**컴퓨터 그래픽스 최종 프로젝트**

김태순, 김준호

* **프로젝트 소개**
  + 해당 프로젝트는 OpenGL을 통해 서로 다른 두개의 공간을 연결하는 포탈을 구현하여 이를 이용한 간단한 게임을 제작하는 프로젝트입니다.
  + 가상 카메라를 이용하여 건너편 공간을 렌더링하여 포탈평면에 텍스처로 입힘으로써 마치 포탈을 통해 건너편 공간과 이어져 있게 보이도록 구현하였습니다.
* **프로젝트의 구조**
  + 프로젝트는 각각의 객체를 클래스로 구현한 다음에 Stage에서 객체들의 상호작용과 렌더링 함수를 호출, Stage의 Update, Render함수를 Main에서 호출하는 형식으로 구현하였습니다.
  + 이렇게 구현하면 각 클래스를 따로 구현한 다음에 Stage에 추가하기만 하면 되기 때문에 공동작업하기에 최적화된 구조라 생각했습니다.
  + 프로젝트의 객체로는 Camera, Object, Player, Portal이 있고 쉐이더 프로그램을 저장하는 Shader가 있습니다.
  + 먼저 Camera에는 카메라의 위치 position, 카메라의 정면벡터 front, 카메라의 위를 표시하는 벡터 up, 정면에서부터 어느정도 회전되어 있는지를 나타내는 yaw와 pitch, 그리고 마우스에 따라 얼마나 회전되는지를 결정하는 민감도 sensitivity가 저장되어 있다. Camera의 주요 함수는 SetMouseLook과 UpdateCamera, GetViewMatrix가 있는데 SetMouseLook은 화면에서 마우스가 이동한 거리를 인자로 입력 받아 카메라의 각도를 수정하는 함수다. UpdateCamera는 그렇게 변한 yaw와 pitch를 통해 front를 업데이트하는 함수이다. 마지막으로 GetViewMatrix는 이름 그대로 카메라의 position, front, up을 이용해 View 행렬을 만들어 리턴해준다.
  + 다음으로 Object에는 obj파일을 읽어서 장하는 Load\_Object함수와 변수들이 있고, 오브젝트를 초기화하는 Set\_Obj, 오브젝트를 그리는 Draw, 오브젝트에 대한 물리법칙을 처리하는 Update함수, 충돌처리에 관한 CheckCollisionWithBox, HandleCollision, 그리고 바닥에 닿았는지 확인하는 IsOnFloor함수가 있다. 클래스에 저장되는 값으로는 위치 position, 크기 size, 속도 velocity, 색상, RGB, 이동가능한 오브젝트인지 표시하는 isMovable, 바닥에 닿아있는지 표시하는 isGrounded, 그리고 질량 mass, 마찰계수 friction이 있다.
  + Player는 실제 사용자가 움직이게 만드는 객체로 저장하는 변수는 위치 position, 속도 velocity, 충돌박스의 크기(실질적으로는 플레이어의 크기) colliderSize, VAO, VBO, 플레이어 객체가 초기화(생성)이 정상적으로 되었는지 확인하는 isInitialized, 바닥에 닿아 있는지 표시하는 isGrounded, 플레이어의 카메라 camera변수가 저장되어 있다.
  + 마지막으로 Portal에는 위치 position, 회전각 rotation, 크기 size, VAO, VBO, framebuffer, textureColorbuffer, renderbuffer, 초기화 되어있는지 표시하는 isInitialized, 연결된 포탈을 가리키는 linkedPortal, 포탈의 폭과 높이를 저장하는 PORTAL\_WIDTH, PORTAL\_HEIGHT, 그리고 반대편 시점을 렌더링 하기 위해 사용되는 가상 카메라 virtualCamera가 변수로 저장되어 있다.
* **프로젝트 진행 사항**
  + 원래의 계획은 포탈을 통과할 때 포탈끼리의 크기 차이만큼 통과한 물체 또한 크기가 변하고 포탈을 직접 플레이어가 생성할 수 있게 만들 계획이었는데 포탈의 구현에 생각보다 많은 시간이 소요되어 구현하지 못했다.
  + 현재 구현되어 있는 내용으로는 플레이어의 이동과 카메라가 마우스에 따라 회전하는 효과와 가상 카메라를 구현하였고, 포탈은 반대편 시점의 렌더링 및 플레이어가 통과 가능하게 구현, 충돌 처리는 충돌 박스를 통한 충돌 감지와 충격량 기법을 적용한 충돌 해결, 오브젝트는 이동가능, 불가능 오브젝트를 구분하고 그에 따른 다른 충돌 처리를 구현하였고 텍스처 매핑은 도입하지 못했다.  
    그 이외에도 스테이지는 틀은 만들었지만 서로 다른 스테이지끼리 연결되는 것은 구현하지 못해 한 개의 스테이지에 여러 스테이지가 있는 것처럼 만들었고 조명 효과 또한 넣지 못했다.
* **팀 원간 작업한 내용**
  + **김태순**
    - 플레이어와 카메라의 이동 및 회전, 가상 카메라를 통한 포탈의 렌더링, 오브젝트의 충돌 및 물리 법칙 적용
  + **김준호**
    - ㅁㄴㅇㄹ
* **프로젝트의 결과**
  + **3차원 공간에서의 이동 및 오브젝트와의 충돌을 구현하였고 포탈을 이용해 연결된 공간을 관찰하고 이동할 수 있는 기능들이 만들어졌고 이를 이용한 간단한 게임을 제작하였다.**
  + **총 3개의 스테이지로 구성되어 있고 각 스테이지에는 3개의 포탈이 있다.  
    포탈 너머를 잘 관찰하여 진짜 땅과 연결되어 있는 하나의 포탈을 찾아내서 건너가는 형식으로 진행되며 틀린 포탈을 건너가게 되면 더 이상 포탈이 없어 진행이 불가능하게 된다.  
    이미 건너온 포탈을 이용하거나 바닥으로 떨어지면 처음으로 돌아오게 된다.  
    총 3개의 진짜 포탈을 찾아 건너가면 게임을 클리어하는 포탈이 있는 공간에 도착한다. 마지막 포탈을 건너가면 게임이 끝나게 된다.**
* **프로젝트의 명령어**
  + **WASD로 이동 마우스를 이용한 카메라 회전**
  + **Q를 누르면 프로그램을 바로 종료할 수 있다.**
* **프로젝트 개발 소감 및 후기**
  + **김태순**
    - 처음에는 참고자료도 있고 인터넷에 관련된 정보들이 많아서 구현이 어렵지 않을 것이라 생각했는데 생각보다 가상 카메라를 설치하고 그 화면을 포탈평면에 렌더링 하는 과정이 복잡했고 그 때문에 시간이 많이 소요되어서 구현하고 싶었던 내용들을 전부 구현하지 못한 것이 아쉽다. 그럼에도 0부터 만들어서 물리법칙이 적용된 3차원 공간과 포탈을 직접 만들어보니 뿌듯했고 그와 관련된 알고리즘이나 이론들을 공부할 수 있어서 재미있고 유익한 시간이었다.
  + **김준호**
    - ㅁㄴㅇㄹ