

TDT 4195 - Lab Assignment 2

Kari Lovise Ness (MTING)
Håvard Kjellmo Arnestad (MTFYMA)

September 2019

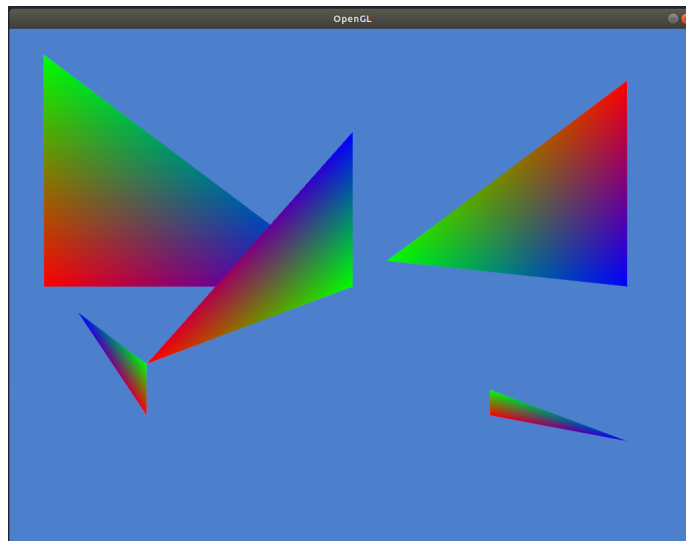
1 Repetition

a)

Not requested.

b)

5 triangles with different coordinates and colours in each vertex, as seen in Fig. 1 with vertex, colour, and index arrays in Fig. 2.



Figur 1: 5 triangles with different colours in each vertex.

```

//Vertex coordinates
std::vector<float> vertices = {
    -0.6f,-0.3f,0.0f,
    0.0f,-0.0f,0.0f,
    0.0f,0.6f,0.0f,

    0.8f,0.8f,0.0f,
    0.1f,0.1f,0.0f,
    0.8f,0.0f,0.0f,

    -0.9f,0.0f,0.0f,
    -0.9f,0.9f,0.0f,
    0.0f,0.0f,0.0f,

    0.4f,-0.5f,0.0f,
    0.4f,-0.4f,0.1f,
    0.8f,-0.6f,0.1f,

    -0.6f, -0.5f,0.3f,
    -0.6f, -0.3f,0.2f,
    -0.8f, -0.1f,0.2f
};

//Index buffer - indices
std::vector<unsigned int> indices = {
    0,1,2,
    3,4,5,
    6,8,7,
    9,11,10,
    12,13,14
};

//vector with colours
std::vector<float> colours = {
    1.0f,0.0f,0.0f,
    0.0f,1.0f,0.0f,
    0.0f,0.0f,1.0f,

    1.0f,0.0f,0.0f,
    0.0f,1.0f,0.0f,
    0.0f,0.0f,1.0f,

    1.0f,0.0f,0.0f,
    0.0f,1.0f,0.0f,
    0.0f,0.0f,1.0f,

    1.0f,0.0f,0.0f,
    0.0f,1.0f,0.0f,
    0.0f,0.0f,1.0f,

    1.0f,0.0f,0.0f,
    0.0f,1.0f,0.0f,
    0.0f,0.0f,1.0f
};

```

Figure 2: Vertex, colour, and index arrays of 5 triangles from Fig. 1.

2 Alpha Blending and Depth

a)

The 3 triangles can be seen in figure 3.

```

//Vertex coordinates
std::vector<float> vertices = {
    -0.6f,-0.6f,0.9f,
    0.6f,-0.6f,0.9f,
    0.0f,0.6f,0.9f,

    0.6f,0.6f,0.5f,
    -0.6f,0.6f,0.5f,
    0.0f,0.0f,0.5f,

    -1.0f,-0.3f,0.1f,
    0.7f,0.3f,0.1f,
    -0.6f,0.3f,0.1f
};

std::vector<float> colors = {
    1.0f, 0.0f, 0.0f, 0.5f,
    1.0f, 0.0f, 0.0f, 0.5f,
    1.0f, 0.0f, 0.0f, 0.5f,

    0.0f, 1.0f, 1.0f, 0.5f,
    0.0f, 1.0f, 1.0f, 0.5f,
    0.0f, 1.0f, 1.0f, 0.5f,

    1.0f, 1.0f, 1.0f, 0.5f,
    1.0f, 1.0f, 1.0f, 0.5f,
    1.0f, 1.0f, 1.0f, 0.5f
};

```

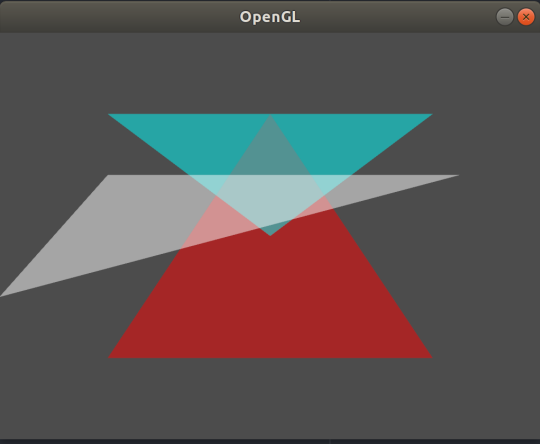


Figure 3: 3 overlapping triangles.

b)

i)

The 3 triangles after changing the VBO containing the colors can be seen in figure 4. **What effects on the blended colour did you observe, and how did exchanging triangle colours cause these changes to occur?** *The overlapping region becomes more red than blue. It happens because now the red triangle is drawn last. Because of the way the new pixel value is calculated the order in which colours are applied will matter: One minus alpha gives how much of the background should contribute in portion to the new color. This assumes the proposed `glBlendFunc` parameters.*

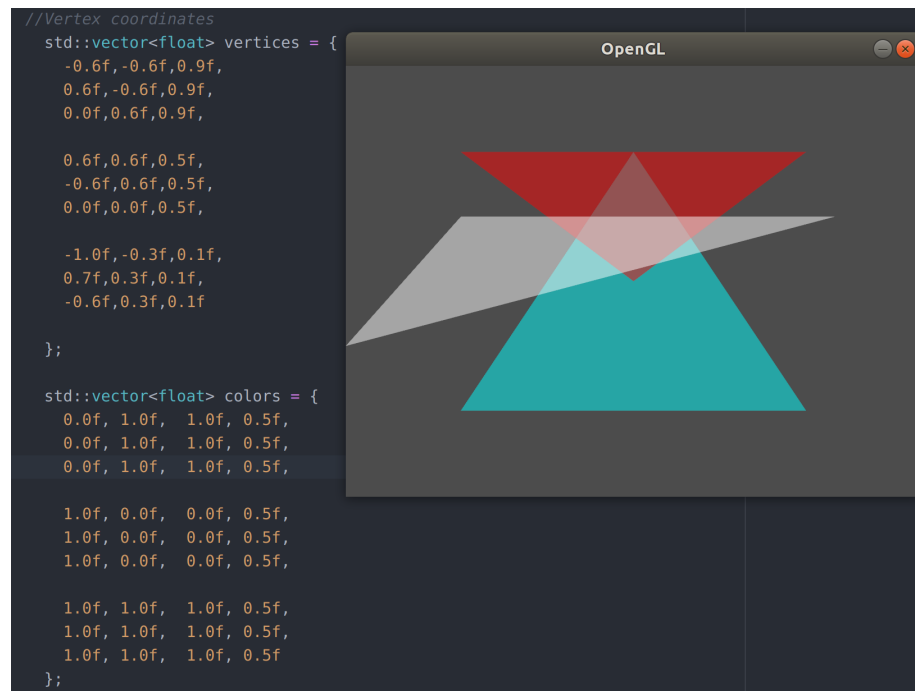
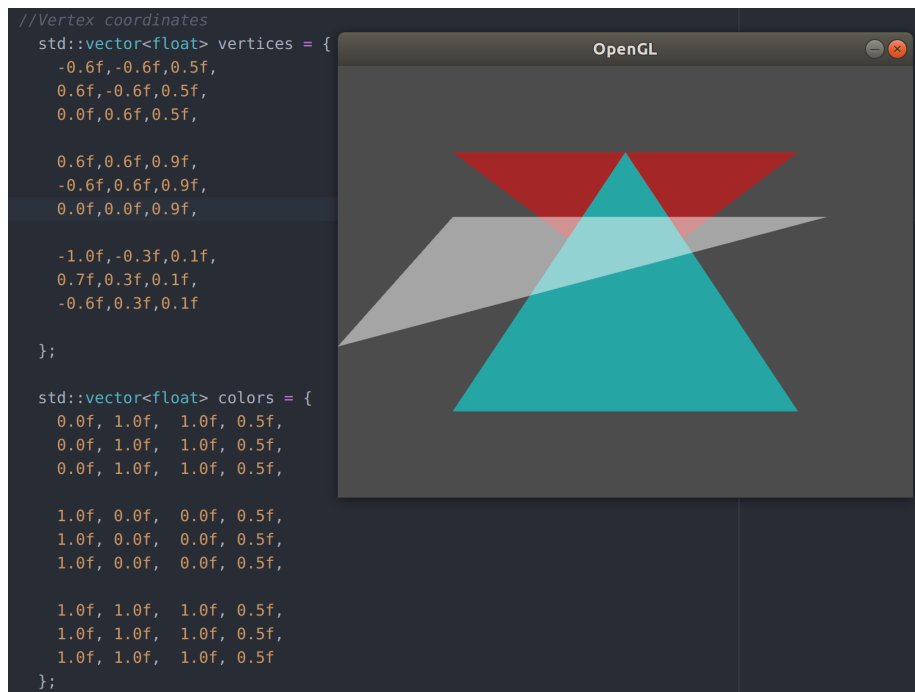


Figure 4: The colors of the top and bottom triangle have changed.

ii)

The 3 triangles after changing the drawing depth can be seen in figure 5.

Which changes in the blended colour did you observe, and how did the exchanging of z-coordinates cause these changes to occur? Why was the depth buffer the cause this effect? *The red triangle is no longer visible through the blue. That happens because the red triangle is now behind the blue and also drawn after the blue triangle. Since the new color is already in front of the red triangle, the red triangle is not included in the mix.*



Figur 5: The blue triangle no longer appears to be transparent because the red is drawn afterwards even though it is behind.

3 The Affine Transformation Matrix

a)

Not asked for.

b)

The four transformations discussed in the lectures are

- Translation
- Scaling
- Rotation
- Shearing

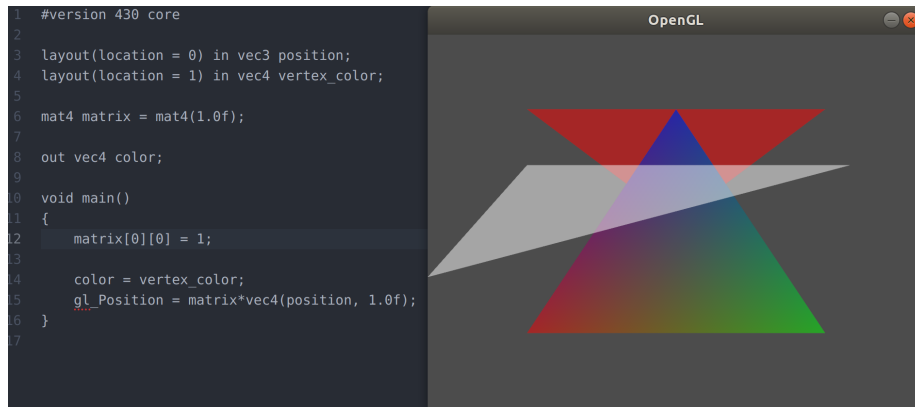


Figure 6: This is the original image, nothing has happened (identity matrix).

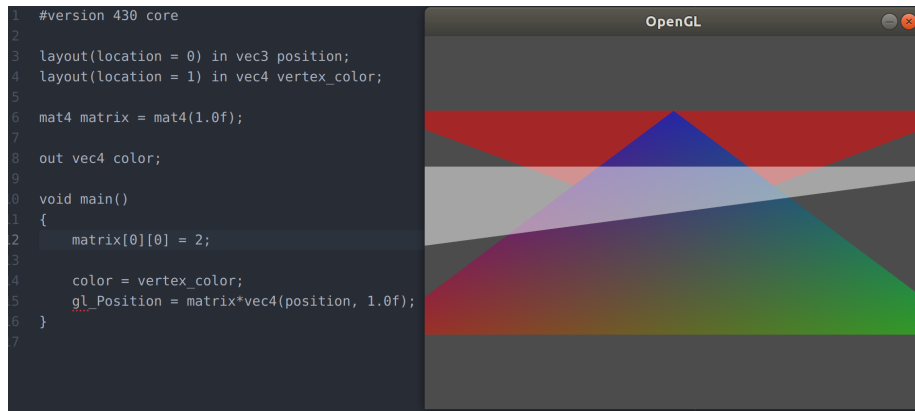


Figure 7: Here the 'a' element is set to 2, meaning that we have a scaling in the x direction by a factor 2.

4 Combinations of Transformations

Not strictly requested, but the keys are as follows (because of bonus challenge):

- Space - move forward
- Left Ctrl - move backward
- Arrow Up - move up (relative to camera)
- Arrow Down - move down (relative to camera)
- Arrow Left - move left (relative to camera)
- Arrow Right - move right (relative to camera)

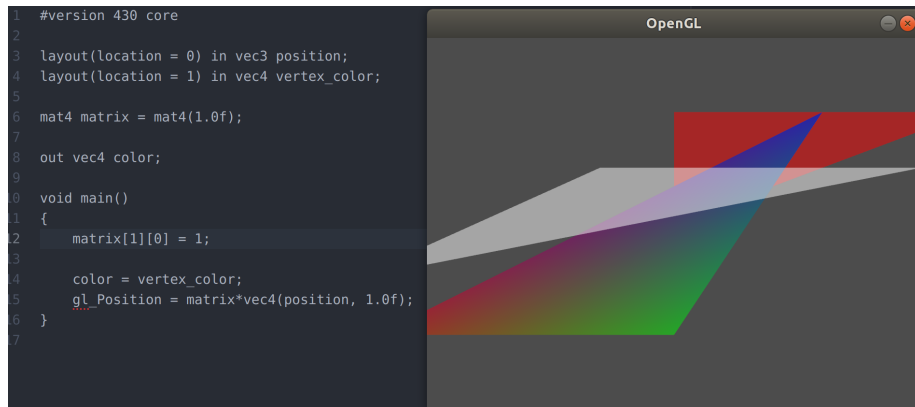


Figure 8: Here 'b' is set to 1, so that the x value now depends on y. This causes a shearing, as pixels higher up (large y value) gets moved to larger x values.

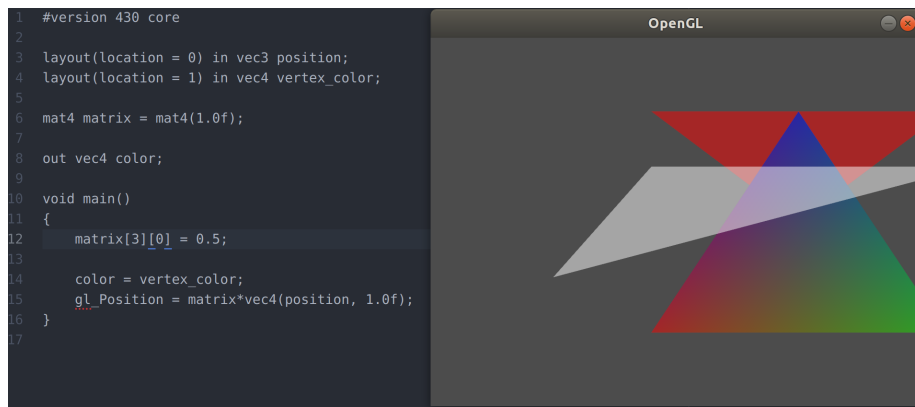


Figure 9: Here 'c' is set to 0.5, so the x values of the triangles are translated to the right.

- W - look up
- A - look left
- S - look down
- D - look right

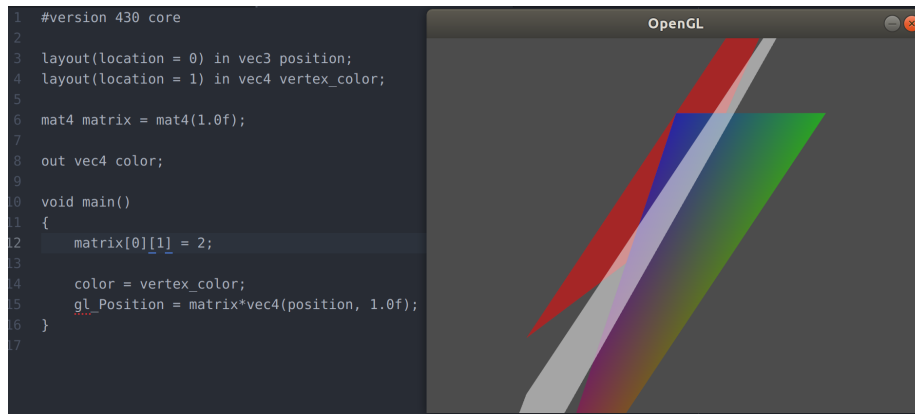


Figure 10: Here the y coordinate depends on x (because 'd' is changed), so we have a shearing effect again, but the direction is different.

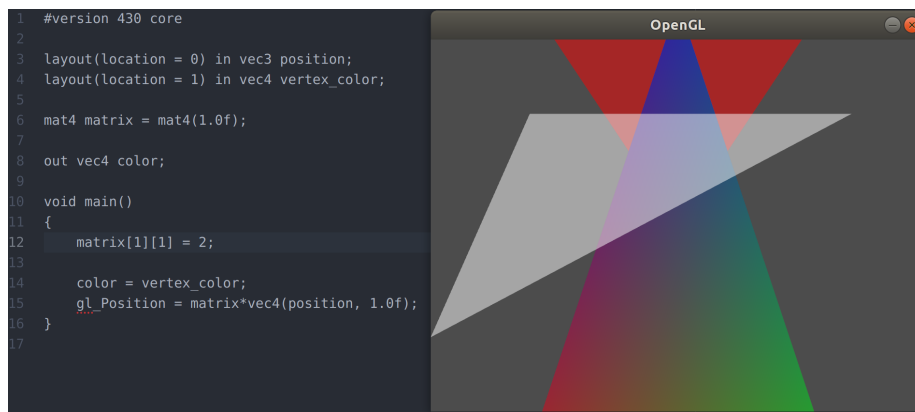


Figure 11: Now 'e' is set to 2, so y is scaled to the double.

5 Optional Bonus Challenges

5.1 5 a)

5.2 5 d)

- i) - This is possible: shear, rotation, translation.
- ii) - Not possible, this is not a simple transformation.
- iii) - Hmm... Not possible as a shear, but it is pretty much like a projection seen from above, so it could be possible by altering the matrix in some way. It requires that parallel vertical lines will meet in a singularity though, and as far as I have been thinking that is not so easy to get with a single

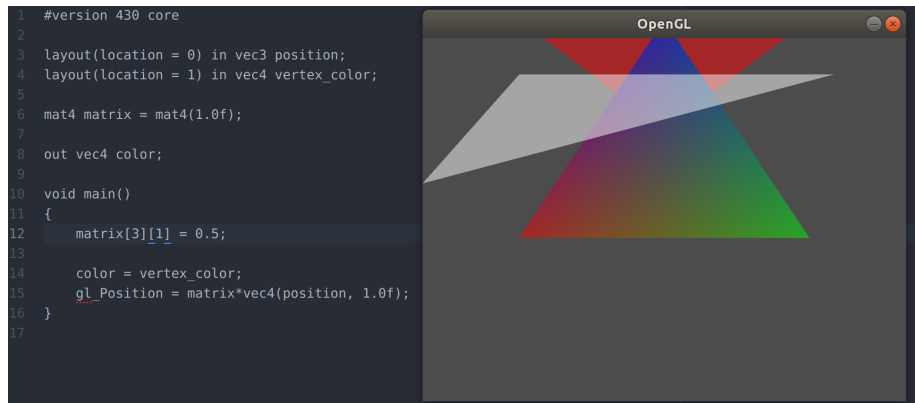


Figure 12: This is a translation by 0.5 in the positive y direction, obtained by changing the 'f' value.



Figure 13: The directions of movement is now altered by the direction we look in. Coming from physics, the formulas were 'guessed upon' based on experience with working in spherical coordinates. The signs were adjusted to give the most natural movement.

matrix.

- iv) - Not possible, it is a translation of only one of the objects.
- v) - Not possible. x goes to x, but y goes to $y*x*x$ (or something similar but non-linear), so the matrix elements have to be variables and can thus not be a single matrix".