Class Assignment 2 Report

2018008395

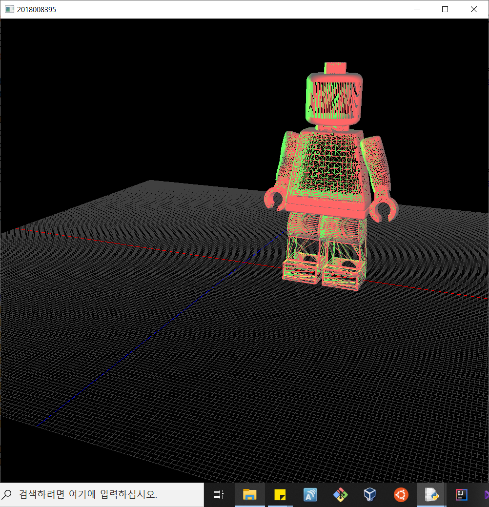
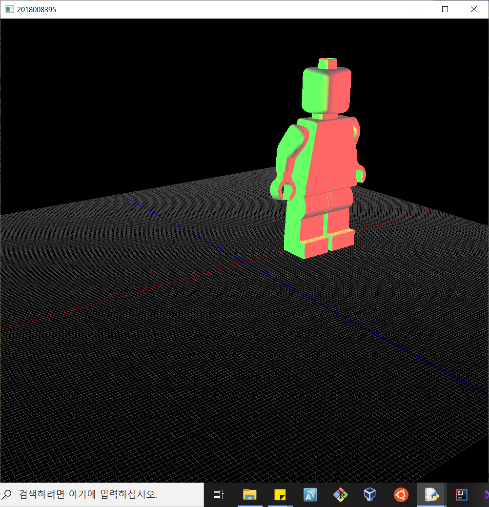
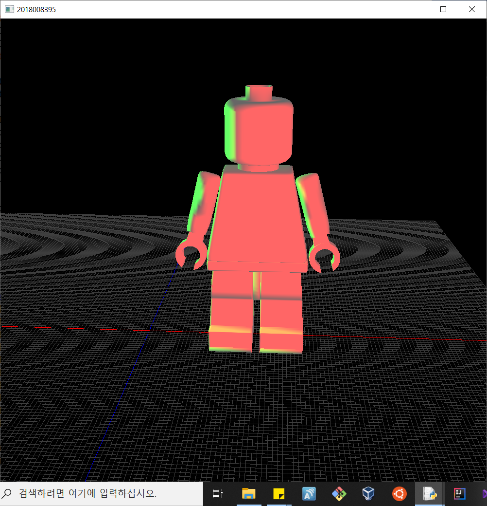
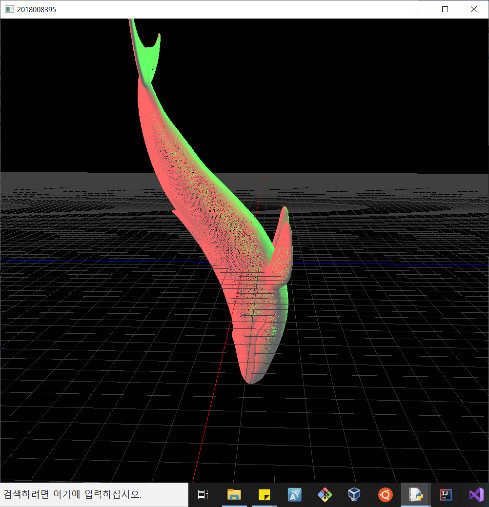
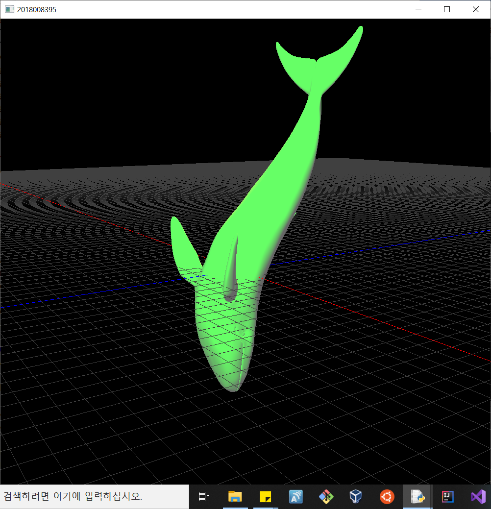
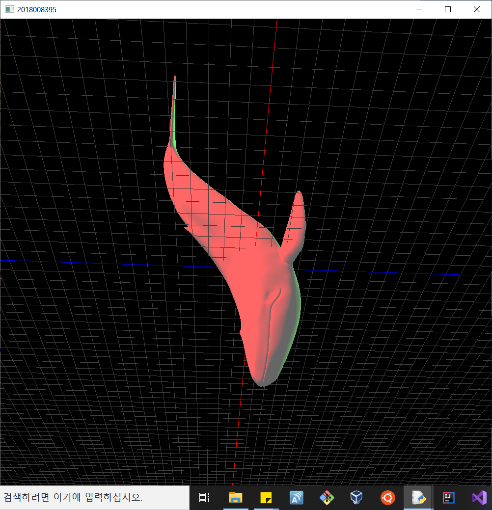
박정호

1. Implemented Requirement
   1. Basic OpenGL Viewer  
      지난 ClassAssignment1과 같다. 구현에 대한 자세한 설명은 생략한다.
   2. Rendering obj file  
      기본적으로 drop callback이 다수의 파일을 drag and drop하는 것을 전제로 하기에, 한번에 하나의 파일만 렌더링할 수 있도록 paths[0]만 파일 I/O의 대상으로 했다. 한줄 한줄 받아오면서, v tag는 vertex 정보를 저장하는 list에 저장하고, vn tag는 vertex normal 정보를 저장하는 list에 넣었다.  
      f 태그의 경우 그다음에 오는 parameter의 형식이 다양했는데 나는 a/\_/c의 경우, 즉 vertex normal이 제공되는 경우만 상정하고 parser를 구현했다. 위에서 저장한 vertex list와 vertex normal list에서 해당 값을 가져와서 실제로 drawArray에 쓸 varr에 저장했다. 또한 기본 제공 정보가 index이므로 이 단계에서 drawElements에서 사용할 iarr에 읽어온 index를 저장했다.  
      모든 파일을 다 읽고, varr과 iarr을 만들었으면, 여기에 담긴 정보를 이용해 drawElements에서 사용할 varr을 만들었다. 굳이 두가지 varr을 만드는 이유는 smooth shading과 flat shading을 위해서 두가지 normal vector의 정보가 필요했고, 이를 일일이 바꾸는 것보다는 처음에 두개를 만드는 게, toggling의 성능을 올리는데 더 좋다고 판단했기 때문이다.

Zooming 후

Panning 후

* 1. Lighting  
     object color는 흰색으로 설정했다. Light source를 복수로 설정해도 그리 티가 나지 않았기에, object color를 희게 설정하고, light color를 다양하게 설정해서 light source가 복수가 있는지 쉽게 확인되게 했다. Light source는 (5, 5, 5)와 (-5, 5, -5)에 각각 하니씩 설치했고, 둘다 점광원이다. Light color는 각각 Red, Green으로 설정했다.
  2. Extra Credit
     1. Toggle shading mode  
        우선 shading의 모드를 바꾸기 위해 obj를 읽어오면서 vertex array와 index array를 모두 만들어 두었으며, 전역변수인 smooth를 key callback에서 True/False로 매번 바꿔주면서 해당 값이 True면 index array를, False면 vertex array를 사용해서 서로 다른 normal vector를 쓰도록 했다.
     2. Triangulation algorithm  
        제공된 polygon이 삼각형이 아니어도 삼각형의 모임으로 그리기 위해 단순한 triangulation algorithm을 구상했다. 주어진 면의 첫번째 vertex를 에서부터 나머지 vertex로 모두 대각선을 그리게 되면 자연히 면이 삼각형들로 나뉘는데, 이 방식을 사용해서 첫번째 vertex를 저장하고 그다음부터 연속하는 두개의 vertex과 첫번째 vertex를 연결해서 삼각형을 만들어내는 방식을 사용했다.

1. Screenshot Image of Program
   1. Lego Figure Model  
      모델의 크기가 너무 커서 x, y, z 모두 0.1로 scaling한 이미지입니다.
   2. Blue Whale Model  
      모델의 크기가 너무 커서 x, y, z 모두 0.001로 scaling한 이미지입니다.