3D 게임 프로그래밍 과제-3

게임공학과 2018180035 장재문

1. **과제3 목표**

* 이전 과제2에서 만들었던 Framework 를 통해 새로운 게임을 만듭니다. (탱크,슈팅,지형)
* 이전 만들고 있던 프레임워크를 보완하고 Heightmap 을 적용해 맵 위에 게임 오브젝트를 띄웁니다. 탱크와 헬리콥터 등 유니티에서 추출한 모델 파일을 이용해 슈팅게임을 만듭니다.

1. **실행결과 / 조작법**

* 실행결과

야외, 하늘, 헬리콥터, 교통이(가) 표시된 사진

자동 생성된 설명

Heightmap.raw 를 통해 추출한 지형위에 탱크들이 랜덤하게 움직이고 있습니다. 플레이어는 헬리콥터 이며 마우스 우측 버튼을 누르면서 화면을 회전하고 WASD로 움직입니다.

이때 카메라가 바라보는 방향이 전진 방향입니다. 마우스 좌측 버튼을 클릭하면 미사일을 발사합니다. 네모난 육면체로 조준합니다. 플레이어 헬리콥터는 하늘을 날 수 있으며 지형과 닿으면 지형을 타고 이동합니다. 지형 아래로 떨어지지 않습니다. 탱크 몬스터들은 일정범위에 있는 플레이어를 향해 포탄을 발포합니다. 플레이어는 지형위에서 움직이는 모든 탱크들을 쏘아 맞춰 없애고 이후 게임이 종료됩니다.

* 조작법

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | |  |  | | --- | --- | | 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 | | RBTN  (마우스 우측) | 카메라를 회전한다 ( PRESSED )  ( 플레이어 기준 3인칭 시점 ) | | LBTN  (마우스 좌측 ) | 플레이어가 포탄을 발포한다. | |

* 플레이어 이동

|  |  |
| --- | --- |
|  |  |

**카메라를 위쪽으로 바라보면 플레이어도 위를 향합니다. 아래로 하강하고 싶다면 아래를 바라보며 이동합니다. 플레이어는 지형위에 닿으면 지형아래로 내려가지 않고 지형 높이를 타면서 이동할 수 있습니다. 다시 위로 날고 싶으면 위를 향해 이동하면 됩니다.**

**항공기, 교통, 헬리콥터, 헬리콥터 로터이(가) 표시된 사진

자동 생성된 설명**

* **포탄 발사**

하늘, 야외, 산, 눈이(가) 표시된 사진

자동 생성된 설명

마우스 좌측버튼을 클릭하면 미사일 포탄이 날라갑니다. 앞에 보이는 네모난 육면체는 조준 점입니다. 조준점을 이용해 몬스터를 향해 미사일을 날릴 수 있습니다.

* **몬스터의 발포**

하늘, 아동 미술, 스크린샷, 야외이(가) 표시된 사진

자동 생성된 설명

몬스터는 Player 를 향해서 포를 움직여 포탄을 발포합니다. Player 의 위치 값을 받아서 몬스터의 위치와 뺀 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**

지형위에서 움직이는 모든 탱크를 맞춰 없애면 게임이 종료됩니다.

야외이(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**

|  |  |
| --- | --- |
| 텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진  자동 생성된 설명 | * Dx12 그래픽 장치 클래스화   Dx12 장치들을 클래스로 구분해서 만들었습니다. 이 클래스들은 CFramework 에서 shared\_ptr 로 만들어졌고 이 안에 실직적인 ID3D12 장치들은 Comptr로 구현했습니다. 엔진을 만들 때 프레임워크에 만들어져있는 장치들을 가져와 사용했습니다.  텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진  자동 생성된 설명 |

1. **리소스 관리**

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

|  |  |
| --- | --- |
| 텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진  자동 생성된 설명 | 리소스들은 전부 ResourceManager 에서 관리하도록 했습니다. Mesh , Material, Description 정보들, Shader 클래스, 파이프라인들을 모두 map 으로 저장해 두었습니다. 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 파싱 한 mesh 와 material 을 저장해 놓고 게임 오브젝트 생성시 원하는 대로 조립하기 위함 이었습니다.  텍스트, 폰트, 스크린샷, 그래픽이(가) 표시된 사진  자동 생성된 설명      파싱 후 리소스 매니저에 저장해 놓은 Mesh 와 Material 을 가져와서 게임 오브젝트를 만들 때 썼습니다.  File Manager 는 파일을 통해 리소스를 로드 하는 모든 함수를 모이 놓은 매니저 입니다. 파일 매니저로 로드 하면서 읽어 들인 Mesh Material 등의 정보는 리소스 매니저에 저장해 관리합니다. |

1. **마지막 : 과제를 하면서 아쉬운 점**

리소스 매니저로 직접 작성한 메쉬를 만들었는데 이때 커맨드 리스트가 closed 된 상태에서 만들었기에 에러가 났었다. 함수 호출 순서만 나중에 부르면 되는 문제였는데 삽질을 하루 정도 오래했다. 이때 이 문제를 알게 된 건 그래픽 디버깅을 해서였다. 이전에는 CPU 디버깅만 했는데 그래픽 디버깅을 통해 Command List 가 Closed 됐다는 에러 문장을 보고 나서야 알았다. 앞으로 이 그래픽 디버깅을 좀 더 활용했으면 좋겠다는 생각을 했다. 또한 엔진을 만들면서 delete 에 소홀했기 때문에 메모리 릭을 모두 없애는데 시간을 많이 썼다. 앞으로는 delete 를 잘 해야겠다는 생각을 했다. 지금까지 만든 이 엔진을 더 보완하고 기본기를 더 공부해서 2학기때 발전된 엔진을 만들고 싶다는 생각을 했다.