Distributed Mandelbrot Set

Introduction

A fractal is defined to be “The graphical representation of a mathematics unlimited curve.” Where “the main property of fractals is the ‘auto-similarity’, meaning that each part of this curve contains the structure of the whole curve.” You can see that as we zoom into the fractal more and more detail is revealed, and eventually we come upon a miniature area that is similar to the original Mandelbrot set.

Complex Numbers

Once you get into more complicated math you start coming across some equations where the answer is neither a real number nor an imaginary number, but the sum of both. An answer that looks like this: 1 + 2i, or 7 + 56i. There's no way of simplifying these equations. You can't reduce them so that there's only one term, you can't derive a sum, you just have to write them down as 1 + 2i. These numbers, part real, part imaginary, are called complex numbers.

If we want to add two complex numbers, say 7 + 4i and 3 + 9i, we simply add the real and imaginary components separately, i.e.; 7 + 3 is 10 and 4i + 9i is 13. This gives us 10 + 13i. What about multiplying them? Do we have to learn brand new weird math to do that? Not really. We'll start with the easy part. 7 * 3 is 21. More complicated is 4i * 9i. Or is it? It's the same as saying 4 * 9 * i * i. 4 * 9 is 36, and we know (since we defined it as such) that i * i is -1. Therefore 4i * 9i is -36. That leaves us with 7 * 9i which is 63i, and 4i * 3 which is 12i. First we add up the real parts which gives us 21 + -36 which is -15. Then we add up the imaginary parts which gives us 63i + 12i which is 75i. So the answer is -15 + 75i. A bit cumbersome and error prone, but not too incomprehensible.

The absolute value of a complex number is given by

\[|a+ib| = (a^2 + b^2)^{1/2}\]

and the following rules are used to add and multiply complex numbers:

\[(a+ib) + (c+id) = (a+c) + i(b+d)\]

\[(a+ib) \times (c+id) = (ac-bd) + i(ad+bc)\]
The Mandelbrot Set

So that brings us to today. The Mandelbrot set is found in the complex plane. Each point on that plane represents a single complex number of the form \( a + bi \), where \( a \) is the distance left or right from centre line (negative when left, positive when right) and \( b \) is the distance above or below the centre line (negative when below, positive when above) and \( i \) is the root of -1.

The Mandelbrot Set is a fractal in the complex plane generated by this simple iteration:

\[
Z_{n+1} \leftrightarrow Z_n^2 + C
\]

Both \( C \) and \( Z \) are complex numbers, which means that they have both real and imaginary components. The multiplication (squaring) and addition are both complex operators. The real component is mapped to the X-axis and the imaginary component is mapped to the Y-axis of the resulting graphs.

\( C \) - The point in the complex plane in question.

\( Z \) - The first \( Z \) (\( Z_0 \)) is set to zero, and because of this, the second \( Z \) (\( Z_1 \)) is simply \( C \), the point in the complex plane. Then, in the second and subsequent iterations, this \( Z \) is squared, and the initial point \( C \) is added to create the next \( Z \) (\( Z_{n+1} \)). After many iterations of this process, one of two things will happen, depending on the initial point \( C \).

\( Z \) will gradually get smaller and smaller, eventually reaching zero.
\( Z \) will begin to grow larger and larger, eventually reaching infinity.

In the first case, the point \( C \) is said to be inside of the Mandelbrot set, and is colored black in the graph. In the second case, the point \( C \) is outside of the Mandelbrot set, and is color-coded based on how fast it approaches infinity.

With the understanding we now have of complex math, we can write out the Mandelbrot formula in a much more compact form. If we have a computer language that understood complex numbers we could write a program in this form.

Complex \( Z = (0 + 0i) \)

Complex \( C = (a + bi) \)

for (count = 0; ABS(Z) <= 2.0 &&
   count < MaxIters; count++)
   
   \( Z = Z * Z + C; \)
That's it. The Mandelbrot set is REALLY simple. All the lines of code at the beginning of this discussion were just teaching the computer how to do the single line of math above. Multiply Z by itself. Add C. The answer is the new value for Z. Repeat until the absolute value of Z is greater than two, or until our counter expires.

So, here's an example of a calculation: Suppose we choose \( n_{\text{max}} = 5 \). Suppose further that we assign colors to values of \( n \) as follows:

| \( n \) | \( |Z| > 2 \) | color          |
|-------|-----------------|---------------|
| 0     | \( |Z_0| > 2 \)   | white         |
| 1     | \( |Z_1| > 2 \)   | red           |
| 2     | \( |Z_2| > 2 \)   | yellow        |
| 3     | \( |Z_3| > 2 \)   | green         |
| 4     | \( |Z_4| > 2 \)   | blue          |

\( n_{\text{max}} = 5 \) \( |Z_5| > \) \( |Z_5| =< 2 \) black

Let's try the point \( C = (0.7 + 0.3i) \):

\[
\begin{align*}
n = 0: & \quad Z_0 = C = (0.7 + 0.3i); \quad |Z_0| = 0.762 < 2; \quad \text{keep going . . .} \\
n = 1: & \quad Z_1 = Z_0^2 + C = (1.1 + 0.72i); \quad |Z_1| = 1.315 < 2; \quad \text{keep going . . .} \\
n = 2: & \quad Z_2 = Z_1^2 + C = (1.392 + 1.884i); \quad |Z_2| = 2.342 > 2; \quad \text{STOP.}
\end{align*}
\]

We'd color that point in yellow. Now try \( C = (0.4 + 0.3i) \):

\[
\begin{align*}
n = 0: & \quad Z_0 = C = (0.4 + 0.3i); \quad |Z_0| = 0.5 < 2; \quad \text{keep going . . .} \\
n = 1: & \quad Z_1 = Z_0^2 + C = (0.47 + 0.54i); \quad |Z_1| = 0.716 < 2; \quad \text{keep going . . .} \\
n = 2: & \quad Z_2 = Z_1^2 + C = (0.329 + 0.808i); \quad |Z_2| = 0.872 < 2; \quad \text{keep going . . .} \\
n = 3: & \quad Z_3 = Z_2^2 + C = (-0.143 + 0.832i); \quad |Z_3| = 0.844 < 2; \quad \text{keep going . . .} \\
n = 4: & \quad Z_4 = Z_3^2 + C = (-0.271 + 0.061i); |Z_4| = 0.278 < 2; \quad \text{keep going . . .} \\
n = 5: & \quad Z_5 = Z_4^2 + C = (0.470 + 0.267i); \quad |Z_5| = 0.541 < 2; \quad \text{keep going . . .} \\
> n_{\text{max}}
\end{align*}
\]

That point looks like it's in the Mandelbrot set, so we color it black.
Performing the above iteration up to 300 times for each pixel generated the graphs you see here.

For a 1024x768 image, this iterative process must be performed 786,432 times for a maximum total of (1024x768x300)235,929,600 iterations. As each of the iterations contain a complex multiplication and addition, this process is computationally expensive. Because each part of the Mandelbrot set can be generated independently, this is a good task to use in experimentation in distributed computing.