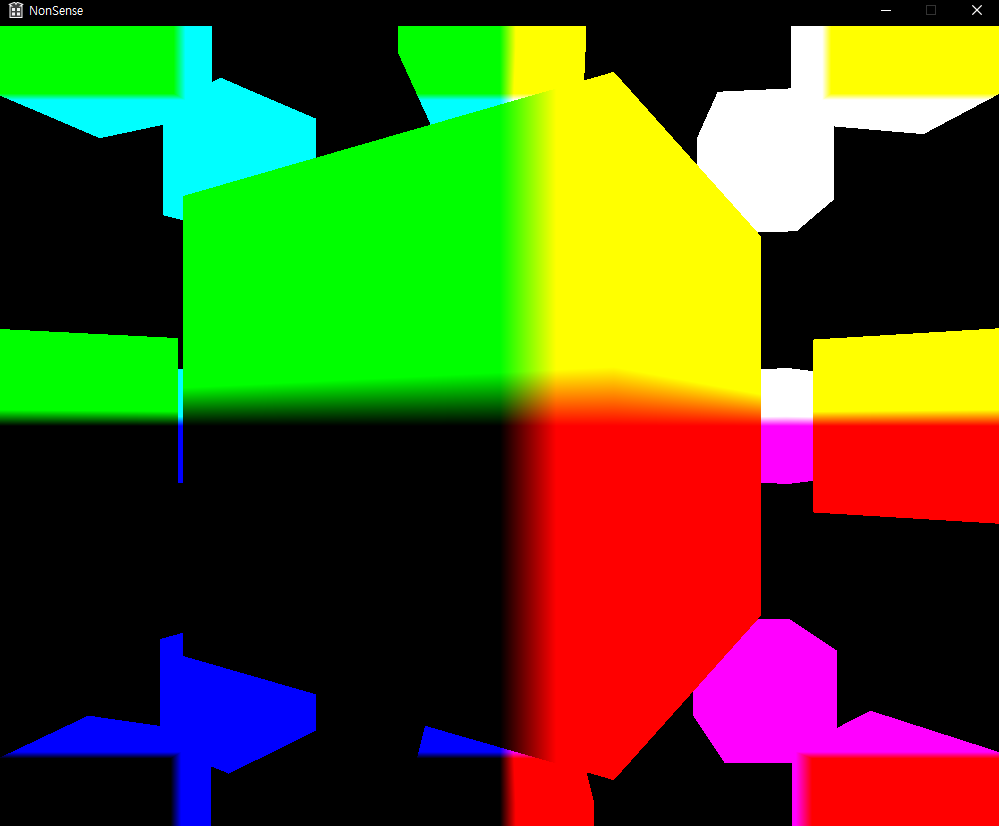
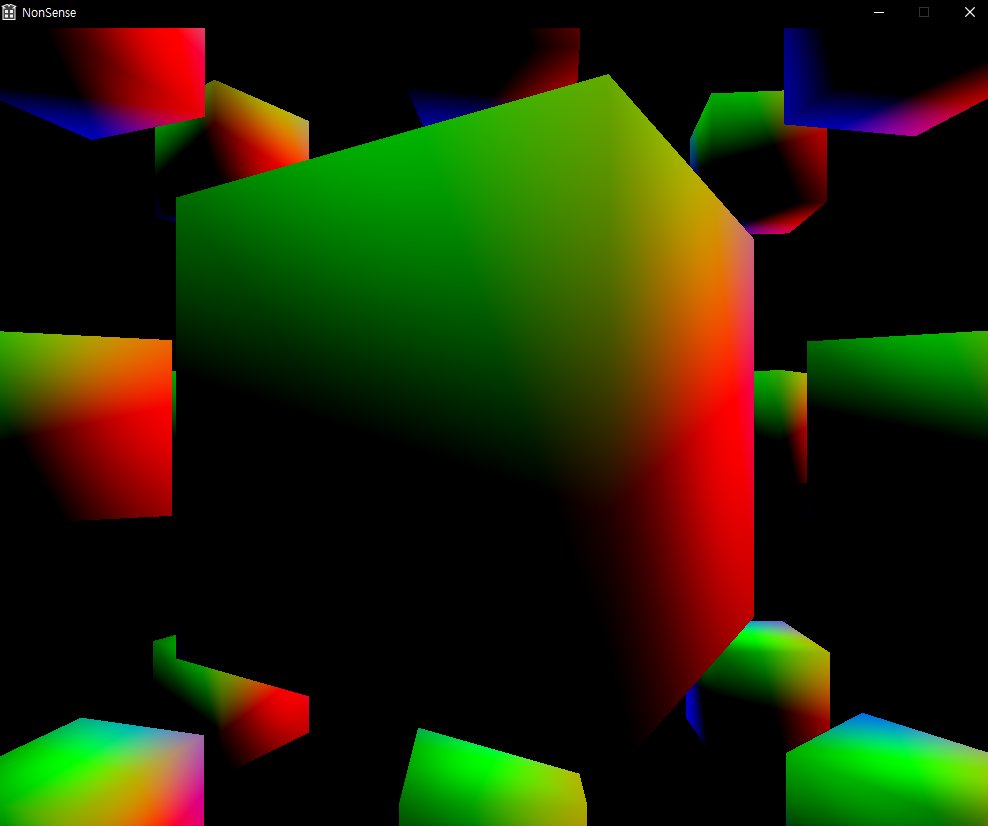
자동 생성된 설명

그리고 Object Shader의 Create Input Layout에도 UV를 추가해주었다. (3번째)

텍스트이(가) 표시된 사진

자동 생성된 설명

이제 새로 만든 Pixel Shader를 넣고 실행을 한 결과 다음과 같은 화면이 나온다.

첫번째가 Position을 MRT의 첫번째로, 두번째가 Normal을 MRT의 첫번째로 넣은 결과이다.

Normal값이 왜인진 모르겠지만 제대로 들어가지 않는다.

여기서 우린 MRT의 TARGET0이 Render된다는 것을 알 수 있고, 이것을 Render하지 않는 방법을 찾아야 한다.

이 결과를 화면에 Render하는 것이 아니라 새로 만들어줄 Texture Resource에 각각담아, 그것을 사용하는 것이 우리의 목표이기 때문이다.

또한 Texture를 적용시키기 위한 Sampler또한 정의해 주었다.

텍스트이(가) 표시된 사진

자동 생성된 설명

WRAP형식으로 적용시켜주는 Sampler이다.

그 다음은 이 MRT를 사용하여 조합을 해 화면에 그려줄 Screen Shader를 만들었다

일단 Screen Shader의 Vertex Shader는 그냥 화면 전체를 나타내는 평면 사각형을 찍어주었다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. 외곽선 구현

텍스트이(가) 표시된 사진

자동 생성된 설명

이것은 외곽선 코드이다.

gtxInputTextures[4]는 Object ID가 들어갈 자리라고 생각하면 된다.

다른 코드에서 만든 외곽선 코드라서 다른 변수가 들어가 있다.

오브젝트 별로 가장 끝 외곽선만 선을 그려주기 위해 Depth나 Normal값은 신경 쓰지 않았다.

원래는 Stencil값을 오브젝트를 그릴 때 마다 하나씩 올려서 그것으로 검사를 하려고 했는데 Object ID라는 변수를 그냥 하나 추가해 주어 Texture로 만들었다.

라플라시안 필터를 적용하여 외곽선을 검출해내는데 이 때 외곽선을 나타내는 Pixel이 외곽선을 중심으로 3x3픽셀을 외곽선으로 채우는데 이것을 몇 개의 픽셀까지 채울 것인지 변수로 넣어주어서, 외곽선의 두께를 조절할 수 있도록 하였다.

1. 막히는 부분들

Texture를 넣으려고 시도를 해보는데 잘 들어가지 않는 것 같다.

Texture를 넣을 때 그냥 정보를 불러와서 루트 파라미터를 이용하여 넣어주면 될 줄 알았는데 생각보다 설정해주어야 하는 것이 많은가 보다.

또한 위에서 본 듯이 Normal값이 제대로 들어가지 않는 버그를 수정해야한다.

분명 Mesh를 생성할 때 Position이 잘 들어가니 Normal도 잘 들어가야 할 텐데 어디서부터 꼬이게 된 건지 잘 모르겠다.

1. 3D 게임 프로그래밍1 복습

2023.2.6 - Direct3D Device (part2) – 04

2023.2.6 - Direct3D Device (part2) – 05 - Fin

