3D 게임프로그래밍

과제01 설명 문서

<과제01 텍스처>

제출일: 2021. 11. 08

2018182021 윤성주

**목차**

1. **프로젝트 소개**
2. **프로젝트 목표**
3. **구현 내용 설명 및 가정**
4. **코드 설명**
5. **사용한 자료구조와 알고리즘, 디자인 패턴**
6. **프로젝트의 매커니즘**
7. **조작법**
8. **프로젝트 플레이 화면, 지형 전경**
9. **프로젝트 소개**

지형, 오브젝트 등을 텍스처를 사용해 렌더링하고, 빌보드와 이펙트를 추가하였다. 프로젝트의 전체적인 분위기는 시골이고, 오브젝트, 빌보드, 텍스처 등을 통해 분위기를 맞췄다.

1. **프로젝트 목표**
2. 지형을 베이스 텍스처와 디테일 텍스처 여러 개를 사용해서 렌더링한다.
3. 스카이박스를 추가해서 화면에 빈틈이 없도록 한다.
4. 지형위에는 알파 텍스처를 사용하여 길을 그린다.
5. 지형 위에 빌보드로 나무, 꽃을 그린다.
6. 물 지형을 추가해 계곡을 만든다.
7. 지형 위에는 플레이어가 움직이며 Ctrl을 누르면 총알이 발사된다.
8. 총알은 지형위에 설치된 오브젝트와 충돌 시 폭발하며 이때 폭발 이펙트가 생성된다.
9. 폭발 이펙트는 빌보드처럼 카메라에 따라 Look 벡터가 바뀐다.
10. 이 프로젝트를 통해 쉐이더로 연결, 쉐이더 코드의 사용 등을 익힌다.
11. **구현 내용 설명 및 가정**

* **가정**
* 플레이어는 모든 오브젝트와 충돌하지 않는다.
* 지형을 타는 플레이어는 지형을 벗어나지 않는다.
* 총알은 10초후 자동으로 삭제된다.
* 총알과 지형은 충돌하지 않는다. (지형과 충돌하는 경우가 많기 때문에 바로 총알이 사라져 버림. 대신 오브젝트와의 충돌은 가능)
* 물에 빠져도 죽거나 타격을 입지 않는다.
* **구현 내용 및 기능 설명**
  + **지형**

1. 하이트 맵 터레인



* 1개의 베이스 텍스처와 3개의 디테일 텍스처를 사용하여 그림.
* 길은 흙으로, 길과 잔디사이의 경계는 연한 잔디색으로, 나머지는 잔디로 디테일 텍스처 매핑함.
* 길 텍스처를 까는 것은 알파맵을 통해 알파값이 1인 곳은 길 텍스처를 깔도록 함.

쿠키틀이(가) 표시된 사진

자동 생성된 설명 -> 길을 까는데 사용한 텍스처

* 베이스 텍스처는 보이지 않고, 디테일 텍스처만 보임.
* 보간을 사용해서 텍스처끼리 경계를 부드럽게 함.

1. 물 지형

지도이(가) 표시된 사진

자동 생성된 설명

* 텍스처를 이동하도록하여 물이 흘러가는 것을 표현
* 텍스처를 여러 개 합쳐서 물을 표현
  + **플레이어**

(플레이어) (플레이어와 총알)

* 지형을 타도록 함
* 마우스로 시점 변경 가능하고 상하좌우 키로 카메라 시점에 따라 이동한다.
* 플레이어는 길 위로 다닐 수 있다.
* ctrl키를 누르면 총알이 발사되고 총알은 오브젝트와 충돌한다.
  + **빌보드 객체**

실외, 식물, 녹색, 정원이(가) 표시된 사진

자동 생성된 설명자연, 평야이(가) 표시된 사진

자동 생성된 설명

식물이(가) 표시된 사진

자동 생성된 설명

* + - 나무, 풀, 꽃을 ObjectsMap에 따라 렌더링한다.
    - 각각은 바람이 부는 것을 표현하기 위해 약간씩 흔들린다.
    - 빌보드 객체이므로 카메라에 따라 Look벡터가 바뀐다.
  + **폭발 이펙트**

****

* + - 총알과 오브젝트가 충돌 시 오브젝트와 총알은 사라지고 충돌한 위치에 생성된다. 빌보드처럼 카메라에 따라 Look벡터가 바뀐다.
    - 폭발 이펙트의 애니메이션이 끝나면 이펙트는 사라진다.

1. **코드 설명**

프로젝트는 샘플 프로젝트 중 LabProject08-2-1을 기반으로 만들었으며 프로젝트 중 프로젝트에 이미 있던 물, 터레인, 플레이어를 제외한 모든 오브젝트는 직접 추가하였다.

**<Player, DDSTextureLoader12, Timer 소스코드 및 헤더파일>**

* 기존 샘플 프로젝트에서 따로 수정하지 않음.

**<Camera 소스코드 및 헤더파일>**

텍스트이(가) 표시된 사진

자동 생성된 설명

* 빌보드 객체는 카메라에 따라 Look벡터가 변하며 쉐이더에서 빌보드의 Look벡터를 결정하므로 쉐이더로 카메라의 위치를 넘겨주어야 한다.
* 쉐이더는 카메라의 위치를 받아서 처리한다. (쉐이더 설명에서 자세히)

**<GameFramework 소스코드 및 헤더파일>**

1. 총알 발사 입력 받기

* OnProcessingKeyboardMessage() 함수에서 컨트롤 입력이 들어올 경우 Scene의 AddBullet() 함수를 호출하여 총알을 추가한다.

**<Mesh 소스코드 및 헤더파일>**

1. CTexturedRectMesh 클래스 추가
   * Skybox를 위한 클래스. 사각형을 텍스처입힌 메쉬
2. CBillboardMesh 클래스 추가
   * 빌보드 객체를 위한 클래스. 정점6개로 만든 사각형 메쉬
3. CRawFormatImage 클래스 추가
   * raw 형식 파일을 위한 클래스. 알파맵 등 raw 형식의 이미지 클래스.

**<Object 소스코드 및 헤더파일>**

1. 구조체
   * MATERIAL, CB\_GAMEOBJECT\_INFO 구조체를 다르게 정의한다. 두 구조체는 쉐이더 코드에서 사용된다.
2. CTexture 클래스
   * 폭발 이펙트를 위한 변수 Row, Col 관련 변수를 추가한다. 애니메이션을 수행할 때 몇 번째 열, 행의 이미지를 그릴지 정해주기 때문.
   * 폭발 이펙트를 위해 AnimateRowColumn() 함수를 추가하여 일정 시간마다 이미지의 열, 행을 바꿔준다.
3. CGameObject 클래스
   * 오브젝트의 이동방향을 정해주는 Dir 변수 추가
   * Rendering 여부를 결정하는 Alive확인 변수 추가 -> Animate, Render 함수 호출 시 이 변수를 통해 수행 여부를 결정한다.
   * 충돌체크를 위한 BoundingOrientedBox 추가
   * 위의 변수들을 위한 Get, Set 함수 추가 및 객체의 상태를 Alive로 바꾸고 위치를 재조정하는 Awake 함수 추가
4. CSkyBox 클래스
   * 스카이박스 오브젝트 클래스. CTexturedRectMesh를 사용한다.
5. CBullet 클래스
   * 총알 클래스로, 총알을 이동하고, 삭제한다.
6. CMultiSpriteObject 클래스
   * 폭발 이펙트와 같은 애니메이션이 있는 객체 클래스. 이미지의 행, 열을 관리한다.
   * 이미지 애니메이션이 모두 끝나면 삭제된다.

**<Scene 소스코드 및 헤더파일>**

* AddBullet(): GameFramework 클래스에서 호출되는 함수로 총알을 추가할 때 호출된다.
* SetPlayer(): m\_pPlayer를 설정하는 함수
* BuildObjects() 함수
  + Terrain, Skybox, Water, objectsShader, billboardShader, multiSpriteObjectShader를 추가한다.
  + m\_nShaders를 3으로 하여 3개의 쉐이더(objectsShader, billboardShader, multiSpriteObjectShader)가 추가된다.
* CreateGraphicsRootSignature()함수
  + 스카이박스, 빌보드가 추가되었으므로 pd3dDescriptorRanges 배열의 크기를 2 증가하고, 4, 5번 인덱스에 각각 스카이박스(t9), 빌보드를 연결시킨다(t10).
  + pd3dRootParameters 배열의 크기를 10으로 하여 추가된 것들을 연결한다. 연결 시, pd3dDescriptorRanges를 맞게 연결해야 한다. 예를들어, 스카이 박스는 4번 인덱스이므로, 스카이박스 파라미터의 8번인덱스의 DescriptorTable.pDescriptorRanges은 pd3dDescriptorRanges[4]을 연결한다. 이 부분을 정확하게 해야 한다.
  + 스카이박스를 위한 샘플러 추가를 위해 pd3dSamplerDescs의 크기를 1증가하고 샘플러를 하나 더 만든다. 기존은 wrap 샘플러였고, 스카이박스는 clamp 샘플러이다.
  + 루트시그니처에 생성한 파라미터, 샘플러를 연결한다.

**<Shader 소스코드 및 헤더파일>**

1. class CObjectsShader
   * 총알, 충돌 오브젝트를 생성하는 쉐이더. 둘은 같은 텍스처를 공유한다.
   * m\_vecObjects로 두 오브젝트를 관리한다.
   * CMultiSpriteObjectsShader을 멤버로 갖게 하여 총알, 오브젝트 충돌 시 폭발 이펙트가 생성되도록 한다.
   * BuildObject()함수
     + 사용할 오브젝트를 최대 개수만큼 미리 모두 만들어 놓는다. 오브젝트의 개수를 디스크립터힙에 저장해놓으므로 중간에 동적으로 생성할 수 없다.
     + 오브젝트를 생성할 때 디스크립터에 연결할 때 인덱스를 주의해야한다. iCreateCnt 변수를 만들어서 연결하도록 했다.



* + - 오브젝트를 생성할 때 충돌체의 센터, 크기를 지정해준다.
  + Collision\_Check()함수
    - 총알과 오브젝트의 충돌을 확인하는 함수
    - 충돌시 Alive 상태를 false로 바꾸고 폭발 이펙트를 생성한다.

1. class CSkyBoxShader, class CBillboardObjectsShader
   * 각각 스카이박스, 빌보드 객체의 쉐이더 생성
2. class CMultiSpriteObjectsShader
   * 애니메이션이 있는 오브젝트의 쉐이더 생성
   * BuildObjects()함수에서 미리 최대 개수의 이펙트를 생성해 놓는다. 이때, 두 개씩 한쌍이다. (이펙트 생성시 0, 1번 인덱스를 같이 생성한다.)
   * 렌더가 끝나면 Alive 상태가 비활성화 된다.

**<Shaders.hlsl>**

* cbCameraInfo에 gvCameraPosition 추가: 빌보드 객체에서 사용
* MATERIAL 구조체 추가
* cbGameObjectInfo에 gMaterial 추가
* gClampSamplerState 샘플러 추가: 스카이박스를 위해
* VSSpriteAnimation() 추가: 폭발 이펙트에서 사용하는 VS쉐이더
* PSTerrain()에서 디테일 텍스처 보간 수정: 더 자연스러운 보간 구현
* VSBillboardInstancing() 추가: 빌보드 객체의 VS쉐이더. 객체의 벡터를 계산
* PSBillboardInstancing() 추가: 빌보드 객체의 텍스처에 맞는 색상 출력

1. **사용한 자료구조와 알고리즘, 디자인 패턴**
   1. Objectsshader에서 벡터를 사용하여 객체 관리. 생성되는 객체의 수는 처음에 정하고 한 번에 모두 생성한다(최대 개수만큼). 이후에 Alive 변수를 통해 렌더링 여부를 결정하여 생성된 객체만 렌더링하고 삭제되면 다시 Alive 변수를 비활성화한다.
   2. 오브젝트의 인덱스 접근은 생성할 때마다 생성된 개수를 증가하고, 생성된 개수가 최대 개수가 되면 다시 0번 인덱스의 객체를 생성하도록 한다. 총알의 경우 생성 후 10초후에 자동으로 삭제되므로 최대 개수만큼 총알을 생성했다면 0번 인덱스의 총알은 이미 삭제되었으므로 인덱스가 꼬이지 않는다.
2. **프로젝트의 매커니즘**
   1. 플레이어가 지형 중간에 생성된다.
   2. 플레이어는 총알을 쏘고 오브젝트를 파괴한다.
   3. 생성된 오브젝트들을 시점을 변경하여 길로 다니면서 파괴한다.
   4. 나무, 꽃, 잔디, 스카이박스를 보면서 프로젝트의 전체적인 분위기를 감상한다.
3. **조작법**
   * 키보드 이동키로 이동
   * 마우스 좌클릭으로 회전
   * F1, F2, F3으로 시점 변경. F2 상태에서는 마우스 우 클릭으로 회전
   * 컨트롤로 총알 발사
4. **프로젝트 플레이 화면, 지형 전경**

지도이(가) 표시된 사진

자동 생성된 설명

****