



StudiumPlus SE-ET Systemsimulation WS23/24 Marvin Müller (5273308)

Hausübung 1 - Koch'sche Schneeflocke

Aufgabe 1

Visualisierung erster Iterationsschritt der Idee aus file K_flake_fun:

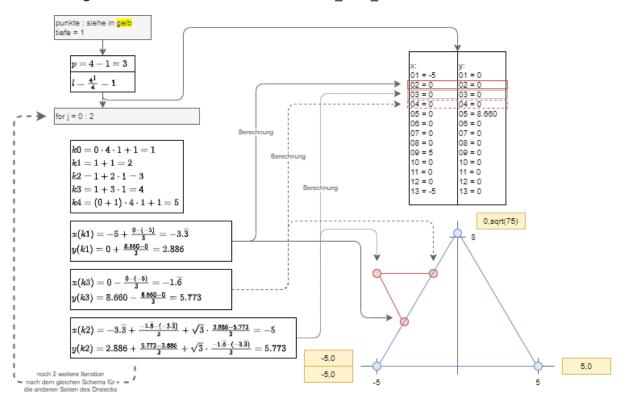


Abbildung 1: Visualisierung erster Iterationsschritt der Idee aus file K_flake_fun





Nassi-Shneiderman-Diagram von function Koch_Flocke_fun:

```
function [x,y]=Koch_Flocke_fun(punkte, tiefe)
[~, n2] = Size(punkte); % Bestimmung Anzahl Punkte durch Anzahl der Spalten
der Ursprungs-Matrix
p = n2-1:
                          % Bestimmung der Punkte (hier: 3)
I = 4^tiefe:
                         % Formel zur Bestimmung der Teilstrecken
n = p*I+1:
                         % Formel zur Bestimmung der Anzahl der Punkte
x = zeros(n,1);
                          % 0-Array fuer die x-Koordinaten initialisieren
y = zeros(n,1);
                          % 0-Array fuer die y-Koordinate initialisieren
% Befuellen der x- & y-Koordinate mit den festgelegten/uebergebenen Punkte des
Grunddreiecks
for i=0:p
      x(i*l+1) = punkte(1,i+1);
      y(i*l+1) = punkte(2,i+1);
% Berechnung der Iterationstiefe beginnend ab der 1. Iterationstiefe
for i=1:tiefe
      I=round(I/4); % Definition des jeweils einen Teilabschnittes
      % Berechnung der jeweiligen Konstruktionsschritte/Streckenzuege
      for j=0:p-1
            % Indizes
            % Konstruktionsschritte
            k0 = j*4*l+1;
            k1 = k0+1:
            k2 = k0 + 2*1:
            k3 = k0+3*1:
            k4 = (j+1)*4*l+1;
            % Position der Streckenzuege (inkl. Spitzen)
            x(k1)=x(k0)+(x(k4)-x(k0))/3;
            y(k1)=y(k0)+(y(k4)-y(k0))/3;
            x(k3)=x(k4)-(x(k4)-x(k0))/3;
            y(k3)=y(k4)-(y(k4)-y(k0))/3;
            x(k2)=x(k1)+(x(k3)-x(k1))/2+sqrt(3)*(y(k1)-y(k3))/2;
            y(k2)=y(k1)+(y(k3)-y(k1))/2+sqrt(3)*(x(k3)-x(k1))/2;
      p=p*4:
                      % Erweiterung der verwendeten Punkte pro Iterationstiefe
```

Abbildung 2: Nassi-Shneiderman-Diagram function Koch_Flocke_fun





Aufgabe 2

Nassi-Shneiderman-Diagram von function length_koch_fun:

```
function perimeter = length_koch_fun(x,y)

perimeter = 0 % Variable initialisieren

% Schleife für alle Punkte bis auf den Letzten
for i = 1:numel(x)-1

% Berechnung Abstand aktuellem und nächstem Punkt und Aufsummierung
perimeter = perimeter + sqrt( ( x(i)-x(i+1) )^2 + ( y(i)-y(i+1) )^2 );
```

Abbildung 3: Nassi-Shneiderman-Diagram function length_koch_fun





Aufgabe 3

Nassi-Shneiderman-Diagram von m-file test Koch Flocke:

m-File test_Koch_Flocke

```
close all:
                                 % Noch geoeffnete Plots schließen
clearvars;
                                 % Variablenspeicher leeren
punkte = [-5 0 5 -5; 0 sqrt(75) 0 0]; % Definition der Startpunkte(x,y)
                                          % Definition Iterationstiefe
% Funktion aufrufen --> Berchnung x und y
[x,y] = Koch_Flocke_fun(punkte, tiefe);
Ausgabe: Koch Flocke
figure:
                                       % Plot generieren
fill(x,y,'b');
                                       % Fuellen des 2-D Polygon in blau
title('Koch^{,}sche Schneeflocke'); % Titel
axis('equal', 'off')
                                       % Achsen-Konfiguration:
                       % -> equal = Seitenverhaeltnis von x und y gleich anpassen
                       % -> off = Achsenbeschriftung ausblenden
```

```
% Funktion aufrufen -> Berechnung des Umfangs
U = length_koch_fun(x,y);

Ausgabe: Umfang Koch Flocke
annotation('textbox', [0 0 1 0.1], 'String', ['Umfang: ' num2str(U)],
'EdgeColor', 'none') % Darstellung Umfang-Information auf Plot
```

```
tiefe max = 7:
                           % max. Iterationstiefe
u=zeros(tiefe_max+1,1); % Anlegen eines Arrays fuer die Umfaenge (je Iterationstiefe)
i=zeros(tiefe_max,1);
                           % Anlegen der Iterationstiefe, da mit 0 gestartet wird
% Berechnung Umfang fuer jede Iterationstiefe
for j = 0:tiefe_max
      % Funktion aufrufen -> Berechnung x und y
      [x,y] = Koch_Flocke_fun(punkte,j);
      % Funktion aufrufen -> Berechnung des Umfangs
      U = length_koch_fun(x,y);
      u(j+1) = U;
                       % Array mit den errechneten Umfangwerten fuellen
      i(j+1) = j
                       % Array fuer die Iterationstiefe anlegen (beginnend mit 0)
Ausgabe: Umfang Koch Flocke / Iterationstiefe
figure:
                                       % Plot generieren
stem(i,u);
               % Darstellung von u(i) -> Umfang in Abhaengigkeit der Iterationstiefe
title('Koch^{,}sche Schneeflocke'); % Titel
xlabel('Iterationstiefe');
                                      % Beschriftung x-Achse
```

Abbildung 4: Nassi-Shneiderman-Diagram m-file test_Koch_Flocke

ylabel('Umfang');

% Beschriftung y-Achse





Aufgabe 4

Beispiel selbstähnliches Objekt aus dem Bereich der Technik:

- Fractal antenna → en.wikipedia
- Fractal landscape → en.wikipedia
- Fractal analysis → en.wikipedia
- Fractal dimension on networks → en.wikipedia
- Bailey Winstanley, Alessandro Principi: Fractal field-effect transistors: Enhanced photodetection and fractal dependent resonances → <u>arxiv.org</u>
- IEEE: Fractal structures for low-resistance large area AlGaN/GaN power transistors \rightarrow ieee.org

GitHub Repository: https://github.com/Osingar/se-et-systemsimulation/tree/main/5273308/H1