Aufgabe 5 Teil 2 Rekursive Grafiken

Lindenmayer Systeme und rekursive Grafiken / Mehrfachrekursion

Pflanzen aber auch andere Naturphänomene lassen sich als Fraktale begreifen. Der Begriff Fraktal geht zurück auf den Mathematikern Benoît Mandelbrot und beschreibt natürliche oder künstliche Muster mit einem hohen Grad an Selbstähnlichkeit. Fraktale entstehen z.B. wenn sich Objekte aus mehreren verkleinerten Kopien zusammensetzen lassen. Fraktale sind daher ein weiterer Vertreter für rekursive Strukturen.

Der Biologe Aristid Lindenmayer entwickelte unter Verwendung von Chomsky Grammatiken einen Formalismus, mit dem sich natürliche Muster mit Hilfe von Grammatiken beschreiben lassen. Dieser Formalismus, genannt Lindenmayer Systeme (kurz L-Systeme) wurde zuerst für die Berechnung primitiver multizellulärer Organismen verwendet. In Zusammenarbeit mit dem Mathematiker Przemyslaw Prusinkiewicz gelang es die mathematischen Grammatiken mit Hilfe von Turtle-Grafiken zu visualisieren. Heutzutage sind L-Systeme soweit, dass sich sehr realistische Darstellungen natürlicher Formen, Pflanzen, Gebirge etc. im Computer erzeugen lassen (vgl. Abbildung 1) [1].

Abbildung 1: Computergenerierter Baum (http://michael.szell.net/fba/kapitel2.html#kapitel21) / Sonnenblume (http://michael.szell.net/fba/kapitel4.html) / Sträucher („Fractal weeds". Lizenziert unter Gemeinfrei über Wikimedia Commons - http://commons.wikimedia.org/wiki/File:Fractal_weeds.jpg#/media/File:Fractal_weeds.jpg)

Das Grundprinzip der L-Systeme

L-Systeme basieren auf der Ersetzung von Teilen eines Objektes mit Hilfe von Regeln. Objekte werden abstrakt als Zeichenketten beschrieben, in denen Zeichen eine bestimmte Bedeutung zugeordnet werden. Formal lässt sich ein L-System als Tupel L=(V,w,P) beschreiben, wobei V die gültigen Zeichen (das Alphabet), w die Startsymbole und P die Produktionsregeln beschreibt.

Tabelle 1: Symbole und deren Bedeutung in L-Systemen

Symbol	Bedeutung
F	Vorwärtsbewegung und Zeichnen
f	Vorwärtsbewegung ohne Zeichnen
+	Linksdrehung
-	Rechtsdrehung

Durch eine geeignete grafische Interpretation der Symbole, können L-Systeme mit Hilfe der Turtle-Grafik visualisiert werden.

Erstes kleines Beispiel

L-System für die Kochkurve

G = (V,w,P)

V= {F,+,-} # die Symbole

w= {F } # das Startsymbol

P= { F-> F+F--F+F }, # Regel für die Kochkurve

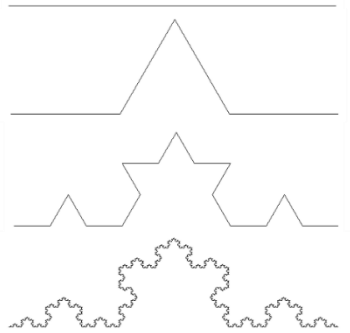

Abbildung 2: Kochkurven der Ordnung 0,1,2 und 5. Erzeugt mit der Turtlegrafik der Toolbox. Vorlage siehe [1].

Bei der Kochkurve wird eine Strecke jeweils durch 4 verkürzte Strecken ersetzt. Wenn der Drehwinkel 60° ist, dann ergeben sich geometrisch die Längen der Teilstrecken als Drittel der ursprünglichen Strecke.

Eine Kochkurve 0'ter Ordnung ergibt sich, wenn keine Ersetzung von F erfolgt.

F

Eine Kochkurve 1'ter Ordnung ergibt sich, wenn 1 Ersetzung von F über Regeln erfolgt.

F+F--F+F

Eine Kochkurve 2'ter Ordnung ergibt sich, wenn 2 Ersetzungen von F über Regeln erfolgen, u.s.w.

F+F--F+F+F+F--F+F--F+F--F+F+F+F--F+F

Turtle Grafik

Die Turtle Grafik erfunden von Seymour Papert und implementiert in der Sprache LOGO, basiert auf der Idee, dass sich beliebige geometrische Formen im 2D Raum durch Aneinandersetzen von Strecken erzeugen lassen. Eine Turtle (Schildkröte) bewegt sich über das Zeichenblatt und hinterlässt eine Spur, wenn der „Schwanz" unten ist (Symbol **F** in Tabelle 2, Symbol **f** steht für Bewegungen ohne zu zeichnen („Schwanz ist oben")). Die Turtle kann sich in beliebige Richtungen drehen. Dabei werden Drehungen gegen den Uhrzeigersinn auf Linksdrehungen (Symbol **+** in Tabelle 2), Drehungen im Uhrzeigersinn als Rechtsdrehungen (Symbol **–** in Tabelle 2) dargestellt.

Dies ist die einfachste Variante der Turtle-Grafik mit der sich bereits bestimmte Kategorien von L-Systemen darstellen lassen.

Tabelle 2: **Symbole und deren Interpretation in der Turtle-Grafik**:

Symbol	**Bedeutung**	**Methoden der Klasse *Turtle***
F(steps)	gehe steps Schritte voraus und zeichne eine Strich	***go_ahead***
f(steps)	gehe steps Schritte voraus ohne zu zeichnen	***go_head***
+(alpha)	drehe dich um alpha nach links	***turn_left***
-(alpha)	drehe dich um alpha nach rechts	***turn_right***

Programmierschnittstelle der Turtle-Grafik

Die Implementierung der Turtle-Grafik, die Klasse ***Turtle***, setzt die Interpretation aus Tabelle 2 in animierte Grafikbefehle um.

Die Schnittstelle der ***Turtle*** ist wie folgt definiert

initialize(x,y,angle)	Setzt ein ***Turtle***-Objekt auf Position (***x,y***) und richtet sie nach Winkel ***angle*** im Uhrzeigersinn aus. Wenn kein Wert für den Winkel übergeben wird, dann schaut die Turtle nach rechts (Osten). Der „Schwanz“ der Turtle ist zu Beginn unten.
turn_left(angle)	Bewege die ***Turtle*** um den Winkel ***angle*** **gegen** den Uhrzeigersinn. Entspricht dem "**+**" aus Tabelle 2
turn_right(angle)	Bewege die ***Turtle*** um den Winkel ***angle*** **im** den Uhrzeigersinn. Entspricht dem "**-**" aus Tabelle 2.
go_ahead(steps,width)	Gehe Anzahl Schritte ***steps*** in Richtung Blickrichtung. ***Width*** gibt die ***Breite*** der Spur vor, also die Breite des Zeichenstifts. Wird ***width*** nicht angegeben, dann ist die Stärke 1. Ob gezeichnet wird, hängt davon ab, ob der „Schwanz“ der Turtle oben oder unten ist. Dies wird über die Methode ***down*** und ***up*** gesteuert. Entspricht dem **F/f** in Tabelle 2.
up()	Bewegt den „Schwanz“ der Turtle nach oben.
down()	Bewegt den „Schwanz“ der Turtle nach unten.

Kochkurve mit TurtleGrafik

Für das L-System der Kochkurve müssen wir noch die *Startlänge*, den *Verkürzungsfaktor* und den *Drehwinkel* festlegen:

Startlänge l: möglichst nicht zu kurz, da in jedem Ersetzungsschritt verkürzt wird (z.B. 600)

Verkürzungsfaktor: 1/3

Drehwinkel: 60°

Dann können wir aus der rekursiven Definition des L-Systems und den Zusatzinformationen Startlänge, Verkürzungsfaktor und Drehwinkel eine rekursive Implementierung für die Kochkurve angeben.

```
class KochKurve

  # wdhl gibt die Anzahl der rekursiven Verzweigungen und damit die Ordnung der
  # Kochkurve an
  # x,y ist die Anfangsposition der Turtle und
  # kl die Kantenlänge zu Beginn
  #
  # Der Drehwinkel ist eine für die Kochkurve feste Größe von 60 im Gradmass
  # @factor ist die Verkürzung in jedem Rekursionsschritt

  def zeichnen(wdhl,x,y,kl)
    @turtle = Turtle.new(x,y,0)
    @angle = 60
    @factor = 3.0
    koch(wdhl,kl)
  end

  # Löscht alle Zeichnungen der Turtle
  def loeschen()
    @turtle.loeschen()
  end

  # rekursive Funktion zum Zeichnen der Kochkurve
  def koch(n,kl)
    # Abbruchbedingung: Alle Ersetzungen wurden abgeschlossen
    # jetzt können die verkürzten Strecken gezeichnet werden
    if n == 0
      @turtle.go_ahead(kl)
      return
    end
    # Rekursiver Aufruf - Anwendung der Ersetzungsregel: F -> F+F--F+F
    koch(n-1,(kl/@factor).round)          # F
    @turtle.turn_left(@angle)             # +
    koch(n-1,(kl/@factor).round)          # F
    @turtle.turn_right(@angle)            # -
    @turtle.turn_right(@angle)            # -
    koch(n-1,(kl/@factor).round)          # F
    @turtle.turn_left(@angle)             # +
    koch(n-1,(kl/@factor).round)          # F
  end

end
```

Aufgabe 5.4.1

Lèvy Fraktal:

Die geometrische Konstruktionsidee für das Lèvy Fraktal lautet: Zeichne eine Strecke der Länge kl. Errichte über dieser Strecke ein rechtwinkliges Dreieck mit Katheten gleicher Länge. Fahre mit der Konstruktionsvorschrift für die beiden Katheten fort. Zeichne nur die Katheten, nicht die Hypotenuse.

Ein Lévy-Fraktal der Ordnung 0 ist eine Strecke. Ein Lèvy-Fraktal der Ordnung 1 ist ein einfacher „Hut“. Damit ein Hut der Ordnung 1 entsteht, muss eine 45° Drehung nach links ausgeführt werden, dann ein Lévy-Fraktal der Ordnung 0 mit Streckenlänge $\frac{1}{\sqrt{2}}$ *kl gezeichnet werden, dann eine 90° Drehung nach rechts ausgeführt werden und schließlich wieder ein Lévy-Fraktal der Ordnung 0 mit Streckenlänge $\frac{1}{\sqrt{2}}$ *kl gezeichnet werden ($\frac{1}{\sqrt{2}}$ *kl ergibt aus dem Satz des Pythagoras).

Diese Vorschrift wird in der Produktionsregel des L-Systems ausgedrückt und lässt sich zur rekursiven Vorschrift für die Konstruktion von Levy-Fraktalen höherer Ordnung verstehen. Da in der Produktionsregel ein **F** durch **2 F** ersetzt wird und nur für Lèvy-Fraktale der Ordnung 0 eine Strecke gezeichnet wird, ist sichergestellt, dass nur die Katheten der kleinsten Dreiecke der Ordnung n gezeichnet werden. Der Verkürzungsfaktor $\frac{1}{\sqrt{2}}$ gibt das Verhältnis zwischen altem **F** und den **2 neuen F** wieder.

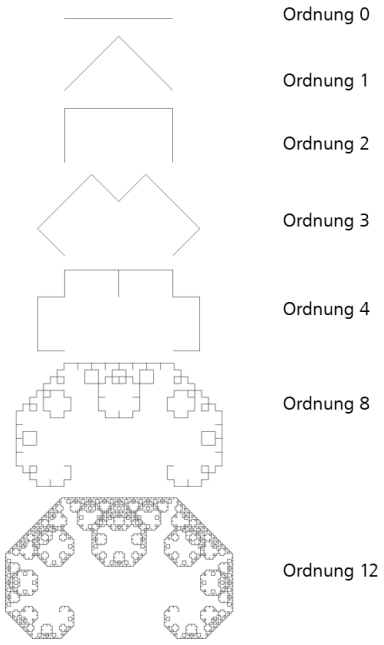

Abbildung 3: Lèvy-Fraktale verschiedener Ordnungen für Kantenlänge 300. Generiert mit der Turtle-Grafik der Toolbox.

L-System für das Lèvy-Fraktal

Definition nach [2]:

G = (V,w,P)
V = {F,+,-}
w = {F}
P = {F → +F--F+}

Drehwinkel: 45°
Verkürzungsfaktor: $\frac{1}{\sqrt{2}}$

Implementieren Sie im Projekt *A5 SoSe2019 RekursiveGrafik* in der Klasse ***LevyFraktal*** die Methode ***levy(n,kl)***, die die rekursive Produktionsregel mit Hilfe der Turtle grafisch umsetzen soll. Winkel und der Nenner des Verkürzungsfaktors werden bereits bei der Initialisierung des Fraktals definiert.

Hinweis:

- Im Projekt *A5 SoSe2019 RekursiveGrafik* finden Sie für 4 weitere Fraktale die vorbereiteten Klassen. Wenn Sie Spaß an solchen Aufgaben haben, versuchen Sie sich an einer Lösung ☺.

Lindenmayer Systeme und Vegetation

Auch das Wachstum/Entstehen „natürlicher" Vegetation lässt sich rekursiv auf Basis von Lindenmayer Systemen beschreiben. Eine Pflanze / ein Baum entsteht, indem an gewissen Knotenpunkten Verzweigungen angesetzt werden, die selbst wieder Pflanzen/Bäume sind.

Da die Verzweigungen am gleichen Knotenpunkt ansetzen, muss es möglich sein, nach Fertigstellung eines Zweiges auf die Startposition dieses Zweiges zurückzusetzen. Dazu müssen wir die Symbole des L-Systems um zwei weitere ergänzen: eines, um uns den Zustand vor der Verzweigung zu merken (das Symbol **[**) und eines um den alten Zustand wiederherzustellen (das Symbol **]**). Die vollständige Liste der Symbole für diese Varianten eines L-Systems sowie die zugehörigen Methoden der Turtle-Grafik zeigt Tabelle 3.

Tabelle 3: **Symbole und deren Interpretation in der Turtle-Grafik**:

Symbol	**Bedeutung**	**Methode der Klasse *Turtle***
F(steps)	gehe steps Schritte voraus und zeichne eine Strich	***go_ahead***
f(steps)	gehe steps Schritte voraus ohne zu zeichnen	***go_head***
+(alpha)	drehe dich um alpha nach links	***turn_left***
-(alpha)	drehe dich um alpha nach rechts	***turn_right***
[	speichere den aktuellen Zustand	***remember***
]	Stelle den letzten Zustand wieder her	***restore***
C0 …CN	Ändere die Farbe des Zeichenstifts	***change_color***

Ein ***Turtle***-Objekt speichert beim Aufruf von ***remember*** als aktuellen Zustand die Position, Farbe und die Blickrichtung auf einem Stack. Mit ***restore*** wird das oberste Element – der letzte gespeicherte Zustand – vom Stack gelesen und Position und Winkel der Turtle auf den letzten gespeicherten Zustand gesetzt. Durch die Verwendung eines Stacks ist es möglich auch geschachtelte Zustände (geschachtelte Klammerung) in den Regeln der L-Systeme zu verwenden.

Aufgabe 5.4.2:

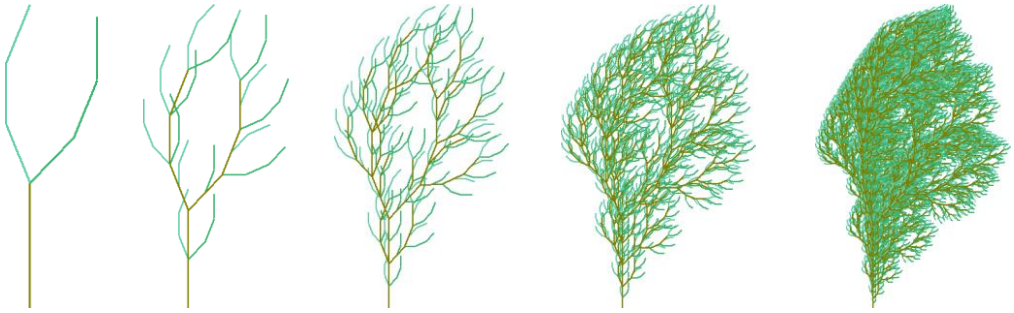

*Abb. 4 Ein mittels eines L-Systems generierter Baum der Ordnung 1-5. Länge berechnet aus 64/2**(n-1), Breite konstant 2, Winkel konstant 22°. Zeichnung verwendet 3 Farben.* ***Quelle****: Eigene Arbeit.*

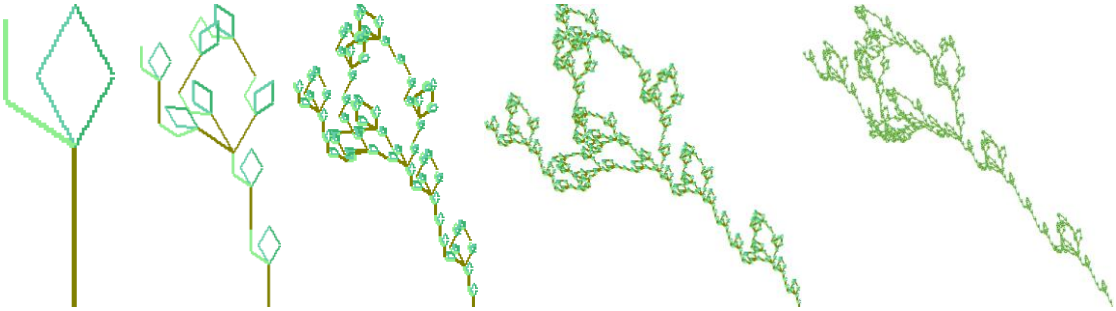

*Abb. 5 Eine mittels eines L-Systems generierte Ranke der Ordnung 1-5. Länge berechnet aus 64/2**n, Breite konstant 2, Winkel konstant 28°. Zeichnung verwendet 4 Farben.* ***Quelle****: Eigene Arbeit.*

Abb. 4 und Abb. 5 zeigen Grafiken für 2 L-Systeme mit Zustandsspeicher an den Verzweigungspunkten mit unterschiedlichen Drehwinkeln und Produktionsregeln.

Übersetzen Sie die nachfolgenden L-Systeme in rekursive Methoden ***baum***, ***ranke*** der Klasse ***Vegetation*** unter Verwendung der Turtle-Grafik.

L-System für Baum (Typ 1) (Abb.)

w = F
P = { F → C0FF-[C1-F+F+F]+[C2+F-F-F] }
Drehwinkel: 22

L-System für Pflanze (Typ 2)

w = F
P = { F → C0FF[C1-F++F][C2+F--F]C3++F--F }
Drehwinkel: 27

Hinweis: Die Symbole C0, C1, C2, C3 stehen für Farben. In dem Ruby-Projekt sind diese bereits definiert.

Quellen:

[1] L-Systeme: http://michael.szell.net/fba/kapitel2.html#kapitel21

[2] Beispiele für die L-Systeme: http://www.kevs3d.co.uk/dev/lsystems/ zuletzt abgerufen am 5.5.2019

Optionaler Aufgabenteil

Anstelle als grafische Anweisungen lassen sich die L-Systeme als Substitutions-Systeme verstehen. In einer Zeichenkette werden dabei alle Zeichen, zu denen einen Produktions-Regel in der Menge P des L-Systems existiert, durch die rechte Seite der Produktionsregel (rekursiv) ersetzt.

Schreiben Sie die Methode `expand(rules,l_word,n,sp=[])` der Klasse `RekursiveSubstitution`, die diese Ersetzungen vornimmt und wie folgt arbeitet (skizziert):

1. Wenn das erste Zeichen von `l_word` als Schlüssel in `rules` enthalten ist, dann ersetzte das Zeichen durch die expandierte rechte Seite der Regel und konkateniere das Ergebnis mit dem Speicher `sp`. Fahre mit den verbleibenden Zeichen von `l_word` fort.
2. Sonst lese das Zeichen, übertrage es in den Speicher `sp` und fahre mit den verbleibenden Zeichen in `l_word` fort. **Achtung**: die Farbkodierungen benötigen 2-Zeichen.
3. Abbruchbedingungen:
    a. `n==0`: dann kann nicht weiter reduziert werden und das Ergebnis steht in `l_word`.
    b. `l_word` ist abgearbeitet: dann steht das Ergebnis im Speicher `sp`.

ERLÄUTERUNG zu den Parametern:

`rules`: ein Hash für die Substitutionsregeln
`l_word`: ein Wort des L-Systems, das abgearbeitet werden muss. Vereinfachung es gibt nur ein Startwort.
`n`: Rekursionstiefe für die Substitution
`sp`: Speicher für das Einsammeln der Ergebnisse

Wenn die Methode wie folgt aufgerufen wird, dann sind die Ergebnisse nach einem `join` auf dem Ergebnis der Methode:

```
RekursiveSubstitution.new().expand({"F"=>"C0FF[C1-F++F][C2+F--F]C3++F--F"},
"F",  1, []).join()
```

Ergebnis: "C0FF[C1-F++F][C2+F--F]C3++F--F"

```
RekursiveSubstitution.new().expand({"F"=>"C0FF[C1-F++F][C2+F--F]C3++F--F"},
"F",  2, []).join()
```

Ergebnis: "C0C0FF[C1-F++F][C2+F--F]C3++F--FC0FF[C1-F++F][C2+F--F]C3++F--F[C1-C0FF[C1-F++F][C2+F--F]C3++F--F++C0FF[C1-F++F][C2+F--F]C3++F--F][C2+C0FF[C1-F++F][C2+F--F]C3++F--F--C0FF[C1-F++F][C2+F--F]C3++F--F]C3++C0FF[C1-F++F][C2+F--F]C3++F--F--C0FF[C1-F++F][C2+F--F]C3++F--F"