3D 게임프로그래밍1(2025) 과제1

2022180024 유영빈

**목표**

게임 화면(시작 화면 → 메뉴 화면 → Level-1 → Level-2)을 구현하여, 3D 게임의 씬 전환, 3D 오브젝트와 충돌 처리, 마우스 및 키보드 입력 처리 등을 이해하며, 그래픽스 프로그래밍의 전체적인 틀을 이해하는데 목표를 한다.

폰트, 텍스트, 그래픽, 로고이(가) 표시된 사진

자동 생성된 설명**3D 객체 회전 및 마우스 클릭 이벤트 처리**: 시작 화면에서 "3D 게임프로그래밍 1"과 이름이 3D로 회전하며, 이름을 클릭하면 폭발 효과와 함께 메뉴 화면으로 전환되는 기능을 구현한다.

**메뉴 UI 및 씬 전환 구현**: 메뉴 화면에서 메뉴 항목("Tutorial", "Level-1", "Level-2", "Start", "End")을 구성하고, 사용자가 "Start"를 선택했을 때 "Level-1", "Level-1"를 선택했을 때 "Level-1" 각 맞는 장면으로 이동하도록 설계한다.

**레벨별 게임 플레이**:

* **Level-1**: 롤러코스터를 3D로 구현하며, 목표 도달 혹은 키 입력('n')으로 Level-2로 진입.
* **Level-2**: 3인칭 시점 탱크 게임을 구성하며, 적 탱크와의 전투, 보호막 토글, 공격 등을 통해 기본적인 게임 플레이를 구성한다.

**3D 렌더링 및 충돌 효과 구현**: 모든 객체는 3D로 모델링되고 렌더링 ( 승리 텍스트를 제외하고 모든 객체 3D 모델링 하였습니다.) 되어야 하며, 포탄 충돌 시 적 탱크의 폭발 이펙트를 포함하여 시각적 효과를 강화한다.

**게임 입력 시스템 구현**: WASD, 마우스 좌/우 클릭, ‘a’, ‘s’, ‘n’, ‘w’, ‘Esc’ 등 다양한 입력을 인식하고 이에 따라 게임 로직이 변화되도록 한다.

**가정**

**렌더링 :**

**Direct3D 12와 같은 고급 GPU 기반 API가 아닌**, **CPU 기반 사용자 정의 3D 렌더링 엔진**을 사용하여 구현되며, **DirectXMath를 이용한 행렬 및 벡터 수학 계산**을 기반으로 3D 변환 및 투영을 처리함.

**카메라 :**

카메라는 시작 화면과 메뉴 화면은 고정된 카메라를 사용하며, Level1, Level2 씬에는 플레이어의 카메라를 사용한다.

**충돌 :**

포탄과 적 탱크 간 충돌은 상자 교차로 판단함.

**적 탱크 :**

AI 행동은 구현하지 않고, 간단한 로직을 사용하여 움직이는 루프를 사용한다.

**실행 결과와 조작법**

1. **시작 화면**

텍스트, 폰트, 스크린샷, 그래픽이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

**기능**: "3D 게임프로그래밍 1", 사용자 이름이 화면 중앙에 3D로 회전하며 표시됨.

**조작법**: 마우스로 사용자 이름(메시) 클릭 → 폭발 이펙트와 함께 **메뉴 화면으로 전환**

**구현 내용**

OBJMeshLoader.cpp – LoadOBJToMesh()함수 - .obj 파일의 정점을 사용하여 정점 위치 값만 가져와 Mesh 생성

파일 로딩 – Assets 파일 경로에 .obj 파일을 열고 텍스트를 한 줄씩 읽음

정점 파싱 – v로 시작하는 정점 데이터를 읽고 vertices변수에 저장

면 파싱 – f로 시작하는 줄은 하나의 면으로, 각 정점의 인덱스로 face변수에 저장 ( obj export 시, 3각형 면으로 내보냈습니다.)

Face에 인덱스를 사용하여 vertices[]에서 가져와 CVertex 생성 -> CPolygon에 정점 생성

모든 면을 반복해서 CMesh에 저장

최소 크기의 정점과 최대 크기의 정점을 기준으로 바운딩 박스 생성 후, pMesh의 m\_xmOOBB에 저장

마우스 피킹 – PickObjectPointedByCursor()함수를 사용하고, 마우스 버튼 클릭 감지 후, 이름 메시와 픽킹 후 폭발 처리

이를 위해서 3D 게임 프로그래밍과 이름 메시를 obj (3D게임), obj2(이름) 변수에 각각 저장

1. **메뉴 화면**

텍스트, 폰트, 스크린샷, 화이트이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

**기능**: 메뉴 항목 각 메뉴 항목에 따른 모델 형태로 표시됨.

**조작법**:

* 키보드 방향키를 사용하여 원하는 메뉴로 이동
* 마우스로 "Start" 좌 클릭 : **Level-1** 씬으로 전환
* Tutorial : - (Scene.cpp로 이동) 부가 구현 X,
* Level-1 : **Level-1** 씬으로 이동
* Level-2 : **Level-2** 씬으로 이동
* End : 프로그램 종료
* Esc키 : 프로그램 종료

**좌/우측 방향키를 사용하여 (예를들어)Start가 화면 가운데로 오게 이동 후, 화면 클릭 시 해당 장면으로 이동합니다.**

1. **Level-1 (롤러코스터 씬)**

**도표, 라인, 지도이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.**

**기능**: 3D 롤러코스터를 따라 카메라가 이동.

**조작법**:

* n 키: **Level-2**로 즉시 이동
* Esc 키: 메뉴 화면으로 되돌아감

**구현 내용**

1) BuildObjects()

* 씬 초기화 시, 114개의 레일 생성
* 각 큐브(레일)은 직접 지정한 위치를 사용하여 직선, 경사, 낙하 경로로 생성 (곡선 구현 x)
* m\_vecRailPositions배열에 해당 레일들의 위치 값을 저장 (롤러코스터의 이동 경로 활용)

2) Animation()

* 매 프레임마다 시간을 사용하여 플레이어 (롤러코스터)를 레일 위를 이동
* 현재 위치에서 다음 목표 지점까지의 방향 벡터를 계산 후, 이동 거리만큼 전진 (현재 위치 변수 - m\_nRailIndex) (이동 속도 변수 - m\_fRailSpeed)
* 경사 경로 구간에서 속도를 감소, 낙하 구간에서 속도 증가
* XMVectorGetX(XMVector3Length(vTarget - vNew)) < 0.1f 를 사용하여 목표 지점 도달 확인 후, 다음 지점으로 인덱스를 갱신하여 롤러코스터 루프를 무한으로 반복

1. **Level-2 (탱크 씬)**스크린샷, 도표, 라인, 텍스트이(가) 표시된 사진

   AI가 생성한 콘텐츠는 부정확할 수 있습니다.

**기능**: 3인칭 탱크 게임 (플레이어 = 탱크)

**조작법**:

* w : 플레이어 전진
* s : 플레이어 후진
* a : 플레이어 좌 이동
* d : 플레이어 우 이동
* 좌 CTRL : 포탄 발사
* 좌 클릭 : 플레이어 회전 ( 카메라 회전 )
* 우 클릭 : 적 탱크 락온 -> 락온 후 ctrl 시, 파란색 포탄 발사
* r : 쉴드 기능
* q : You Win! 텍스트 생성
* Esc : 메뉴 화면 이동

**구현 내용**

1. **맵 및 벽 구성**

Y축 하단으로 이동은 불가능하기 때문에 추가적인 충돌처리는 구현 x

외곽에 벽은 렌더링 하지 않고, 맵 전체 크기에 (100.0) 플레이어가 외부로 이동하게 되면 0,0,0으로 이동하게 된다.

1. **적 탱크 초기화**

총 10대의 적 탱크를 서로 다른 색상, 시작 위치, 이동 방향 속도로 지정했다.

적 탱크 또한 y축 이동 불가.

각 탱크는 폭발 가능 오브젝트로 구성하였으며 (CExplosiveObject) 포탄에 피격 되었을 시, 폭발 애니메이션이 실행된다.

1. **포탄 발사**

스크린샷, 라인, 텍스트, 도표이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

좌 CTRL 키를 입력하여 포탄을 발사할 수 있다.

FireBullet()함수를 사용하였고 추가적인 구현은 하지 않았다.

1. **포탄 발사 (락온)**

그림, 스케치, 도표, 직사각형이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

PickObjectPointedByCursor()함수를 사용 하였으며, 락온된 오브젝트를 향해서 파란색 포탄 발사.

1. **폭발 이팩트**

스케치, 그림, 도표, 라인이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

1. **쉴드 시스템**

스크린샷, 라인, 도표, 평행이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

m\_bShieldActive – 토글 사용

r키를 입력하여 1초간 쉴드를 생성할 수 있다.

GameObject.h -> CShieldObject.class가 있다. 해당 클래스는 플레이어의 쉴드 클래스이다.

사각형 Mesh로 시각화 하였으며 1초 후, 자동으로 비활성화 된다.

1. **장애물과 충돌 처리 & 포탄과 탱크 충돌 처리**

**스크린샷, 도표, 라인, 스케치이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.**

장애물과 충돌 처리 함수 – CheckPlayerByObstacleCollision()

m\_xmf3PrevPlayerPos 변수를 사용하여 플레이어의 이전 위치를 저장

플레이어의 바운딩 박스 업데이트 함수 호출 – UpdateBoundingBox() ( 바운딩 박스 동기화 )

포탄과 적 탱크(오브젝트) 충돌 처리 함수 - CheckObjectByBulletCollisions()

추가적으로 변경하거나 구현 X

1. **적 탱크 애니메이션과 회전 처리**

탱크의 이동 방향에 맞춰 모델의 회전 방향을 실시간으로 계산하여 전방을 향하도록 회전 시킴 ( 180도 회전 시, 틀어지는 문제점)

1. **플레이어 카메라**

플레이어의 카메라는 m\_pPlayer->Rotate(0.0f, cxMouseDelta, 0.0f) 를 사용하여 항상 정면을 바라보며 좌우만 방향 변경이 가능하게 하였다.

1. **승리 텍스트**스크린샷, 텍스트, 라인, 스케치이(가) 표시된 사진

   AI가 생성한 콘텐츠는 부정확할 수 있습니다.

m\_bGameWin 변수를 사용하여 현재 탱크의 남은 계수를 확인하여 텍스트를 렌더링 한다.

m\_bTankDestroyed 변수를 사용하여 현재 제거된 탱크를 저장 후, 따로 계산 한다. ( 객체 자체를 제거 x )