3D 게임 프로그래밍 과제-2

게임공학과 2018180035 장재문

1. **과제2 목표**

* 이전에 만들었던 과제1을 DirectX12를 통해 렌더링 하는 것과 과제1을 3D 모델 파싱을 통해 구현하는 것이 목표이다.
* 이번 과제를 하면서 저는 DirectX12 의 그래픽 장치들이 동작하는 과정과 구동되는 큰 흐름을 보는 것에 집중했습니다. 그래서 시간이 오래 걸리더라도 교수님께서 짜주신 LabProject 코드를 최대한 뜯어서 재배치 하는 것에 목표를 두었습니다. DirectX12 장치 초기화와 이를 통한 렌더링 과정을 클래스를 통해 나누어서 구동하게 만들었습니다.

1. **실행결과 / 조작법**

* 실행결과

만화 영화이(가) 표시된 사진

자동 생성된 설명

* 조작법

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ( Jump 키는 구현하지 못했습니다. )  텍스트, 스크린샷, 폰트이(가) 표시된 사진  자동 생성된 설명 | |  |  | | --- | --- | | W | 현재 카메라가 바라보는 방향이 나아가는 방향이 되어 앞으로 이동 | | A | 현재 카메라가 바라보는 방향기준 좌측 방향으로 이동 | | S | 현재 카메라가 바라보는 방향기준  뒤 방향으로 이동 | | D | 현재 카메라가 바라보는 방향기준  우측 방향으로 이동 | | Q | 포를 시계반대방향으로 회전 - Yaw | | E | 포를 시계방향으로 회전 - Yaw | | R | 포를 위 방향으로 회전 - pitch | | F | 포를 아래 방향으로 회전 - pitch | | RBTN | 카메라를 회전한다 ( PRESSED )  ( 플레이어 기준 3인칭 시점 ) | | LBTN | 플레이어가 포탄을 발포한다. | |

* 게임 실행
* Wasd 키로 움직입니다. 현재 카메라가 바라보는 방향이 Forward 방향이 됩니다.

마우스 우측버튼을 누르며 화면을 움직일 수 있습니다(상하좌우). 마우스 좌측 버튼을 누르면 포탄을 발사합니다.

* **포탄 발사**

하늘, 무기, 탱크, 전투차량이(가) 표시된 사진

자동 생성된 설명

포탄은 탱크의 포의 끝 위치에서 발사하도록 구현했습니다.

Body – Turret(자식) – Gun(자식) 으로 탱크가 구성이 되어있습니다. 이동행렬이 부모의 영향을 받도록 설정해 놨으므로 포 오브젝트위치에서 포가 바라보는 Forward 방향으로 생성 시작지점을 이동시켰습니다. 또한 포 오브젝트와 회전행렬을 동일하도록 구현하였습니다. 이 탱크는 회전행렬들은 부모의 영향이 없게 독립적으로 회전행렬을 적용할 수 있게 구현했습니다.

* **몬스터의 발포**

하늘, 배, 보트, 만화 영화이(가) 표시된 사진

자동 생성된 설명

몬스터는 Player 를 향해서 포를 움직여 포탄을 발포합니다. Player 의 위치값을 받아서 몬스터의 위치와 뺸 후 Normalize해 플레이어를 향하는 방향벡터를 구했습니다. 포탄은 플레이어를 바라보는 방향이 Forward 가 되도록 회전을 시켜 발포하게 구현하여 모델이 플레이어에 향하게 쏘는 것을 구현했습니다.

* **게임 종료 조건**

적 몬스터가 플레이어의 포탄과 30번 충돌하게 되면 게임이 종료됩니다.

만화 영화, 군용 차량이(가) 표시된 사진

자동 생성된 설명

1. **변경한 구조 설명**
2. **게임 오브젝트**

* **텍스트, 스크린샷, 폰트, 직사각형이(가) 표시된 사진

  자동 생성된 설명**

게임 오브젝트 입니다. 게임 오브젝트는 Mesh Material 을 갖고 있고 Component 와 Script 를 통해 업데이트 됩니다. Component 는 Transform 과 Camera 를 갖고 있습니다.

1. **Dx12 그래픽 장치**

|  |  |
| --- | --- |
|  | * Dx12 그래픽 장치 클래스화   Dx12 그래픽 장치들을 클래스로 나누었습니다. 각 그래픽 장치 클래스들을 싱글톤으로 구현하여 다른 장치 클래스 내부에서 해당 장치를 사용해야 할때는 싱글톤으로 가져왔습니다. 또한 내부 DX12D 객체들은 스마트 포인터를 활용해서 Release에 유의하여 구현했습니다. 해당 장치들은 게임 프레임워크에서 초기화 하고 렌더링 합니다. |

1. 리소스 관리

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

|  |  |
| --- | --- |
|  | 리소스들은 전부 ResourceManager 에서 관리하도록 했습니다. Mesh , Material, Description 정보들, Shader 클래스, 파피프라인들을 모두 map 으로 저장해두었습니다. map 으로 저장해둔 이유는 검색에 용이했기 때문에 썼습니다.  리소스 매니저가 저장해둔 쉐이더와 DESC 를 이용해서 파이프라인을 조립하는 구조로 만들었기 때문에 검색에 용이한 자료구조를 사용했습니다.  게임 오브젝트를 생성할 때도 이 리소스 매니저가 만들어 놓은 Mesh 와 Material을 사용했습니다. 이런 구조로 만든 이유는 bin 폴더에서 파싱한 mesh 와 material 을 저장해놓고 게임오브젝트 생성시 원하는 대로 조립하기 위함이었습니다.  텍스트, 폰트, 스크린샷, 그래픽이(가) 표시된 사진  자동 생성된 설명      파싱후 리소스 매니저에 저장해 놓은 Mesh 와 Material 을 가져와서 게임오브젝트를 만들 때 썼습니다. |

1. **마지막 : 과제를 하면서 아쉬운 점**

일단 맵 위에 오브젝트를 띄워보지 못한 점이 아쉬웠습니다. 하이트맵을 배운 상태이고 씬을 로드하는 방법을 배웠지만 실행해 보지 못했습니다. 저번 시간에 하이트맵 이미지가 없을 시 어떻게 구현할 것인가 하는 것에 해당 지점을 카메라로 찍어서 깊이버퍼에 저장한 후 이 데이터를 사용한다고 발표했었는데 교수님께서 사용해도 좋다고 하셔서 이번 과제에 사용해보고 싶었습니다. 하지만 이 기술을 사용해 보지 못했습니다. 그 점이 아쉽습니다. 또한 교수님의 코드를 다시 짜면서 시간이 너무 많이 소요된 점이 아쉽습니다. 좀 더 시간을 많이 투자하고 기본기 있는 공부를 많이 했더라면 코드 재배치도 수월히 했겠지만 그러지 못했습니다.