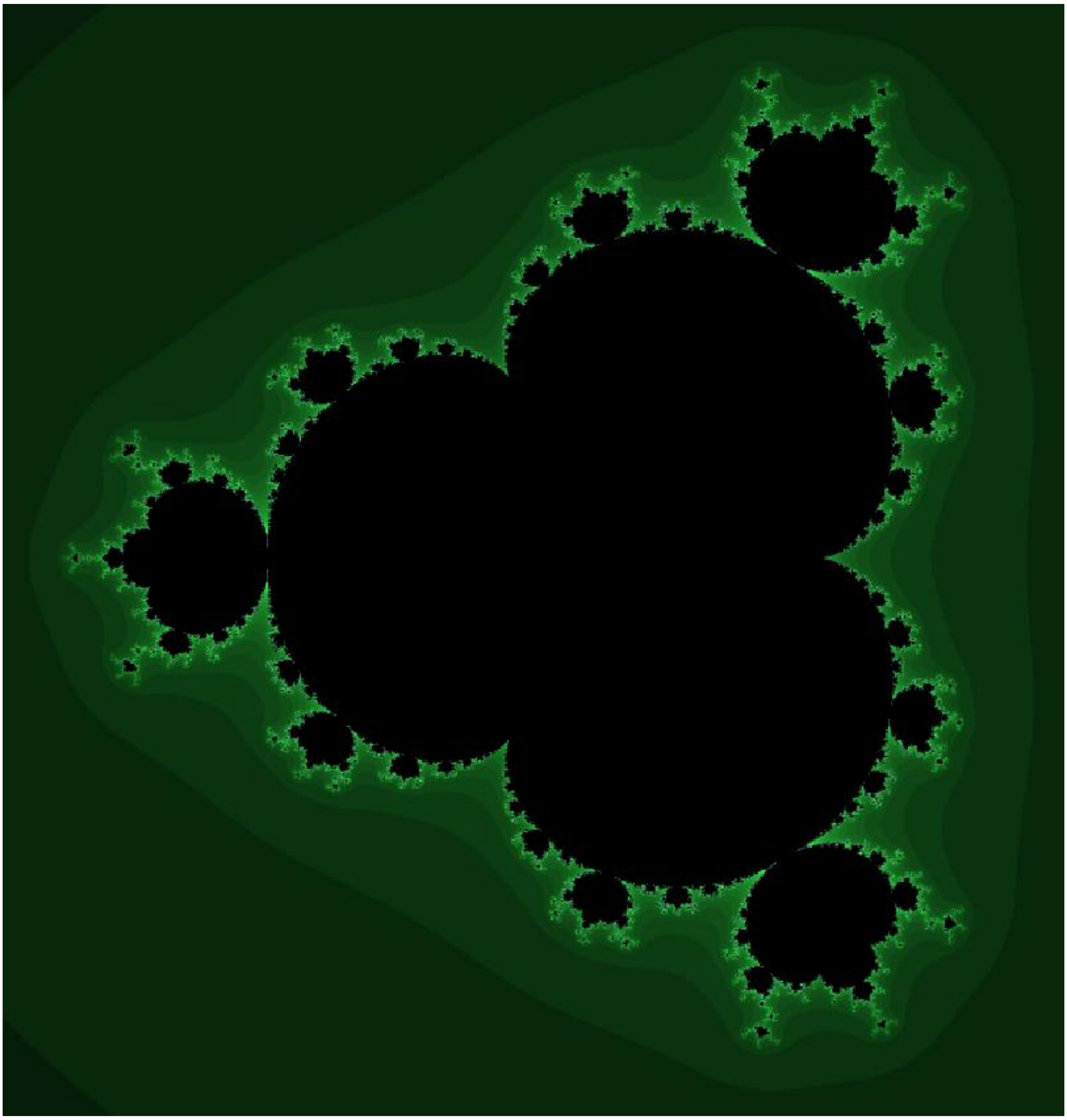
Nolix

Nolix fractals

2023-12-09



Nolix

Table of contents

1 Introduction

1.1 What Nolix fractals are

A Nolix fractal is a **definition** of a specific fractal. From a Nolix fractal an image can be generated.

1.2 Why to use Nolix fractals

- A Nolix fractals is very **general**. There can be chosen any explicit or implicit fractal function, view section, number of iterations, decimal number precision and coloring funtion.
- Nolix fractals can be calculated using **multi-threading**. This makes the generation of fractal images much faster.

1.3 Where the Nolix fractals are

The Nolix fractals are in the Nolix library. To use Nolix fractals, import the Nolix library into your project.

1.4 Structure of this document

Chapter 2 describes the mathematical context for Nolix fractals. Chapter 3 shows Nolix fractals can be built.

2 Mathematical Context

2.1 Motivation

This chapter describes the principle how Nolix fractals work. This chapter explains **all parameters** of a Nolix fractal.

There are **different** ways to create fractals. Nolix fractals are defined by **sequences of complex numbers**.

For this chapter, you need to know the following things.

- What **complex numbers** are and how calculations with complex numbers are done.
- What **sequences** are and what explicit and implicit definitions of sequences are.

2.2 Parametrized complex sequences

Definition (parametrized complex sequence)

For a Nolix fractal there is given a complex sequence $\left(a_n(c)\right): \mathbb{N} \longrightarrow \mathbb{C}$, whereas c is a complex number. We call $a_n(c)$ a **parametrized complex sequence**.

Example (parametrized complex sequence)

$\left(a_1(c)\right) \coloneqq 0$

$\left(a_n(c)\right) \coloneqq {a_{n-1}}^2 + c$

$\boldsymbol{n}$	$\boldsymbol{a_n(0)}$	$\boldsymbol{a_n(1)}$	$\boldsymbol{a_n(\mathrm{i})}$	$\boldsymbol{a_n(1+\mathrm{i})}$
1	0	0	0	0
2	0	1	i	$1+\mathrm{i}$
3	0	2	$-1+\mathrm{i}$	$1+3\mathrm{i}$
4	0	5	$-\mathrm{i}$	$-7+7\mathrm{i}$
5	0	26	$-1+\mathrm{i}$	$1+97\mathrm{i}$

We see that:

- $a_1(c) = 0$ for all c
- $a_2(c) = c$ for all c
- $a_3(c) = c * (c+1)$ for all c
- $a_n(0) = 0$ for all n

2.3 Fractals from parametrized complex sequences

Motivation (iteration count for divergence)

For painting a Nolix fractal, we take a 2-dimensional coordination system. We interpret a point (x, y) in the coordination system as the complex number $x + y\mathrm{i}$. Note that x and y can be any decimal number or real number and do need to be integers.

For a complex number $z = x + y\mathrm{i}$, we will calculate a so-called **iteration count for divergence**.

Definition (iteration count for divergence)

Let $a_n(c)$ be a parametrized complex sequence, $mfc \in \mathbb{R}$ a magnitude for convergence, $ic_{max} \in \mathbb{N}$ a max iteration count and $z \in \mathbb{C}$.

- The **iteration count for divergence of** $a_n(c)$**,** mfc**,** ic_{max} **in** z is the smallest natural number n with $|a_n(z)| > mfc$ **and** $n \leq ic_{max}$ **if** such an n exists.
- The **iteration count for divergence of** $a_n(c)$**,** M**,** m **in** z is -1 **otherwise**.

Definition in other words (iteration count for divergence)

Let $a_n(c)$ be a parametrized complex sequence, $mfc \in \mathbb{R}$ a magnitude for convergence, $ic_{max} \in \mathbb{N}$ a max iteration count and $z \in \mathbb{C}$. n is the **iteration count for divergence of** $a_n(c)$**,** mfc**,** ic_{max} **in** z.

$$n := \begin{cases} \min(n \in \{1,2, \dots, ic_{max}\} \; with \; |a_n(z)| > mfc) \; if \; exists \\ -1 \quad else \end{cases}$$

Painting fractals

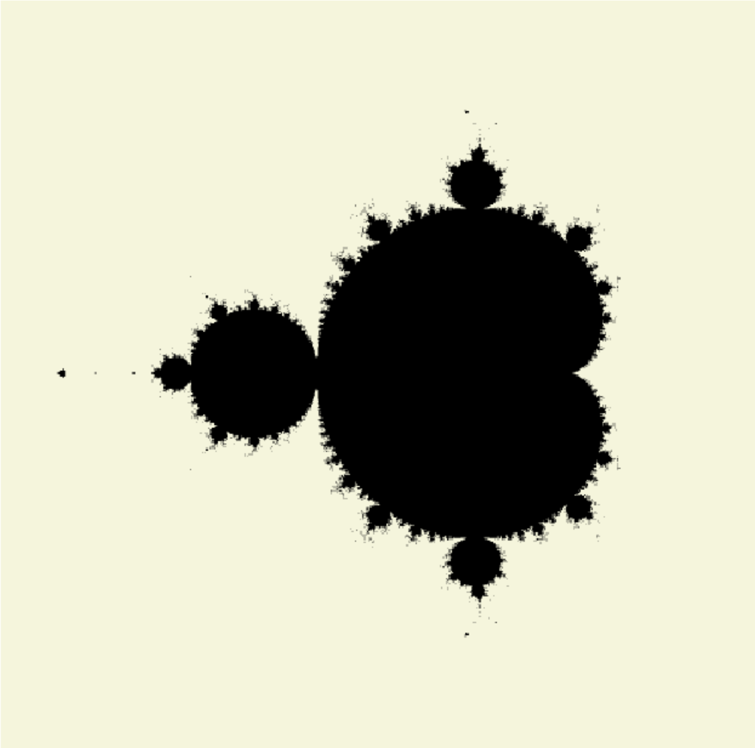
First, we define a function that assigns a color to all iteration counts for divergence, whereas:

- If the iteration count for divergence is -1, the color is black.
- If the iteration count for divergence is $n \neq -1$, the color is the one that was chosen for n.

For a parametrized complex sequence, converge magnitude, maximum iteration count for divergence, a color function and a chosen section in a 2-dimensional coordination system we can paint fractal images when we do the following steps.

1. We interpret each point in the chosen section as a complex number.
2. We calculate the iteration count for divergence of each of the complex numbers.
3. We determine the color of the iteration counts for divergence from the color functions.
4. We paint the points in the determined colors.

Example (Bicolored Mandelbrot fractal)



parametrized complex sequence	$a_1(c) := 0 \quad a_n(c) := {a_{n-1}}^2 + c$
maximum magnitude	10
maximum iteration count	50
color function	$n \mapsto \begin{cases} \text{black } if\ n = -1 \\ \text{beige} \quad\quad else \end{cases}$
coordination system section	$\left\{(x, y) \middle\vert \begin{matrix} x \in (-2,\ -1.99, \dots, 0.99, 1) \\ y \in (-1.5, -1.49, \dots, 1.49, 1) \end{matrix} \right\}$

Definition (Mandelbrot fractal)

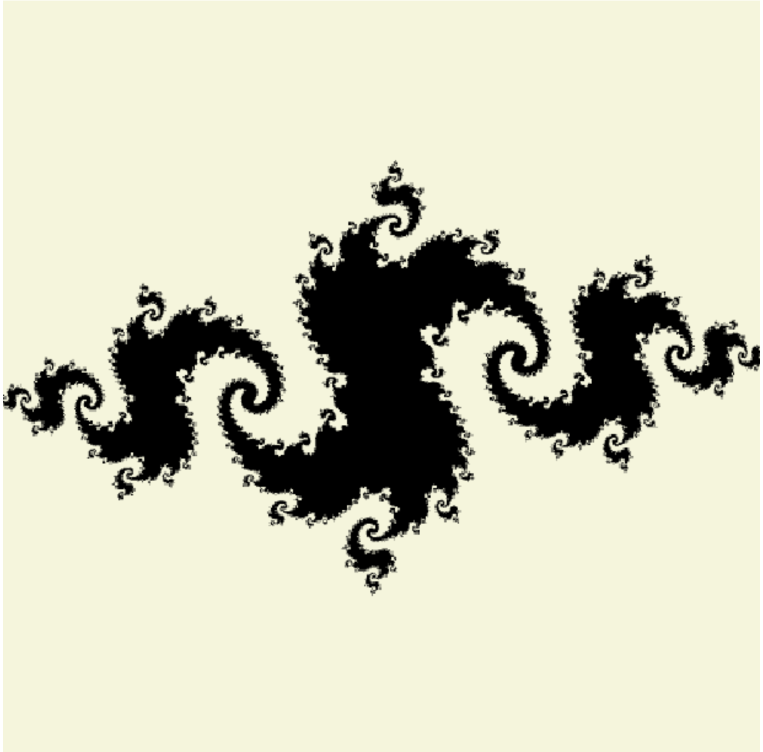
A fractal that is defined by a sequence $(a_n)(c)$ with $a_n(c) := {a_{n-1}}^2 + c$ is called **Mandelbrot fractal**. The Mandelbrot fractal is a very popular fractal.

Definition (Mandelbrot set)

The following set is called the **Mandelbrot set**.

$$\{z \in \mathbb{C} \mid \exists N \in \mathbb{N}: \forall n \in \mathbb{N}: |a_n(z)| < N\}$$

Example (Bicolored Julia fractal)



parametrized complex sequence	$a_1(c) \coloneqq c \quad a_n(c) \coloneqq {a_{n-1}}^2 - 0.8 + 0.15\mathrm{i}$
maximum magnitude	10
maximum iteration count	50
color function	$n \longmapsto \begin{cases} \text{black } if\ n = -1 \\ \text{beige} \quad else \end{cases}$
coordination system section	$\left\{ (x, y) \middle\vert \begin{matrix} x \in (-1.5, -1.49, \dots, 1.49, 1.5) \\ y \in (-1.5, -1.49, \dots, 1.49, 1.5) \end{matrix} \right\}$

About Julia fractals

A fractal that is defined by a sequence $(a_n)(c)$ with $a_1(c) = c$ and $a_n(c) \coloneqq {a_{n-1}}^2 + j$ whereas $j \in \mathbb{C}$ is called **Julia fractal**. j is called **Julia constant** of the Julia fractal.

3 FractalBuilder

3.1 Types for fractals

The ch.nolix.tech.math.fractal package contains types for fractals.

Type	Meaning
Fractal	Represents a fractal.
FractalBuilder	Can build Fractals.
ComplexNumber	Represents a complex number.
ClosedInterval	Represents a closed interval.
ComplexExplicitSequence	Represents a complex sequence that is defined explicitly.
ComplexSequenceDefinedBy1Predecessor	Represents a complex sequence that is defined recursively with 1 predecessor.
ComplexSequenceDefinedBy2Predecessor	Represents a complex sequence that is defined recursively with 2 predecessors.

3.2 Create a Fractal from a FractalBuilder

```
import ch.nolix.tech.math.fractal.Fractal;
import ch.nolix.tech.math.fractal.FractalBuilder;
…
var fractalBuilder = new FractalBuilder();
var fractal = fractalBuilder.build();
```

The build method of a FractalBuilder builds a new Fractal. On a FractalBuilder, properties for fractals can be set. The properties of a FractalBuilder will always be applied to the next Fractal the FractalBuilder builds. A Fractal is immutable.

3.3 Generate an image from a Fractal

```
Fractal fractal = …;
var image = fractal.toImage();
```

The toImage method of a Fractal generates a new image of the Fractal.

3.4 Set the number precision scale of a Fractal

```
FractalBuilder fractalBuilder = …;
fractalBuilder.setBigDecimalScale(20);
```

The setBigDecimalScale method of a FractalBuilder sets the number precision of the Fractals the FractalBuilder will build. The numbers of a Fractal are BigDecimals. The scale of a BigDecimal is the number of decimal places. All numbers of a Fractal will have the same scale. The default number precision scale of a Fractal is 10. The bigger the number precision scale of a Fractal, the bigger the precision of the calculations. But the costs for the calculation increase.