DH2323 Lab 1

Ted Klein Bergman tedber@kth.se
Sruti Bhattacharjee srutib@kth.se
Elias Alkvist Cetin ecetin@kth.se

Introduction

First we met on Zoom to talk about how to set up the lab, environment and process strategy, to then try to do the lab partially individually so each member got a grasp on the theory and the code. Later on we synchronized our work and helped each other to fix bugs and discuss theory. We also coded a bit using some mob programming to finalize the lab.

To complete the lab we started by just pasting in all given files and code into a project.

Building the skeleton proved some difficulty, needing manual configuration on Windows like changing how the C-runtime library was linked. On MacOS it went smoothly after downloading SDL through homebrew, although the library had problem generating a graphics context for Cocoa.

Method

Rainbow

In our initial implementation of the Interpolate function we modified the input vector's elements, that also were vectors, by index (e.g. result[i].x =a.x + stepX * i;) and calculating the step between the pixels. So that the next pixel got the previous pixel's value plus the stepvalue, in each direction. To get the value of each step we calculated the difference of the input vectors b and a, (b - a) and divided by the result.size()-1. We later abandoned this method.

Instead to implement the interpolation for the rainbow, we used the formula for linear interpolation (lerp) from the previous course, which states `start * (t - 1) + stop * t` where t is time.

Starfield

As previously mentioned we pasted in the code that was given in the instructions and modified it accordingly to conform with the given formulas.

We transformed the 3D points with coordinates (x, y, z) to 2D with coordinates (u, v) to get the projection of them on the screen. This was done to every star to later implement motion on each of them.

We used the following formulas for the projection:

$$u^i = f\frac{x^i}{z^i} + \frac{W}{2}$$

$$v^i = f\frac{y^i}{z^i} + \frac{W}{2}$$

Then to get the stars moving we implemented an update function with shifting z-values to create the motion.

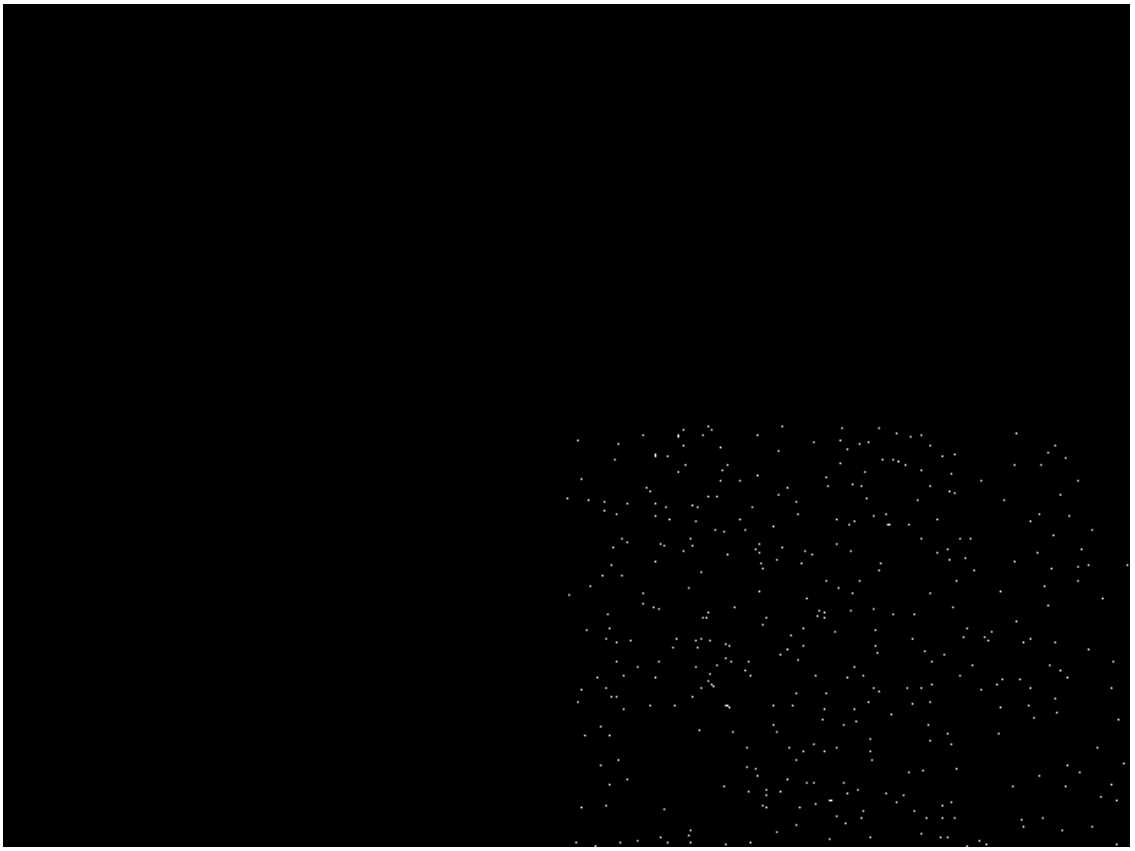
Errors and Challenges

Rainbow

When having the initial method of the interpolation function we made the mistake of first calculating the glm::length of the step vector, for some reason, and received the output shown in the picture below.

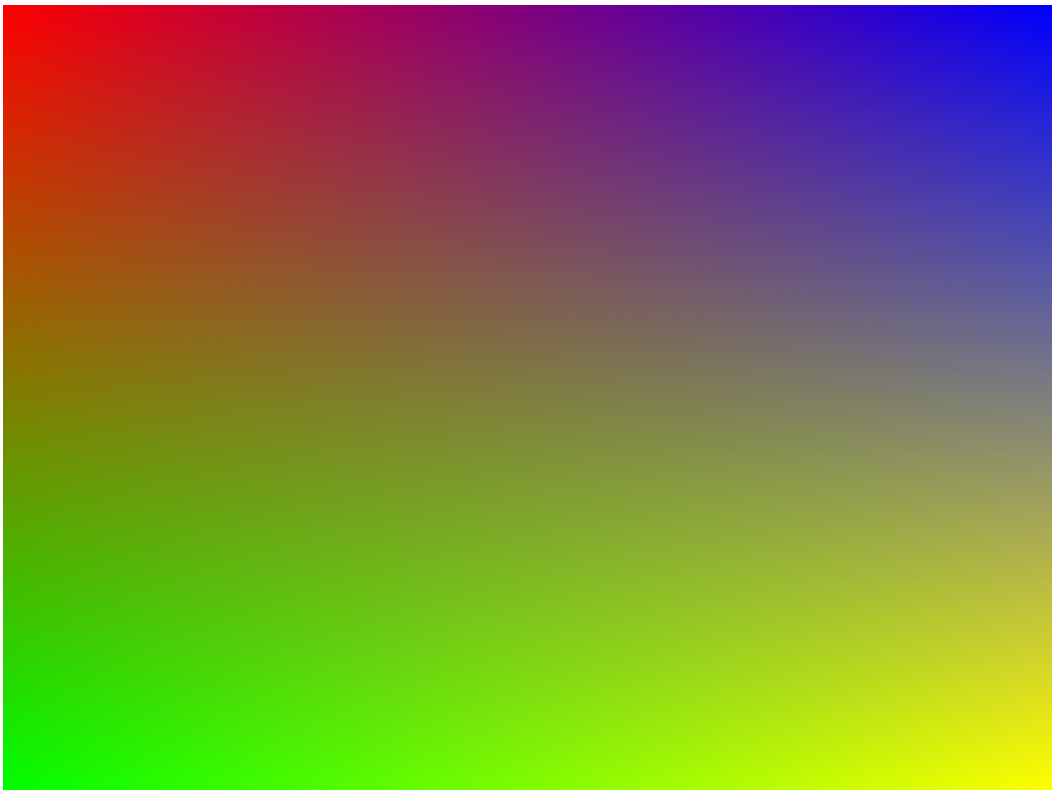
Starfield

Initially, we forgot to generate random x and y coordinates in the range -1 to 1, and instead generated coordinates between 0 to 1 (as the z coordinate should be). This was easily fixed by remapping the random value by multiplying by 2 and subtracting 1.



Final Results

Rainbow



Starfield

Source code

https://github.com/Naxaes/DH2323-Computer-Graphics