3D Mandelbrot Set Ray Tracer

Charles Underhill

This project requires no additional hardware or software other than the standard Raspberry Pi setup for Physics 129L.

The project is a program that computes the Mandelbrot Set in a user specified region and renders a 3D image of the region with a ray tracing algorithm. The program starts by asking the user to enter in the coordinate range and a number of points along the x and y (real and imaginary) axes to compute the Mandelbrot set. The program requires that the same amount of points be computed along both axes so the user is only prompted once for N points in both x and y directions.

If the user inputs the previous values correctly, they will be prompted for a view angle and a light height. The view angle is the angle from the x-y plane that the screen which holds the projected image will be situated at. The angle must be in between 0 and 90 degrees. The light height is the height above the x-y plane where the light is positioned in the scene. Any value can be input here but something between 0 and 2 is recommended for the scene to be visible.

From here the program computes the Mandelbrot Set and stores the stability of each point in the range as a number which will be interpreted as a height above ground. Using this array the program computes surface normals with an approximate formula in order to calculate the trajectory of light rays off of the surface. The function below can then be used to calculate light intensities.

$$I(\vec{p}) = \begin{cases} I_a & \cos(\theta) < 0 \\ I_a + f_d \cos(\theta) + f_s * (\cos(\alpha))^b & \cos(\theta) \geq 0 \end{cases}$$

The function holds weights for ambient, diffuse, specular, and shiny components of light. The parameter θ refers to the reflection angle at each point on the surface and α is the angle that the surface normal makes with the view vector, which is the vector from surface environment to the screen. These intensities are then projected onto the screen using the floating horizon method and the following function:

$$P(y, z) = \text{int}((N-1)(y\sin(\phi) + z\cos(\phi))$$

The floating horizon method involves projecting one vertical line at a time and keeping track of the maximum y value projected. Any points projected above the horizon are considered not visible by the light and are set to black. The pixel values on the screen are saved in an array and are plotted with a colormap using PyPlot.

Here are some interesting results from using this program. To get these results the user simply has to input the coordinate ranges, view angle, and light heights specified by the figure captions.

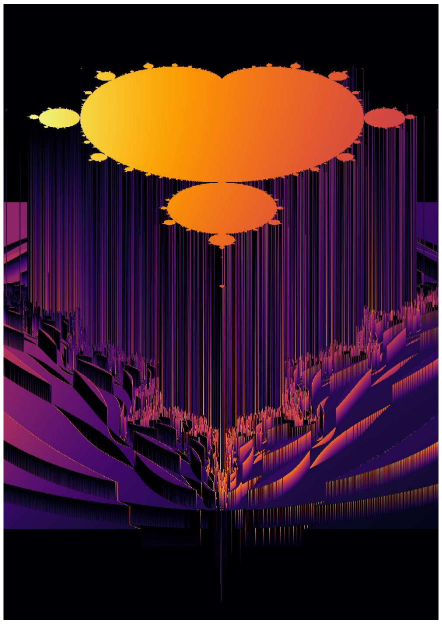

Figure 1: The full Mandelbrot Set using coordinates x0 = -2, x1 = 1, y0 = -1, y1 = 1. The view angle is set to 45 degrees and the light height is 0.2.

As you can see in Fig. 1, lighting effects are visible on the Mandelbrot Set due to the nature of ray tracing techniques. To get this output the user must enter -2, 1, -1, 1, 300, 0.2, and 45 in that order after running the program. The same instructions apply for the next pictures. (x0, x1) define the x range, and (y0, y1) define the y range. The number of data points in each of these pictures is 1000, but the user can enter 300 to get a picture of similar quality with far less computing time.

As you can see from comparing Fig. 2 and Fig. 4, changing the position of the light can have dramatic effects on the image produced by the program. The taller sections of the image appear much darker when the light is positioned higher like in Fig. 2. Additionally, comparing Fig. 1 with the other figures, the difference in view angle is apparent with a higher view angle showing more of the scene but a lower view angle showing more contrast in height.

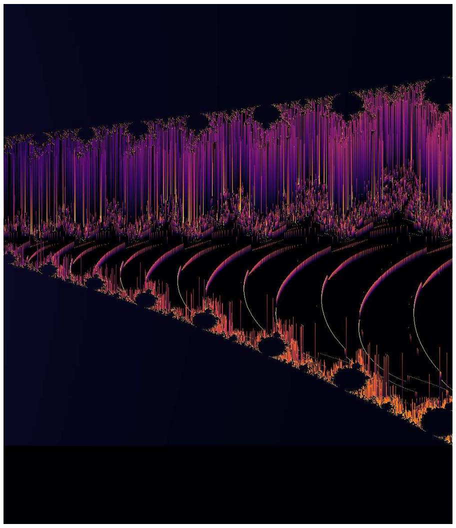

Figure 2: The coordinates are x0 = -0.80, x1 = -0.72, y0 = 0.09, and y1 = 0.15. View angle is 80 degrees and height is 0.95.

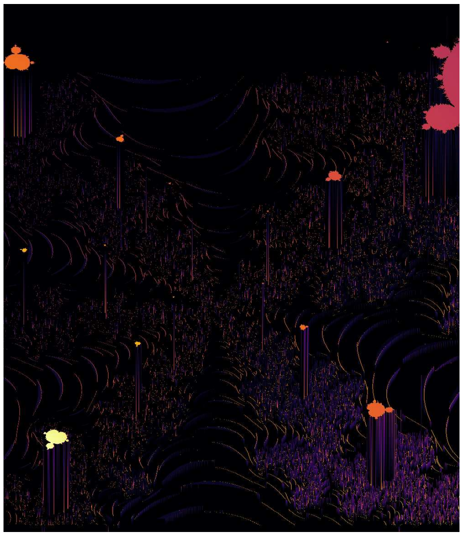

Figure 3: The coordinates are x0 = -0.75, x1 = -0.737, y0 = -0.13, and y1 = -0.12. View angle is 85 degrees and height is 0.1.

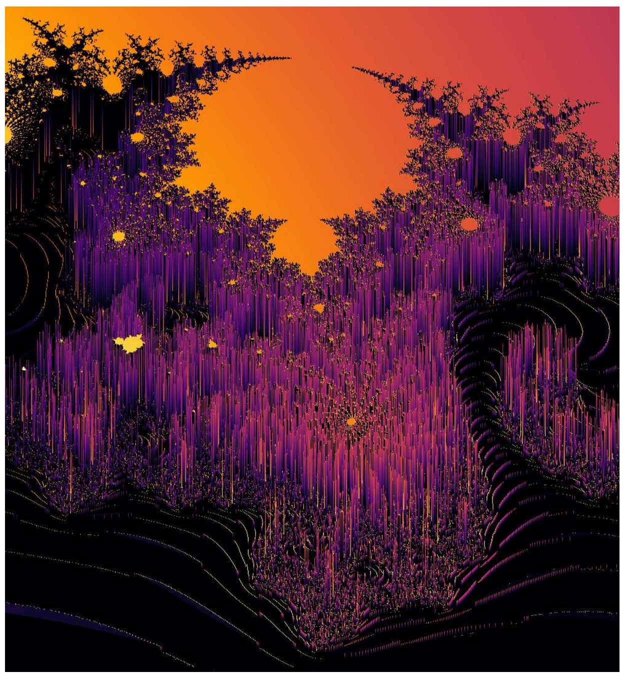

Figure 4: The coordinates are x0 = -0.60, x1 = -0.52, y0 = -0.67, and y1 = -0.61. View angle is 85 degrees and height is 0.1.