Name: Nicholas Ung
NUID: 002336960

CS 5310
Summer 2024

Lab 2 Report: Fractals

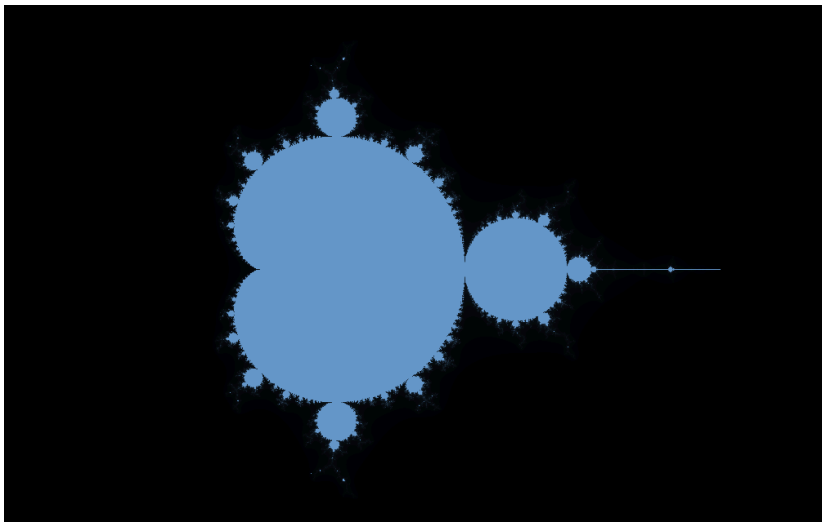
Summary

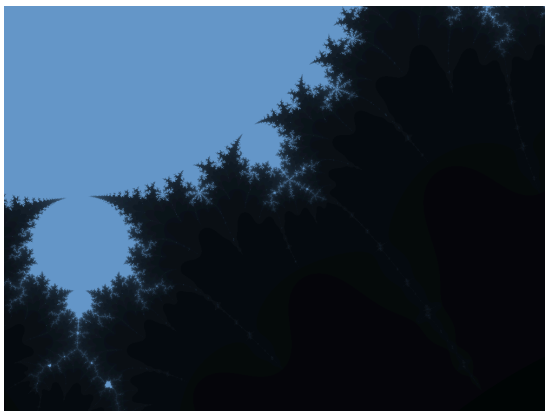
The purpose of this project was to develop a series of fractal images by creating and utilizing data structures such as ***Image*** and ***FPixel***. Tasks included generating Mandelbrot and Julia sets, in addition to implementing fractal noise (Perlin noise). The assignment aimed to test and demonstrate the functionality of these basic classes and their use in creating visually interesting fractal images.

Required Images

1. ***Mandelbrot Set***

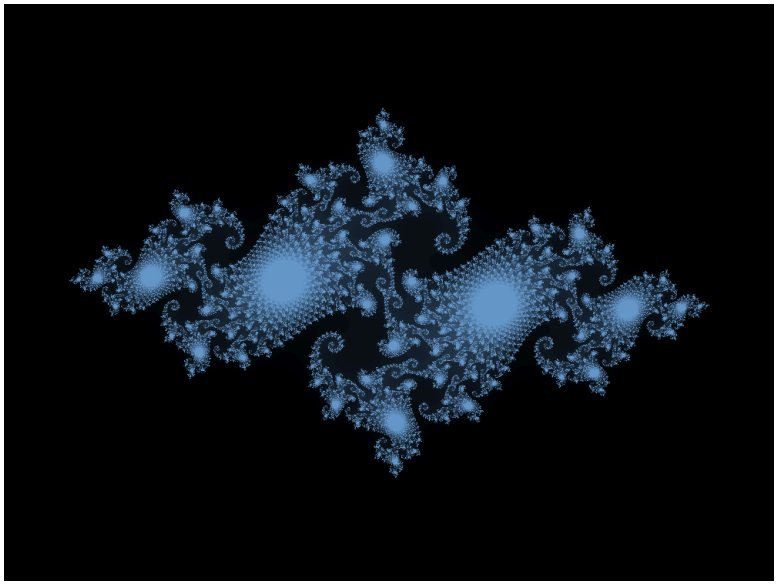
   Description: The Mandelbrot set image was generated using the function ***mandelbrot***, which iterates over each point in the complex plane to determine if it belongs to the Mandelbrot set. I generated this image by calling the function with the parameters x0 = -1.5, y0 = -1.5, and dx = 4.0 to define the rectangle in the complex plane to be visualized. My code was written to color points within the set as sky blue.





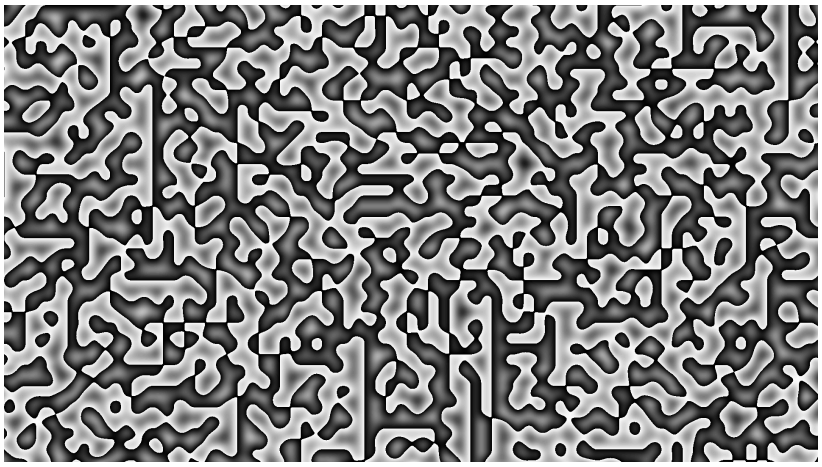
2. *Julia Set*

Description: The second task involved creating a Julia set image using the function ***julia***, which iterates over each point in the complex plane using the complex parameter *c = 0.7454054 + 0.1130063i*. I generated this image by calling the function with the parameters x0 = -1.5, y0 = -1.5, and dx = 3.0 to define the rectangle in the complex plane to be visualized.



3. *Fractal Noise*

Description: The third task was to create a fractal noise texture and use it to generate a procedural noise image. I implemented a function to model Perlin noise, which takes in parameters for scale and a seed, then fills an image with continuous gradient noise, creating a realistic texture for use in effects like fog, fire, or dust. The noise values were calculated with linear interpolation and gradient vectors that helped build a natural looking texture by adding randomness and smoothness.

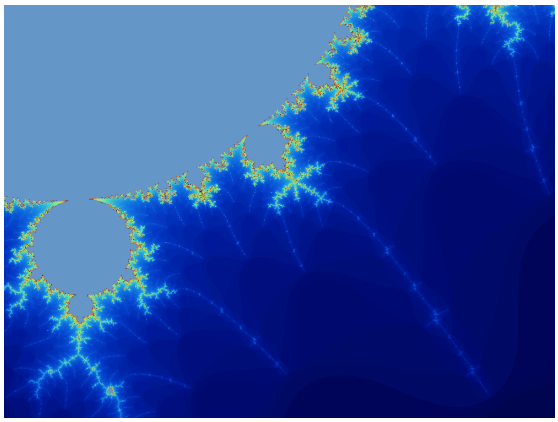


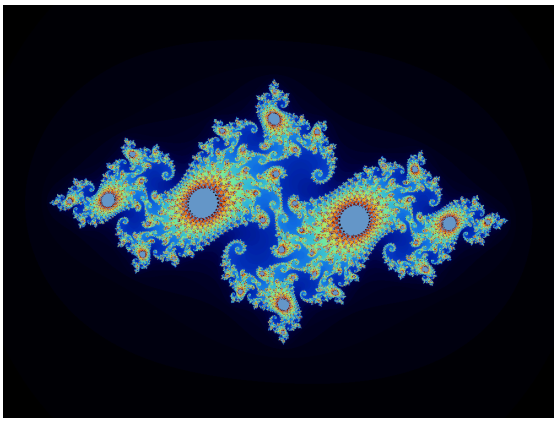
Name: Nicholas Ung
NUID: 002336960

Extensions

1. *Coloring Mandelbrot & Julia Sets*

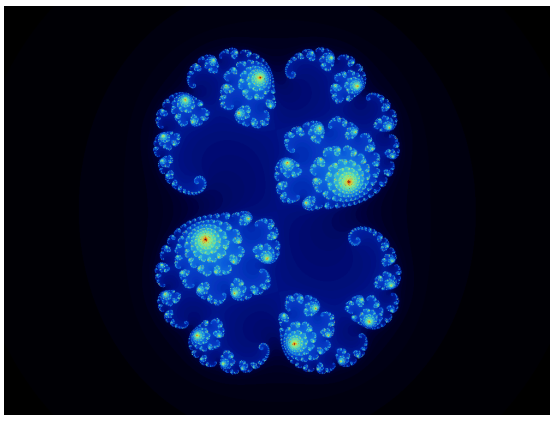
Description: I modified the helper function ***getColor*** to map the number of iterations in the set to an RGB color rather than pixel intensity, in order to produce some dynamic coloring for the Mandelbrot and Julia set methods. The result is a more vibrant image that pops more than the blue/black images that were previously generated.

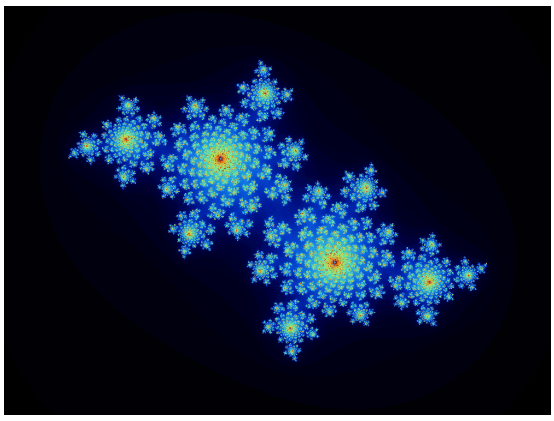




2. *Trying More Julia Sets* + Coloring

Description: My second extension included experimenting with different constants in Julia sets while including the coloring from my first extension. To produce the following images, I used $c = 0.285 + 0.01i$ and $c = -0.4 + 0.6i$, respectfully. I found it interesting how different constants can affect the disconnectedness and chaos of the fractal structure.



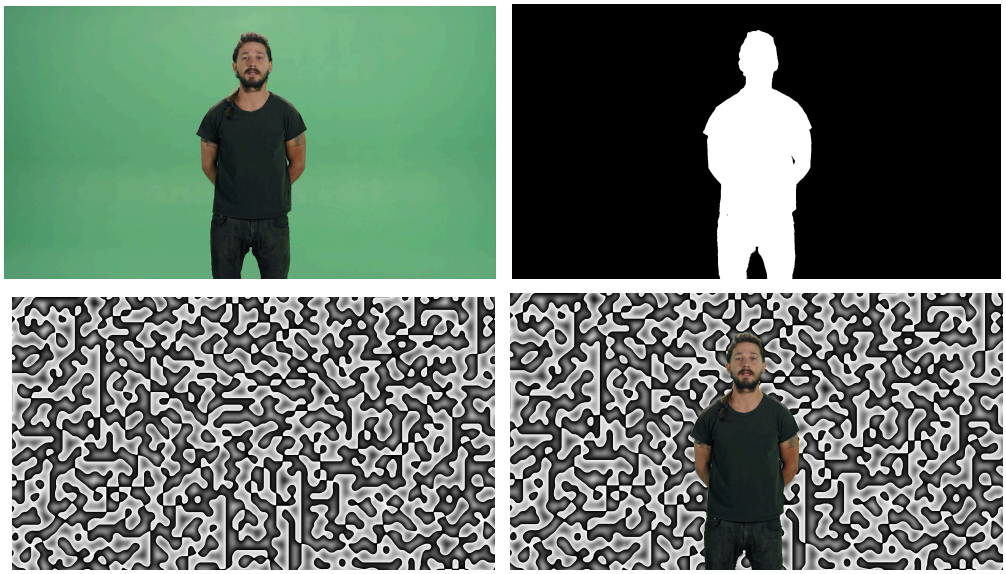


Name: Nicholas Ung
NUID: 002336960

Portfolio Images

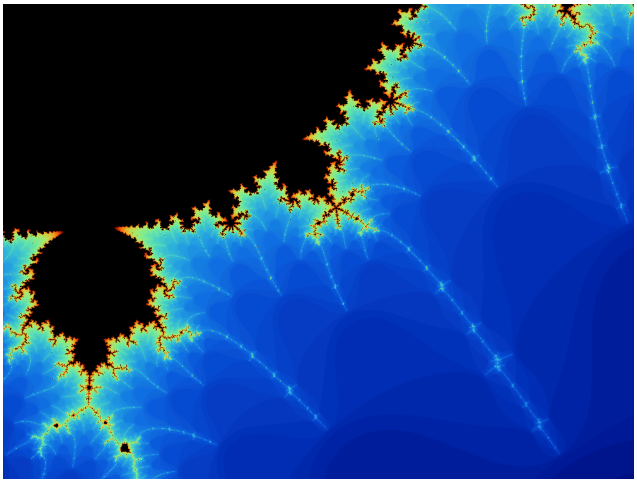
1. ***Perlin Noise and Composited Image (Extension 3)***

   Description: My first portfolio image was also my third extension. I combined my compositing technique from Lab 1 with the Perlin noise image from Lab 2 to produce an image of actor Shia LaBeouf in front of a static screen.



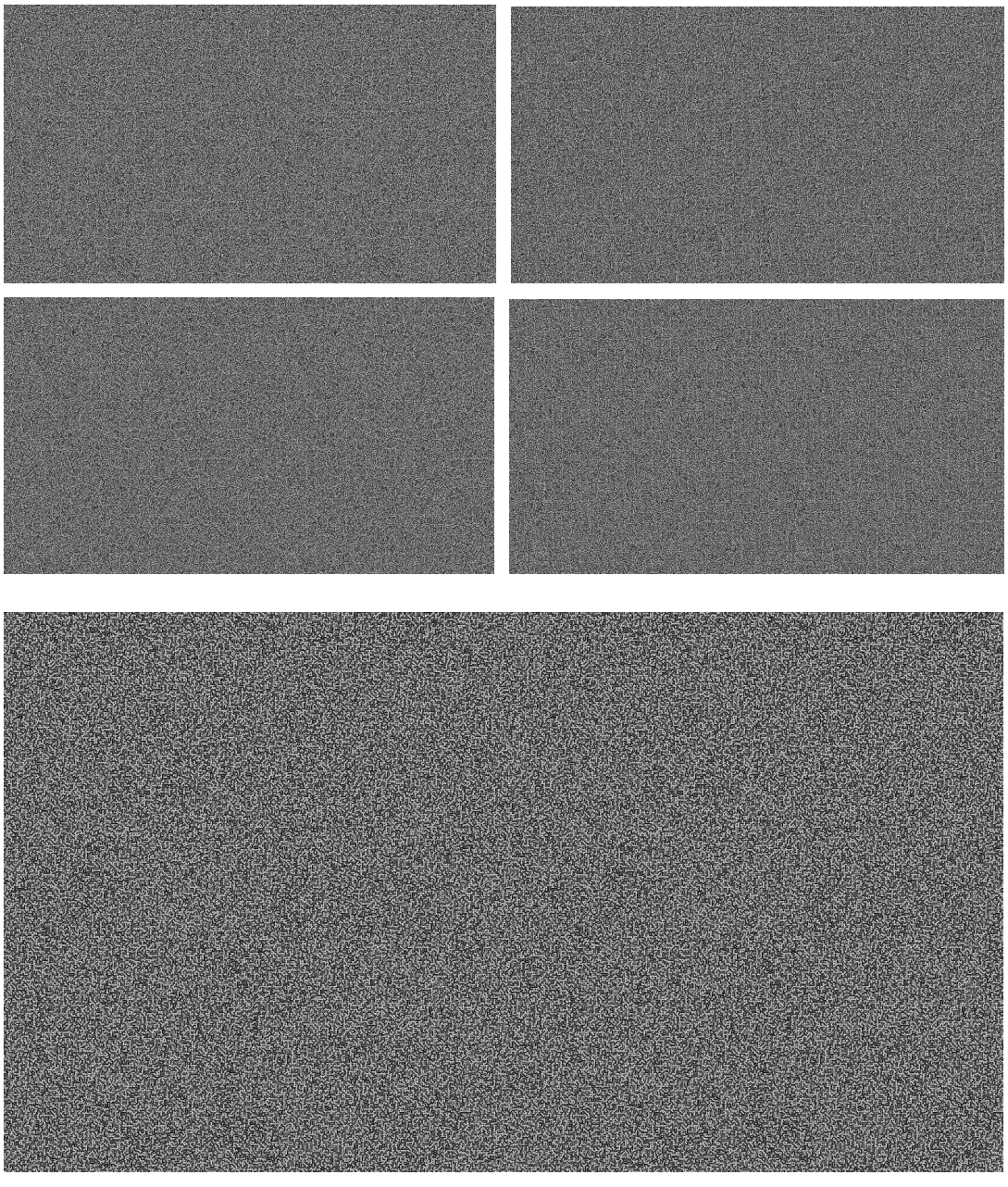
2. ***Colored Mandelbrot Set***

   Description: I used my extension for coloring Mandelbrot sets and made it so the points within the set were colored black in addition to cutting the maximum iterations of the algorithm in half from 256 to 128. Reducing the iterations smooths the rough edges a bit. The up close image demonstrates the beauty of coloring a Mandelbrot Set and presents a striking scene.



3. Perlin Noise

Description: I entered a smaller scale factor and different seed values to produce four unique patterns of Perlin noise. This noise is much finer compared to the first picture I created. I then compiled all of the images into a GIF to produce something reminiscent of TV static.



Name: Nicholas Ung
NUID: 002336960

Reflection

This project helped better understand the intricacies of fractal generation and procedural textures. I learned how to implement and use basic data structures in C, as well as how to create visually appealing images with mathematical functions and color mapping. The assignment also highlighted the importance of modularity and code organization in managing complexity.

Acknowledgements

I would like to thank my peers (Jiafeng Du) for their valuable insights and online resources such as W3schools for providing useful material for improving my C programming. This article on github was also helpful in learning how to write a program for Perlin noise. I consulted the lecture notes and recordings from Professor Maxwell to understand the techniques for solving all the problems in this assignment.