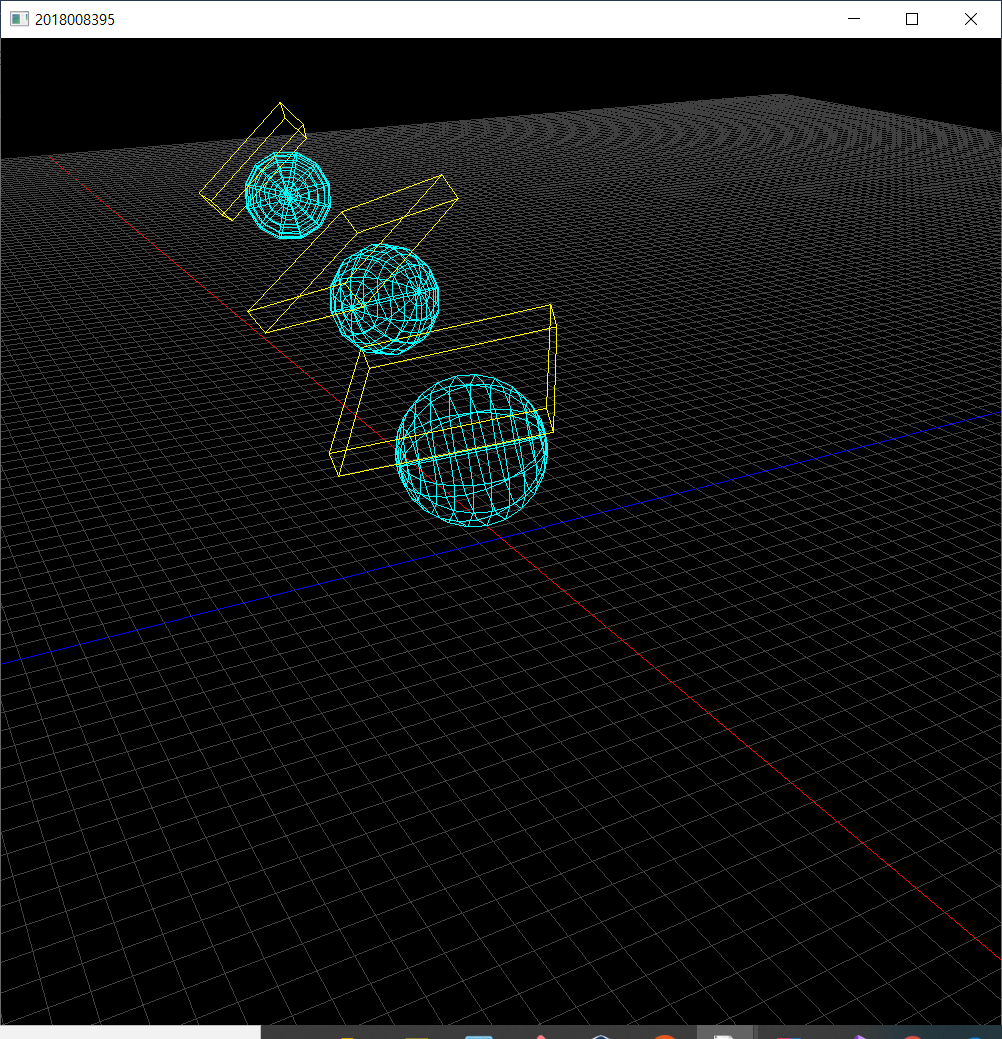
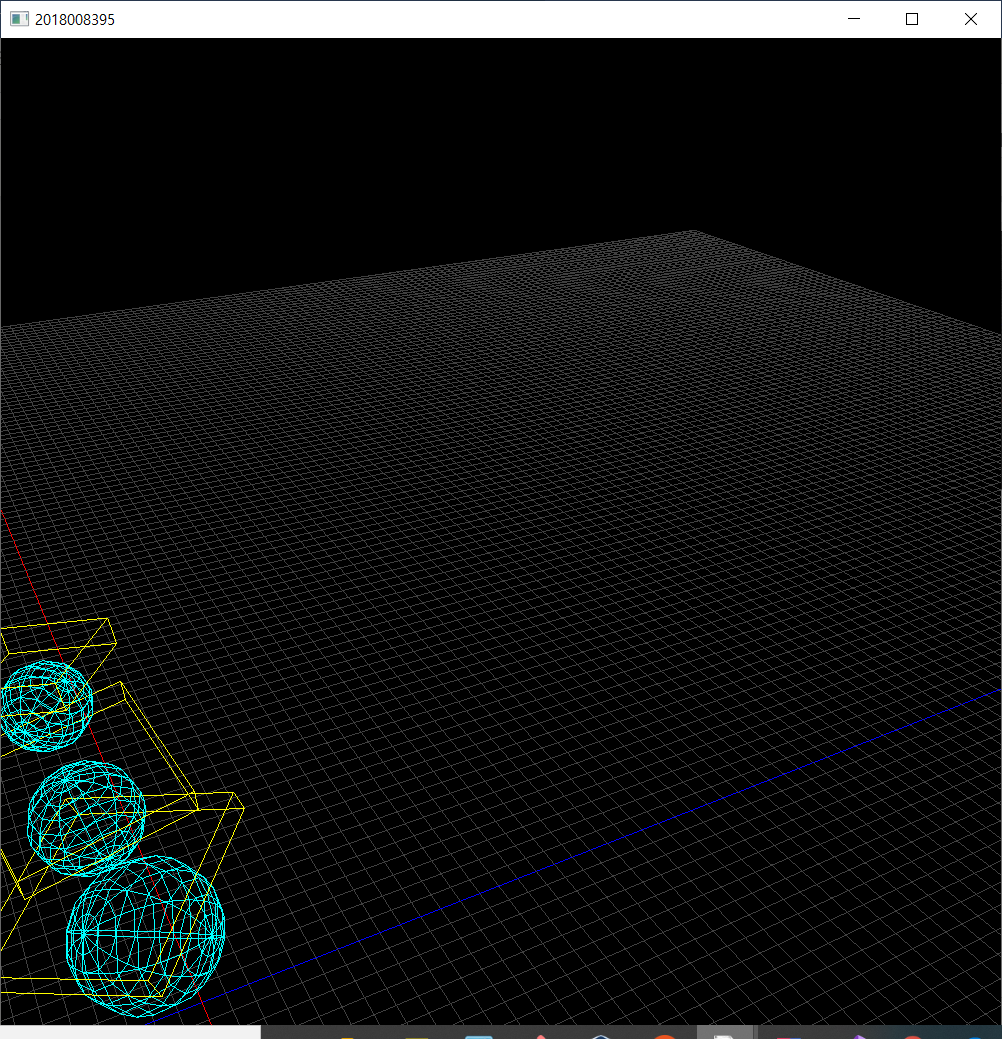
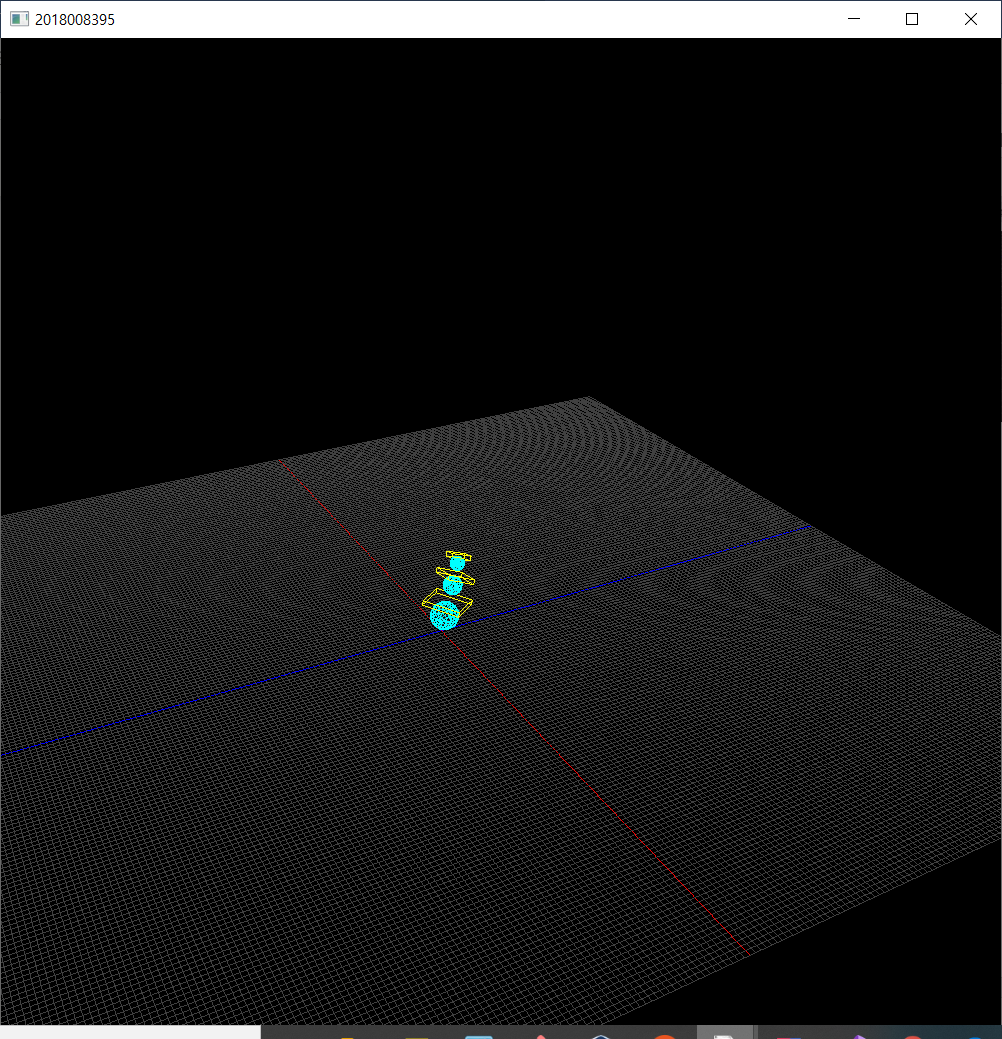
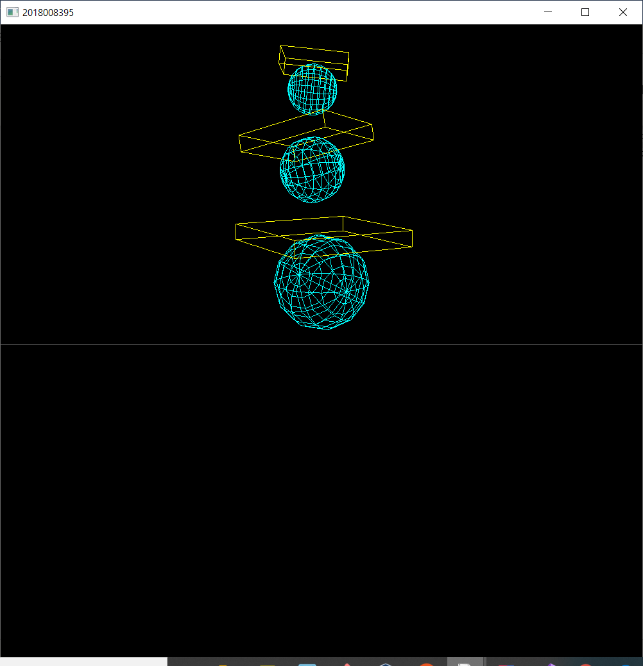
Class Assignment 1 Report

2018008395

박정호

1. Implemented Requirement
   1. Basic OpenGL Viewer  
      기본적으로 render는 이렇게 동작한다. elevation과 azimuth를 이용해서 w벡터를 구하고, 이와 up 벡터를 이용해서 u와 v벡터를 구한다. 여기서 w는 origin, 즉 원점에서 카메라까지의 벡터이기 때문에 카메라의 좌표는 단순히 w를 원점 좌표에 더하기만 하면 된다. 또한 up 벡터는 기본적으로 (0,1,0)이지만, elevation이 커지면서 카메라의 y좌표가 음수가 되면 up벡터도 (0,-1,0)으로 변경한다. 이렇게 해야 rotation이 자연스럽게 수행된다. elevation과, azimuth, origin의 값 변화는 아래에서 설명하겠다.
      1. Orbit  
         드래그 자체는 OpenGL의 마우스 버튼 이벤트에 없었기 때문에 마우스 버튼 콜백과 커서 콜백을 같이 사용했다. 마우스 버튼 콜백으로 왼쪽 버튼의 PRESS를 받으면, 전역변수인 enableOrbit을 True로 변경하고, RELEASE를 받으면 False로 변경했다.   
         커서 콜백에서는 enableOrbit이 True일 때만 다음과 같은 연산을 수행했다.  
         기존 커서 위치와 현재 커서 위치의 차이를 구해서 X 방향 차이에 비례하게 azimuth를, Y 방향 차이에 비례하게 elevation을 변경했다.
      2. Panning  
         위의 Orbit과 같은 이유로 마우스 버튼 콜백과 커서 콜백을 같이 사용했다. Orbit과 같은 방법으로 전역변수인 enablePanning의 값을 변경해서 드래그의 시작과 끝을 구별했다.  
         커서 콜백에서는 enablePanning이 True일 때만 다음의 연산을 수행했다.  
         기존 커서 위치와 현재 커서 위치의 차이를 구해서, X 방향 차이에 비례하게 u벡터를 origin(카메라가 보는 점의 좌표)에 더하고, Y 방향 차이에 비례하게 v 벡터를 origin에 더했다.
      3. Zooming  
         스크롤 콜백을 이용해서 yoffset에 비례하게 distance의 값을 변경했다.
   2. Hierarchical model  
      기본적으로 자동으로 움직여야 하기 때문에 시간에 따른 삼각함수 연산으로 주기적 움직임을 만들었다. 실행시키면 구와 육면체를 쌓은 것이 통통 튀는데, 이는 각 파트에 삼각함수의 절댓값만큼을 translate해서 구현했고, 각 파트가 y축과 x축을 기준으로 회전하도록 했다.   
      구와 육면체 한 쌍이 하나의 파트이고, 아래의 파트가 위의 파트의 parent이다. 즉, 각 파트에 적용된 rotate함수가 중첩되면서 위의 파트가 더 많이 회전하는 것처럼 보인다. 또한 각 파트에 적용된 translate 함수도 중첩되어서 위의 파트가 더 많이 튀는 것처럼 보이게 되고, 그 방향 또한 매번 rotate 방향에 따라 다르게 돌게 된다.

Zooming 후

Panning 후

Orbit 후

처음 시작 화면