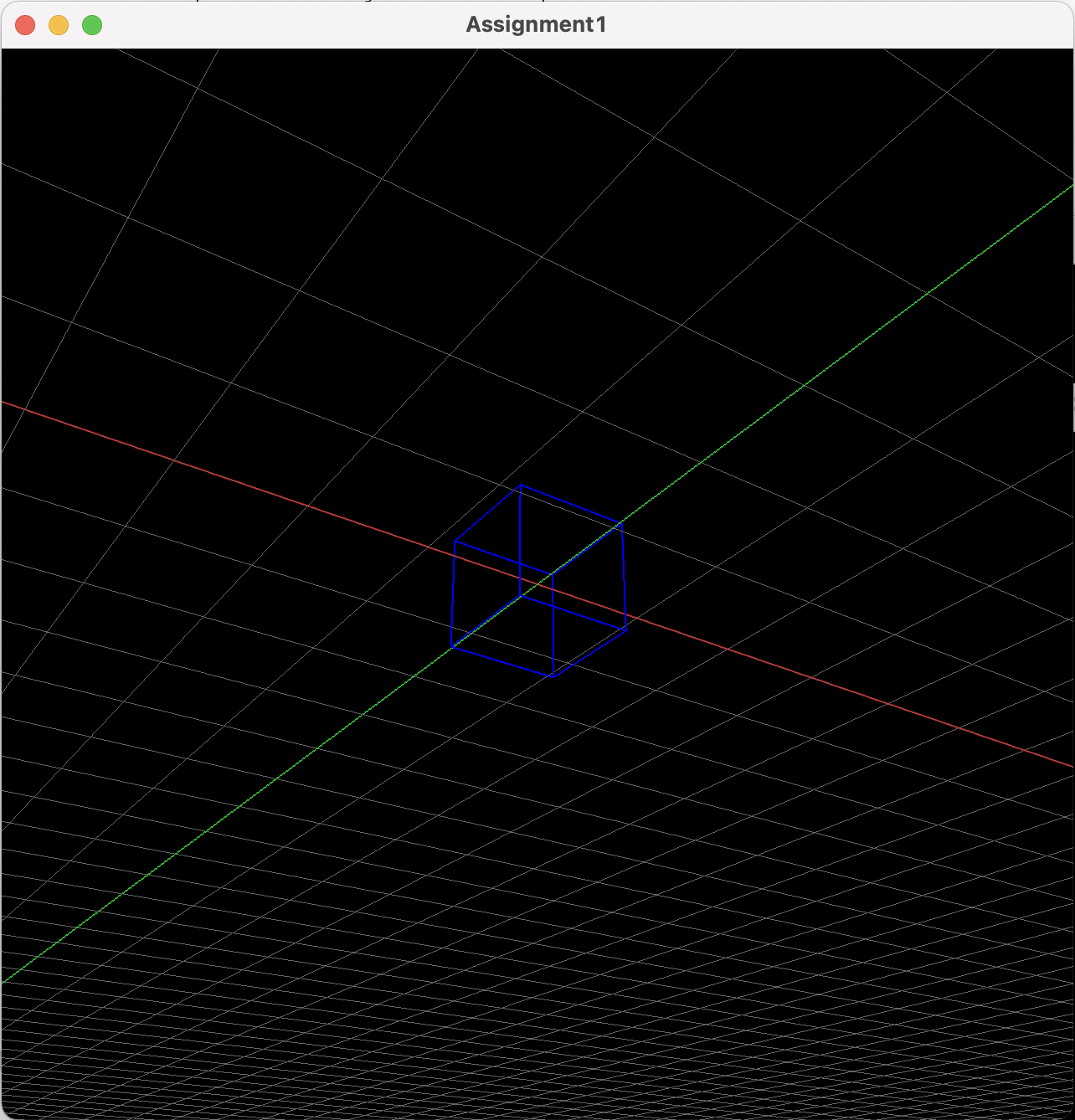
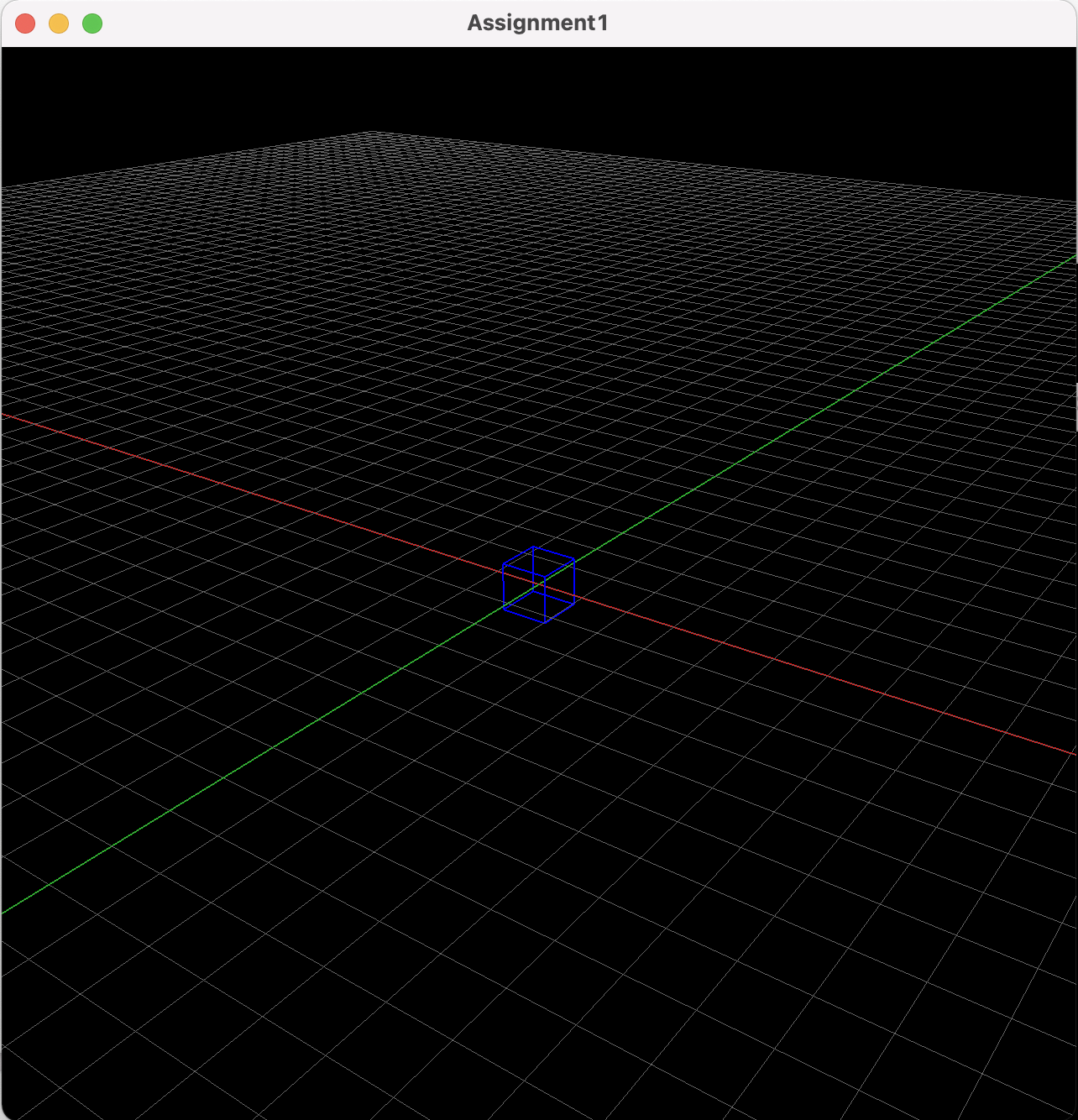
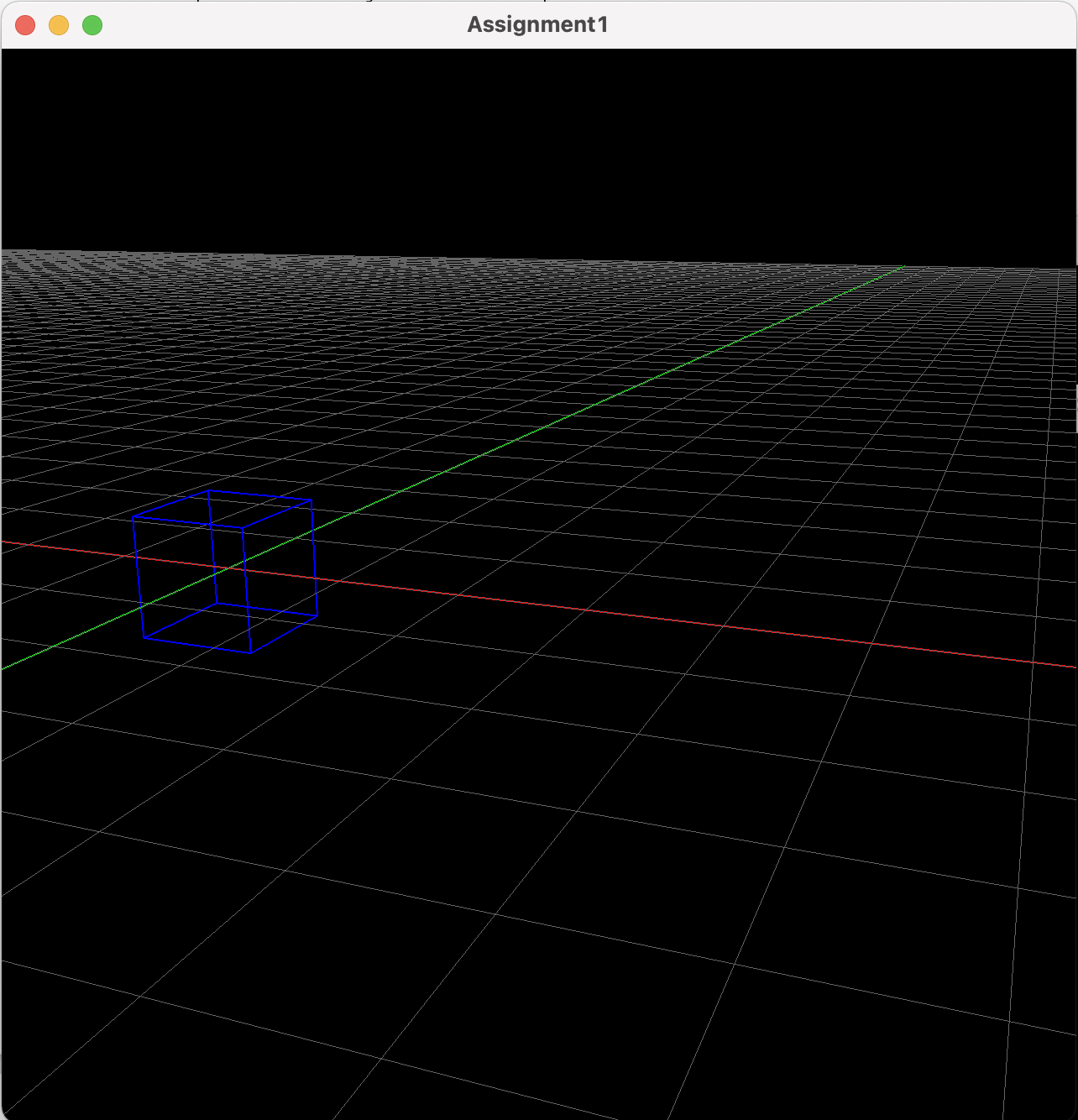
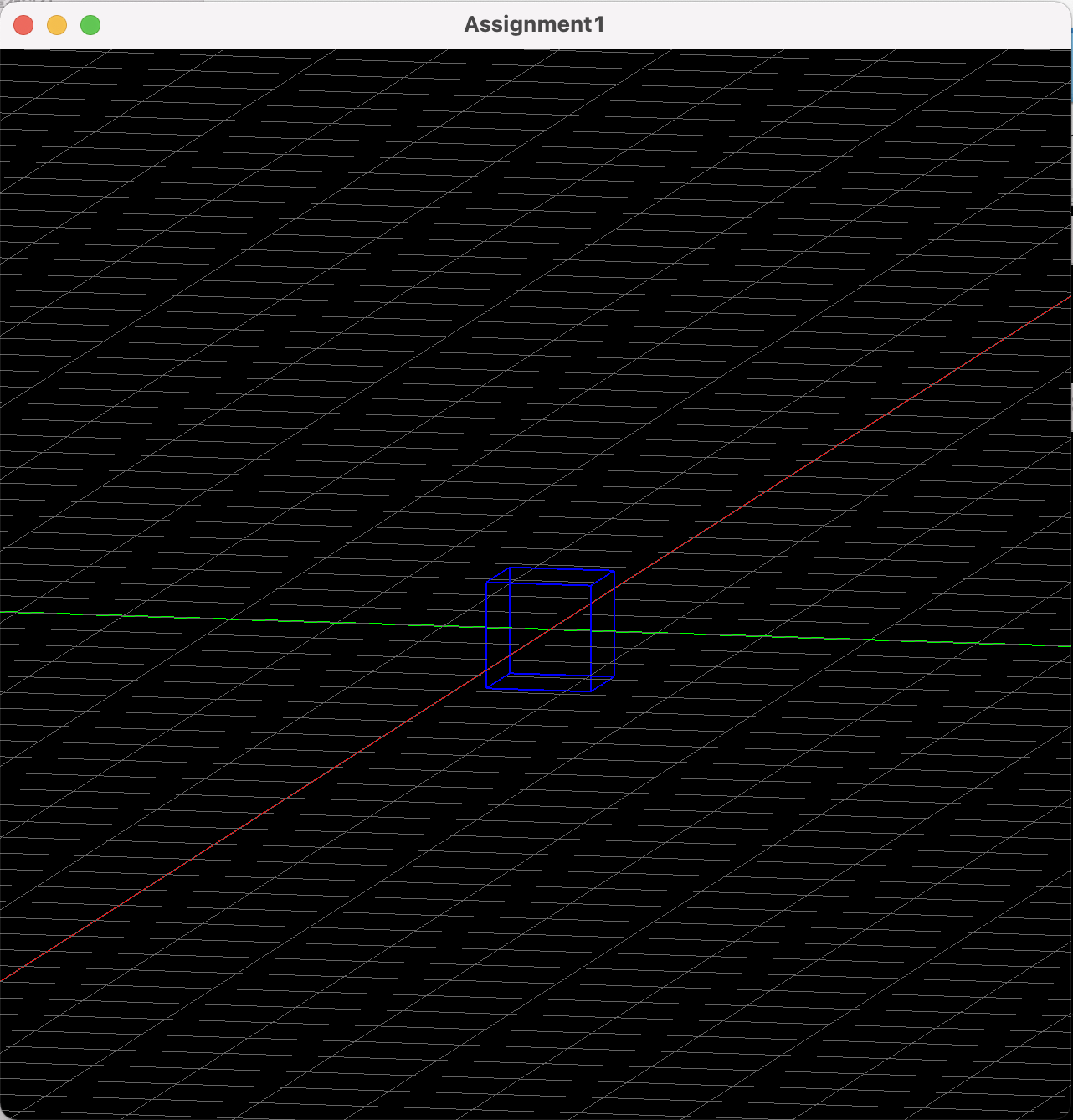
**Class Assignment 1 Report**

2017029589 류지범

1. Manipulate the camera with mouse movement
   1. 처음 카메라의 세팅은 타켓포인트는 (0,0,0), 거리는 10으로 설정했고, z축 선상에서 -z쪽을 보도록 했다.
   2. **Camera control operations**
      1. Orbit
         1. Orbit은 마우스 왼쪽 버튼을 누른 채로 드래그 하면 작동한다.
         2. 카메라를 회전시키지 않게 하기 위해, 물체를 구 궤적으로 translate 하도록 한 후 rotate 시킴으로써 마치 카메라가 타켓을 회전하는 것 처럼 보이도록 했다.
         3. 구 궤적은 glTranslatef(-gCamDistance \* np.cos(elevation) \* np.sin(azimuth), -gCamDistance \* np.sin(elevation), gCamDistance \* np.cos(elevation) \* np.cos(azimuth)) 으로 움직이게 했다.
         4. 마우스의 왼쪽 버튼을 누른 채로 드래그하면 elevation과 azimuth가 마우스 커서의 x, y 좌표 값을 기반으로 바뀐다.
      2. Panning
         1. Panning은 마우스 오른쪽 버튼을 누른 채로 드래그하면 작동한다.
         2. 카메라의 현재 위치와 타켓 포인트, up 벡터를 통해, w, u, v 벡터를 구한다.
         3. Orbit과 마찬가지고 마우스의 x, y 좌표 값을 기반으로 좌우로는 u 벡터 방향으로, 위아래로는 v벡터 방향으로 움직이도록 했다.
      3. Zooming
         1. Zoom은 distance라는 global variable을 통해 결정되고, 이는 구 궤적의 반지름이 된다.
         2. Scroll\_callback 함수를 통해 마우스 스크롤의 좌표를 반환하도록 했으며, 스크롤을 내리면 distance가 증가해서 zoom out 되고, 스크롤을 올리면 distance가 감소해서 zoom in 되도록 구현했다.
   3. **Toggle perspective projection / orthogonal projection**
      1. 프로그램이 실행 됐을 때는 perspective 상태로 볼 수 있도록 했고gluPerspective(45, 1, 1, 100)를 사용했다.
      2. Orthogonal 상태는 glOrtho(-5, 5, -5, 5, -100, 100)과 같이 보이도록 설정했다.
      3. v키를 누르면 projection 상태가 전환된다.
   4. **Draw a rectangular grid with lines on xz plane**
      1. Grid의 범위는 -50 ~ 50까지로 했으며, np.linspace(-50, 50, 99)를 통해 row와 column에 각각 99줄씩 나오도록 그었다.
      2. 도형과 겹쳐보이지 않게 하기 위해 선의 굵기는 0.2로 했으며 색상은 연한 회색으로 설정했다.

Perspective\_mode

Zoom-out

Panning

Orbit