Computer Graphics Project 2

2019092306 구본준

1. Implemented requirements

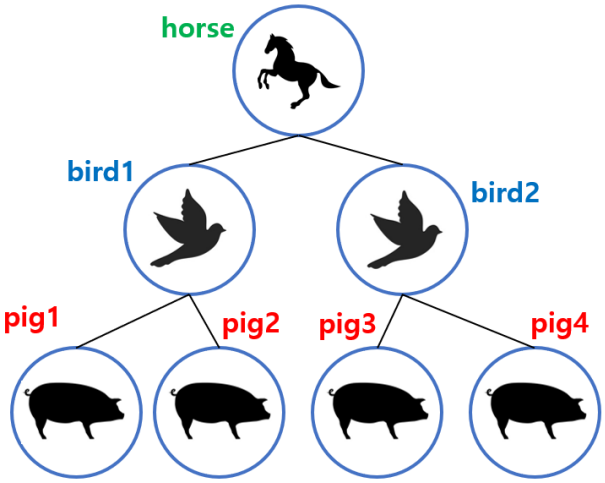
- Camera and grid

카메라의 경우 project1과 동일한 viewing matrix로 구현하였으며 grid는 빛의 영향을 받지 않도록 하기 위하여 별도의 shader을 사용해 렌더링하였다.

- Single mesh rendering mode

glfwSetDropCallback 함수를 이용하여 파일을 드래그 후 창에 올려놓았을 때 그 파일의 위치를 알아낼 수 있도록 구현하였다. obj 파일 정보를 바탕으로 glm array를 만들어 vao를 생성해주는 *prepare_vao_objfile* 함수를 구현하여 드랍한 파일을 처리할 수 있도록 하였다. *prepare_vao_objfile* 함수의 vertex와 face 데이터는 그대로 대입을 해주고 vertex normal 데이터의 경우 vertex가 포함 되어있는 face들의 vertex normal 정보를 평균 내어 대입을 해주었다. 이 때 하나의 면이 여러 개의 삼각형으로 이루어져 있는 경우 한 vertex가 같은 면에서 같은 vertex normal을 여러 개 가질 수 있으므로 이를 해결하기 위해 중복을 제거해주었다. 이후 삼각형 mesh들은 그대로, 그 이상의 다각형은 삼각형으로 쪼개 glm array에 넣어주었다. 마지막으로 화면에 렌더링을 할 때 필요한 vertex 개수 정보, 출력해야 할 정보를 담아 놓은 리스트와 함께 vao를 반환해준다. 이렇게 생성한 vao를 바탕으로 drop한 파일을 띄워주고 obj 파일의 정보를 출력해주었다.

- Animating hierarchical model rendering model



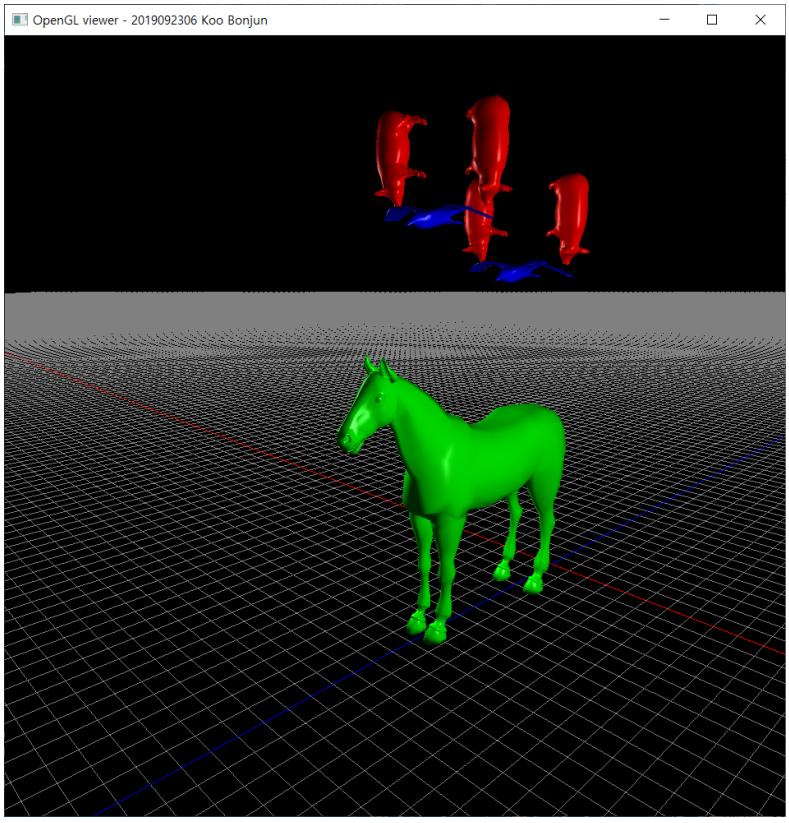
말, 새, 돼지 형태의 오브젝트를 위 그림의 가계도 형태로 구성하였다. 총 세 level로 되어있으며 non-leaf node 위치의 오브젝트들은 각 두 개의 자식 node를 가지고 있다. 말

은 앞뒤(z축 방향)로, 새는 위아래(y축 방향)로 움직이며 돼지는 y축을 중심으로 회전한다. *prepare_vao_objfile* 함수를 호출하여 obj 파일에 대한 vao를 생성하였고 node class를 통해 hierarchical한 구조를 구성하였다. 각 node가 갖고 있는 modeling matrix, color 정보를 uniform variable에 대입하여 그림을 그려주는 *draw_node* 함수를 구현하여 각 동물 오브젝트를 화면에 띄워주었다.

- Lighting & Etc

phong illumination과 phong shading 방법을 사용하여 lighting을 구현하였다. 이때 material color은 uniform 변수를 통해 설정을 하였으며 lighting color은 (0.7, 0.7, 0.7)으로 설정하였고 위치의 경우 하나는 (2, 1.5, 1)에 다른 하나는 (-2, 1.5, 1)에 배치하였다. View position의 경우 viewing matrix 역행렬의 첫 번째 column이므로 이를 계산하여 대입해주었다. Wireframe/solid mode는 *glPolygonMode* 함수를 통해 변하도록 구현하였다. 돼지 오브젝트의 경우 vertex가 너무 촘촘하여 wireframe 모드에서도 solid형태인 듯 보이나 자세히 보면 wireframe 형태로 나타나는 것을 확인할 수 있다.

2. The animating hierarchical model video.



https://youtu.be/5XU7HpjU73E