**ClassAssignment2 보고서**

**2017030119 이유돈**

**모든 기능을 구현하였고 Extra credit의 구현 요구사항또한 완성하였습니다.**

**Animating Hierarchical model youtube :** [컴퓨터그래픽스 ClassAssignment2 - YouTube](https://www.youtube.com/watch?v=1mBWn8dWFvc)

**https://www.youtube.com/watch?v=1mBWn8dWFvc**

**1.구현된 기능**

**A. draw grid the reference grid plane**

- ClassAssignment1의 기존의 코드에서 이미 구현된 drawfame() 함수를 이용하여 grid를

그렸으며 Materialfv의 색깔은 파란색으로 설정하여, 제 hierarchical modeling에서의 배와

바다라는 컨셉에 맞추었습니다.

**B. Single mesh rendering mode**

i**.** single mesh rendering mode는 singleMesh()라는 함수를 호출하여 표현합니다.

ii. viewer에서는 하나의 obj파일만을 표현하도록 구현되었으며, glfwSetDropCallback 호출될때 마다 drop 되는 새로운 obj파일을 rendering 하도록 구현했습니다.

iii. obj파일에서 vertex, vertex normal, 그리고 f의 정보를 파싱하여 global 변수에 저장하였고, draw 함수를 통해 flat shading일땐 glDrawArray를 이용해 표현하고 smooth shading일땐 glDrawElements함수를 이용해 mesh를 그리게 됩니다. 이는 draw()함수 내부에 구현했습니다.

iv. obj file에서 파일이 drop될 시, filename, total number of faces, number of faces with 3, vertices number of faces with 4 vertices, number of faces with more than 4 vertices이 출력되게 됩니다. 이때 hierarchical mode일시, file의 입력을 받고 h키가 입력되면 마지막으로 drop된 파일이 출력되게 됩니다.

**C. Animating hierarchical model rendering mode**

ii. key ‘h’를 누를 시 key\_callback을 통해 boolean 변수 hierarchical이 true가 되며 ‘animating hierarhcical mode’가 작동됩니다.

iii. 상대경로를 위해 rendering에 사용될 파일들은 같은 디렉토리에 넣어져있고 open함수를 통해 열게 됩니다. 사용될 obj 파일들은 총 4개로 이루어져 있습니다.

v. matrix stack을 통해서 전체 model이 움직일때는 같이 움직이도록 하고, 개별동작이 필요할때에도 각각의 child들이 움직일수 있도록 했습니다.

**vi.** Animate the model to show the hierarchical structure

1. 본 hierarchical model은 boat, motor\_fan, rope, anchor로 이루어져 있으며 배는 최상위 부모이고, 그 밑의 자식은 회전하는 모터 팬과 밧줄이며, 배의 닻은 밧줄의 child로 설정하였습니다. 부모인 배가 움직이면 matrix stack을 통해 전체 모든 child는 움직이게 됩니다. Child인 모터팬은 독자적으로 회전을 하여 모터를 이용해서 배를 움직이는 효과를 표현했습니다. 또다른 child인 밧줄은 물살에 의해 흔들리는 모습을 표현하고자 최대 45도 회전하며, 이때 밧줄의 child인 닻또한 함께 움직입니다. 밧줄의 child은 닻은 바닷속 깊은 곳에 존재하기 때문에 더 강한 물살을 받는다는 가정으로 5도가량 더 회전합니다. 이로서 배의 모든 child는 parent와 함께 움직이나, 각각의 child는 독자적인 움직임을 가지게 됩니다
2. 제가 선택한 hierarchical model은 닻과 모터를 가진 배이며, 모터 팬을 통해서 앞으로 나가고, 닻과 밧줄을 바닷속에서 끌고 향해합니다. 닻은 배를 정박하는 용도또한 있지만 작은 배에게는 중심이 되는 점을 이용하여 닻이 완전히 내려져 있지 않는 한 배는 앞으로 나가는 것이 가능합니다

**D. Lighting & Etc**

**i.** 1. Light source는 2가지를 사용했습니다. 1번 light는 GL\_LIGHT0을 사용하고 (0.5, 0, 1)에 위치하고 point light를 사용해주었으며 색깔은 (1, 1, 0)인 노란색을 사용했습니다. 2번 light는 GL\_LIGHT1을 사용하고 (-0.5, -0.5, -1)에 위치하고 색깔은 (116/255, 73/255, 108/255,) 보라색을 사용했습니다.

Singlemesh mode에서는 material의 색깔은 (0.627, 0.321 , 0.176) 갈색을 사용했습니다.

Hierarchical mode에서는 배와 motor fan은 material 색상은 갈색을 사용해주었고, 닻과 밧줄은 파란색으로 해주었으며 grid와 frame은 파란색을 사용했습니다.

**ii.** rendering 될때 default는 glPolygonMode에서 GL\_LINE으로 설정된 wireframe mode이고, key ‘z’를 입력받을 때 GL\_FILL이 활성화 되어 도형이 채워지도록 구현했습니다.

**4.Extra Credits**

**A.** default로는 flat shading을 glDrawArrays를 이용하여 구현했습니다. Key ‘s’가 입력된다면 smooth shading mode로 되어, 각 vertex에 해당되는 모든 normal vertex들을 다 더한 뒤, 벡터의 크기를 나눈것으로 normal vertex를 설정하였고 glDrawElements를 이용하여 rendering됩니다.

**B.** 삼각형이 아닌 quad나 그 이상의 polygon들은 triangulation algorithm을 사용해서, 한 개의 face가 네개 이상의 vertex로 이루어져 있을경우 3개로 다시 표현 하도록 만들고 이들을 glDrawArrays와 glDrawElements에 사용하도록 만들었습니다