TDT4195 - Visual Computing Fundamentals - Assignment 2

Group 119
Martin Eek Gerhardsen

September 20th, 2019

Department of Engineering Cybernetics

Contents

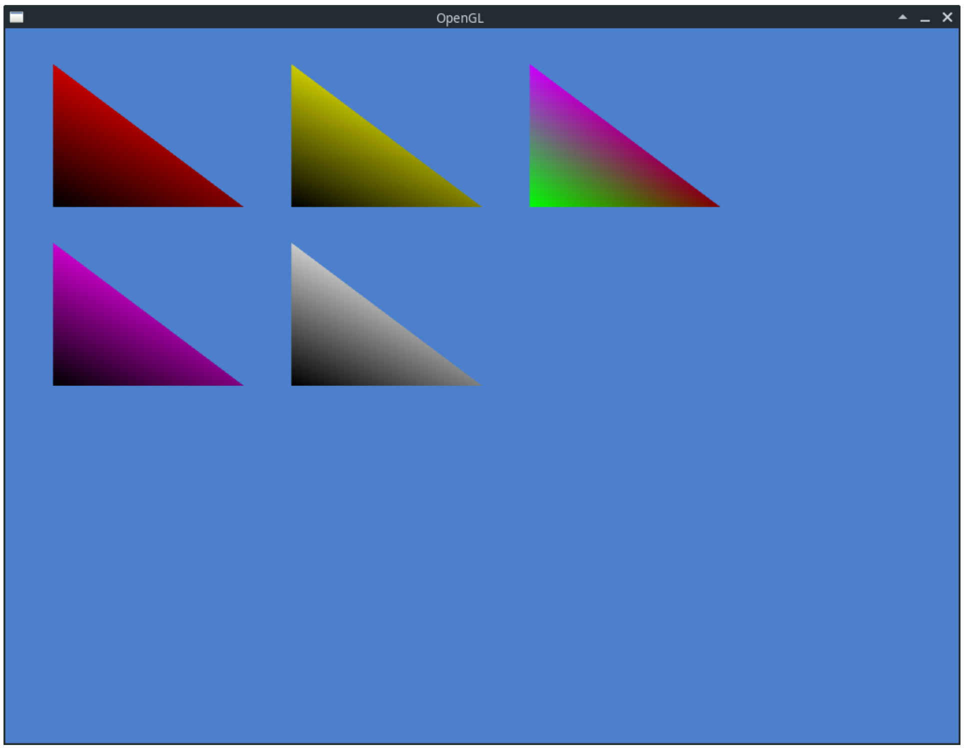

Figure 1: Five triangles with different coloured vertices.

1 Task 1

1.1 b)

See the figure for the result:

The values used were the following:

```
std::vector<float> triangleVertices =
{
     -0.9,  0.5, 0.0,
     -0.5,  0.5, 0.0,
     -0.9,  0.9, 0.0,

     -0.9 + 0.5,  0.5, 0.0,
     -0.5 + 0.5,  0.5, 0.0,
     -0.9 + 0.5,  0.9, 0.0,

     -0.9,  0.5 - 0.5, 0.0,
     -0.5,  0.5 - 0.5, 0.0,
     -0.9,  0.9 - 0.5, 0.0,

     -0.9 + 0.5,  0.5 - 0.5, 0.0,
     -0.5 + 0.5,  0.5 - 0.5, 0.0,
     -0.9 + 0.5,  0.9 - 0.5, 0.0,

     -0.9 + 1.0,  0.5, 0.0,
     -0.5 + 1.0,  0.5, 0.0,
     -0.9 + 1.0,  0.9, 0.0,
};

std::vector<float> triangleColours =
{
     0.0, 0.0, 0.0, 1.0,
     0.0 + 0.5, 0.0, 0.0, 1.0,
     0.0 + 0.8, 0.0, 0.0, 1.0,

     0.0, 0.0, 0.0, 1.0,
     0.0 + 0.5, 0.0 + 0.5, 0.0, 1.0,
     0.0 + 0.8, 0.0 + 0.8, 0.0, 1.0,

     0.0, 0.0, 0.0, 1.0,
     0.0 + 0.5, 0.0, 0.0 + 0.5, 1.0,
     0.0 + 0.8, 0.0, 0.0 + 0.8, 1.0,

     0.0, 0.0, 0.0, 1.0,
     0.0 + 0.5, 0.0 + 0.5, 0.0 + 0.5, 1.0,
     0.0 + 0.8, 0.0 + 0.8, 0.0 + 0.8, 1.0,

     0.0, 1.0, 0.0, 1.0,
    0.0 + 0.5, 0.0, 0.0, 1.0,
    0.0 + 0.8, 0.0, 1.0, 1.0,
};

std::vector<unsigned int> triangleIndices
{
    0, 1, 2,
    3, 4, 5,
    6, 7, 8,
    9, 10, 11,
    12, 13, 14
};
```

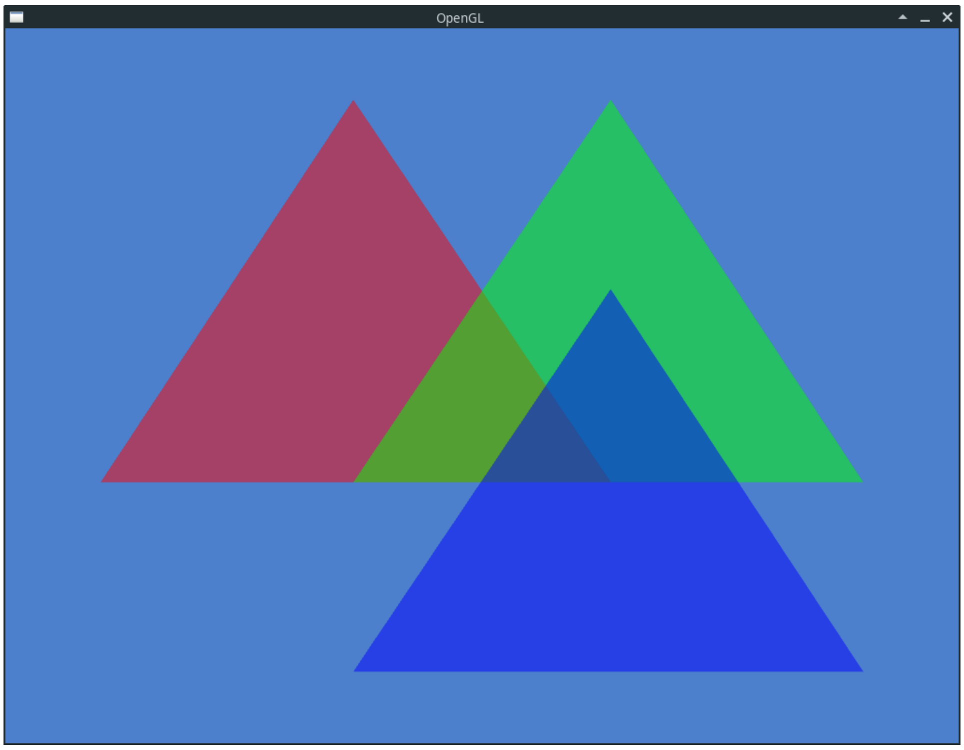

Figure 2: Three triangles, intersecting at one area. Hex: #294F99

2 Task 2

2.1 a)

Here is the resulting figure. The colours are all simple, pure red, green and blue for the three triangles. Furthermore, all their alphas are 0.5. As it's interesting, we can note that the colour value of the intersection of all the triangles have the RGB hex-value #294F99.

2.2 b)

2.2.1 i)

When alternating the arrangement of the colours, we can see that the colour of the intersection of all the triangles is not the same. Reading out the RGB hex-values, we can further note that the colour is off.

What we are seeing here is the effect of eq. 1 in the assignment task. When a traingle of one colour is behind another (both with with alphas less than 1, greater than 0), the new colour is a mixture of the old. If we expand this equation for the several layers of colours we have in this task, we may note that the more colours are on top of each other, the more *washed out*

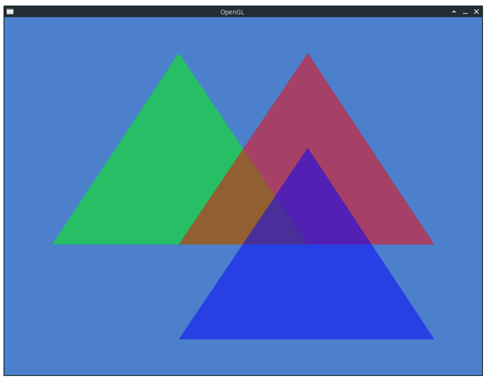

Figure 3: One alternative arrangement of colours. Hex: #492F99

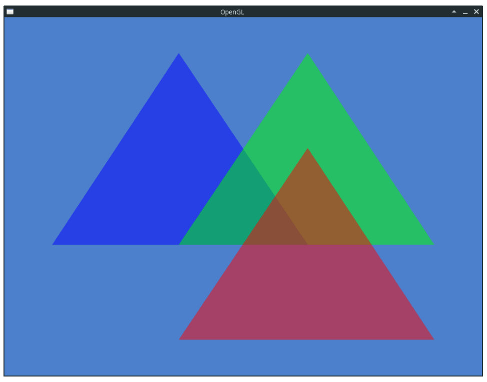

Figure 4: Another alternative arrangement of colours. Hex: #894F39

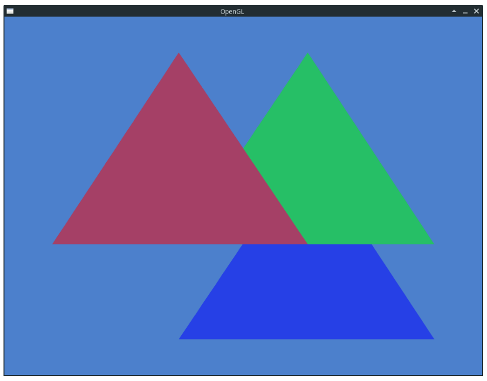

Figure 5: Z-coordinates reversed

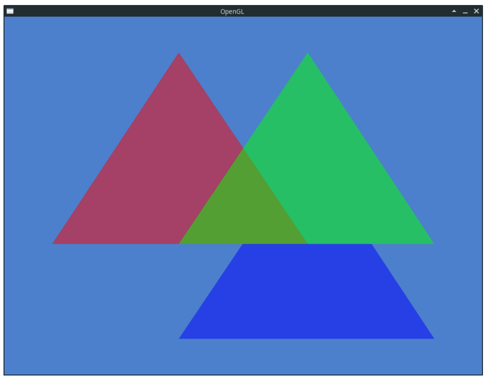

Figure 6: Not all z-coordinates reversed

the colour furthest back will appear, as it will be multiplied several times by the transparency value for the different colours.

2.2.2 ii)

Reverting the colour arrangement to the same as fig. 2.

When generating the new colour, the idea is that we only check for what is behind the current index in the index buffer. Then, when something which is *supposed* to be behind an object, but is further behind in the index buffer, it will not be mixed with the front object and we see the effect shown in fig. 5 and fig. 6.

3 Task 3

3.1 b)

Changing all values in the range $[-0.5, 0.5]$.

- a is a scaling in the x direction.
- b is a shear on the x-axis.
- c is a translation in the x-axis.
- d is similar to b and is a shear on the y-axis.
- e is similar to a and is a scaling in the y direction.
- f is similar to c and is a translation of the y-axis.

We can note that there are no pure rotations in these examples, as the rotation matrix requires more than one element to change. I.e., rotation about the z-axis requires $-b = d = \sin\theta$ and $a = e = \cos\theta$, which we never can achieve by changing only one of these variables at a time.

For fun, her are all the changes applied after each other

4 Task 4

Key map (relative for the camera):

- W: forwards
- S: backwards
- A: left
- D: right
- Q: upward
- E: downward
- U: pan left (yaw)
- O: pan right (yaw)
- I: pan up (pitch)
- K: pan down (pitch)
- J: rotate left (roll)
- L: rotate right (roll)

J and L (roll) added for fun.