**게임개발자전문가 과정**

**1.과목명: 게임 일반 프로그래밍**

**2.능력단위: 게임 그래픽 프로그래밍**

**3.제출일자: 2022년 04월 26일**

**4.포트폴리오: 노말맵, 환경맵, 깊이 맵 쉐도우 및 조명 구현 프로젝트**

**5.작성자: 최지원**

**<제출내역>**

1. **노말맵, 환경맵, 깊이 맵 쉐도우 및 조명 구현 프로젝트**
2. **노말맵, 환경맵, 깊이 맵 쉐도우 및 조명 구현 프로젝트 분석 및 세부 문서**

**<증거자료>**

|  |
| --- |
| * **반드시 폰트 크기= 10** * **줄 2칸 이상 엔터 금지** * **페이지 여백 조정 불가**   **< 현페이지 수정 불가 및**  **증거자료 공란 유지 >** |

**<목차>**

1. **프로젝트의 소개 및 개요**
   1. 프로젝트 소개
      1. 노말 매핑에 대한 개요
      2. 쉐도우 매핑에 대한 개요
2. **프로젝트의 설계 및 세부적 기능 단위로 다이어그램 기술**
   1. UML 다이어그램
      1. 깊이 맵 쉐도우 및 조명 프로젝트의 시퀀스 다이어그램 (첨부)
      2. 깊이 맵 쉐도우 및 조명 프로젝트의 클래스 다이어그램 (첨부)
3. **프로젝트의 구현된 주요 기술 단위로 분석 및 설계 기술**
   1. 구현된 노말 매핑
      1. 주요 함수 설계
   2. 구현된 깊이 맵 쉐도우
      1. 주요 함수 설계
4. **최종 결과 및 추가된 내용**
   1. 결과물 화면 (첨부)
   2. 범용성/ 유연성/ 확장성/ 간결성 고려하여 추가된 내용
5. **프로젝트의 소개 및 개요**
6. **프로젝트 소개**

- Visual Studio 2019 환경에서 DirectX11 SDK를 활용한 게임 엔진 라이브러리를 제작한다.

- 디퓨즈 매핑, 스페큘러 매핑, 노말 매핑, 환경 매핑을 적용한다.

- 조명을 구현하고, 깊이값을 저장한 깊이맵으로 그림자를 구현한다.

* + 1. **노말 매핑에 대한 개요**

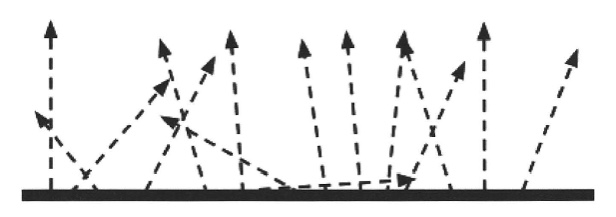
노말 맵은 픽셀에 사용할 법선 정보를 담고 있다.

그림 2 평평하지만 노말맵을 적용한 면

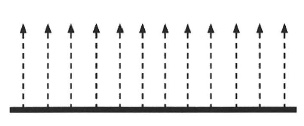


그림 1 평평한 면의 법선

노말 매핑을 사용하면 버텍스의 노말(법선) 정보가 아닌 텍스쳐의 노말(법선)정보를 사용하므로, 텍스쳐의 공간을 사용해야 한다. 그것을 접선 공간이라고 한다.

접선 공간은 텍스쳐 위의 좌표계라고도 한다. 법선벡터(Normal), 접선벡터(Tangent), 바이노멀(Binormal) 벡터 축으로 하는 공간이다. 이러한 벡터들로 한 픽셀이 어떤 기울기를 갖고 있는지 알 수 있다.

**법선(Normal) : 면에서 수직인 벡터**

**접선(Tangent) : 정점 표면의 정보**

**종법선(Bi-Normal) : 접선과 법선의 외적 된 결과, 노말의 수직인 벡터**

변환을 하기 위해서는 TBN(tangent, binormal, normal) 행렬을 만들어야 한다.

행 기준 행렬( Row major matrix ) - d3d

| Tx Ty Tz |

| Bx By Bz |

| Nx Ny Nz |

* + 1. **쉐도우 매핑에 대한 개요**

텍스트이(가) 표시된 사진

자동 생성된 설명그림자란 광원의 경로 상에서 불투명한 물체가 있을 때, 빛의 직진성 때문에, 물체에 빛이 통과하지 못하여 생기는 어두운 부분을 말한다.

그림 1 출처 : https://learnopengl.com/Advanced-Lighting/Shadows/Shadow-Mapping

왼쪽 사진에서는 그림자가 없을 때의 모습인데, 3D 공간에서 그림자가 없으면 물체가 떠있는지 바닥에 있는지 구별하기가 어렵다. 최소한 3D 공간라면, 원형 그림자이라도 깔아 놔야 한다.

게임에서는 그림자를 흉내내기 위한 기법이 여러 가지 있지만, 대표적으로 평면 투영 쉐도우, 투영 쉐도우, 깊이 맵 쉐도우가 있다.

|  |  |
| --- | --- |
| 구분 | 설명 |
| 평면 투영 쉐도우 | 그림자를 만들 오브젝트를 빛 방향에서 바닥으로 납작하게 눌러서 바닥에 그려주는 방식이다. 평면 투영 행렬을 구해서 곱한다. 단점으로는 평면이 아니면 그림자가 드리워질 수 없다. |
| 투영 텍스쳐 쉐도우 | 울퉁불퉁한 지형에 그림자를 그리기 위해 투영 텍스처 쉐도우를 사용한다.  광원 위치에서 바라본 장면, 빈 텍스처에 그 장면을 텍스처로 저장한다.  텍스처를 생성해, 빈 텍스처에 렌더 타깃 RenderTargetView()을 그려서 리소스를 바닥 텍스처에 붙이기 위해 전달해야 한다. |
| 깊이 맵 쉐도우 | 전방향 쉐도우(Omnidirectional Shadow Maps), 계단식 쉐도우(Cascaded Shadow)등  기초가 되는 그림자 기법이며 자기 그림자인, Self-Shadow가 가능하다. |

깊이 맵 쉐도우를 적용하기 위한 단계는 다음과 같다.

1. 광원 위치에서 바라보는 뷰 및 투영 행렬 연산

2. 깊이 맵 저장할 텍스쳐 생성

3. 깊이 맵 텍스쳐에 깊이 값 렌더링 (랜더타겟으로 광원(카메라)에서 바라본 깊이 값 텍스쳐에 저장)

4. 오브젝트 렌더링 시 깊이 값 이용해 그림자 판정

깊이 맵에서 멀리 있으면, 흰색, 가까울수록 검은색으로 렌더링하여, 텍스쳐의 배경은 흰색으로 구현했다.

어두운이(가) 표시된 사진

자동 생성된 설명

그림 2 깊이 맵 텍스쳐

깊이 맵을 만들 때, 고려해야 할 것이 깊이 바이어스(편향치)이다. 저장 시에 약간의 실수 값의 오차가 발생되어서 나타난다.

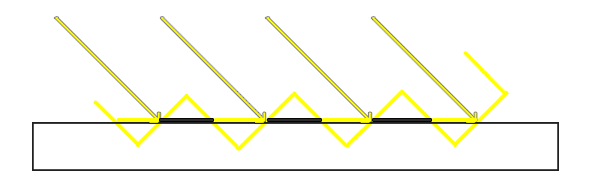


그림 3 출처 : https://learnopengl.com/Advanced-Lighting/Shadows/Shadow-Mapping

광원의 각도로 텍스처를 저장해서 기울어진 깊이 텍스처가 되었기 때문이다.

이 문제를 깊이 바이어스, 깊이 값을 오프셋 함으로서 문제를 해결할 수 있다.

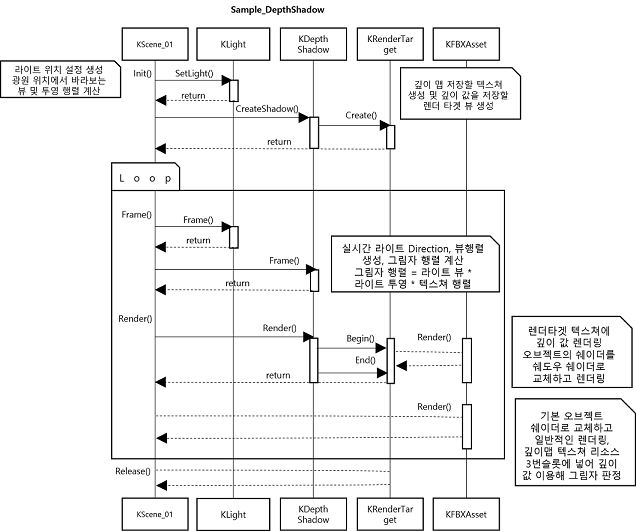
1. **프로젝트의 설계 및 세부적 기능 단위로 다이어그램 기술**
   1. UML 다이어그램
      1.  **깊이 맵 쉐도우 및 조명 프로젝트의 시퀀스 다이어그램**

그림 4 깊이 맵 쉐도우 및 조명 프로젝트의 시퀀스 다이어그램

|  |  |
| --- | --- |
| **깊이 맵 쉐도우 및 조명 프로젝트의 시퀀스 다이어그램** | |
| 1 | 라이트 위치 설정, 광원 위치에서 바라보는 뷰 및 투영 행렬 계산 |
| 2 | 깊이 맵을 저장할 텍스처 생성 및 깊이 값을 저장할 렌더 타겟 뷰 생성 |
| 3 | 실시간 라이트 방향, 뷰 행렬 생성, 그림자 행렬 계산 |
| 4 | 렌더타겟 텍스처에 깊이 값 렌더링 오브젝트의 쉐이더를 쉐도우 쉐이더로 교체 후 렌더링 |
| 5 | 기본 오브젝트 쉐이더로 교체 후 일반적인 렌더링, 깊이 맵의 깊이 값 이용해 그림자 판정 |

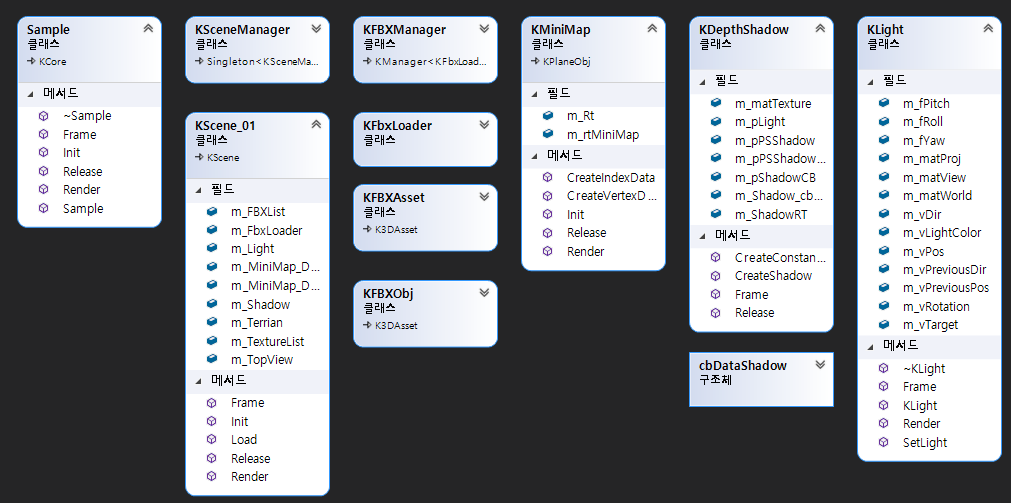
* + 1.  **깊이 맵 쉐도우 및 조명 프로젝트의 클래스 다이어그램**

그림 5 깊이 맵 쉐도우 및 조명 프로젝트의 클래스 다이어그램

그림 7 동적 정적 지형 LOD 클래스 다이어그램

|  |  |
| --- | --- |
| 구현된 기능의 주요 클래스, 구조체 설명 | |
| Sample클래스 | 윈도우, DirectX11 COM객체 관리, 엔진에 필요한 기능 객체 관리 클래스 KCORE를 상속받는 클래스 |
| KSceneManager 클래스 | 여러 씬을 관리하는 클래스 |
| KScene\_01 클래스 | 해당 씬 클래스 |
| KMiniMap 클래스 | 렌더타겟 텍스쳐 리소스를 받아 미니맵 형태로 출력하는 클래스 |
| KDepthShadow 클래스 | 렌더타겟 객체를 갖고 있는 그림자 객체 클래스, |
| KLight 클래스 | 라이트 객체 클래스 |
| KFBXAsset 클래스 | KFBXLoader로 불러온 정보를 가진 FBX 파일 객체 클래스 |

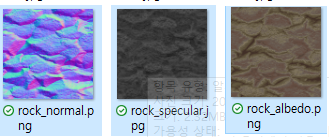
1. **프로젝트의 구현된 주요 기술 단위로 분석 및 설계 기술**
   1. 구현된 노말 매핑
      1. 주요 함수 설계

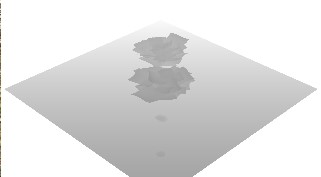
|  |  |  |
| --- | --- | --- |
| **클래스**  **/구조체** | **함수** | **내용** |
| K3DAsset | CreateTangentSpace() | 노말맵 적용을 위한 접선 ,법선, 종법선 계산 함수  어떤 면이든 3점을 얻어서 법선, 종법선, 접선을 구하는 함수이다.  3점의 벡터를 얻어서 가장자리의 벡터를 구한다.  UV값도 마찬가지로 가장자리의 벡터를 구해서  접선과 종법선은 탄젠트 공간에서 수직 관계인 점을 이용해서 구하고, 접선과 종법선을 구하면 외적의 결과로 법선 값을 얻는다. |
| VSPS\_Default.hlsl | PS(VS\_OUTPUT Input) : SV\_TARGET | 버텍스 쉐이더에서 월드 행렬을 곱한 법선, 접선, 종법선을 받아온다.    노말맵을 저장할 때, 범위는 -1~1으로 0~1 범위로 바꿔줘야한다.  텍스처의 각 채널에서 가질 수 있는 값의 범위는 0~1이기 떄문이다.   |  |  | | --- | --- | | 법선벡터->법선맵 | 법선맵->법선벡터 | | 법선맵 RGB = 법선벡터 XYZ \* 0.5 + 0.5 | 법선벡터 XYZ = 법선맵 RGB \* 2 - 1 |     접선 공간 행렬 : Z는 정점의 법선(Normal),  X는 UV중 하나의 값 접선(Tangent), Y는 두 벡터의 외적 결과인 종법선(Binormal) 이렇게 접선 공간 변환에 사용하는 행렬을 만들 수 있다. |

* 1. 구현된 깊이 맵 쉐도우
     1. 주요 함수 설계

|  |  |  |
| --- | --- | --- |
| **클래스**  **/구조체** | **함수** | **내용** |
| KLight | SetLight() | 라이트 생성 함수로 위치 벡터와 타겟 벡터를 매개변수로 받아  월드 행렬, 뷰행렬, 원근 투영 행렬을 생성한다. |
| Frame() | 매 프레임 마다 타겟 벡터와 위치 벡터를 뺀 결과를 정규화해 방향 벡터를 만든다.  라이트의 이동에 따라 뷰행렬을 실시간으로 계산한다. |
| KDepthShadow | CreateShadow() | 라이트 객체를 매개변수로 받아 깊이 값을 저장할 렌더타겟을 생성하고, 그림자 행렬을 생성,각 오브젝트 객체의 쉐도우 쉐이더를 생성한다. |
| Frame() | 그림자 행렬을 실시간 계산한다.  그림자 행렬 = 라이트 뷰행렬 \* 라이트 투영행렬 \* 텍스쳐 행렬 |
| VSPS\_Default.hlsl | VS(VS\_INPUT Input) | 오브젝트 버텍스 쉐이더로 깊이 값을 계산한다.  버텍스 컬러 값에 저장한다. |
| PSDepth(VS\_OUTPUT Input) : SV\_TARGET | 쉐도우 픽셀 쉐이더 내용이다. 깊이값을 컬러로써 리턴한다.  이 값으로 렌더링하면 깊이 맵을 만들 수 있다. |
| PS(VS\_OUTPUT Input) : SV\_TARGET | 그림자 행렬로 쉐도우 투영 행렬을 계산해, 깊이 맵 텍스처를 오브젝트에 그린다. 현재 픽셀 그림자가 깊이보다 작다면 그림자를 그린다. 0.006f 그림자 바이어스 값으로 그림자 여드름 문제를 해결한다. |
| KRenderTarget | Create() | 너비 높이를 매개변수로 받아 렌더타겟을 생성한다.  뷰 포트 설정 투영 행렬 생성, 렌더타겟 생성, 뎁스 스텐실뷰 생성 |
| CreateRenderTargetView() | 렌더타겟을 뿌리기 위한, 빈 텍스처를 생성한다. 빈 텍스처를 가지고 렌더타겟을 생성한다. |
| Begin() | 이전 뷰포트를 저장한다. 화면을 초기화하고  RSSetViewports()함수로 현재 뷰포트를 설정한다. |
| End() | 렌더타겟을 초기화하고 저장해뒀던  뷰포트를 다시 복귀 시켜준다. |
| KMiniMap | Render() | 미니맵은 평면 오브젝트 객체인 KPlane 클래스를 상속받은 클래스이다. 렌더타겟의 텍스쳐를 평면 픽셀 쉐이더로 텍스쳐를 넘겨서 렌더링한다. |

1. **최종 결과물 및 추가된 내용**
   1. 결과물 화면 (첨부)

* 환경맵, 디퓨즈맵, 스페큘러맵, 노말맵이 적용된 오브젝트 
* 텍스트, 표지판, 거석이(가) 표시된 사진

  자동 생성된 설명라이트의 방향에 따라 바뀌는 그림자, 깊이맵 텍스쳐
* 렌더타겟으로 저장된 깊이맵 텍스쳐

1. **범용성/ 유연성/ 확장성/ 간결성 고려하여 추가된 내용**

- 동적으로 라이트 위치를 움직이기 위해 사용자 인터페이스 라이브러리 IMGUI 적용

- 디퓨즈맵, 스페큘러맵, 노말맵 환경맵 적용

- FBX 파일 로드 시, 연결된 매터리얼 텍스처 정보 파일 입출력으로 로드 기능 구현

-라이브러리를 구축해서 다른 속성의 응용프로그램에서도 쉽게 이식할 수 있게 제작함

-전방향 쉐도우, 계단식 쉐도우 기초가 되는 그림자 기법 깊이 맵 쉐도우 구현, 깊이 바이어스로 그림자 여드름 문제 개선