3D게임1 과제4 설명 문서

2018182025 이승준

1. 조작법

* **레이싱 게임 Scene**

***2***: 건물 Scene으로 전환

**상하좌우 방향키**: 플레이어 이동

***Space****:* 부스터

***F5***: 시작 위치로 이동

* **건물 Scene**

**1**: 레이싱 게임 Scene으로 전환

***N***: 밤 모드/낮 모드로 전환

***R***: 플레이어 조작 모드로 전환

**상하좌우 방향키**: 플레이어 전후좌우 이동

***PageUp/PageDown***: 플레이어 상하 이동

1. 과제에 대한 목표 및 가정

본 과제는 1. Scene을 전환할 수 있는지 여부와 2. 오브젝트의 배치 상태와 Material이 저장된 Scene.bin 파일을 읽어 충돌체크와 조명 등을 구현할 수 있는지 여부를 묻는 과제이다.

Scene 전환의 경우, 두 프로젝트가 동일한 구조로 구현되어 있다면 구현하기 매우 수월하겠으나 파일로부터 Mesh를 읽어 들이는 방법, 조명과 재질의 처리, 계층 구조로 이루어진 오브젝트의 차이 등부터 사소한 부분까지 프레임워크 상에서 서로 다른 부분이 너무 많았기 때문에 어느 한 프로젝트를 다른 프로젝트에 이식시키기 매우 어려울 것이라고 생각되었다. 따라서 프로젝트를 이식시키기 보다는, 객체 내에서 겹치는 부분을 클래스로 분리시켜 주고, 각 프로젝트에서 해당 클래스를 상속받아 서로 다른 부분을 각각 구현해주는 식으로 제작하고자 하였다.

충돌체크의 경우 .bin 파일 내에 바운딩 박스에 대한 정보가 포함되어 있으므로, 그것을 활용하여 플레이어와 오브젝트의 바운딩 박스를 설정해 주고, 플레이어의 Update 시마다 이전 위치를 저장해둔 후, 충돌이 발생하면 이전 위치로 되돌리는 식으로 검사하기로 하였다.

조명의 경우는 오브젝트를 순회하며 객체의 이름을 검사한 후, 건물이나 가로등 등 조명을 넣고 싶은 객체를 발견할 경우 그 오브젝트의 위치를 받아와 해당 위치에 조명을 설정해 주기로 계획하였다. 플레이어에게 달린 조명과 달리 위치가 변하는 경우는 없으므로 Scene의 생성자에서만 설정해 주었고, 밤/낮 전환 시에는 조명을 꺼 주었다.

또한 중간 과제에서 오브젝트를 배치할 때마다 파일을 새로 읽어 와 프로그램 실행이 매우 늦어지는 문제가 있었는데, 이번 과제에서는 해당 문제를 해결해 보고자 하였다

주어진 .bin 오브젝트들을 배치하기 위하여 Unity 엔진을 활용하여야 했는데, Unity 엔진을 설치하고 오브젝트들을 배치하였으나 실제 Scene.bin으로 만드는 과정에서 어려움을 겪어 부득이하게 샘플 프로젝트에서 제공된 Scene.bin을 사용해 제작하였다.

1. 실행 결과

다채로운이(가) 표시된 사진

자동 생성된 설명레이저이(가) 표시된 사진

자동 생성된 설명

*(Scene 1 실행 화면)*

프로그램을 실행하면 먼저 레이싱 게임 Scene이 출력된다. 중간 과제와 동일하게 방향키를 사용하여 이동할 수 있으며, Space를 이용하여 부스터 역시 사용 가능하다.

다채로운, 회의실이(가) 표시된 사진

자동 생성된 설명옅은, 실외, 교통, 켜진이(가) 표시된 사진

자동 생성된 설명

*(Scene 2 낮 모드) (Scene 2 밤 모드)*

2를 누르면 건물 Scene으로 이동한다. 다시 1을 눌러 레이싱 게임 Scene으로 이동할 수도 있으며, 플레이어의 이동 정보 등은 그대로 유지된다. 초기 건물 Scene 모드는 전체 Scene을 볼 수 있는 상태에서 출력되며, 해당 상태에서는 화면 전환 등 조작이 불가하다.

건물 Scene에서 N을 누르면 밤 모드로 전환한다. 밤 모드에서는 Diffuse 조명의 세기가 약해지고 배경색이 변경되며, 가로등과 일부 건물에 조명 효과가 추가된다.

실내, 다채로운, 여러개이(가) 표시된 사진

자동 생성된 설명하늘, 옅은, 어두운, 켜진이(가) 표시된 사진

자동 생성된 설명

*(플레이어 이동) (충돌 처리)*

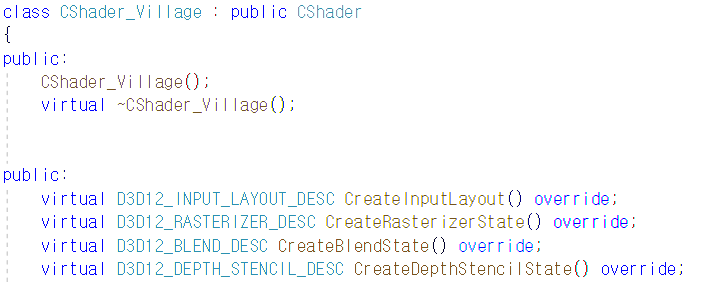
건물 Scene에서 R 버튼을 누르면 플레이어 이동 모드로 변경할 수 있다. 방향키와 PageUp / PageDown 버튼을 통해 플레이어를 이동시킬 수 있으며, 모든 오브젝트에 기본적인 충돌 처리를 하여 오브젝트가 벽을 뚫고 지나가지 않는다.

밤 모드에서 다시 N 버튼을 눌러 낮 모드로 바꿀 수 있다.

1. 구현 내용

텍스트이(가) 표시된 사진

자동 생성된 설명



*(Scene 전환)*

*Scene* 전환의 경우, Scene, Shader, Objects, Player, Mesh 등 두 프로젝트 상에서 서로 다르게 구현되어 있는 클래스들을 각자 사용할 수 있도록 하기 위해 하나의 부모 클래스를 만들어 두고, 공통되는 함수와 변수 등은 해당 클래스에 배치한 후, 그렇지 않은 함수들은 순수 가상 함수로 지정하여 Shader\_Village, Shader\_Racing과 같이 클래스를 분리하여 재정의해주는 방식으로 구현하였다.

텍스트이(가) 표시된 사진

자동 생성된 설명

*(충돌 처리)*

충돌 처리의 경우는 Update가 발생하기 전 미리 플레이어의 위치를 저장해 두고, 충돌이 발생하면 LoadPosition() 함수를 호출해 이전 위치로 플레이어를 되돌려주는 방식으로 구현하였다.

텍스트이(가) 표시된 사진

자동 생성된 설명

*(조명)*

Scene의 생성자에서 전체 Object를 순회하며 c4\_lamp와 같이 조명을 넣고 싶은 인스턴스의 이름을 확인하여, 해당 오브젝트의 Position 정보와 Look 정보를 받아와 객체에 맞는 적절한 조명을 설정해 주었다.