Politechnika Krakowska im. Tadeusza Kościuszki w Krakowie

Fraktale - paproć Barnsleya

Ewelina Adamus

Paulina Drąg

Magdalena Gwarda

26 stycznia 2017

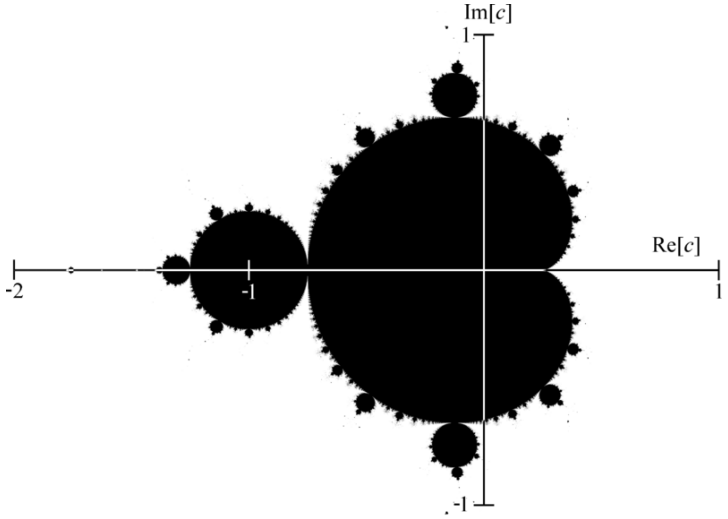
1 Wprowadzenie

1.1 Fraktale

Fraktal w znaczeniu potocznym oznacza zwykle obiekt samo-podobny, czyli taki gdzie występuje efekt podobieństwa całego zbioru do jego elementów. Ich najciekawszą grupą, są te stworzone na płaszczyźnie zespolonej. W przyrodzie fraktale to struktury o budowie fraktalnej są powszechnie spotykane w przyrodzie. Przykładem mogą być płatki śniegu, system naczyń krwionośnych, systemy wodne rzek, błyskawica lub kwiat kalafiora rzymskiego.

Rysunek 1.1: Kalafior rzymski jako przykład występowania fraktali w przyrodzie.

Pojęcie fraktala zostało wprowadzone do matematyki przez francuskiego informatyka i matematyka polskiego pochodzenia Benoita Mandelbrota w latach siedemdziesiątych XX wieku. Odkryty przez niego **zbiór Mandelbrota** wygląda następująco:



Rysunek 1.2: Zbiór Mandelbrota.

1.2 Zastosowanie fraktali

- Kompresja fraktalna, która należy do kompresji stratnej i jest głównie używana w grafice.
- Powiększanie obrazów, najczęściej obrazy będące grafiką rastrową.
- Medycyna i biologia (uważa się, że chromosomy mają strukturę drzewiastą i tym samym są fraktalami).
- Ekonomia, fraktale używa się do analizy trendów spółek.
- Opisywanie układów dynamicznych, które używana się np. do opisu ruchu planet Układu Słonecznego

2 Przedstawienie problemu

2.1 Paproć Barnsleya

Paproć Barnsleya to fraktal znany ze względu na uderzające podobieństwo do liści paproci występujących w naturze, spopularyzowany przez Michaela F. Barnsleya. Jest to przykład złożonego obiektu, który może być opisany za pomocą zaledwie czterech przekształceń afinicznych jako atraktor (czyli zbiór w przestrzeni fazowej do którego w miarę upływu czasu zmierzają trajektorie rozpoczynające się w różnych obszarach przestrzeni fazowej) następującego systemu funkcji zwężających (IFS). IFS to zbiór funkcji za pomocą, których konstruuje się fraktale.

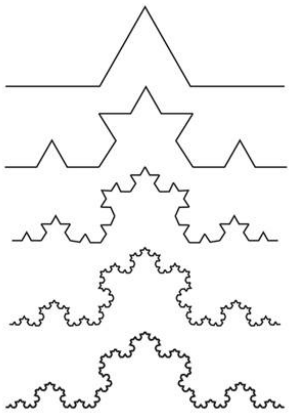
Rysunek 2.1: Paproć Barnsleya.

2.2 IFS - system funkcji iterowanych

Fraktal IFS stworzony jest z kilku bądź nawet kilkunastu kopii samego siebie. Każda kopia zostaje przekształcona przez funkcję, ściślej przez system funkcji iterowanych. Funkcje te można nazwać za kurczliwe, co oznacza, że kształty poddane ich działaniom stają się coraz mniejsze. Podsumowując fraktal IFS stworzony jest z (prawdopodobnie) nakładających się samo-podobnych kształtów, które także są kopiami samego siebie itd.

Dobrym przykładem może okazać się krzywa Kocha, gdzie z odcinka podzielonego na 3 części powstaje tzw. płatek Kocha. Stworzenie takiej krzywej przebiega następująco:

- Odcinek podzielony na 3 równe części, z których środkowa część zostaje zastąpiona przez dwa równe odcinki, tak aby tworzyły z „nieobecną" podstawą trójkąt równoboczny (nachylone względem niej pod kątem 60°).
- W kolejnym kroku krzywa podzielona jest na 4 odcinki każdy o długości 1/3 L.
- Następnie odcinki te dzielone są tak samo jak odcinek pierwszy, czy na 3 równe części.
- W kolejnych krokach powtarza się czynność, co skutkuje pojawieniem się poniższego wzoru:



Rysunek 2.2: Etapy powstawania krzywej Kocha.

Opisane translacje można przedstawić za pomocą IFS w następujący sposób:

$$f_1(\mathrm{x}) = \begin{bmatrix} 1/3 & 0 \\ 0 & 1/3 \end{bmatrix} x$$

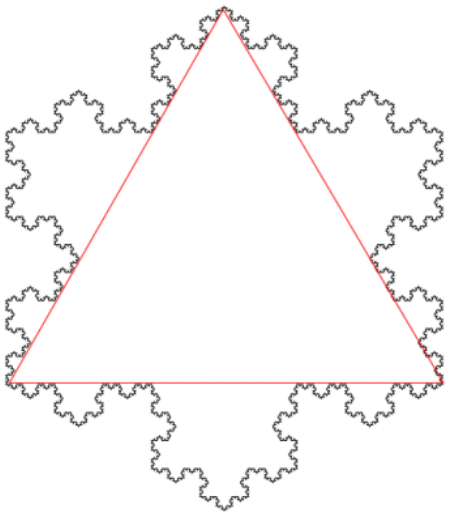
$$f_2(\mathrm{x}) = \begin{bmatrix} 1/6 & -\sqrt{3}/6 \\ \sqrt{3}/6 & 1/6 \end{bmatrix} x + \begin{bmatrix} 1/3 \\ 0 \end{bmatrix}$$

$$f_3(\mathrm{x}) = \begin{bmatrix} 1/6 & \sqrt{3}/6 \\ -\sqrt{3}/6 & 1/6 \end{bmatrix} x + \begin{bmatrix} 1/2 \\ \sqrt{3}/6 \end{bmatrix}$$

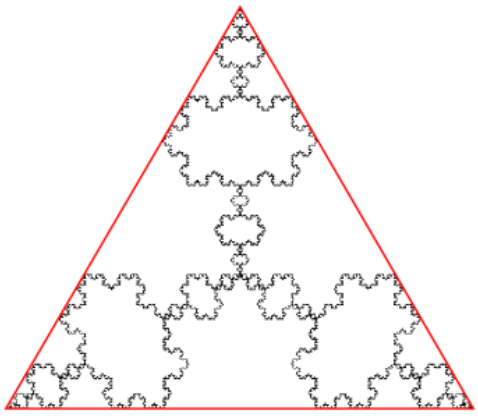
$$f_4(\mathrm{x}) = \begin{bmatrix} 1/3 & 0 \\ 0 & 1/3 \end{bmatrix} x + \begin{bmatrix} 2/3 \\ 0 \end{bmatrix}$$

f_1(x) – podział odcinka na 4 równe części
f_2(x) – podział oraz obrót konkretnego odcinka o 60°
f_3(x) – podział oraz obrót konkretnego odcinka o -60°
f_4(x) – podział odcinka na 4 równe części

Trzy krzywe Kocha będące ramionami trójkąta tworzą fraktal nazywany Płatkiem Kocha:



Zaś krzywe „wpisane” w trójkąt tworzą następujący fraktal:



IFS zostały opisane przez John'a Hutchinson'a w 1981 natomiast spopularyzował je Micheal Barnsley. Formalnie system funkcji iterowanych zapisać można jako zbiór n funkcji przekształcający zbiór S:

$$\begin{gathered} \mathrm{S}=f_1(S) \\ \mathrm{S}=f_2(S) \\ \ldots \\ \mathrm{S}=f_n(S) \end{gathered}$$

Zbiór S będzie reprezentowany jest przez płaszczyznę dwuwymiarową (opisaną przez współrzędne x oraz y), a funkcje to przekształcenia afiniczne tej płaszczyzny, czyli pojedyncza funkcja f(x_n;y_n) będzie dokonywać przekształcenia:

$$\begin{gathered} x_{n1}=ax_n + by_n + e \\ y_{n1}=cx_n+dy_n+f \end{gathered}$$

2.3 Chaos game - generacja fraktali

Generowanie fraktali ze zbioru funkcji z poprzedniego slajdu przedstawił Barnsley, pod nazwą "gra chaos". Polega to na tym że dla aktualnego punktu (x; y) wybierana jest losowo funkcja ze zbioru, a następnie wykonuje się przekształcenie otrzymując nowy punkt.

Zaznacza się punkt na płaszczyźnie i po raz kolejny losuje funkcję i dokonując dalszego przekształcenia. Proces ten powtarza się określoną liczbę razy. Im więcej przetworzonych punktów, tym bardziej dokładny fraktal.

Początkową wartość punktu wybierana jest losowo. Nie dla wszystkich IFS, prawdopodobieństwo wylosowania każdej z funkcji powinno być takie samo, dlatego oprócz zbioru funkcji definiuje się również prawdopodobieństwo p_i z jakim dana funkcja powinna zostać wylosowana. Suma tych prawdopodobieństw musi wynosić $p_1 + p_2 + ... + p_n = 1$.

Prawdopodobieństwa dla paproci Barnsleya: $p_1 = 0.01$, $p_2 = 0.85$, $p_3 = 0.07$, $p_4 = 0.07$

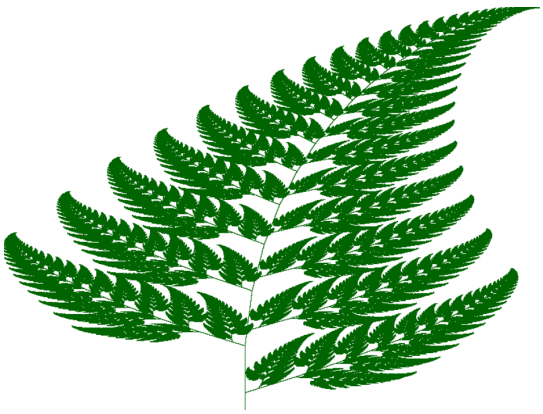
Przekształcenia:

$$\begin{gathered} f_1(0.0x + 0.0y + 0.0,\ 0.0\mathrm{x} + 0.16\mathrm{y} + 0.0) \\ f_2(0.85\mathrm{x} + 0.04\mathrm{y} + 0.0;\ -0.04\mathrm{x} + 0.85\mathrm{y} + 1.6) \\ f_3(0.2\mathrm{x} - 0.26\mathrm{y} + 0.0;\ 0.23\mathrm{x} + 0.22\mathrm{y} + 1.6) \\ f_4(-0.15\mathrm{x} + 0.28\mathrm{y} + 0.0;\ 0.26\mathrm{x} + 0.24\mathrm{y} + 0.44) \end{gathered}$$

3 Program

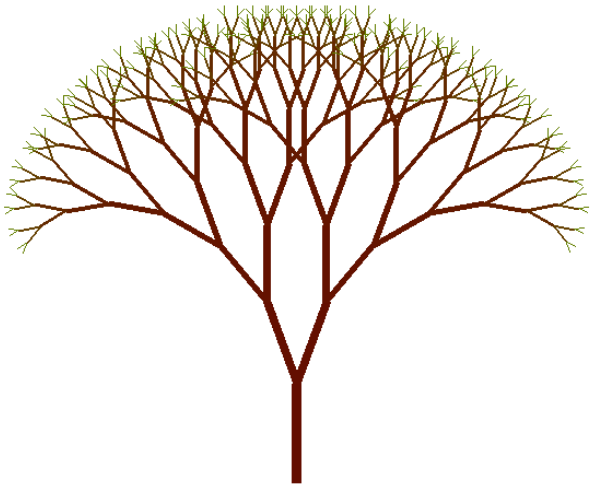
Wszystkie programy zostały stworzone w języku Python.

Program nr 1 to program generujący paproć Barnsleya. Jako produkt końcowy otrzymujemy następujący obraz:



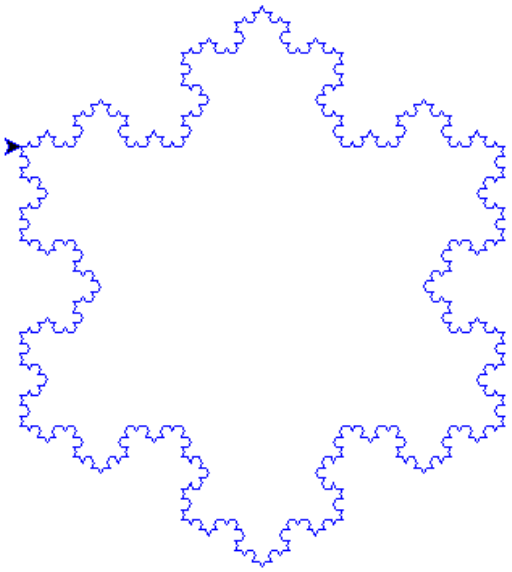
Rysunek 3.1: Wynik działania programu 1.

Program nr 2 to program generujący drzewo. Jako produkt końcowy otrzymujemy poniższy obraz:



Rysunek 3.2: Wynik działania programu nr 2.

Program nr 3 to program generujący płatek Kocha. Jako produkt końcowy otrzymujemy:



Rysunek 3.3: Wynik działania programu nr 3.

3.1 Kod źródłowy

3.1.1 Program nr 1

```
import random
from PIL import Image


class Barnsley(object):
    def __init__(self, szerokosc, wysokosc, pedzel=(0, 100, 0),
                 tlo=(255, 255, 255)):
        self.img_width=szerokosc
        self.img_height=wysokosc
        self.paint_color = pedzel
        self.x=0
        self.y=0
        self.age = 0

        self.paproc = Image.new('RGB', (szerokosc, wysokosc),
                                tlo)
        self.pix = self.paproc.load()
        self.pix[self.skaluj(0, 0)] = pedzel

    def skaluj(self, x, y):
        h = (x + 2.182)*(self.img_width - 1)/4.8378
        k = (9.9983 - y)*(self.img_height - 1)/9.9983
```

```
        return h, k

    def transformuj(self, x, y):
        rand = random.uniform(0, 70)
        if rand <= 1:
            return 0, 0.16*y
        elif rand<=8:
            return 0.2*x - 0.26*y, 0.23*x + 0.22*y + 1.6
        elif rand<=15:
            return -0.15*x + 0.28*y, 0.26*x + 0.24*y + 0.44
        else:
            return  0.85*x + 0.04*y, -0.04*x + 0.85*y + 1.6

    def iteruj(self, iteracja):
        for _ in range(iteracja):
            self.x, self.y = self.transformuj(self.x, self.y)
            self.pix[self.skaluj(self.x, self.y)]
                = self.paint_color

        self.age += iteracja

paproc = Barnsley(800, 600)
paproc.iteruj(50000000)
paproc.paproc.show()
```

3.1.2 Program nr 2

```
import pygame, math

pygame.init()
window = pygame.display.set_mode((600, 600))
pygame.display.set_caption("Drzewo")
screen = pygame.display.get_surface()
screen.fill((255, 255, 255))
screen.blit(screen, (0, 0))

pygame.display.update()

def rysujDrzewo(x1, y1, kat, poziom):
    if poziom:
        x2 = x1 + int(math.cos(math.radians(kat)) * poziom
                * 10.0)
        y2 = y1 + int(math.sin(math.radians(kat)) * poziom
```

```
                * 10.0)
        pygame.draw.line(screen, (100,255/(2*poziom),0),
                (x1, y1), (x2, y2), 1*poziom)
        rysujDrzewo(x2, y2, kat - 20, poziom - 1)
        rysujDrzewo(x2, y2, kat + 20, poziom - 1)

def input(event):
    if event.type == pygame.QUIT:
        exit(0)

rysujDrzewo(300, 550, -90, 9)
pygame.display.flip()
while True:
    input(pygame.event.wait())
```

3.1.3 Program nr 3

```
from turtle import *

def Koch(dlugosc, poziom):
        if poziom ==0:
                forward(dlugosc)
                return
        Koch(dlugosc/3, poziom-1)
        left(60)
        Koch(dlugosc/3, poziom-1)
        right(120)
        Koch(dlugosc/3, poziom-1)
        left(60)
        Koch(dlugosc/3, poziom-1)

def caly_platek(dlugosc, poziom):
        for i in range(3):
                Koch(dlugosc,poziom)
                right(120)

if __name__ =="__main__":
        speed(0)
        pensize(1)
        pencolor("blue")
        dlugosc=300.0
        penup()
        backward(dlugosc/2.0)
```

```
pendown()
title('Platek Kocha')
caly_platek(dlugosc,4)
mainloop()
```

4 Literatura

1. http://home.agh.edu.pl/ zobmat/2016/1_gorskipawel/zastosowanie.html
2. https://pl.wikipedia.org/wiki/Fraktal
3. http://www.algorytm.org/fraktale/system-funkcji-iterowanych-ifs.html
4. http://ecademy.agnesscott.edu/ lriddle/ifs/ifs.htm
5. https://en.wikipedia.org/wiki/Iterated_function_system
6. https://rosettacode.org/wiki/Barnsley_fern