Computer Graphics Project2 :

Obj viewer & drawing a hierarchical model

2019073181 심기호

1. Which requirements you implemented

A : manipulating camera and grid line : 전 과제 보고서에 있으니 자세한 설명은 생략한다.

B : Single mesh rendering mode : drop_callback()함수가 불리면 os.path.basename을 통해 파일의 경로 중 파일 이름만 추출하고 파일을 readLines함수를 통해 한줄 한줄 읽으면서, v로 시작하는 것들은 vertex list에 저장, vn으로 시작하는 것들 것 vertex_normal list에 저장했다.

F로 시작하는 것들은 정규표현식 re.findall(r'd₩+')로 숫자들을 파싱했고 숫자들의 개수가 6개 초과인 것과 아닌 것으로 구분해서, 6개인 것들은 0번째, 2번째, 4번째만 vertex정보만 담고 잇으니 그것들만 faces list에 저장했고, 6개 초과인 것들은 0번째 vertex와 연속된 vertex들의 정보들을 쌍으로 총 0, 2i, 2i+2 이렇게 3개의 vertex 정보들을 faces list에 저장했다.

그리고 다시 line을 읽어들이면서 face 정보들 중에서 vertex마다의 face 정보들을 추출해서 그 번호에 해당하는 vertex_normal 중 xyz 값들을 tempnormal이란 list에 더해주었다. 그리고 vertexcnt라는 list에서 face 정보 중 그 vertex가 몇번 등장하는지 세주었다. 그래서 다 읽어 들이고 나서는 tempnormal에 들어있는 xyz값을 vertexcnt로 나누어서 vertexnormal의 평균 값들을 구해주었다. 그 후 vertex_info라는 list를 새로 만들어서 vertex의 xyz value들, 그 점의 vertex_normal xyz value들을 차례대로 총 6개씩 넣어주었다.

이렇게 하면 drop_callback을 호출했을 때 점들의 정보와 normal정보들, index정보들이 모두 추출되고, prepare_obj함수를 통해 vertex_info와 face를 인자로 넘겨주어 vao를 만들어서 obj파일을 그리게 된다. 처음에 obj파일을 읽을 때에 face정보를 읽으면서 길이가 6개인 건 face정보가 3개, 8개인 건 face정보가 4개, 그 이상은 4개 초과를 의미하기 때문에 그 정보를 저장했다가 drop_callback이 불리면 obj file의 정보들을 출력하게 해주었다.

C : Animating hierarchical model rendering mode
Hierarchical render()함수에서는 각각의 obj file들을 os.path.join을 통해서 읽어들이고, open한 후 parsing_for_common_obj함수를 통해서 vertex_info정보와 face정보들을 저장하게 둔다.

Parsing_for_common_obj()함수에서는 drop_callback함수에서 처럼 동일한 과정을 진행하지만, f로 시작하는 face정보들의 형식이 v/vt/vn이기 때문에 for문에서 3씩 jump를 하면서 i(vertex)와 i+2(vertex_nomral)의 정보들을 읽었다. 그 후 drop_callback에서처럼 vertex_normal의 평균값을 낸 후 veretex_info list에 vertex의 xyz 값들과 vertex_normal의 평균 xyz값을 저장해주었다. 그리고 main문에서 미리 그려놓을 obj model의 obj file을 읽으면서 정보들을 추출해놓고, Node정보들에다가 계층 구조 정보를 입력해놓는다.

본 과제에서는 coral이 base가 되고, starfish4개와, shark 3개가 coral의 자식이 된다. Starfish는 따로 자식은 없고, shark3개들은 각각 2개의 fish 자식을 가진다. Fish와 starfish가 leafnode가 된다.

h_mode라는 변수를 통해 hierarchical mode를 변경할 수 있게 해주었고, 그 안에서는 각각의 obj file의 transform정보들을 입력해주었고, prepare_vao_obj함수를 통해 vao를 준비하고, draw_node를 통해서 각각의 obj file을 그려준다.

D: Lighting & etc

Lighint은 fragment shader에서 phong illumination과 phong shading을 통해서 구현하였고, multiple light은 light_pos, light_color, material_shininess를 2개씩 만들어 총 2개의 광원을 구현했다.

Wireframe과 solidmode를 구분하는 것은 wiremode라는 변수를 통해 main문에서 glPolygonmode(GL_FRONT_AND_BACK,GL_LINE 또는 GL_FILL)을 통해 wiremode와 solidmode를 구분해주었다.

2. Hyperlink to video

https://www.youtube.com/shorts/k9VEku3rh4Q