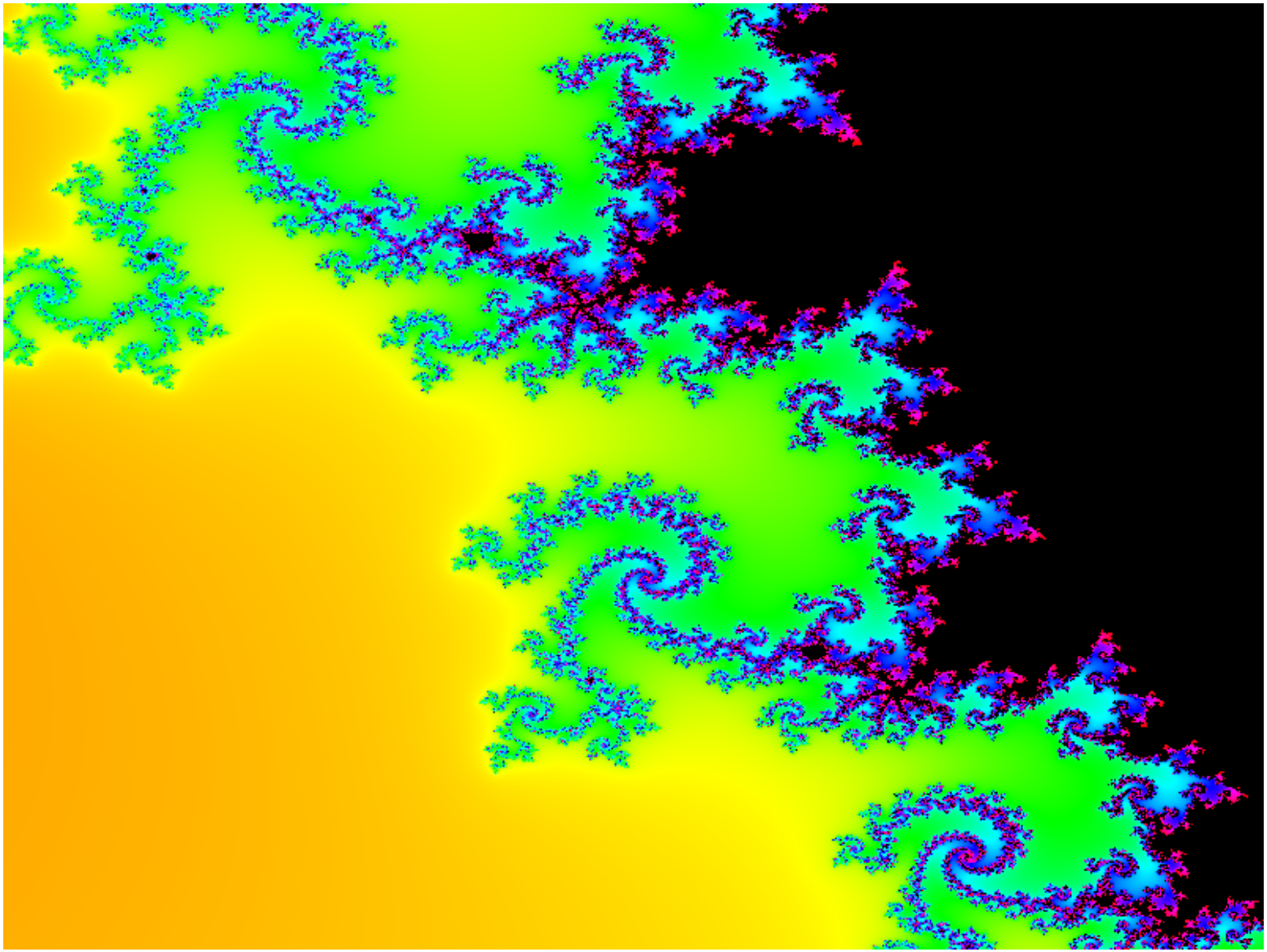
Exercises - Programming

Mandelbrot set computed and visualized on GPU



Exercises - Programming

Mandelbrot set computed and visualized on GPU

Definition: $P_c : z \rightarrow z^2 + c$
For all complex numbers c, for which z does not tend towards infinity.
- If the absolute value of z is larger than 2, we tend towards infinity.

How to compute:

1. Use pixel coordinate as complex c
2. Set $z_0 = 0$
3. Compute $z_{n+1} = {z_n}^2 + c$
4. If $|z_n| > 2$ for any n, abort
5. If $n > N$, where N is the max number of iterations, abort.

GLSL does not support a complex datatype, use a 2D vector type instead. We compute the absolute value of a complex number the same way as we compute the magnitude of a 2D vector, so we can use the length() function in GLSL.

Implement computing and visualizing of the mandelbrot set using the fragment shader, you can start from `ex15/mandelbrot.frag.template`

Exercises - Programming

Mandelbrot set computed and visualized on GPU

```
//Initialize z to be c, the complex coordinate
complex z = c;

//Iterate until the length of z is larger than
//two, or until we have reached max_iterations
while (|z| < 2 && i < max_iterations) {
  //Update the value of z according to the
  //formula
  z = z^2 + c
  ++i;
}

if (|z| < 2) {
  out_color = 1; //Assumed to be part of the set
}
else {
  out_color = 0; //Not part of the set
}
```

Exercises - Programming

Mandelbrot set computed and visualized on GPU

Black and white is dull, we can use linear interpolation to visualize the fractal with colors

```
if (i < max_iterations) {
  float t = (i-log(log(|z|)/log(2))/log(2)) / max_iterations;
  out_color = mix(red,green,t); //Assumed to be part of the set
}
else {
  out_color = 0; //Not part of the set
}
```

Exercises - Programming

Mandelbrot set computed and visualized on GPU

Interpolating over the RGB color space doesn't produce very interesting results, instead we can interpolate over the HSV color space, then convert it to RGB.
http://en.wikipedia.org/wiki/HSL_and_HSV

Try to interpolate between $[0, 1, 1]$ and $[2\pi, 1, 1]$ in HSV, then convert it to RGB