Gimnazija Novo mesto

Seidlova cesta 9, 8000 Novo mesto

Raziskovalna naloga

ZVOKI FRAKTALOV

Raziskovalno področje: MATEMATIKA

Avtorji:

JAN IGOR KRŽAN

ANDRAŽ SENIČAR

LARA VALENČIČ

Mentorja:

JERNEJ BAN, prof.

dr. BORUT JURČIČ ZLOBEC

Novo mesto, marec 2019

Raziskovalna naloga je bila opravljena na Gimnaziji Novo mesto, pripravili smo jo Jan Igor Kržan, Andraž Seničar in Lara Valenčič.

Jan Igor Kržan

____

Andraž Seničar

____

Lara Valenčič

____

Mentorja: Jernej Ban, prof. matematike in dr. Borut Jurčič Zlobec

Datum predstavitve:

KLJUČNA DOKUMENTACIJSKA INFORMACIJA

ŠD Gimnazija Novo mesto, 2018/2019

KG fraktali/ zvok/ preproga Sierpinskega/ Kochova snežinka

AV KRŽAN, Jan Igor/ SENIČAR, Andraž/ VALENČIČ, Lara

SA BAN, Jernej/ JURČIČ ZLOBEC, Borut

KZ 8000 Novo mesto, SLO, Seidlova cesta 9

ZA Gimnazija Novo mesto

LI 2019

IN **FRAKTALI**

TD Raziskovalna naloga

OP XII, 65 strani, 48 sl., 36 spektrogramov, 28 sinusoid, 19 vir.

IJ SL

JI sl/ en

AI Dandanes so fraktali prisotni povsod. Internet je poln generiranih slik fraktalov, primerov, kje naj bi se nahajali v naravi, v glasbi itd. Čeprav je napisanega veliko o povezavi med fraktali in glasbo, je presenetljivo malo informacij o tem, kako bi zveneli. Raziskovalna naloga je usmerjena v dva različna načina prevajanja slike fraktala v zvočne frekvence. V obeh primerih se določi položaj vsakega piksla s pomočjo Hilbertove krivulje, nato pa se lahko vsakemu pikslu določi frekvenco glede na njegovo barvno vrednost. Razlika med obema načinoma prevajanja slike fraktala v glasbo je, kako se seštejejo sinusoide frekvenc vseh pikslov. V prvem oziroma simultanem načinu računalniški program vse sinusoide sešteje v eno samo. V drugem oziroma konsekutivnem načinu pa program sinusoide postavi v zaporedje in jih predvaja. Rezultati v nalogi ustrezajo zastavljenim hipotezam, z manjšimi odstopanji, ki so tudi pojasnjena.

KEY WORD DOCUMENTATION

ND Gimnazija Novo mesto, 2018/2019

CX fractals/ sound/ Sierpinski carpet/ Koch snowflake

AU KRŽAN, Jan Igor/ SENIČAR, Andraž/ VALENČIČ, Lara

AA BAN, Jernej/ JURČIČ ZLOBEC, Borut

PP 8000 Novo mesto, SLO, Seidlova cesta 9

PB Gimnazija Novo mesto

PY 2019

TI **FRACTALS**

DT RESEARCH WORK

NO XII, 65 p., 48 fig., 36 spectrograms, 28 sin., 19 ref.

LA SL

AL sl/ en

AB Nowadays fractals are present everywhere. The Internet is full of images of generated fractals, examples of where they are supposed to be in nature, in music, etc. Although much has been written about the connection between fractals and music, there is surprisingly little information on how would a fractal sound. The research task is focused on two different methods of translating the fractal image into sound frequencies. In both cases, the position of each pixel is determined using the Hilbert curve, and then each frequency can be defined by/based on its colour value. The difference in both ways of translating the fractal image into music is how to sum up the sinusoids of the frequencies of all the pixels. In the first or simultaneous mode the computer program all sinusoids sums together. In the second or consecutive mode, the sinusoid program is sequenced one after the other and plays in that sequence. The results in the task correspond to the set hypotheses, with minor deviations, which are also explained.

Zahvala

Iskreno se zahvaljujemo svojemu mentorju prof. Jerneju Banu, ki nas je pri našem raziskovanju spodbujal, nas usmerjal, nam predajal svoje znanje ter posvečal svoj čas. Zahvalili bi pa se tudi našemu zunanjemu mentorju dr. Borutu Jurčiču Zlobcu, ki nas je usmerjal in dajal nasvete, brez katerih ta naloga ne bi ugledala sončne svetlobe.

Zahvaliti se moramo tudi profesorici Sonji Simčič, ki je našo nalogo lektorirala in poskrbela, da so v naši nalogi vse vejice na pravem mestu.

Zahvaljujemo se še profesorju Janezu Gorencu za pomoč pri prevodu avtorskega izvlečka.

Zahvaljujemo se tudi staršem za veliko razumevanja in potrpljenja ter neprestane spodbude pri pisanju raziskovalne naloge.

Povzetek

O fraktalih je dandanes napisane že veliko literature in zanje je slišalo že mnogo ljudi, vendar pa je predstava o njih marsikdaj zgrešena. Najbolj znani so slike fraktalov, saj ne moremo spregledati njihovih čudovitih oblik in vzorcev, in vizualno privlačni posnetki, v katerih se s približevanjem ponavljajo vedno iste oblike.

Mi pa smo se vprašali, kako bi fraktale prevajali v glasbo in iz slike dobili zvok. V raziskovalni nalogi obravnavamo dva načina prevajanja fraktala iz slike v zvok.

Narisali smo preprogo Sierpinskega in Kochovo snežinko v črni barvi na belo podlago in s pomočjo Hilbertove krivulje vsakemu pikslu na sliki fraktala določili njegovo pozicijo. V programu Python smo napisali dva različna programa, ki sta nam fraktal prevedla v zvok. Prvi program je vsakemu pikslu določil sinusoido; le-te je na koncu seštel v eno ter jo predvajal. Drugi program pa je predvajal vsako sinusoido v zaporedju glede na položaj piksla na sliki.

Kazalo vsebine

Kazalo slik

Kazalo spektrogramov

Kazalo sinusoid

1. Uvod

O fraktalih je dandanes napisane že veliko literature. Izraz fraktalna geometrija postaja vse bolj prisoten v raznih populacijskih znanstvenih člankih. V takšnih člankih nemalokrat fraktalom pripisujejo lastnosti, ki so absurdne, kot na primer, da držijo ključ do razumevanja vesolja, da predstavljajo odgovore na vsa vprašanja ipd.

Ker pa imajo fraktali mnogo uporab, so način za opisovanje mnogih oblik v naravi in ker so res prisotni povsod okoli nas, nas je zanimalo, ali jih lahko opazujemo tudi v glasbi.

Ene izmed prvih študij, ki so raziskovale fraktalne lastnosti v glasbi, sta izvedla fizika Richard F. Voss in John Clarke z Univerze v Kaliforniji. V svoji raziskavi sta analizirala klavirske skladbe Scotta Joplina in Bachov Brandenburški koncert. Določila sta, da te skladbe vsebujejo nekaj fraktalnih lastnosti. [1]

Tej študiji so v prihodnjih letih sledile še mnoge, ki so razpravljale o vlogi fraktalnih vzorcev v glasbi.

Nas je zanimalo, kaj se zgodi, če storimo ravno obratno in poskusimo 'uglasbiti' fraktal. Kakšne glasbene vzorce lahko prevedemo iz slike fraktala in kako bi fraktal zvenel?

V naši raziskovalni nalogi smo se posvetili vprašanju, kako bi prevedli fraktal oziroma sliko fraktala v zvočne frekvence. Posvetili smo se analizi dveh načinov prevajanja slike fraktala v zvočne frekvence, ki sta se nam zdela najbolj zanesljiva.

2. Namen raziskovalne naloge in hipoteze

Naš cilj je prevesti fraktal oziroma sliko fraktala v zvočni zapis, ki ga lahko poslušamo.

Glede na to, da načina, s katerim smo prevajali fraktal oziroma sliko fraktala, nismo našli še nikjer na internetni sferi, smo ustvarili hipoteze glede na naše poznavanje lastnosti fraktalov in Hilbertove krivulje.

Hipoteza 1: Vsaka višja iteracija preproge Sierpinskega se bo bližala frekvenci zvoka, ki smo jo določili beli barvi.

Hipoteza 2: Zvok Kochove snežinke se bo z vsako iteracijo znižal (če pripišemo točkam znotraj Kochove snežinke črno barvo), ker pa je njena površina omejena, se bo zvok z vsako višjo iteracijo manj spreminjal.

3. Zaporedja

Zaporedje elementov neke množice *A* je preslikava iz množice naravnih števil ℕ v množico *A*, ki vsakemu naravnemu številu priredi element množice *A:*

$$f: i \longmapsto a_i, \text{ kjer je } i \in \mathbb{N} \text{ in } a_i \in A.$$

Razporejeni so tako, da je prvi element zaporedja označen z a_1, drugi z a_2, tretji z a_3 itn. Če za primer vzamemo zaporedje naravnih števil, bi bil a_1 enak 1, a_2 enak 2 in tako naprej.

Vsakemu elementu zaporedja smo torej zmožni določiti, kje znotraj zaporedja stoji.

3.1. Geometrijsko zaporedje

Geometrijsko zaporedje je zaporedje števil, pri katerem dobimo vsak naslednji člen tako, da prejšnjega množimo z isti številom. Zaporedje je torej geometrijsko, če je količnik dveh zaporednih členov konstanten, označimo ga s k. [2]

Za člene a_1, a_2, a_3 … geometrijskega zaporedja je:

$a_2 = a_1 * k$

$a_3 = a_2 * k = a_1 * k^2$

…

$a_n = a_{n-1} * k = a_1 * k^{n-1}$

Število k imenujemo količnik geometrijskega zaporedja in zanj velja: $k = \frac{a_{n+1}}{a_n}$

3.2. Končna geometrijska vrsta

Končna geometrijska vrsta je vsota n členov geometrijskega zaporedja. [2]

Označimo jo z S_n in zapišemo kot:

$$S_n = \frac{a_1(k^n - 1)}{k - 1}$$

Končna geometrijska vrsta ima znan končni člen in tako se razlikuje od neskončne geometrijske vrste.

3.3. Neskončna geometrijska vrsta

Neskončna geometrijska vrsta je vsota neskončno mnogo členov geometrijskega zaporedja. [3]
Označimo jo z S in zapišemo kot:

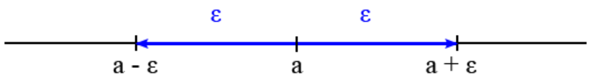
$$S = \frac{a_1}{1 - q}$$

Pogoj za vsoto:

$|q| < 1\ ali - 1 < q\ < 1$

3.4. Limita zaporedja

ε (okolica števila *a)* je odprt interval okoli a, širine 2ε.



Slika 1 – ε-okolica točke a

Kadar je v poljubno majhni ε okolici nekega števila neskončno mnogo členov zaporedja, se to število imenuje stekališče zaporedja. Zaporedje ima lahko eno stekališče, več stekališč ali nobenega. [2]

Če je stekališče eno samo in tako, da so v ε okolici tega števila vsi členi zaporedja od nekega člena naprej, potem se to število imenuje limita zaporedja.

$$a = \lim_{n \to \infty} a_n$$

Limita zaporedja je torej tisto število, kateremu se bližajo členi zaporedja. Zaporedje, ki ima limito, imenujemo konvergentno zaporedje.

3.5. Vsota neskončne vrste

Zapišimo vsote prvih n členov zaporedja:

$S_1 = a_1$

$S_2 = a_1 + a_2$

$S_3 = a_1 + a_2 + a_3$

…

$S_n = a_1 + a_2 + \cdots + a_n$

Dobljene vsote imenujemo delne vsote zaporedja in jih označimo z S_n.

Zaporedje S_1, S_2, S_3 …, S_n pa imenujemo zaporedje delnih vsot. [2]

Vsota neskončne vrste a_1, a_2, a_3 … je limita zaporedja delnih vsot, ko gre n proti neskončno. Limito zaporedja delnih vsot označimo z S in zapišemo kot:

$$S = \lim_{n \to \infty} S_n$$

4. Fraktali

Fraktali so katerakoli skupina kompleksnih geometričnih oblik, ki posedujejo fraktalno dimenzijo.

Na začetku so tako poimenovali zvezno in nikjer odvedljivo krivuljo. To je krivulja, ki ji v nobeni točki ne moremo položiti tangente, v vsaki točki je prelomljena. [4]

Idejo je prvič predstavil matematik Felix Hausdorff.

Naravni svet okoli nas je definiran z nepravilnimi oblikami in površinami in ne s klasičnimi evklidskimi, idealnimi primeri. Mandelbrot v svoji knjigi Fraktalna geometrija narave piše: *”Oblaki niso krogle, gore niso stožci, obale niso krogi, lubje ni gladko in strela ne potuje v ravni črti.”* [2] Fraktali pa so geometrija, s katero opisujemo nepravilne oblike narave.

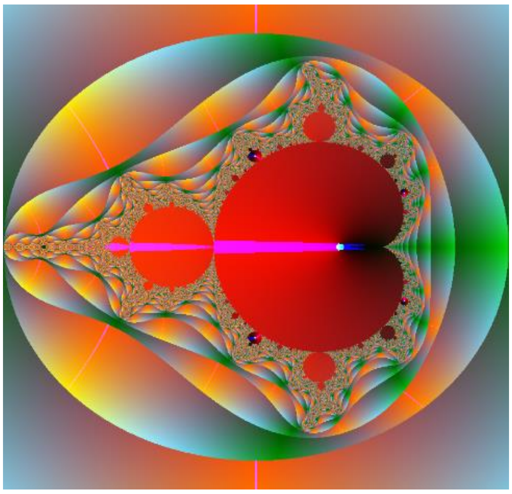
4.1. Zgodovinsko ozadje

Ideje o obstoju fraktalov so bile znane že v 17. stoletju, ko so matematiki začeli razmišljati o ponavljajoči se samopodobnosti.

Prve primere fraktalov je ustvaril Georg Cantor, danes njegov fraktal poznamo kot Cantorjevo množico. V naslednjih desetletjih so ustvarili še nove primere fraktalov, ki pa so bili omejeni na skice in slike, ki so bile enostavne in ročno narisane.

V 70. letih 20. stoletja je francoski matematik Benoit Mandelbrot združil vse predhodne ideje in si izmisli izraz fraktal. Ugotovil je njihove skupne značilnosti in jih javnosti predstavil v knjigi z naslovom The Fractal Geometry of Nature (Fraktalna geometrija narave), 1982. V njej je opisal mnoge primere fraktalov, najdene v naravi, in ugotovil, da se fraktali v naravi pojavljajo zelo pogosto. Zato Mandelbrotu dostikrat pripišemo naziv začetnik fraktalov.

Do preboja na področju fraktalov pride z uporabo računalnika in s tem so se odprle nove možnosti risanja fraktalov; do neverjetnih podrobnosti na poljubno majhnih skalah, iz katerih lahko nastanejo slike in primeri fraktalov, kot jih lahko vidimo danes. [6]

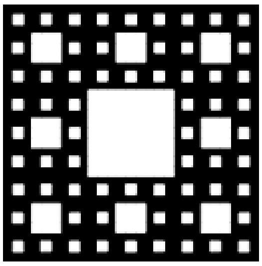


Slika 2 – Mandelbrotova množica

4.2. Samopodobnost

Sodobna definicija fraktala vsebuje dva osnovna pojma, samopodobnost in fraktalno dimenzijo. V matematiki je samopodoben lik ali telo točno podobno delu samega sebe. To pomeni, da ko začnemo povečevati del celote v neskončno majhne dele, morajo biti ti manjši deli podobni celoti. Z drugimi besedami, manjši del mora biti kopija celote. Ta lastnost je skupna vsem fraktalom. Ker imajo fraktali to lastnost, lahko dokaj preprosto ustvarimo izjemno zapletene strukture.

Samopodobnost je lahko lepo razvidna na primeru preproge Sierpinskega.



Slika 3 - preproga Sierpinskega

4.3. Fraktalna dimenzija

Najbolj presenetljivo dejstvo je to, da dimenzija fraktalov ni ena, dve ali tri, temveč je njihova dimenzija med dvema celima številoma. Ideja »dimenzije« ima veliko matematičnih definicij, ki pa niso nujno ekvivalentne.

Ena izmed definicij fraktalne dimenzije je tako imenovana *dimenzija samopodobnosti*, ki jo lahko določimo samo za sebi podobne objekte. Opazujmo daljico, kvadrat in kocko, tipične objekte dimenzij 1, 2 oziroma 3. Vsakega od teh objektov razdelimo na enake dele, ki so na primer za faktor $\frac{1}{4}$ manjši od originala. Daljica razpade na 4 dele, $N(4) = 4 = 4^1$. Kvadrat se razdeli na $N(4) = 16 = 4^2$ delov in kocka na $N(4) = 64 = 4^3$ delov. Namesto 4 bi lahko izbrali kakšen drug faktor, na primer *k*. Objekt razpade na $N(k) = k^n$ delov, kjer je *n* enak dimenziji objekta. [4]

Sedaj pa vzemimo poljuben objekt *X* iz realnega *d*-dimenzionalnega vektorskega prostora $\mathbb{R}^d$, ki dopušča delitev na $N(k) = k^d$ kongruentnih delov (to pomeni, da lahko s pomočjo zasuka in vzporednega premika enega preslikamo na drugega). [4] Vsak objekt je kopija originala, skrčena za faktor *k*. Dimenzija samopodobnosti objekta *X* je tista vrednost *d*, za katero velja:

$$N(k) = k^d.$$

Od tod sledi:

$$d = \frac{logN(k)}{\log(k)}$$

4.4. Primeri fraktalov

Kot primere fraktalov smo navedli dva, ki ju bomo uporabili v praktičnem delu naše raziskovalne naloge. To sta preproga Sierpinskega in Kochova snežinka.

Ukvarjali se bomo s prvimi petimi iteracijami.

Preproga Sierpinskega

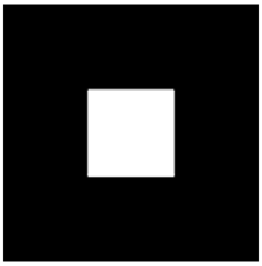
Pri konstrukciji preproge Sierpinskega kvadrat razdelimo na devet manjših kvadratov in izločimo srednjega, seveda brez njegovega roba. Na vsakem od ostalih kvadratov ponovimo opisan postopek. [4]

Postopek nadaljujemo v nedogled. V limiti nam ostane množica, ki jo imenujemo preproga Sierpinskega. [4]

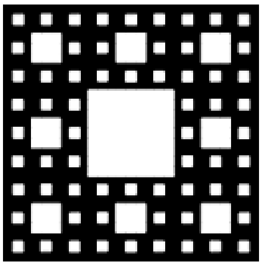
Skupna ploščina izločenih kvadratov v limiti:

$$\sum_{n=1}^{\infty}\left(\frac{1}{3^n}\right)^2 8^{n-1} = \sum_{n=1}^{\infty}\frac{8^{n-1}}{9^n} = \frac{1}{9}\cdot\frac{1}{1-\frac{8}{9}} = 1$$

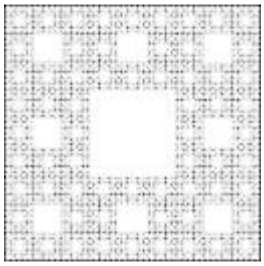
Dimenzija: $\frac{\log 8}{\log 3} = 1{,}8928\ldots$ [4]



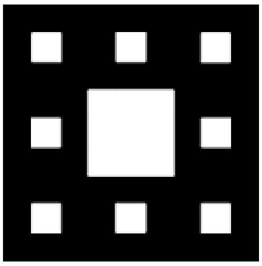
Slika 4 - preproga Sierpinskega, 1. iteracija



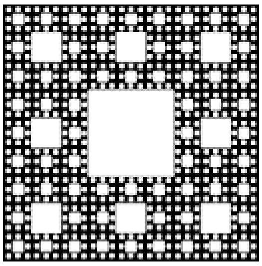
Slika 6 - preproga Sierpinskega, 3. iteracija



Slika 8 - preproga Sierpinskega, 5. iteracija



Slika 5 - preproga Sierpinskega, 2. iteracija



Slika 7 - preproga Sierpinskega, 4. iteracija

Kochova snežinka

Konstrukcijo ene izmed stranic Kochove snežinke poteka identično kot konstrukcija Kochove krivulje. Začnemo z daljico dolžine 1. Dolžina ni pomembna, izbrali smo jo le zaradi preprostejšega računa. V prvem koraku izločimo srednjo tretjino daljice in jo nadomestimo z dvema daljicama dolžine $\frac{1}{3}$, tako da ti dve daljici s tisto, ki smo jo izločili, tvorita enakostranični trikotnik. Vrh, kjer se stikata dodani daljici, se nahaja nad prvotno daljico. [4]

Rezultat prvega koraka je lomljena črta, sestavljena iz štirih enako dolgih stranic. Ta postopek spet ponovimo na vsaki izmed njih. [4]

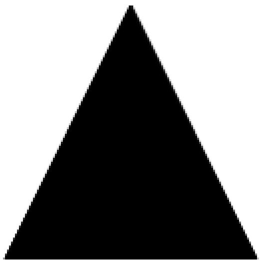
Postopek nadaljujemo v nedogled. V limiti dobimo Kochovo krivuljo. [4]

Za konstrukcijo Kochove snežinke pa namesto daljice vzamemo trikotnik. Omejujejo ga tri daljice, zato je postopek identičen konstrukciji Kochove krivulje.

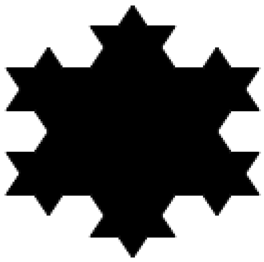
Enačba za ploščino Kochove snežinke:

$$1 + 3\sum_{n=1}^{\infty} \frac{4^{n-1}}{9^n} = \frac{8}{5}$$

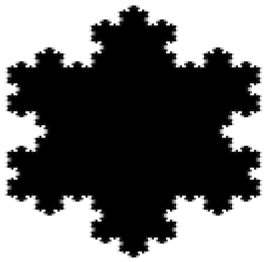
Dimenzija: $\frac{\log 4}{\log 3} = 1{,}2618 \ldots$ [4]



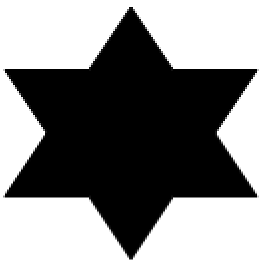
Slika 9 - Kochova snežinka, 1. iteracija



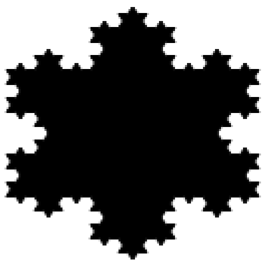
Slika 11 - Kochova snežinka, 3. iteracija



Slika 13 - Kochova snežinka, 5. iteracija



Slika 10 - Kochova snežinka, 2. iteracija



Slika 12 - Kochova snežinka, 4. iteracija

5. L-sistemi

Lindenmayerjeve sistemi ali L-sisteme je leta 1986 uvedel madžarski biolog Aristid Lindenmayer. Sprva so bili namenjeni opisu rasti mnogoceličnih organizmov, pozneje pa so jih posplošili za vizualizacijo rastlin in drugih sebi podobnih struktur. [8]

L-sistemi delujejo kot tvorba nizov znakov oziroma besed s ponavljajočim se vstavljanjem novih podnizov po določenih pravilih oziroma slovnici. Torej, če bi gledali rast mnogoceličnih organizmov, bi nize predstavljale celice, pravila pa bi določala način rasti organizma. [8]

Pravila:

L-sistem tvorijo:

- množica spremenljivk in konstant V ter množica posebnih znakov T, obe množici tvorita abecedo A,
- množica nizov znakov abecede A,
- aksiom, ki je niz simbolov iz množice A in opisuje začetno stanje sistema,
- pravila iteracij P, ki vsakemu znaku iz množice V priredijo niz znakov iz množice A.

Aksiom in njegove izpeljanke imenujemo besede L-sistema.

Primer L-sistema – rast alg:

Spremenljivke: A, B

Aksiom: A

Pravila iteracij: $(A \rightarrow AB), (B \rightarrow A)$

Formalni zapis:

$n = 0$: A

$n = 1$: AB

$n = 2$: ABA

$n = 3$: $ABAAB$

$n = 4$: $ABAABABA$

$n = 5$: $ABAABABAABAAB$

$n = 6$: $ABAABABAABAABABAABABA$

$n = 7$: $ABAABABAABAABABAABABAABAABABAABAAB$

Grafični prikaz:

```
n=0:             A
                / \
n=1:           A   B
              / |    \
n=2:         A B      A
            / |  |     | \
n=3:       A B A      A  B
          / |  |  | \     | \   \
n=4:    A B A A B   A B  A
```

Opazimo, da so števila alg členi Fibonaccijevega zaporedja. [9]

6. Hilbertova krivulja

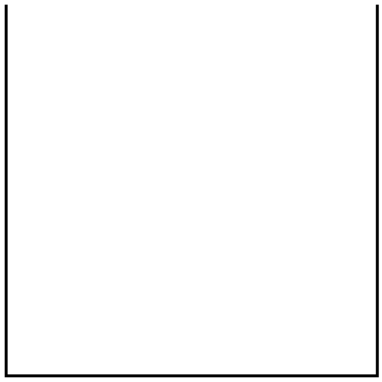
Hilbertova krivulja je zvezna fraktalna krivulja. Je ena izmed Peanovih krivulj, kar pomeni, da je njena Hausdorffova dimenzija točno dve in napolni cel prostor. [10]

Evklidska dolžina *n*-te stopnje Hilbertove krivulje je enaka $2^n - \left(\frac{1}{2}\right)^n$. [11]

To pomeni, da dolžina narašča eksponentno, področje krivulje pa je omejeno s kvadratom, torej likom s končno ploščino.

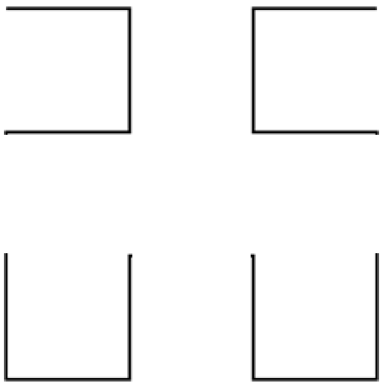
Poglejmo si, kako narišemo Hilbertovo krivuljo.

Osnovni element krivulje je lik na sliki 14, to je psevdo-Hilbertova krivulja prvega reda.

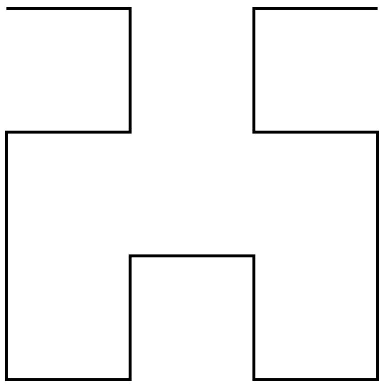


Slika 14 - Hilbertova krivulja 1. reda

V naslednjem koraku uporabimo štiri osnovne elemente in jih, orientirane kot na sliki 15, med seboj povežemo s tremi daljicami, kot je prikazano na sliki 16. Tako dobimo psevdo-Hilbertovo krivuljo drugega reda.



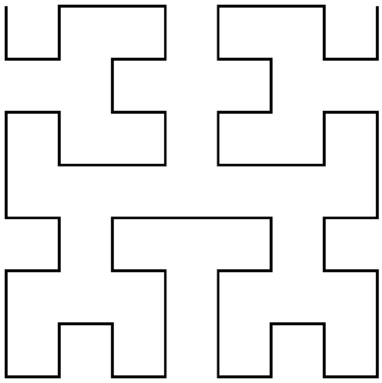
Slika 15 - risanje Hilbertove krivulje 2. reda



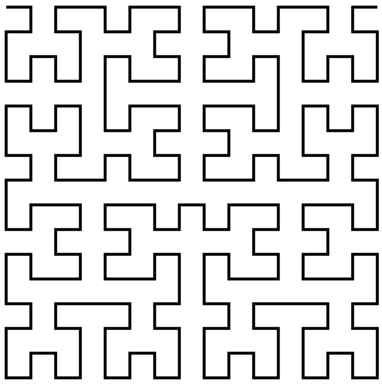
Slika 16 - Hilbertova krivulja 2. reda

Za vsak naslednji korak uporabimo zadnjo dobljeno krivuljo kot osnovni element. Po neskončno mnogo korakih dobimo psevdo-Hilbertovo krivuljo, ki zapolni vse piksle.

Hilbertova krivulja pa v neskončnosti zapolni celoten kvadrat.



Slika 17 - Hiblertova krivulja 3. reda



Slika 18 - Hiblertova krivulja 4. reda

Še enostavneje pa njen nastanek ponazorimo s pomočjo L-sistemov: [8]

Abeceda: $A\ B$

Konstante:

$F\ (premik\ naprej)$

$+\ (zasuk\ za\ 90'\ v\ pozitivno\ smer)$

$-\ (zasuk\ za\ 90'\ v\ negativno\ smer)$

Aksiom: A

Pravila iteracij:

$A\ \rightarrow\ +B\ F-\ A\ F\ A-\ F\ B+$

$B\ \rightarrow\ -A\ F+\ B\ F\ B+\ F\ A-$

7. Prevajanje fraktala

Ko smo se odločali, kako bi prevajali fraktal v zvok, smo prišli do sklepa, da moramo najti način, ki bi nam omogočal, da lahko določimo vse točke na sliki. To smo dosegli s pomočjo psevdo-Hilbertove krivulje.

Potem pa smo spoznali, da lahko sliko fraktala prevajamo v zvok na dva različna načina.

Prvi način je, da smo vsem točkam določili frekvenco in jih seštelі. Tako smo ustvarili eno frekvenco cele slike. To smo poimenovali simultan način.

V drugem načinu pa smo vsako točko postavili v zaporedje, glede na njihovo lego na Hilbertovi krivulji, in jih prevajali v zvok enega za drugim. Ta način smo poimenovali konsekutiven način prevajanja slike fraktala.

Zvočni posnetki so naloženi tudi na Githubu. Za optimalen pregled rezultatov naše naloge si celotni projekt naložite s pomočjo GIT programa:

ukaz git clone https://github.com/raziskovalnagimnm/ZvokiFraktalov.git

Pazite le na to, da je projekt večji od 40GB in boste ob navadni internetni povezavi potrebovali kar nekaj časa, da se bo Github repozitorij prenesel.

Za odpiranje in predvajanje datotek je optimalna uporaba brskalnika Google Chrome in VLC Media Playerja.

7.1. Razlaga prevajanja fraktala

7.1.1. Simultano zaporedje

Program 1 smo poimenovali simultano zaporedje, saj nam naenkrat predvaja celotno sliko fraktala, torej vse piksle in njihove frekvence hkrati.

7.1.1.1. Program

Zapis v Pythonu:

```
#! /usr/bin/env python
```

Uvoz ukazov za izvedbo programa:

```
import hilbert_curve as hc
from PIL import Image
import math
import wave
import numpy as np
from scipy.io import wavfile
import scipy.io
```

Ime datoteke, ki jo hočemo prevajati v zvok:

```
input_f='SC64x64-2'
filler=''
simultano='-S-'
input_file = input_f+'.png'
output_wav = input_f+simultano+filler+'.wav'
```

Določitev časovne dolžine končne wav datoteke:

```
num_seconds = 16
```

Funkcija, ki naredi sinusoide:

```
def generateSine(f,t):
        return ((math.sin(t/44100*2*math.pi*f)))
```

Zapis slike v spomin:

```
img = Image.open(input_file).convert("L")
pixels = img.load()
x,y = img.size


output = []
i = 0
```

Umeščanje pikslov po Psevdo-Hilbertovi krivulji:

```
for i in range(0,x**2):
        hilbert = hc.d2xy(math.log(x*y,2),i)
        output.append(pixels[hilbert])
```

Ustvarjanje zaporedja...:

```
outputAudio = [0]*(44100*num_seconds)
```

Generiranje sinusoid za različne frekvence:

```
pixel_count = 0
for pixel in output:
        if pixel_count % 10 == 0:
                print (pixel_count,"pikslov kočanih od", x*y, end="\n")
        pixel_count += 1
        frequency = (pixel*8)+220
        for t in range(0,len(outputAudio)):
                outputAudio[t] += generateSine(frequency,t)
```

Izračun maksimuma zaporedja:

```
maxx=max(outputAudio)
minn=abs(min(outputAudio))
if maxx>=minn:
        ma=maxx
else:
        ma=minn
```

Umeščanje vrednosti med -1 in 1:

```
for i in range(0,len(outputAudio)):
        if i % 10000 == 0:
                print(i,"od", len(outputAudio), end="\n")
        outputAudio[i] = outputAudio[i]/ma
```

Generiranje wav datoteke:

```
scipy.io.wavfile.write(output_wav,44100,np.asarray(outputAudio,dtype=fl
oat))
```

7.1.1.2. Povzetek programa

V kodo smo vstavili sliko fraktala in določili trajanje ter pogoj, da program opravi svoje delo le, če je slika enaka kvadratu potence števila dve (2^n).

Vsakemu pikslu smo priredili točko na psevdo-Hilbertovi krivulji. Vsaki točki je bila na podlagi svetlosti barvne lestvice od 0 do 255 določena neka frekvenca. Tukaj 0 pomeni črno in 255 belo barvo.

Najvišja frekvenca je 2260 Hz (predstavlja bel piksel na sliki), najnižja frekvenca je 220 Hz (predstavlja črn piksel na sliki).

Vsem frekvencam so prirejene sinusoide, ki so seštete ena na drugo. To pomeni, da se vsi piksli predvajajo hkrati.

Program nato izračuna absolutni maksimum in nato celotno zaporedje deli s to vrednostjo, da se vse vrednosti zaporedja nahajajo med 1 in -1.

Nato se izpiše WAV zvočna datoteka.

7.1.2. Konsekutivno zaporedje

Program 2 smo poimenovali konsekutivno zaporedje, saj predvaja vsak piksel posebej, v določenem zaporedju.

7.1.2.1. Program

Zapis v Pythonu:

```
#! /usr/bin/env python
```

Uvoz ukazov za za izvedbo programa

```
import hilbert_curve as hc
from PIL import Image
import math
import wave
import numpy as np
from scipy.io import wavfile
import scipy.io
```

Ime datoteke, ki jo hočemo prevajati (skozi program) v zvok:

```
input_f='SC64x64-5'
filler=''
konsekutivno='-K-'
input_file = input_f+'.png'
output_wav =  input_f+konsekutivno+filler+'.wav'
output_txt = input_f+konsekutivno+filler+'-Before.txt'
output_txt1 = input_f+konsekutivno+filler+'-After.txt'
```

Določitev časovne dolžine končne wav datoteke:

```
num_seconds = 16
length = (num_seconds*44100)
```

Funkcija, ki naredi sinusoide:

```
def generateSine(f,t):
       return ((math.sin(t/44100*2*math.pi*f)))

*Zapis slike v spomin
img = Image.open(input_file).convert("L")
pixels = img.load()
x,y = img.size

output = []
i = 0
```

Prevajanje pixlov iz slike v zaporedje s pomočjo psevdo-Hilbertove krivulje:

```
for i in range(0,x**2):
        hilbert = hc.d2xy(math.log(x*y,2),i)
        output.append(pixels[hilbert])

outputAudio = [0]*length

startA=0
stopA=0
chop1 = int(length/len(output))
pixel_count = 0
```

Generiranje sinusoid za različne frekvence:

```
for pixel in output:
        if pixel_count % 10 == 0:
                print   (pixel_count,"pixels   completed   out   of",   x*y,
end="\n")

        pixel_count += 1
        frequency = (pixel*8)+220

        startA = (pixel_count-1)*chop1
        stopA = startA + chop1 - 1

        if stopA>length:
                stopA = length
```

Zaporedna postavitev sinusoid:

```
        for t in range(startA,stopA):
                outputAudio[t] += generateSine(frequency,t)
```

Izpis zaporedja, ki zapiše zaporedje pred umeščanjem vrednosti med -1 in 1

```
#np.savetxt(output_txt,outputAudio)
```

Izračun maksimuma zaporedja:

```
maxx=max(outputAudio)
minn=abs(min(outputAudio))
if maxx>minn:
        mapp = maxx
else:
        mapp = minn
```

Umeščanje vrednosti med -1 in 1:

```
for i in range(0,len(outputAudio)):
        outputAudio[i] = outputAudio[i] / mapp
```

Izpis zaporedja, ki zapiše wav datoteko:

```
#np.savetxt(output_txt1,outputAudio)
```

Generiranje wav datoteke:

```
scipy.io.wavfile.write(output_wav,44100,np.asarray(outputAudio,dtype=fl
oat))
```

7.1.2.2. Povzetek programa

Program deluje na skoraj enak način kot prvi, le da ne predvaja vseh pikslov oz. sinusoid hkrati, temveč jih postavi v zaporedje in jih predvaja kot zaporedje frekvenc. Torej predvaja vsak piksel posebej v zaporedju, kot se nahajajo na Hilbertovi krivulji.

V prvem poglavju smo to zapisali na naslednji način:

```
for t in range(startA,stopA):
                        outputAudio[t] += generateSine(frequency,t)
```

v drugem programu pa:

```
for t in range(0,len(outputAudio)):
                        outputAudio[t] += generateSine(frequency,t)
```

7.2. Črna in bela

7.2.1. Bela

Beli barvi smo določili frekvenco 2260 Hz. Ko torej predvajamo sliko, ki je v celoti bele barve, dobimo enoten zvok s to frekvenco.

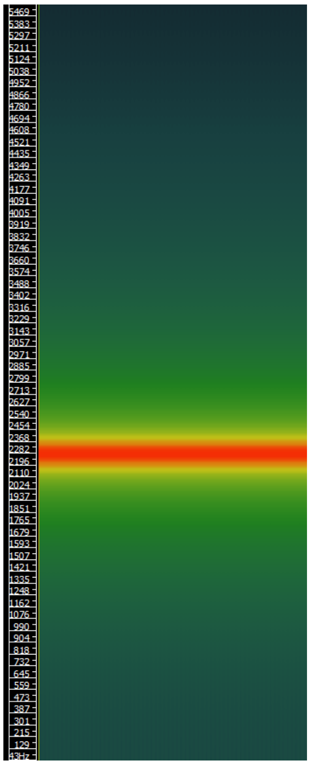
V obeh programih dobimo iz bele slike enak zvok in to, ali predvajamo vse piksle skupaj ali vsakega posebej, na frekvenco nima vpliva, saj je celotna slika v eni barvi.

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%201%20-%20Simultano/128x128W-S-.wav

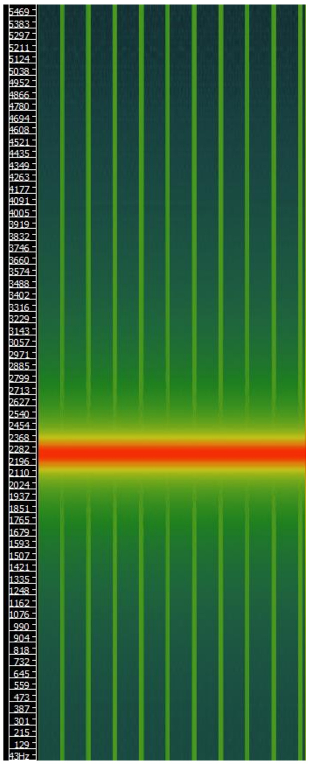
https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%202%20-%20Konsekutivno/DolgeWavDatoteke/128x128W-K-4096s.wav

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%202%20-%20Konsekutivno/128x128W-K-4096s-16s.wav

Na spektrogramih vidimo, da je res zastopana le frekvenca 2260 Hz, sinusoide pa so goste, ker je frekvenca visoka.

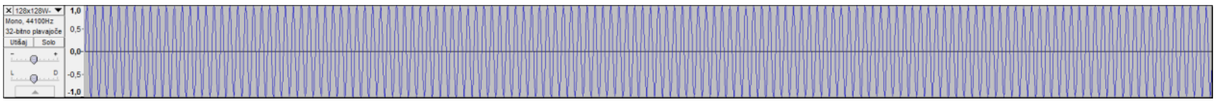


Spektrogram 1 - bela barva, simultano

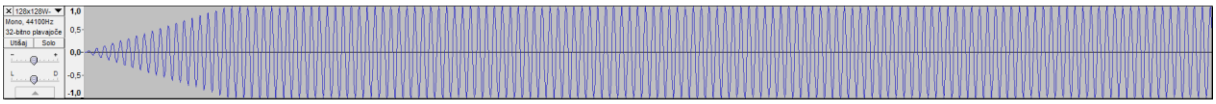


Spektrogram 2 - bela barva, konsekutivno

Slika 19 - bela



Sinusoida 1 - bela barva, simultano



Sinusoida 2 - bela barva, konsekutivno

7.2.2. Črna barva

Črni barvi smo določili frekvenco 220 Hz. Ko torej predvajamo sliko, ki je v celoti črne barve, dobimo enoten zvok s frekvenco 220 Hz.

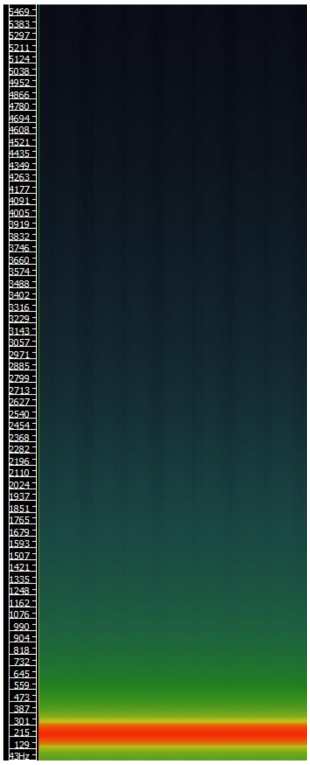
V obeh programih dobimo iz črne slike enak zvok in to, ali predvajamo vse piksle skupaj ali vsakega posebej, na zvok nima vpliva, saj je celotna slika le v eni barvi.

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%201%20-%20Simultano/128x128B-S-.wav

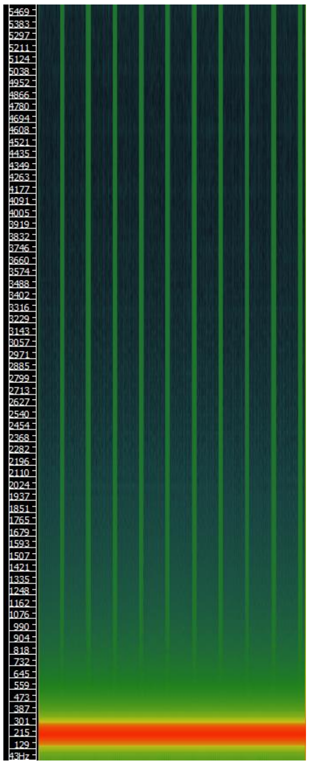
https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%202%20-%20Konsekutivno/DolgeWavDatoteke/128x128B-K-4096s.wav

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%202%20-%20Konsekutivno/128x128B-K-4096s-16s.wav

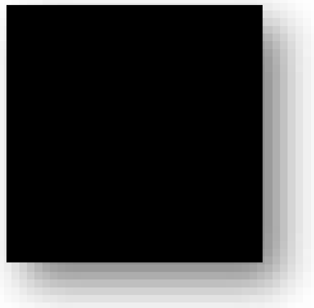
Na spektrogramih vidimo, da je zastopana frekvenca 220 Hz, na sinusoidah pa lahko lepo vidimo, da je frekvenca v primerjavi s sinusoidami oz. frekvenco bele barve dosti nižja.



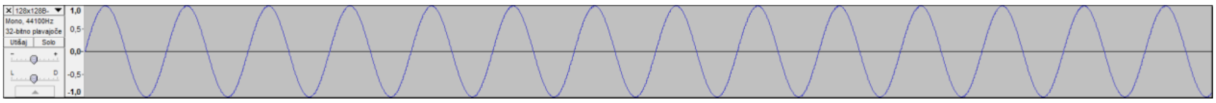
Spektrogram 3 - črna barva, simultano



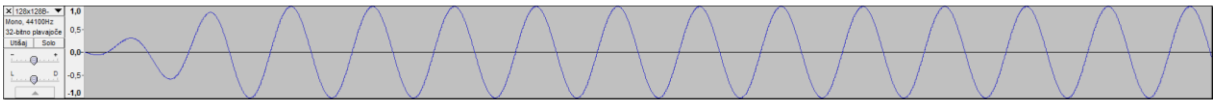
Spektrogram 4 - črna barva, konsekutivno



Slika 20 - črna



Sinusoida 3 - črna barva, simultano



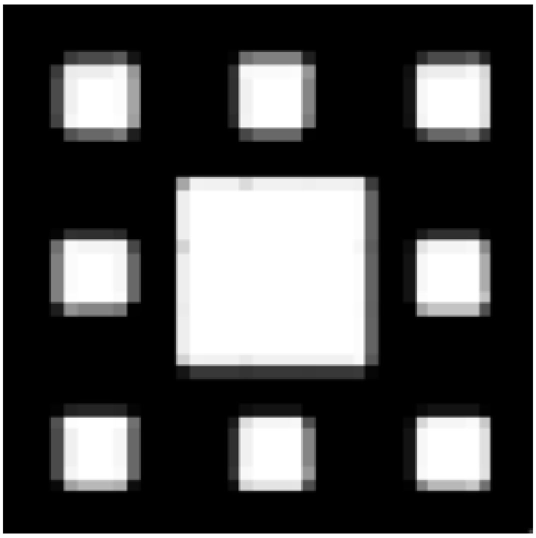
Sinusoida 4 - črna barva, konsekutivno

7.2.3. Sivina

Razlog, da se pri slikah preproge Sierpinskega in Kochove snežinke na prehodu iz črne v belo barvo pojavijo tudi odtenki sive barve, je v tem, da je meja med črnimi in belimi pikami določena s potencami števila tri, naša slika pa je dimenzije $2^n \times 2^n$, stranica torej ni deljiva s tri. Zato na robovih pride do popačitve – anti-aliasinga.

Na slikah sta izseka iz slike naših fraktalov toliko približana, da lahko vidimo sivino na prehodu iz črne v belo barvo.

Slika 21 - izsek iz približane slike Kochove snežinke



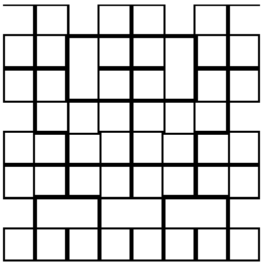
Slika 22 - izsek iz približane slike preproge Sierpinskega

7.3. Hilbertova krivulja

Tu so predstavljeni rezultati prevajanja psevdo-Hilbertove krivulje. Za lažjo predstavo smo rezultatom priložili sliko fraktala, ki smo ga prevajali, in spektrogram.

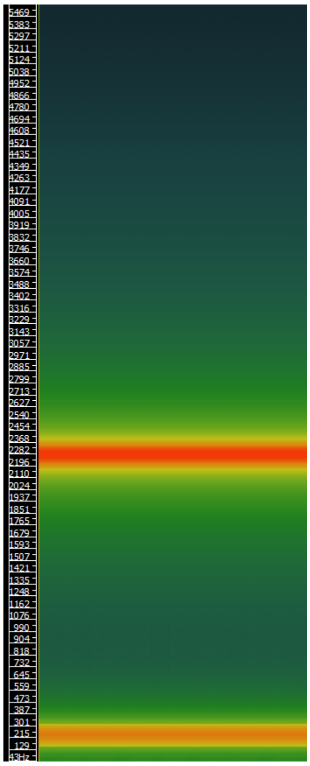
7.3.1. Simultano zaporedje – prvi način predvajanja

a) 4. iteracija



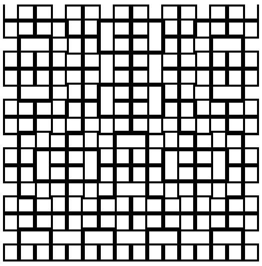
Slika 23 - Hilbertova krivulja, 4. iteracija

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/Hilbert%20Curve/WAV%20Datoteke/Program%201%20-%20Simultano/16x16HC-128x128-S-.wav



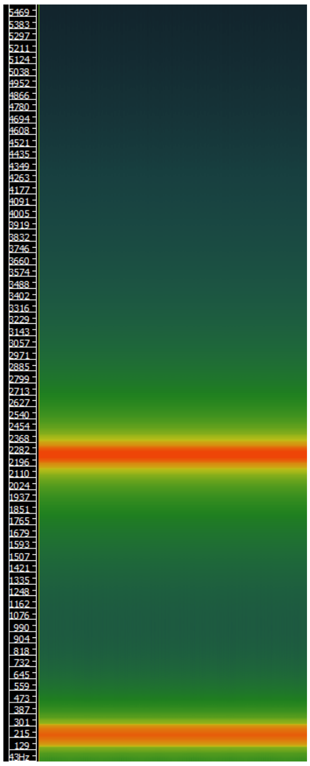
Spektrogram 5 - Hilbertova krivulja, 4. iteracija, simultano

b) 5. iteracija



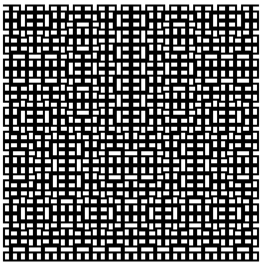
Slika 24 - Hilbertova krivulja, 5. iteracija

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/Hilbert%20Curve/WAV%20Datoteke/Program%201%20-%20Simultano/8x8HC-128x128-S-.wav



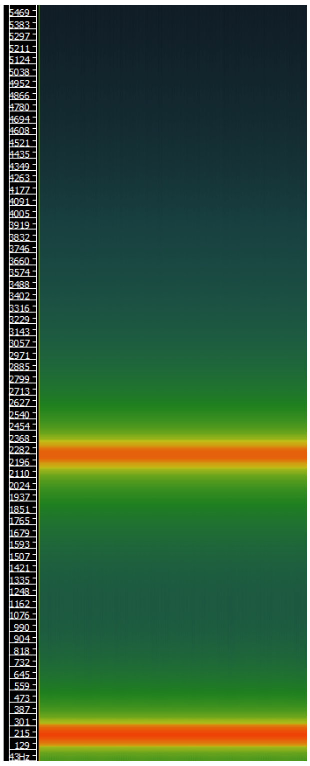
Spektrogram 6 - Hilbertova krivulja, 5. iteracija, simultano

c) 6. iteracija



Slika 25 - Hilbertova krivulja, 6. iteracija

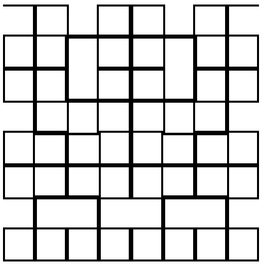
https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/Hilbert%20Curve/WAV%20Datoteke/Program%201%20-%20Simultano/4x4HC-128x128-S-.wav



Spektrogram 7 - Hilbertova krivulja, 6. iteracija, simultano

7.3.2. Konsekutivno zaporedje – drugi način predvajanja

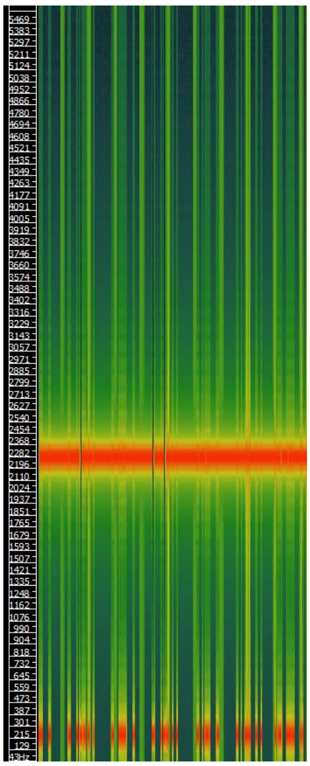
a) 4. iteracija



Slika 26 - Hilbertova krivulja, 4. iteracija

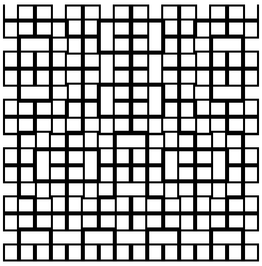
https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/Hilbert%20Curve/WAV%20Datoteke/Program%202%20-%20Konsekutivno/DolgeWavDatoteke/16x16HC-128x128-K-4096s.wav

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/Hilbert%20Curve/WAV%20Datoteke/Program%202%20-%20Konsekutivno/16x16HC-128x128-K-4096s-16s.wav



Spektrogram 8 - Hilbertova krivulja, 4. iteracija, konsekutivno

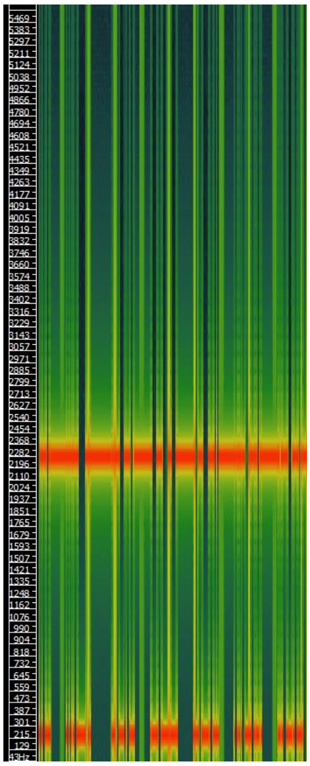
b) 5. iteracija



Slika 27 - Hilbertova krivulja, 5. iteracija

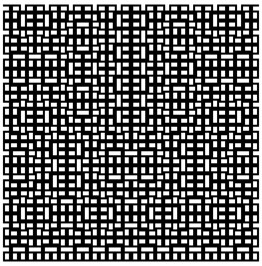
https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/Hilbert%20Curve/WAV%20Datoteke/Program%202%20-%20Konsekutivno/DolgeWavDatoteke/8x8HC-128x128-K-4096s.wav

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/Hilbert%20Curve/WAV%20Datoteke/Program%202%20-%20Konsekutivno/8x8HC-128x128-K-4096s-16s.wav



Spektrogram 9 - Hilbertova krivulja, 5. iteracija, konsekutivno

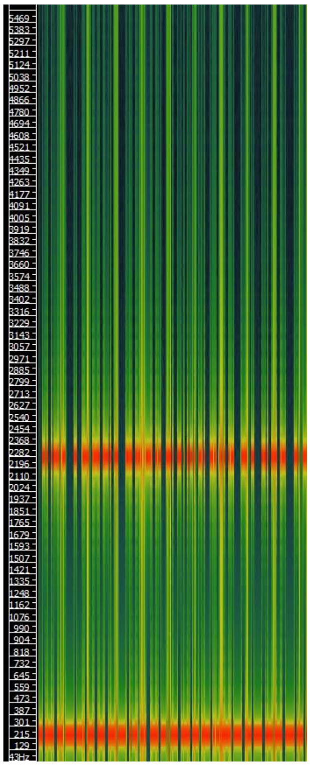
c) 6. iteracija



Slika 28 - Hilbertova krivulja, 6. iteracija

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/Hilbert%20Curve/WAV%20Datoteke/Program%202%20-%20Konsekutivno/DolgeWavDatoteke/4x4HC-128x128-K-4096s.wav

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/Hilbert%20Curve/WAV%20Datoteke/Program%202%20-%20Konsekutivno/4x4HC-128x128-K-4096s-16s.wav

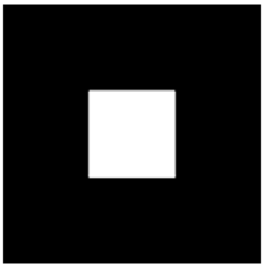


Spektrogram 10 - Hilbertova krivulja, 6. iteracija, konsekutivno

7.4. Preproga Sierpinskega

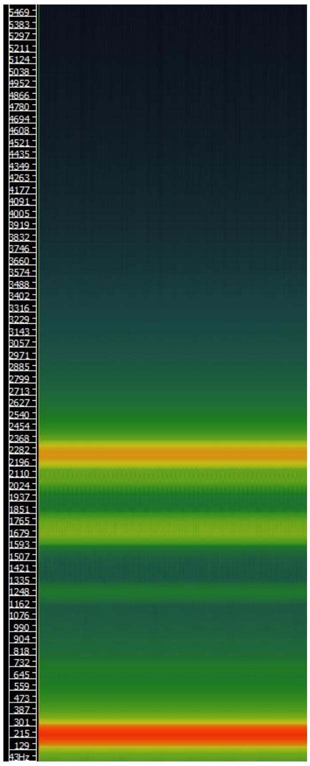
7.4.1. Simultano zaporedje – prvi način predvajanja

a) 1. iteracija

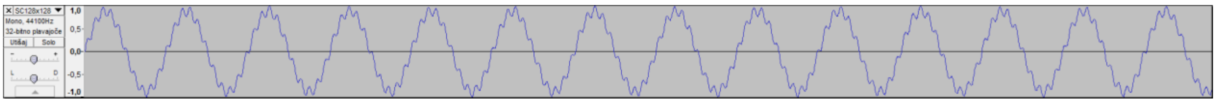


Slika 29 - preproga Sierpinskega, 1. iteracija

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%201%20-%20Simultano/Preproga%20Sierpinskega/SC128x128-1-S-.wav

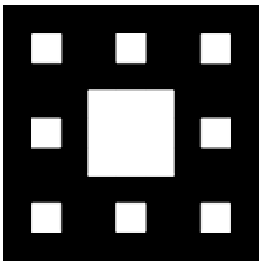


Spektrogram 11 - preproga Sierpinskega, 1. iteracija, simultano



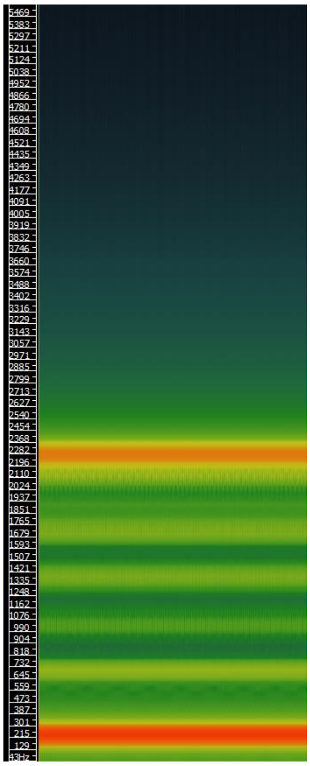
Sinusoida 5 - preproga Sierpinskega, 1. iteracija, simultano

b) 2. iteracija

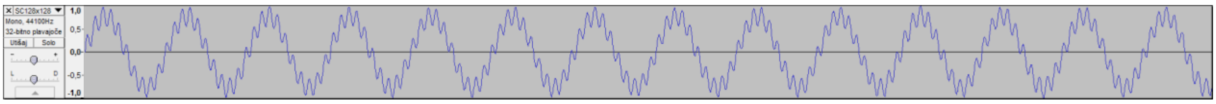


Slika 30 - preproga Sierpinskega, 2. iteracija

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%201%20-%20Simultano/Preproga%20Sierpinskega/SC128x128-2-S-.wav

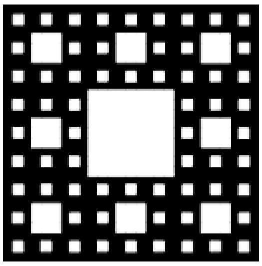


Spektrogram 12 - preproga Sierpinskega, 2. iteracija, simultano



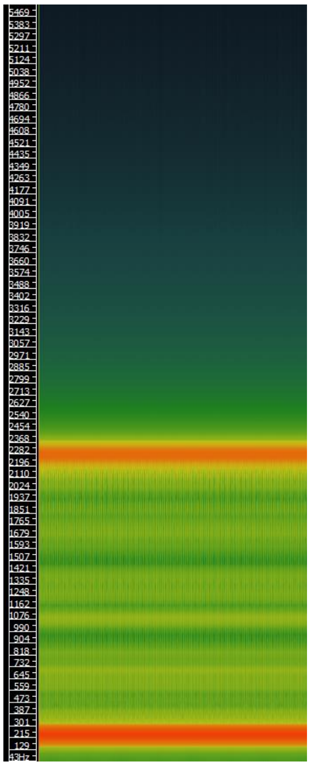
Sinusoida 6 - preproga Sierpinskega, 2. iteracija, simultano

c) 3. iteracija

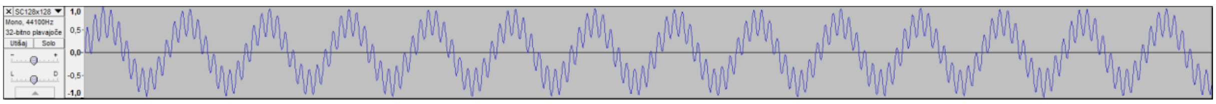


Slika 31 - preproga Sierpinskega, 3. iteracija

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%201%20-%20Simultano/Preproga%20Sierpinskega/SC128x128-3-S-.wav

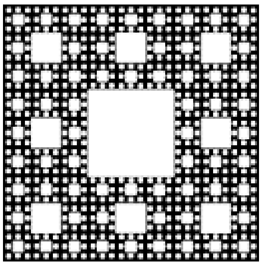


Spektrogram 13 - preproga Sierpinskega, 3. iteracija, simultano



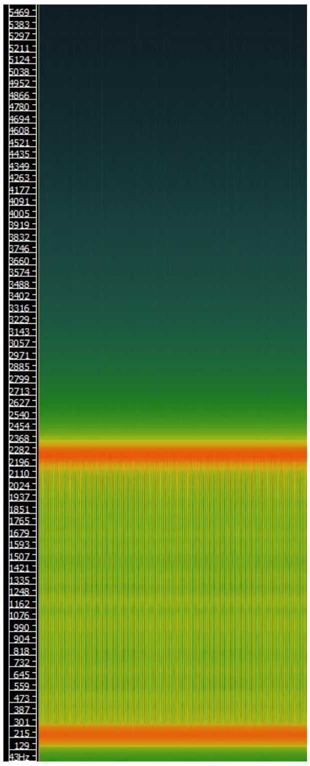
Sinusoida 7 - preproga Sierpinskega, 3. iteracija, simultano

d) 4. iteracija

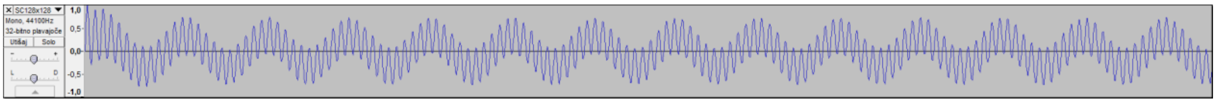


Slika 32 - preproga Sierpinskega, 4. iteracija

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%201%20-%20Simultano/Preproga%20Sierpinskega/SC128x128-4-S-.wav

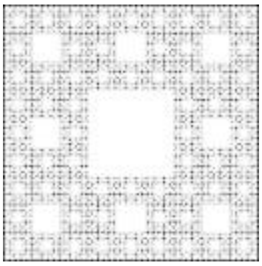


Spektrogram 14 - preproga Sierpinskega, 4. iteracija, simultano



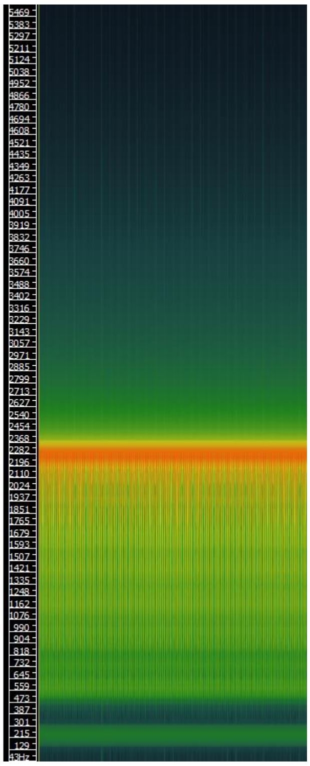
Sinusoida 8 - preproga Sierpinskega, 4. iteracija, simultano

e) 5. Iteracija

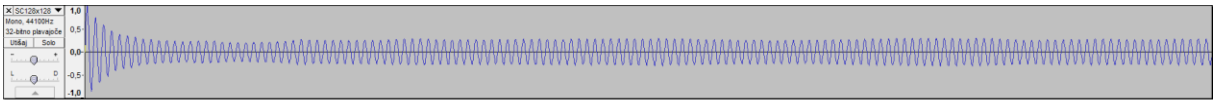


Slika 33 - preproga Sierpinskega, 5. iteracija

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%201%20-%20Simultano/Preproga%20Sierpinskega/SC128x128-5-S-.wav



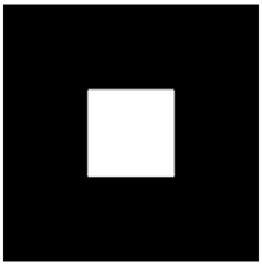
Spektrogram 15 - preproga Sierpinskega, 5. iteracija, simultano



Sinusoida 9 - preproga Sierpinskega, 5. iteracija, simultano

7.4.2. Konsekutivno zaporedje – drugi način predvajanja

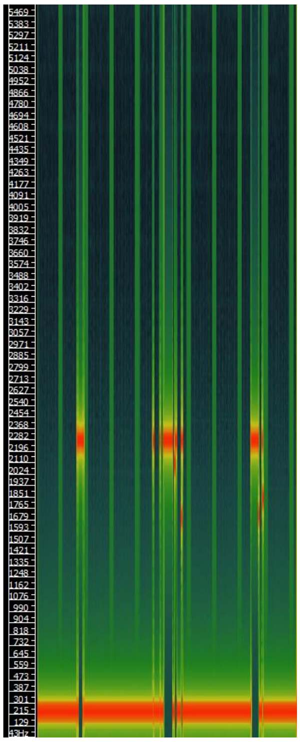
a) 1. iteracija



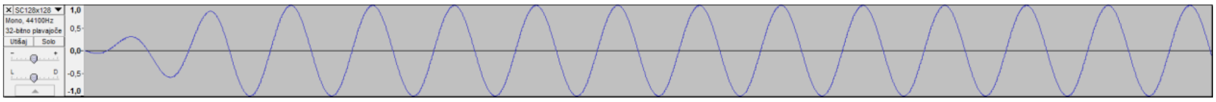
Slika 34 - preproga Sierpinskega, 1. iteracija

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%202%20-%20Konsekutivno/DolgeWavDatoteke/SC128x128-1-K-4096s.wav

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%202%20-%20Konsekutivno/Preproga%20Sierpinskega/SC128x128-1-K-4096s-16s.wav

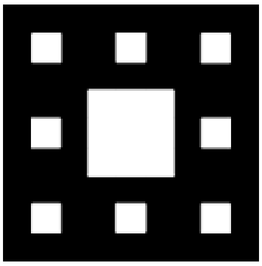


Spektrogram 16 - preproga Sierpinskega, 1. iteracija, konsekutivno



Sinusoida 10 - preproga Sierpinskega, 1. iteracija, konsekutivno

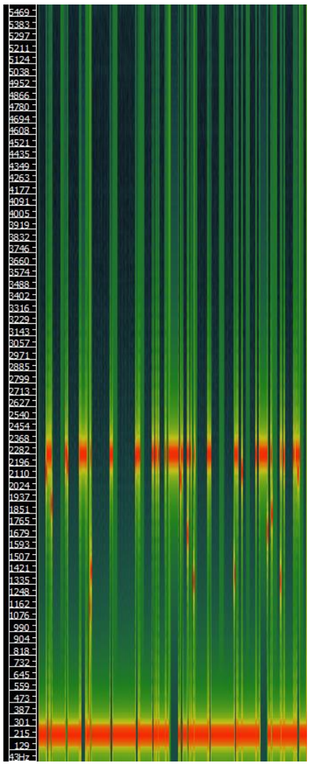
b) 2. iteracija



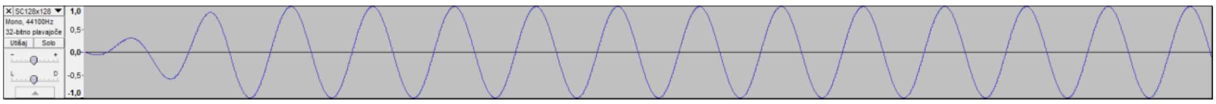
Slika 35 - preproga Sierpinskega, 2. iteracija

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%202%20-%20Konsekutivno/DolgeWavDatoteke/SC128x128-2-K-4096s.wav

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%202%20-%20Konsekutivno/Preproga%20Sierpinskega/SC128x128-2-K-4096s-16s.wav

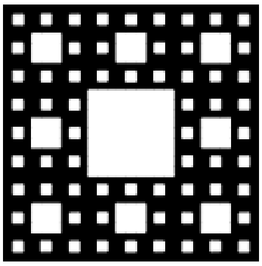


Spektrogram 17 - preproga Sierpinskega, 2. iteracija, konsekutivno



Sinusoida 11 - preproga Sierpinskega, 2. iteracija, konsekutivno

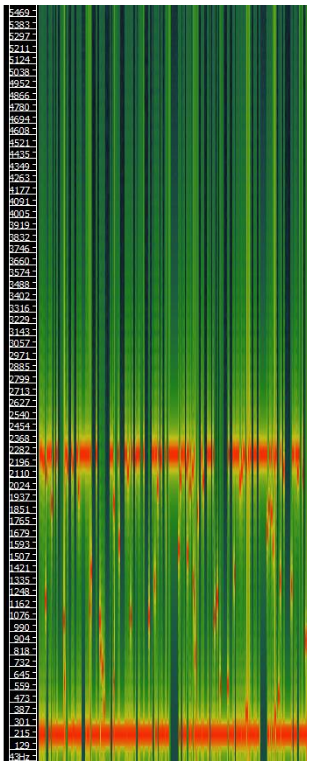
c) 3. iteracija



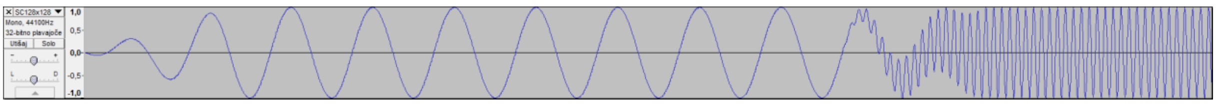
Slika 36 - preproga Sierpinskega, 3. iteracija

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%202%20-%20Konsekutivno/DolgeWavDatoteke/SC128x128-3-K-4096s.wav

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%202%20-%20Konsekutivno/Preproga%20Sierpinskega/SC128x128-3-K-4096s-16s.wav

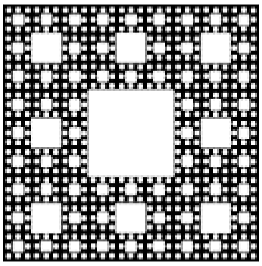


Spektrogram 18 - preproga Sierpinskega, 3. iteracija, konsekutivno



Sinusoida 12 - preproga Sierpinskega, 3. iteracija, konsekutivno

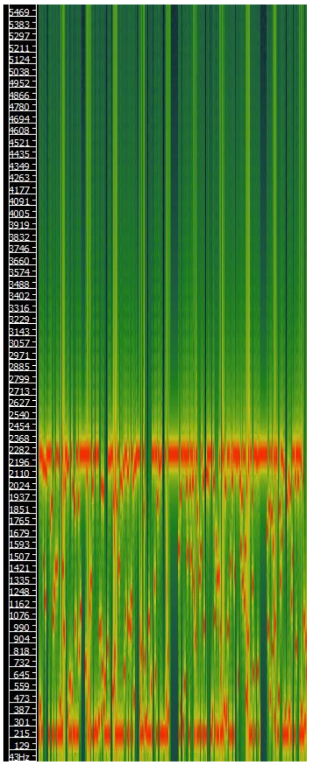
d) 4. iteracija



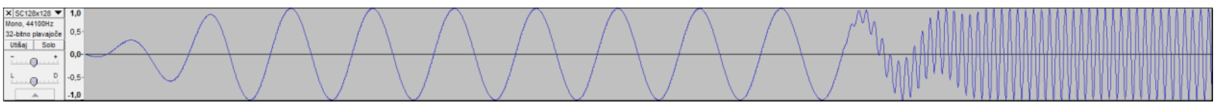
Slika 37 - preproga Sierpinskega, 4. iteracija

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%202%20-%20Konsekutivno/DolgeWavDatoteke/SC128x128-4-K-4096s.wav

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%202%20-%20Konsekutivno/Preproga%20Sierpinskega/SC128x128-4-K-4096s-16s.wav

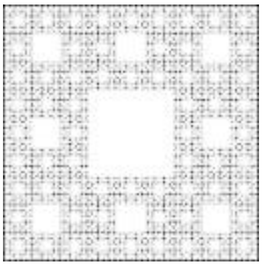


Spektrogram 19 - preproga Sierpinskega, 4. iteracija, konsekutivno



Sinusoida 13 - preproga Sierpinskega, 4. iteracija, konsekutivno

e) 5. iteracija

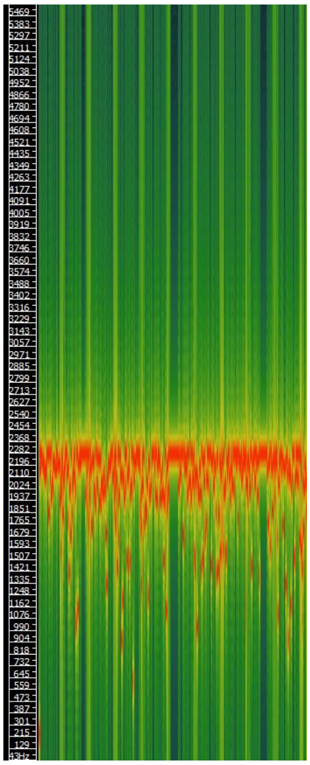


Slika 38 - preproga Sierpinskega, 5. iteracija

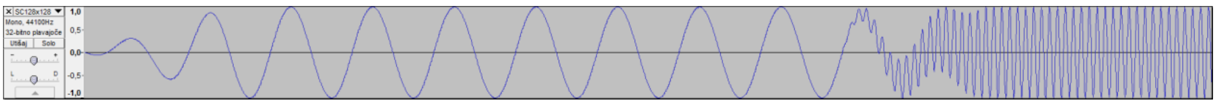
https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%202%20-%20Konsekutivno/DolgeWavDatoteke/SC128x128-5-K-4096s.wav

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%202%20-%20Konsekutivno/Preproga%20Sierpinskega/SC128x128-5-K-4096s-16s.wav

Spektrogram:



Spektrogram 20 - preproga Sierpinskega, 5. iteracija, konsekutivno

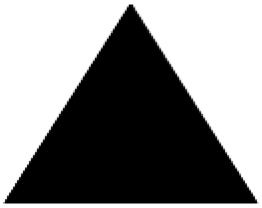


Sinusoida 14 - preproga Sierpinskega, 5. iteracija, konsekutivno

7.5. Kochova snežinka

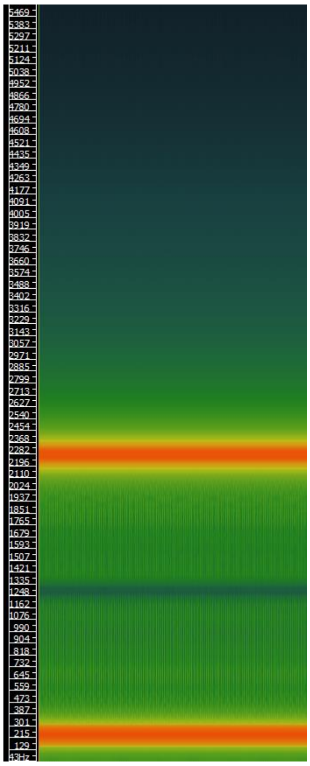
7.5.1. Simultano zaporedje – prvi način predvajanja

a) 1. iteracija

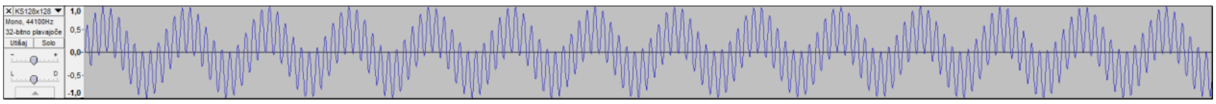


Slika 39 - Kochova snežinka, 1. iteracija

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%201%20-%20Simultano/Kochova%20sne%C5%BEinka/KS128x128-1-S-.wav

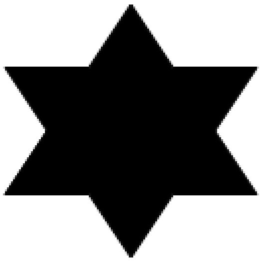


Spektrogram 21 - Kochova snežinka, 1. iteracija, simultano



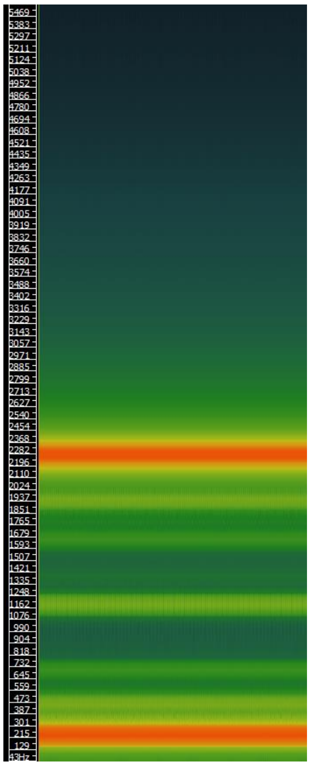
Sinusoida 15 - Kochova snežinka, 1. iteracija, simultano

b) 2. iteracija

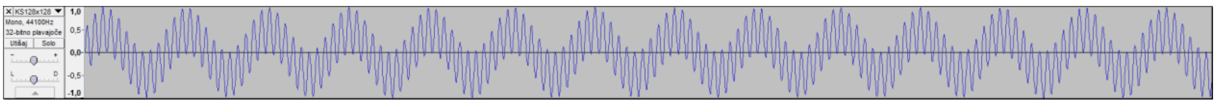


Slika 40 - Kochova snežinka, 2. iteracija

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%201%20-%20Simultano/Kochova%20sne%C5%BEinka/KS128x128-2-S-.wav

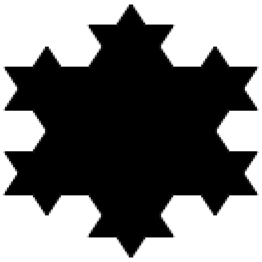


Spektrogram 22 - Kochova snežinka, 2. iteracija, simultano



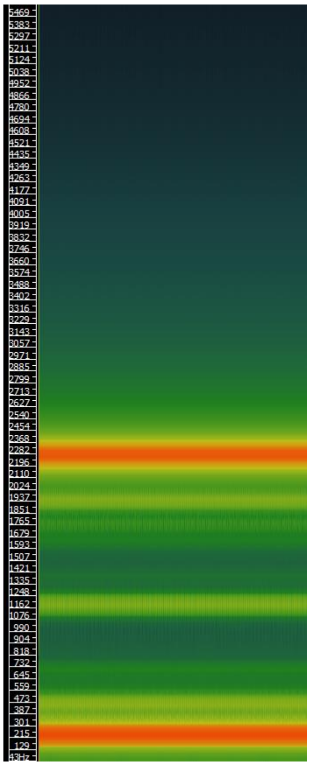
Sinusoida 16 - Kochova snežinka, 2. iteracija, simultano

c) 3. iteracija

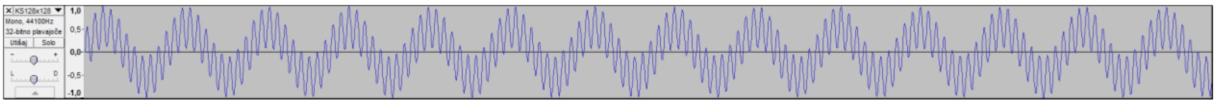


Slika 41 - Kochova snežinka, 3. iteracija

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%201%20-%20Simultano/Kochova%20sne%C5%BEinka/KS128x128-3-S-.wav

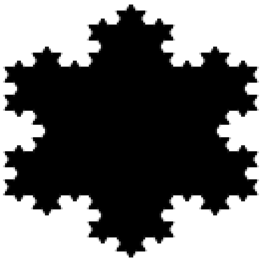


Spektrogram 23 - Kochova snežinka, 3. iteracija, simultano



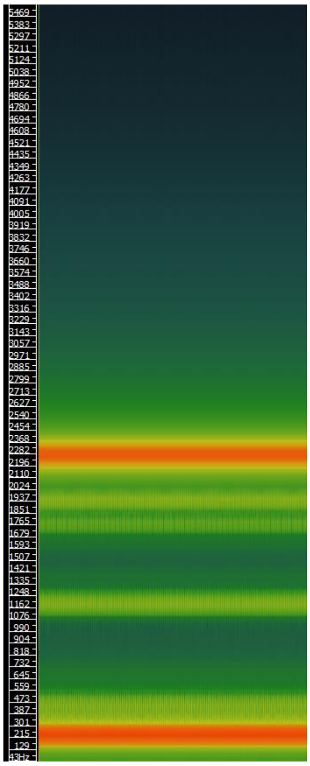
Sinusoida 17 - Kochova snežinka, 3. iteracija, simultano

d) 4. iteracija

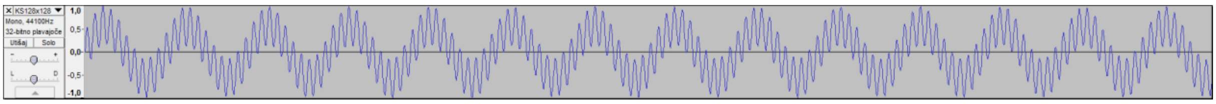


Slika 42 - Kochova snežinka, 4. iteracija

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%201%20-%20Simultano/Kochova%20sne%C5%BEinka/KS128x128-4-S-.wav

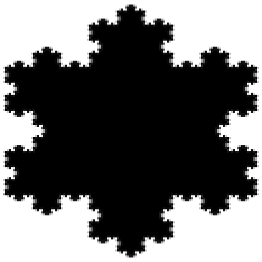


Spektrogram 24 - Kochova snežinka, 4. iteracija, simultano



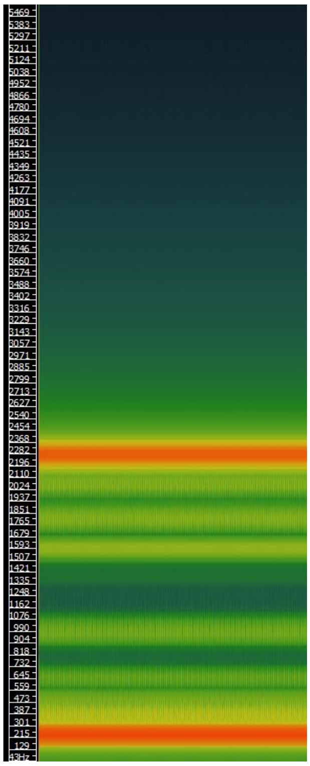
Sinusoida 18 - Kochova snežinka, 4. iteracija, simultano

e) 5. iteracija

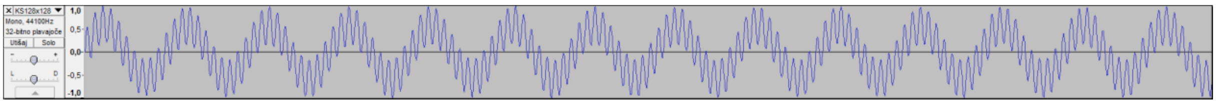


Slika 43 - Kochova snežinka, 5. iteracija

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%201%20-%20Simultano/Kochova%20sne%C5%BEinka/KS128x128-5-S-.wav



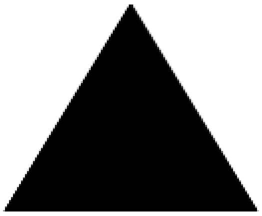
Spektrogram 25 - Kochova snežinka, 5. iteracija, simultano



Sinusoida 19 - Kochova snežinka, 5. iteracija, simultano

7.5.2. Konsekutivno zaporedje – drugi način predvajanja

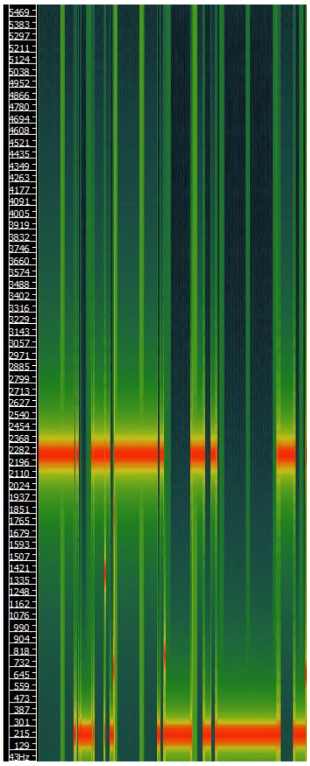
a) 1. iteracija



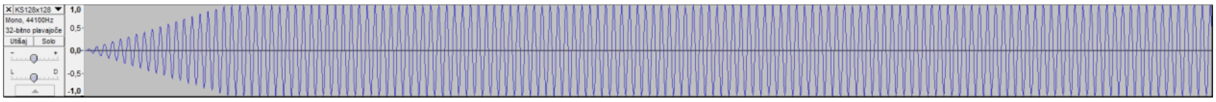
Slika 44 - Kochova snežinka, 1. iteracija

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%202%20-%20Konsekutivno/DolgeWavDatoteke/KS128x128-1-K-4096s.wav

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%202%20-%20Konsekutivno/Kochova%20sne%C5%BEinka/KS128x128-1-K-4096s-16s.wav

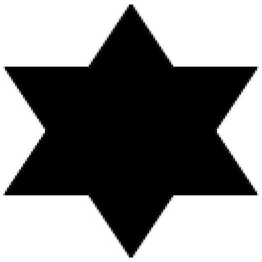


Spektrogram 26 - Kochova snežinka, 1. iteracija, konsekutivno



Sinusoida 20 - Kochova snežinka, 1. iteracija, konsekutivno

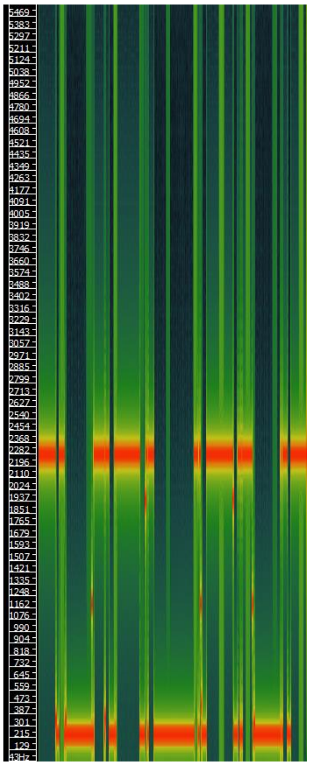
b) 2. iteracija



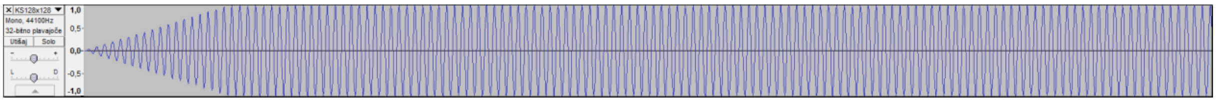
Slika 45 - Kochova snežinka, 2. iteracija

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%202%20-%20Konsekutivno/DolgeWavDatoteke/KS128x128-2-K-4096s.wav

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%202%20-%20Konsekutivno/Kochova%20sne%C5%BEinka/KS128x128-2-K-4096s-16s.wav

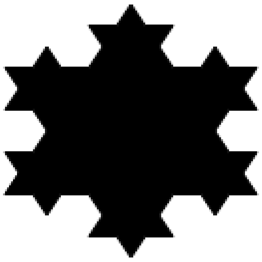


Spektrogram 27 - Kochova snežinka, 2. iteracija, konsekutivno



Sinusoida 21 - Kochova snežinka, 2. iteracija, konsekutivno

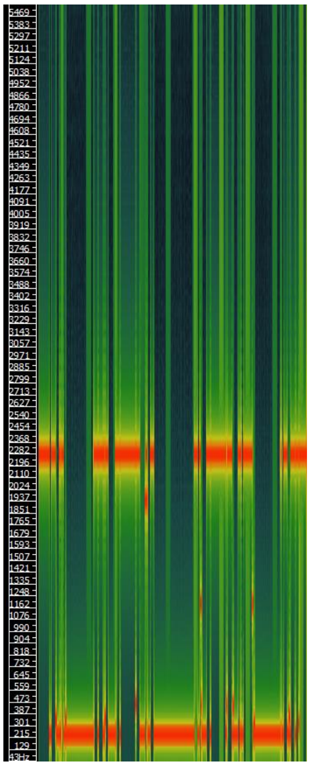
c) 3. iteracija



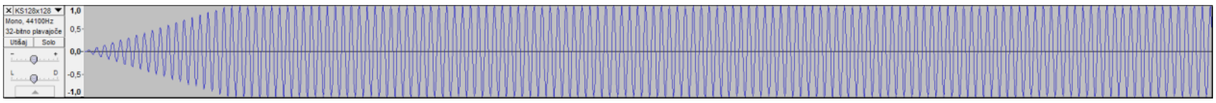
Slika 46 - Kochova snežinka, 3. iteracija

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%202%20-%20Konsekutivno/DolgeWavDatoteke/KS128x128-3-K-4096s.wav

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%202%20-%20Konsekutivno/Kochova%20sne%C5%BEinka/KS128x128-3-K-4096s-16s.wav

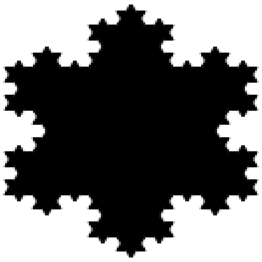


Spektrogram 28 - Kochova snežinka, 3. iteracija, konsekutivno



Sinusoida 22 - Kochova snežinka, 3. iteracija, konsekutivno

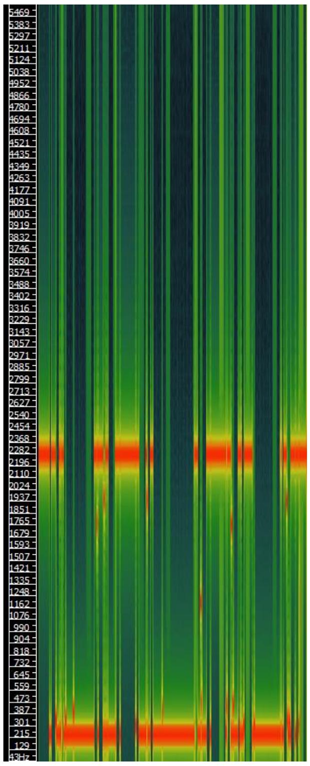
d) 4. iteracija



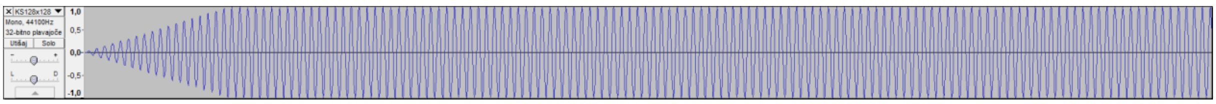
Slika 47 - Kochova snežinka, 4. iteracija

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%202%20-%20Konsekutivno/DolgeWavDatoteke/KS128x128-4-K-4096s.wav

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%202%20-%20Konsekutivno/Kochova%20sne%C5%BEinka/KS128x128-4-K-4096s-16s.wav

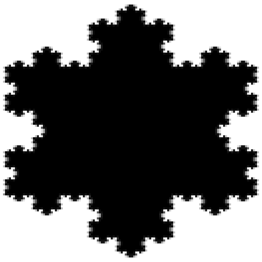


Spektrogram 29 - Kochova snežinka, 4. iteracija, konsekutivno



Sinusoida 23 - Kochova snežinka, 4. iteracija, konsekutivno

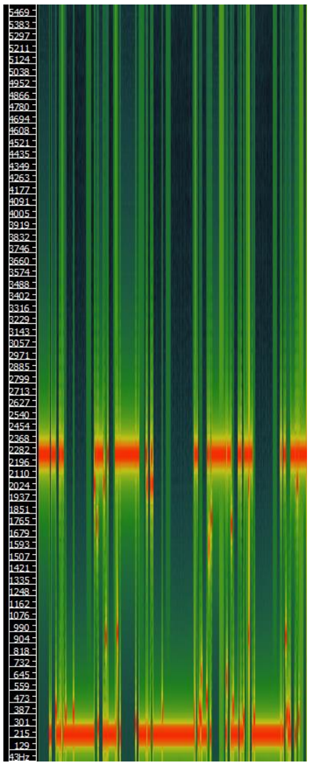
e) 5. iteracija



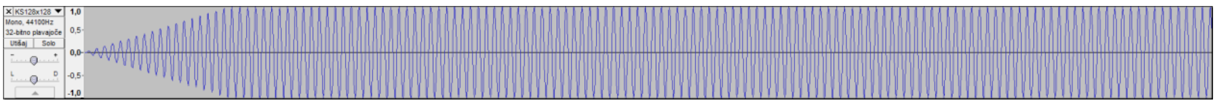
Slika 48 - Kochova snežinka, 5. iteracija

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%202%20-%20Konsekutivno/DolgeWavDatoteke/KS128x128-5-K-4096s.wav

https://github.com/raziskovalnagimnm/ZvokiFraktalov/blob/master/WAV%20Datoteke/Program%202%20-%20Konsekutivno/Kochova%20sne%C5%BEinka/KS128x128-5-K-4096s-16s.wav



Spektrogram 30 - Kochova snežinka, 5. iteracija, konsekutivno



Sinusoida 24 - Kochova snežinka, 5. iteracija, konsekutivno

Vse zvočne datoteke, slike fraktalov, spektrograme in sinusoide smo naložili na spletno mesto Github, kjer si jih je možno bolj podrobno ogledati. V raziskovalni nalogi smo zaradi preglednosti predstavili pomanjšane in skrajšane datoteke in grafe, ki pa v nekaterih primerih naših rezultatov ne pokažejo dovolj jasno. Zato si lahko celotne datoteke ogledate na povezavi:

https://github.com/raziskovalnagimnm/ZvokiFraktalov

Za prikazovanje rezultatov smo uporabili dva programa.

S programom Audacity smo preverili krivuljo določene WAV datoteke in jo primerjali s krivuljami drugih datotek enakih fraktalov različnih iteracij.

S programom Sonic Visualiser smo za vsako WAV datoteko izrisali spektrograme, vsakemu spektrogramu smo frekvenco in časovno dolžino omejili. Nato smo izrisane grafe med seboj primerjali.

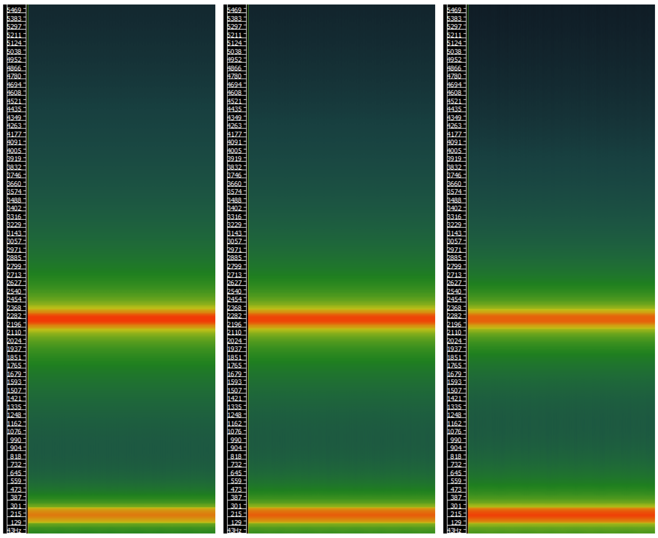
8. Razprava

8.1. Simultano zaporedje

8.1.1. Hilbertova krivulja

Iz spektrogramov lahko razberemo, da se z vsako višjo iteracijo delež bele barve na sliki manjša in se delež črne barve povečuje. Na spektrogramih opazimo tudi, da nastopata samo dve frekvenci, in sicer frekvenca 220 Hz (črne barva) ter frekvenca 2260 Hz (bela barva).

Pri sliki psevdo-Hilbertove krivulje ne pride do popačitve slike (7.2.3), ker je njena Hausdorffova dimenzija točno dve.



Spektrogram 31 - primerjava spektrogramov 4., 5. in 6. iteracije

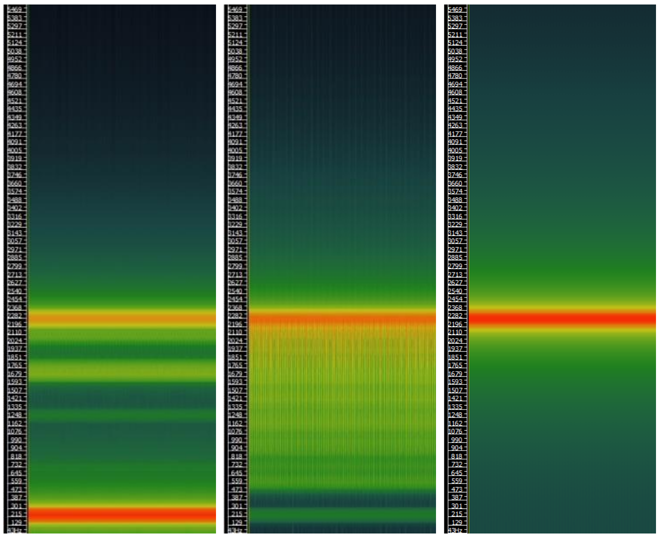
8.1.2. Preproga Sierpisnkega

Z vsako nadaljnjo iteracijo se delež črne barve fraktala zmanjšuje, povečuje pa se delež bele barve. Iz tega smo sklepali, da se bo z vsako nadaljnjo iteracijo frekvenca zvoka približevala frekvenci bele barve in naša teza je bila potrjena.

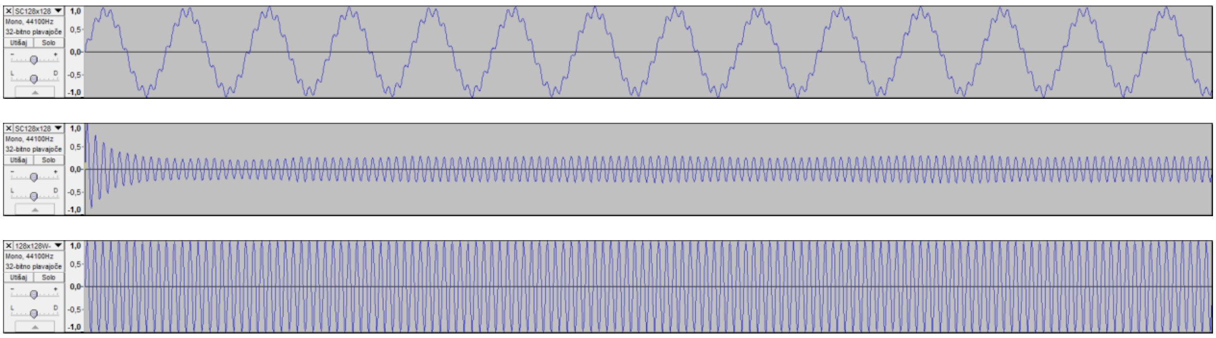
Če primerjamo spektrograme, lahko vidimo, da imamo pri prvi iteraciji v večji meri zastopano frekvenco 220 Hz (črne barve), z vsako naslednjo iteracijo pa je vse bolj zastopana frekvenca 2260 Hz (bele barve).

Tudi sinusoida z višjo stopnjo iteracije postaja vse bolj in bolj podobna sinusoidi, ki jo dobimo iz bele barve, na začetku pa je podobna sinusoidi črne barve. Z vsako iteracijo je sinusoida bolj gosta, torej ima zvok višjo frekvenco, kar se ujema z našimi pričakovanji.

Barve, ki so v ozadju (zelena, rumena, oranžna…), nastanejo zaradi popačitve slike. (Sivina)



Spektrogram 32 - primerjava spektrogramov 1. in 5. iteracije s frekvenco bele barve



Sinusoida 25 - primerjava sinusoid 1. in 5. iteracije s frekvenco bele barve

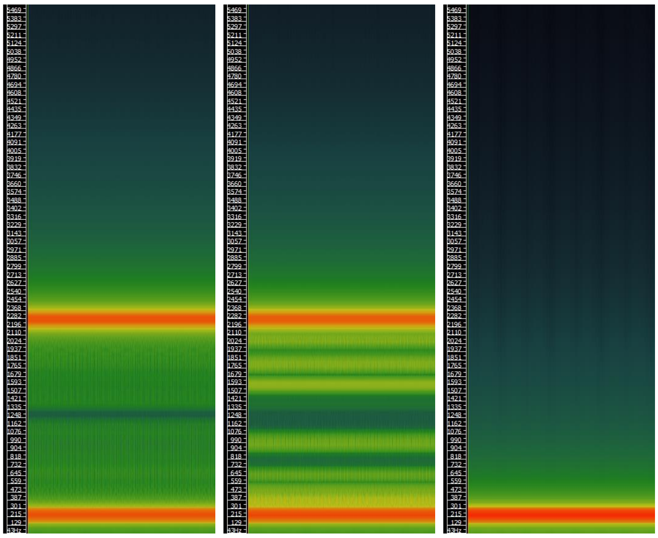
8.1.3. Kochova snežinka

Naša hipoteza je bila, da se zvok z vsako iteracijo Kochove snežinke zniža, saj je vedno več fraktala, ki je črne barve, spremembe pa so z vsako iteracijo manjše, saj je površina Kochove snežinke omejena.

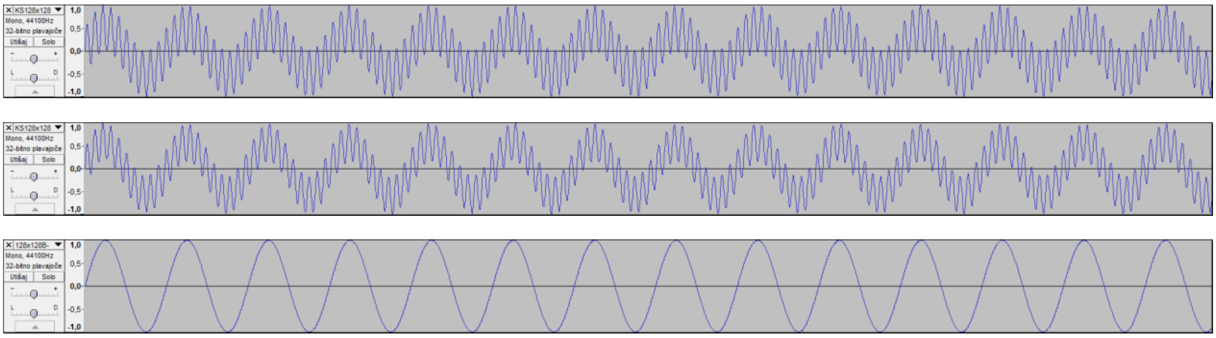
Spremembe so majhne, a jih vseeno lahko opazimo na spektrogramih. Nižja frekvenca (220 Hz – črna barva) postaja bolj zastopana, a z vsako iteracijo manj opazno.

Isto stvar lahko opazimo tudi na sinusoidi.

Barve, ki so v ozadju (zelena, rumena, oranžna…), nastanejo zaradi popačitve slike. (Sivina)



Spektrogram 33 - primerjava spektrogramov 1. in 5. iteracije s frekvenco črne barve

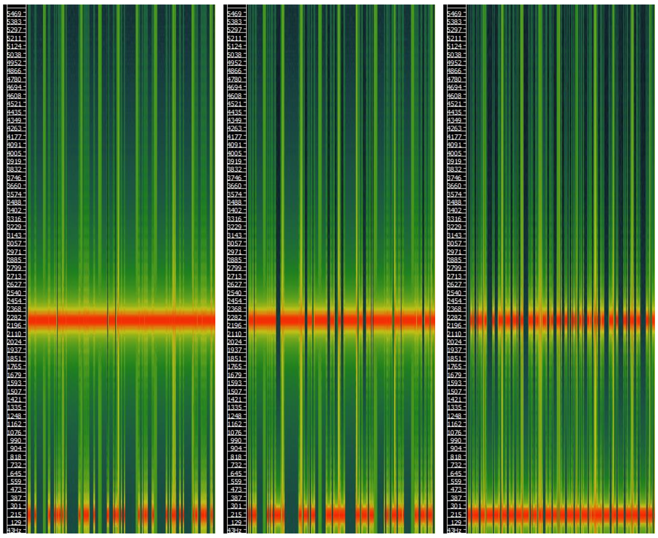


Sinusoida 26 - primerjava sinusoid 1. in 5. iteracije s frekvenco črne barve

8.2. Konsekutivno zaporedje

8.2.1. Hilbertova krivulja

Ker se slika z vsako višjo iteracijo bolj napolni s črnimi piksli, vidimo tudi, da je vedno več zastopanih frekvenc pikslov 220 Hz. Spet slišimo le dva tona, saj ne pride do nikakršne popačitve slike oz. sivine (7.2.3), ker ima psevdo-Hilbertova krivulja Hausdorffovo dimenzijo točno dve.



Spektrogram 34 - primerjava spektrogramov 4., 5. in 6. iteracije

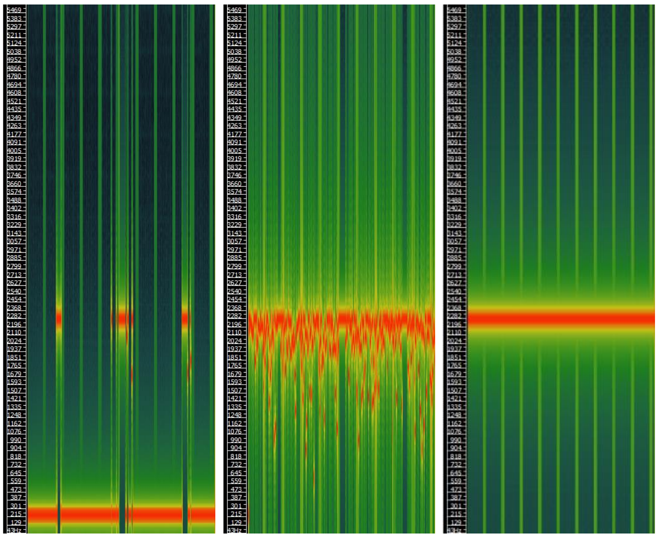
8.2.2. Preproga Sierpinskega

Ker se slika s preprogo Sierpinskega z vsako višjo iteracijo vedno bolj napolni z belimi piksli, je tudi zvok vsake slike višje iteracije preproge Sierpinskega bližji frekvenci 2260 Hz oziroma spektrogramu bele slike. Vidimo, da se rdeča črta, ki predstavlja najbolj zastopane frekvence, z vsako višjo iteracijo približuje polni neprekinjeni črti s frekvenco 2260 Hz. Nizke frekvence na dnu spektrograma pa predstavljajo črno barvo in vidimo lahko, da v 1. iteraciji prevladujejo, z vsako iteracijo pa je nizka frekvenca manj prisotna. Zato je na spektrogramu pete iteracije skoraj ne vidimo več.

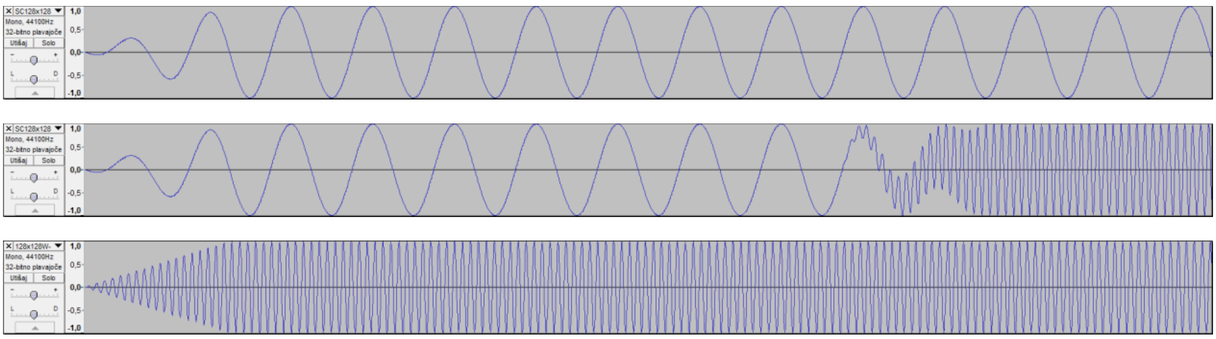
Na sinusoidi 1. iteracije lahko opazimo, da je frekvenca valov manjša in posledično je tudi zvok nižji (nižja frekvenca), saj je na fraktalu prve iteracije večina pikslov črnih.

Sinusoidni val 5. iteracije in sinusoida bele slike pa sta skoraj identična, saj je fraktal pete iteracije skoraj popolnoma bele barve.

Barve, ki so v ozadju (zelena, rumena, oranžna…), nastanejo zaradi popačitve slike. (Sivina)



Spektrogram 35 - primerjava spektrogramov 1. in 5. iteracije s frekvenco bele barve



Sinusoida 27 - primerjava sinusoid 1. in 5. iteracije s frekvenco bele barve

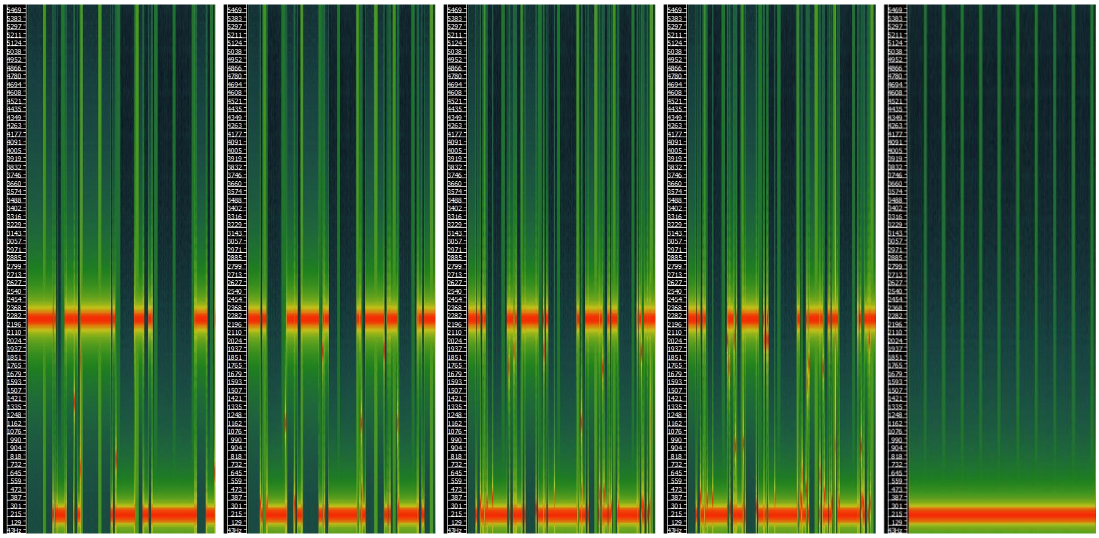
8.2.3. Kochova snežinka

Ker se slika Kochove snežinke z vsako višjo iteracijo vedno bolj napolni s črnimi piksli, je tudi zvok vsake slike višje iteracije Kochove snežinke vedno bližji frekvenci 2260 Hz oziroma spektrogramu črne slike. Ker pa ima Kochova snežinka omejeno površino, se na spektrogramu ohranita tako frekvenca bele barve kot frekvenca črne barve, spremembe pa so z vsako višjo iteracijo manjše, saj se površina fraktala z vsako višjo iteracijo manj poveča.

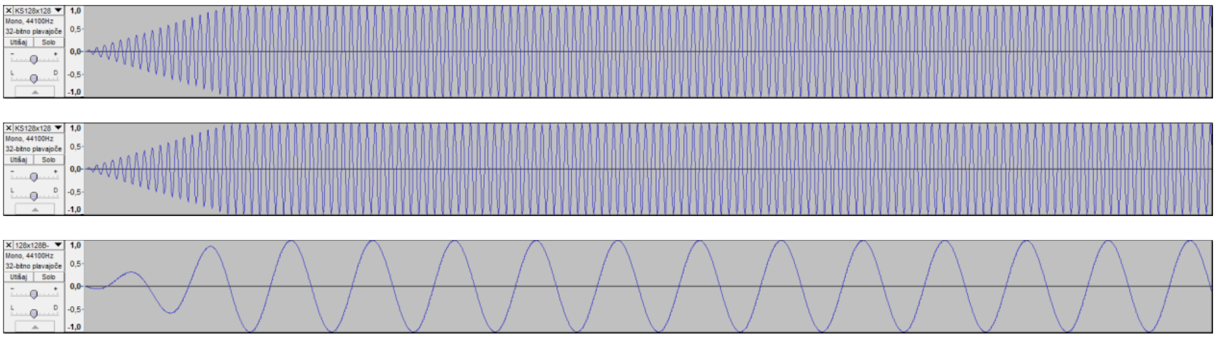
Na spektrogramih lahko vidimo, da je med 1. iteracijo in 2. iteracijo večja razlika kot med spektrogramoma 4. in 5. iteracije, frekvenca pa se do neke mere približuje 220 Hz (črna barva).

Na grafih sinusoid se sprememba frekvence ne vidi dobro, saj je prikazan le odsek grafa. Če bi pogledali celotno sinusoido, bi videli, da pride do spremembe frekvence valovanja, kar pomeni, da je v tistem časovnem odseku psevdo-Hilbertova krivulja prešla iz ene barve v druge. Odseki valovanja z visokimi frekvencami, ki pripadajo belim pikslom, se ujemajo z grafom sinusoide bele barve in odseki grafa z nižjimi frekvencami, ki pripadajo črnim pikslom, so identični grafu sinusoide črne barve.

Barve, ki so v ozadju (zelena, rumena, oranžna…), nastanejo zaradi popačitve slike. (Sivina)



Spektrogram 36 - primerjava spektrogramov 1., 2., 4. in 5. iteracije s frekvenco črne barve



Sinusoida 28 - primerjava sinusoid 1. in 5. iteracije s frekvenco črne barve

9. Zaključek

Namen naše raziskovalne naloge je bil najti način, kako bi prevajali fraktal oziroma sliko fraktala v frekvence, da bi iz njega dobili zvok. Za delo smo si izbrali preprogo Sierpinskega in Kochovo snežinko.

Med našim raziskovanjem smo se bili prisiljeni večkrat usmeriti v drugo smer, pojavila so se nova območja raziskovanja. Naša prvotna ideja za temo raziskovanja je bila iz glasbe, torej iz zvočnih posnetkov, ustvariti fraktal, a smo se potem raje odločili za ravno nasprotno in iskali način prevajanja slike fraktala v zvok, saj naša prvotna ideja ni bila izvedljiva.

Delo je večinoma potekalo gladko, nekaj težav nam je predstavljalo jasno prikazovanje rezultatov, predvsem zvoka in sinusoid.

Pri svojem raziskovanju smo se seznanili s fraktali, ki smo jih sicer poznali že prej, toda šele med našim raziskovanjem smo začeli zares odkrivali njihove čudovite oblike in jih začeli opažati v vsakdanjem življenju, povsod okoli nas. Naučili smo se razlikovati med nekaterimi miti o fraktalih in resničnimi dejstvi, ugotovili smo tudi njihovo uporabnost na mnogih področjih.

Večina našega dela je temeljila na uporabi programa Python, rezultate pa smo si ogledali v programih Audacity (sinusoide) in Sonic Visualiser (spektrogrami). Za bolj natančen ogled rezultatov smo vse dobljene rezultate naložili tudi na spletno mesto Github, saj smo v raziskovalni nalogi zaradi preglednosti predstavili pomanjšane in skrajšane datoteke.

Obravnavana tema je še vedno odprta za nadaljnje raziskovanje, saj še nismo raziskali, kako bi zveneli 3D fraktali, lahko bi se posvetili predvajanju različnih in bolj kompleksnih fraktalov ali pa bi prevajali fraktale v več različnih barvah. Ob raziskovanju so se nam porajala še mnoga zanimiva vprašanja, za katere upamo, da jih bomo lahko v našem nadaljnjem izobraževanju še bolj raziskali in našli odgovore nanje.

10. Viri in literatura

[1] S. Ornes, „Science and culture: Hunting fractals in the music of J. S. Bach," 22 julij 2014. [Elektronski]. [Poskus dostopa februar 2019].

[2] G. Pavlič, D. Kavka, M. Rugelj in J. Šparovec, Tempus novum, matematika za gimnazije, Ljubljana: Modrijan založba d.o.o., 2014, pp. 24-47.

[3] Nauk.si, „Neskončna geometrijska vrsta," [Elektronski]. Available: http://www.nauk.si/materials/200/out/#state=1. [Poskus dostopa februar 2019].

[4] B. Zlobec Jurčič in N. O. Gustavo Rubiano, Iteracije in fraktali, Ljubljana: Jožef Školč, 2016.

[5] B. Mandelbrot, The fractal geometry of nature, New York: W. H. Freeman, 1983.

[6] J. Gleick in S. Kuščer, Kaos: rojstvo nove znanosti, Ljubljana: DZS, 1991.

[7] B. Jurčič Zlobec, „L-sistemi," februar 2019. [Elektronski]. Available: http://matematika.fe.uni-lj.si/html/people/borut/scratch/Dokumenti/Lsistemi.pdf. [Poskus dostopa februar 2019].

[8] Wikipedia, „L-system," Wikipedia, 2019. [Elektronski]. Available: https://en.wikipedia.org/wiki/L-system. [Poskus dostopa februar 2019].

[9] 3Blue1Brown, „Hilbert's Curve: Is infinite math useful?," 21 julij 2017. [Elektronski]. Available: https://www.youtube.com/watch?v=3s7h2MHQtxc. [Poskus dostopa februar 2019].

[10] Wikipedia, „Hilbertova krivulja," [Elektronski]. Available: https://en.wikipedia.org/wiki/Hilbert_curve. [Poskus dostopa februar 2019].

[11] Mathigon, „Fractals," Mathigon, [Elektronski]. Available: https://mathigon.org/world/Fractals. [Poskus dostopa februar 2019].

[12] TED, „Benoit Mandelbrot: Fractals and the art of roughness," 6 julij 2010. [Elektronski]. Available: https://www.youtube.com/watch?v=ay8OMOsf6AQ. [Poskus dostopa februar 2019].

[13] M. Rojko, „Fraktali," 2015. [Elektronski]. Available: http://eprints.fri.uni-lj.si/3051/1/63110265-MOJCA_ROJKO-Fraktali.pdf. [Poskus dostopa februar 2019].

[14] onlinemathtools, „onlinemathtools," 2019. [Elektronski]. Available: https://onlinemathtools.com/. [Poskus dostopa februar 2019].

[15] Numberphile, „The Mandelbrot Set - Numberphile," 25 julij 2014. [Elektronski]. Available: https://www.youtube.com/watch?v=NGMRB4O922I. [Poskus dostopa februar 2019].

[16] MITK12Videos, „What Is A Fractal (and what are they good for)?," 11 junij 2015. [Elektronski]. Available: https://www.youtube.com/watch?v=WFtTdf3I6Ug. [Poskus dostopa februar 2019].

[17] FractalFoundation, „What are fractals?," [Elektronski]. Available: https://fractalfoundation.org/resources/what-are-fractals/. [Poskus dostopa februar 2019].

[18] BritannicaEditors, „Fractal," [Elektronski]. Available: https://www.britannica.com/science/fractal. [Poskus dostopa februar 2019].

[19] Electronic Musician, „Fractals and music," [Elektronski]. Available: https://www.emusician.com/gear/fractals-and-music. [Poskus dostopa februar 2019].