Mandelbrot Zoom

Sarthak Mittal

Contents

1 INTRODUCTION

1.1 What is Mandelbrot set?

The Mandelbrot Set is the set of complex numbers c for which the function $f_c(z) = z^2 + c$ does not diverge to infinity when applied iteratively for $z = 0$, i.e., for which the sequence $f_c(0), f_c(f_c(0)), \ldots$ remains bounded in absolute value.

1.2 Examining the pattern

The elaborate and infinitely complicated boundary of the Mandelbrot set is a fractal curve having progressively ever-finer recursive details at increasing magnifications. The pattern followed by this repeating detail depends on the region of set being examined.

2 MATHEMATICAL DETAILS

2.1 Formal definition

The set of values c in the complex plane for which the orbit of the critical point $z = 0$ under iteration of the quadratic map $z_{n+1} = z_n^2 + c$ remains bounded.

2.2 Sample values

c = 1: the sequence obtained is 0,1,2,5,26,... which does not remain bounded.
c = -1: the sequence obtained is 0,-1,0,-1,... which is bounded.

2.3 Useful property

The Mandelbrot set is closed and contained in a closed disk of radius 2 around the origin. So the absolute value of $|z_n|$ must be at most 2 for a point c to be in the Mandelbrot set. If the value exceeds 2, the sequence is bound to escape to infinity.

3 CREATING THE PATTERN

For each sample point c, we test wheter the sequence $f_c(0), f_c(f_c(0)), \ldots$ goes to infinity. Treating the real and imaginary parts of c as image coordinates on the complex plane, we colour the pixels according to how soon the sequence $|f_c(0)|, |f_c(f_c(0))|, \ldots$ crosses an arbitrarily chosen threshold $t \geq 2$.

4 GEOMETRY

We have $f_c(z) = z^2 + c$. Take a point $z = x + iy$ on the complex plane, where $i^2 = -1$. We scale the values on the display (pixel board) into our rectangle of interest, that is $-2 \leq x \leq 2$ and $-2 \leq y \leq 2$. Then we iteratively calculate coordinates of the next point, using $z_{n+1} = z_n^2 + c$. We assign colour and intensity at each point depending on the number of iterations taken for the sequence to either diverge to infinity or complete a threshold number of iterations.

5 DATA STORAGE AND PLOTTING DESIGN

I have used Binary Search Tree to store values of colours at different nodes, and recorded the point (in Complex Number format) inside each node. For the reverse mapping, I have used Graph with a bipartite vertex set comprising of the set of points (x, y) mapped to the corresponding colour on the plot.

For zooming, I have found out the nearest boundary (using Heap for finding the smallest ratio) and then accordingly adjusted the other boundaries along with re-scaling.

6 CODE INSTRUCTIONS

6.1 System

Preferably use Linux. Run in "Release" mode on an x64 system otherwise it will be very slow.

6.2 Dependencies

g++ (GNU C++ compiler) (Windows/Linux/OSX)

SDL 2.0 (Windows/Linux/OSX)

6.3 Compile/Run

Command to compile:
`g++ Heap.cpp Display.cpp Colour.cpp Graph.cpp BST.cpp Complex.cpp MBZ.cpp -lSDL2main -lSDL2`

Command to run:
`./a.out`

Controls:
`q` to quit
`z` to zoom (towards point below mouse)