Christian Maurer

Lindenmayer Systeme

v. 17. September 2011

Dr. Christian Maurer
Keithstr. 16
10787 Berlin

http://murus.org/

Satz: Autor mit $\mathcal{A}\mathcal{M}\mathcal{S}$-TeX

Eine Rose ist eine Rose ist eine Rose . . .

Gertrude Stein (1874–1946)

Aus ihrem Gedicht „Sacred Emily“, 1913

Vorwort

Die Beschäftigung mit Lindenmayer-Systemen erscheint mir aus mehreren Gründen attraktiv für den Informatikunterricht. Das hatte mich vor vielen Jahren zur Entwicklung eines Simulationsprogramms veranlaßt, um das Thema für die Schule zugänglich zu machen.

Dieses Programm ist vor einigen Jahren im Softwarepraktikum eines Kurses der Lehrerweiterbildung Informatik an der Freien Universität Berlin auf dreidimensionale Systeme erweitert worden. Den damaligen Kursteilnehmerinnen und -teilnehmern gebührt Dank für ihre vielfältigen Anregungen und ihre solide Arbeit.

Christian Maurer

Inhaltsverzeichnis

Allgemeines

Arbeitsverzeichnis 1

Einführung

Zur Bedeutung des Konzepts 2
Sprachen und L-Grammatiken 3

Lindenmayer-Systeme

Deterministische kontextfreie L-Systeme 5
Visualisierung 6
Einfache Beispiele 7
Raumfüllende Kurven 9
Verzweigte L-Systeme 11
Dreidimensionale L-Systeme 13

Anhang

Das Programm `LS` 14
Literatur 16
Das Programm `zeig` 17
Zentralprojektion 18

ALLGEMEINES

LSysteme ist als Bestandteil des Modularen Universums nach dessen Installation oder Aktualisierung ebenfalls installiert.

Voraussetzung für die Installation des Modularen Universums ist die Installation von **Mocka**, dem Modula-2-Compiler der GMD (Hinweise dazu finden sich unter `lwb.mi.fu-berlin.de/inf/mocka`).

Arbeitsverzeichnis

Das Arbeitsverzeichnis, aus dem heraus `LS` aufgerufen wird und in dem ggf. Daten abgelegt werden, ist `$HOME/.LS` (es sei denn, daß durch die Umgebungsvariable `LS` ein anderes Arbeitsverzeichnis definiert ist). Damit ist sichergestellt, daß Benutzer/innen mit dem Aufruf von `LS` ihre eigenen Daten verwalten.

Wenn dieses Verzeichnis noch nicht existiert, wird es beim ersten Aufruf von `LS` selbsttätig angelegt und es werden Beispieldaten aus dem Modularen Universum dorthinein kopiert, falls es welche gibt.

EINFÜHRUNG

Zur Bedeutung des Konzepts

Der Biologe ARISTID LINDENMAYER hat *Grammatiken einer ganz bestimmten Form* zur formalen Beschreibung des *Wachstums von Pflanzen* eingesetzt. Die *Visualisierung von Wörtern* der von diesen Grammatiken erzeugten Sprachen liefert verblüffende *geometrische Gebilde von eigenartiger Schönheit*:

Neben fraktalen Kurven insbesondere – wie von LINDENMAYER beabsichtigt – ansprechende Modelle von Pflanzen.

Reizvolle Aufgaben des Themas bestehen u. a. darin, systematisch Grammatiken zur Erzeugung hübscher Graphiken zu „entdecken"; eine mögliche Variante ist das Herausfinden von Konstruktionsregeln zu vorgegebenen Graphiken oder für einfache Modelle realer Pflanzen.

Die Fragestellung ist hochmotivierend zur Einführung in die Beschäftigung mit grundlegenden informatischen Themen; mit Blick auf die attraktiven Ergebnisse ist es ausgesprochen kreativitätsfördernd. Das Thema ist daher nicht nur an geeigneter Stelle zur Einführung in Konzepte oder im Kontext der Theoretischen Informatik, sondern durchaus auch für eine Unterrichtsreihe im ersten Informatik-Jahr der Oberstufe, gerade auch für den Anfangsunterricht, insbesondere für Profilkurse, sehr geeignet, da es *keine Vorkenntnisse* voraussetzt und mit minimalem syntaktischen Aufwand zu bewältigen ist.

- *Formale Sprachen*:
  Einführung in Grundbegriffe der Theoretischen Informatik (*Sprache*, *Grammatik*),
- *Rekursion*:
  Erfassung dieses grundlegenden Sprachmittels zur Beschreibung von Wiederholungen,
- *kein Vorgriff auf Programmiersprachen*:
  minimaler Programmieraufwand durch „Spielerei" mit sehr einfachen Grammatiken,
- *weites Aufgabenfeld*:
  größtenteils auch mit Tafel und Kreide bzw. Papier und Bleistift bearbeitbar.

GRUNDBEGRIFFE

Sprachen und L-Grammatiken

Ein *Alphabet* ist eine endliche Menge von Symbolen, *Wörter* sind endliche Folgen aus diesen Symbolen (einschl. des leeren Worts ε), eine *Sprache* ist eine Menge von Wörtern.

Das Wort $xx\ldots x$, bestehend aus n-mal dem Symbol x, wird mit x^n bezeichnet (mit den Spezialfällen $x^1 = x$ und $x^0 = \varepsilon$).

Eine *L-Grammatik* besteht

- einem *Alphabet*,
- einem nichtleeren *Startwort* und
- und einer endlichen *Menge von Regeln* der Form $x \to x'$, wobei x und x' nichtleere Wörter sind (x heißt *linke Seite der Regel*, x' *rechte Seite*).

Eine *Regel* $x \to x'$ wird auf ein Wort *angewendet*, indem *sämtliche Teilwörter* des Wortes, die mit der linken Seite der Regel übereinstimmen, durch die *rechte Seite der Regel* ersetzt werden, wodurch ein neues Wort gebildet wird:

$$y_0\, x\, y_1\, x\ \ldots\ x\, y_{n-1}\, x\, y_n \Rightarrow y_0\, x'\, y_1\, x'\ \ldots\ x' y_{n-1}\, x'\, y_n,$$

wenn die linke Seite x in keinem der Teilwörter y_i enthalten ist.

Ein Wort v wird aus einem Wort u *abgeleitet*

$$u \Rightarrow^* v,$$

wenn v aus u durch sukzessives Anwenden der Regeln gebildet werden kann, d. h. wenn es eine Folge von Wörtern $x_1, x_2, \ldots, x_n$ mit

$$u \Rightarrow x_1 \Rightarrow x_2 \Rightarrow \ldots \Rightarrow x_n \Rightarrow v$$

gibt. $\Rightarrow^*$ ist die reflexive transitive Hülle der Relation $\Rightarrow$; n darf dabei auch 0 sein: für $u \Rightarrow v$ gilt auch $u \Rightarrow^* v$.

Die Menge aller Wörter, die aus dem *Startwort* einer Grammatik *abgeleitet* werden kann, wird als *die von dieser Grammatik erzeugte Sprache* bezeichnet.

Beispiele

Die vom Alphabet $\{a\}$ mit der Regel $a \to aa$ aus dem Startwort aaa erzeugte Sprache ist $\{\,\varepsilon, a^3, a^6, a^{12}, a^{24}, a^{48}, a^{96}, \ldots\}$.

Die vom Alphabet $\{a, b\}$ mit den Regeln $a \to ab$ und $b \to a$ aus dem Startwort *ba* erzeugte Sprache hat als Wörter $\{\,\varepsilon$, *ba*, *aab*, *ababa*, *abaabaab*, *abaababaababa*, *abaababaabaababaabaab*, *abaababaabaababaababaabaababaababa*, *abaababaabaababaababaabaababaabaababaab-abaabaababaabaab*, $\ldots\}$ (Ratschlag für „ausgebrannte“ Kollegen: Die nächsten sieben Wörter dieser Sprache ausrechnen, auswendig lernen und in den großen Pausen bei der Hofaufsicht immer laut rezitieren.)

Abgrenzung zu Chomsky-Grammatiken

Zur Vermeidung von Mißverständnissen sei auf die *Unterschiede zwischen L-Grammatiken und* CHOMSKY-*Grammatiken* ausdrücklich hingewiesen:

- Es gibt nur *ein* Alphabet, d. h. es wird nicht zwischen *Variablen* und *Terminalsymbolen* unterschieden.
- Es sind Start*wörter* zugelassen, nicht nur Start*variable*.
- Eine Regel anzuwenden, heißt *nicht*, *ein* Teilwort, sondern *alle Teile* eines Wortes *simultan* zu ersetzen.

Der letzte Punkt macht gerade LINDENMAYERS Intention deutlich: Modellierung von *Wachstum* bedeutet *parallele*, d. h. *gleichzeitige* Weiterentwicklung von Teilen eines Organismus gemäß den Entwicklungsregeln.

LINDENMAYER-SYSTEME

Deterministische kontextfreie L-Systeme

Die einfachste Klasse der L-Systeme ist durch Grammatiken mit folgenden Eigenschaften definiert:

- Ihr *Alphabet* besteht aus allen Großbuchstaben A, B, C, $\dots$, Z, dem kleinen Buchstaben f sowie den Symbolen $+$, $-$ und $|$.
- *Startwort* kann jedes nichtleere Wort sein.
- Zu jedem Symbol S gibt es *genau eine Regel* mit S als linker Seite, wobei zur Vereinfachung folgendes vereinbart ist:
- Zu jedem Symbol S ohne *explizite* Angabe einer Regel mit S als linker Seite gibt es die „triviale“ Regel $S \to S$.

Die Anwendung der Regeln läuft auf die simultane Ersetzung jedes vorkommenden Symbols durch die rechte Seite der entsprechenden Regel hinaus. Diese Grammatiken sind

- *kontextfrei*, da die linke Seite jeder Regel nur aus einem Symbol besteht, und
- in dem Sinne *deterministisch*, daß es keine zwei Regeln mit gleicher linker, aber verschiedener rechter Seite gibt.

Erstes Beispiel

Mit dem Startwort F und der Regel $F \to F+F-F-F+F$ ergibt sich bei der dritten Anwendung der Regel das Wort $F+F-F-F+F+F+F-F-F+F-F+F-F-F+F-F+F-F-F+F+F+F-F-F+F$, nach nochmaliger Anwendung $F+F-F-F+F+F+F-F-F+F-F+F-F-F+F-F+F-F-F+F+F+F-F-F+F+F+F-F-F+F+F+F-F-F+F-F+F-F-F+F-F+F-F-F+F+F+F-F-F+F-F+F-F-F+F+F+F-F-F+F-F+F-F-F+F-F+F-F-F+F+F+F-F-F+F-F+F-F-F+F+F+F-F-F+F-F+F-F-F+F-F+F-F-F+F+F+F-F-F+F+F+F-F-F+F+F+F-F-F+F-F+F-F-F+F-F+F-F-F+F+F+F-F-F+F$.

Es wäre dröge, weitere Wörter der Sprache in diesem Stil weiterzugeben – schon das vierte Wort sieht denkbar uninteressant aus. Der „Witz“ der Angelegenheit besteht vielmehr darin, die Wörter *anders* darzustellen.

Visualisierung

Die Wörter der Sprache dieser Grammatiken lassen sich in der Ebene visualisieren, indem die in ihnen vorkommenden Symbole als Beschreibungen von Zustandswechseln interpretiert werden. Zustände sind dabei Tripel aus

- Punkten der Ebene (gegeben durch ihre (x, y)-Koordinaten),
- Richtungswinkeln α ($0° \leq \alpha < 360°$) und
- Farben.

Die Symbole werden wie folgt als Anweisungen zur Konstruktion von – ggf. unterbrochenen – Polygonzügen aufgefaßt:

- F: ein Schritt einer gegebenen festen Schrittlänge in die aktuelle Richtung, dessen Weg mit einem Strich in der aktuellen Strichfarbe markiert wird,
- $A, \ldots, E, G, \ldots, Z$: keine Interpretation,
- f: ein Schritt in die aktuelle Richtung ohne Strich,
- $a, \ldots, e, g, \ldots, z$: Umschaltung der aktuellen Strichfarbe auf eine andere Farbe (für jeden kleinen Buchstaben – ausgenommen f – ist eine Farbe festgelegt, z. B. r, g, b, y, c, m, o, s bzw. w für rot, grün, blau, gelb, türkis, lila, orange, schwarz bzw. weiß),
- $+$: eine Drehung um einen gegebenen festen Drehwinkel nach links,
- $-$: eine Drehung um diesen Drehwinkel nach rechts,
- $|$: eine Kehrtwendung, d. h. eine Drehung um 180°.

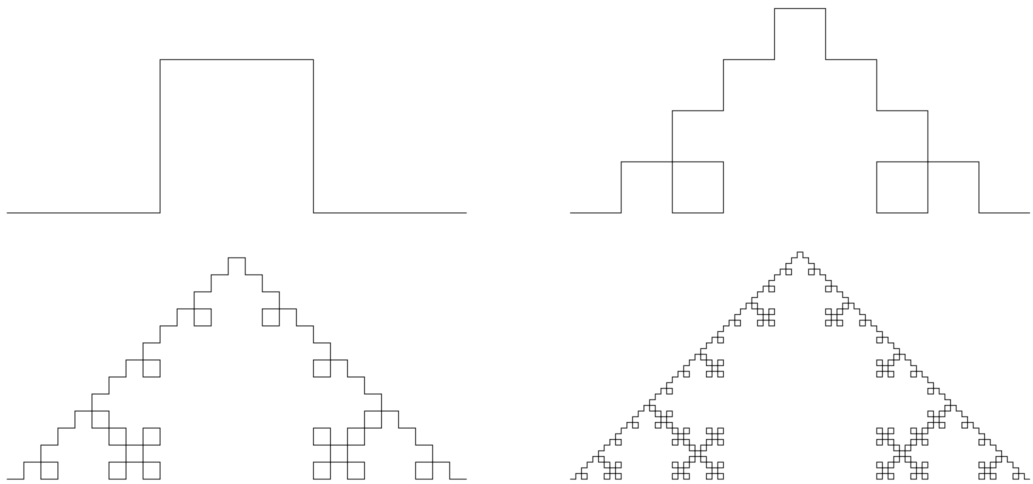
Die Richtung ist durch den Richtungswinkel im „mathematischen Sinn“ festgelegt (z. B. $0° =$ nach rechts, $90° =$ nach oben, $225° =$ nach links unten). Wenn keine anderen Werte festgelegt werden, sind Richtungs- und Drehwinkel anfangs 90°.

Die Koordinaten sind anfangs $(0, 0)$.

Die Schrittlänge wird zweckmäßig – entsprechend der gewünschten Größe der Graphik auf dem Ausgabemedium zu ihrer Darstellung – gewählt; die aktuelle Strichfarbe ist anfangs sinnvoll vorgegeben: je nach Hintergrund schwarz auf weiß oder weiß auf schwarz.

Visualisierung des ersten Beispiels

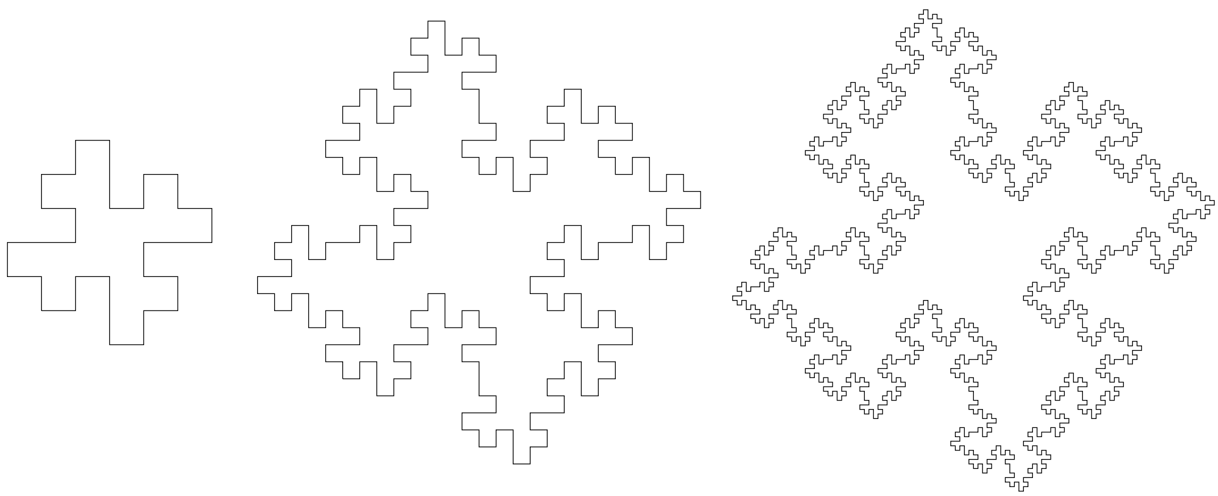
Mit dem Startwinkel 0° und dem Drehwinkel 90° ergeben sich bei der Visualisierung für das zweite bis fünfte Wort der Sprache des oben angeführten Beispiels die folgenden Graphiken (jede nachfolgende Graphik ist gegenüber der vorigen um den Faktor 3 verkleinert):



– weniger langweilig als die Wiedergabe der Wörter als Zeichenketten.

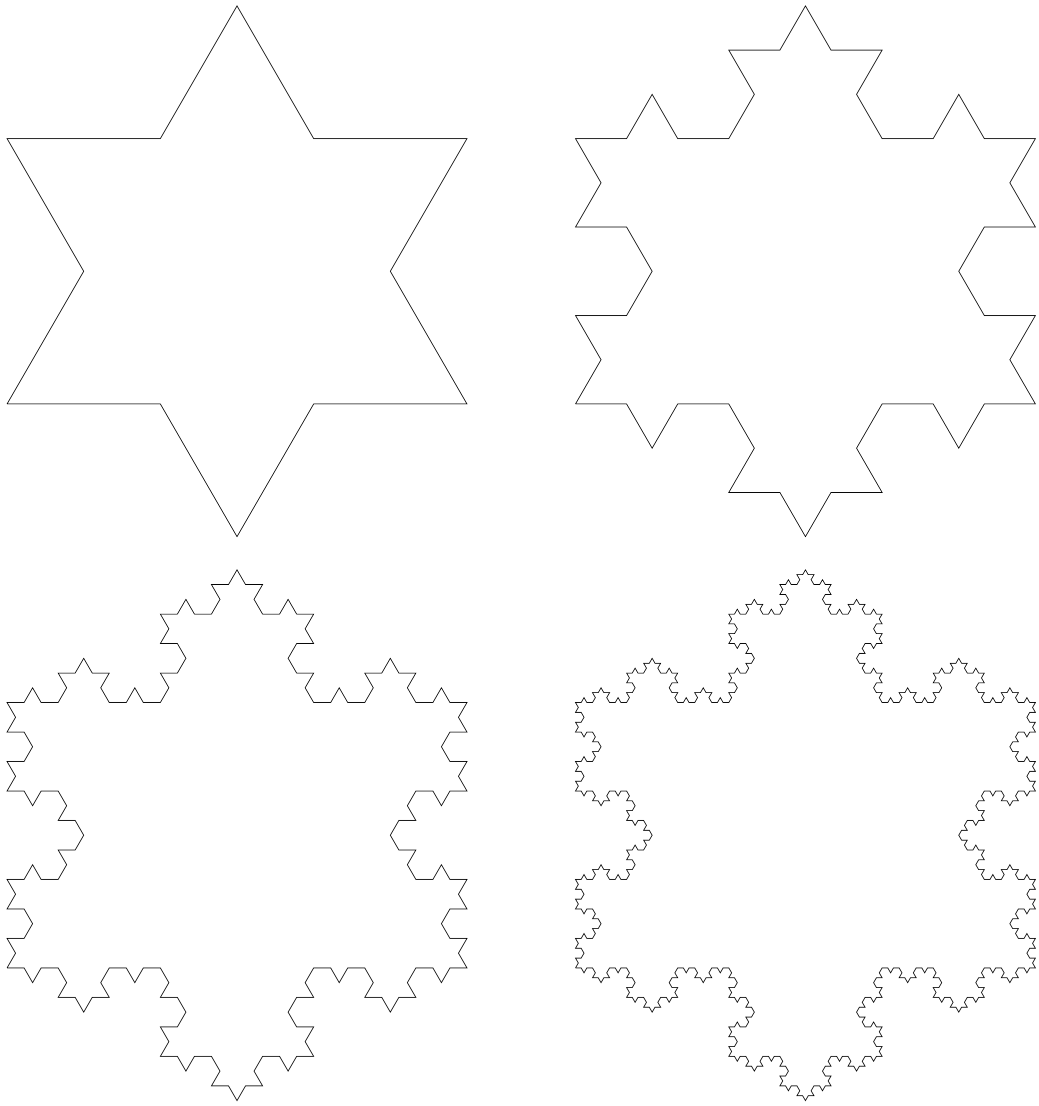
Beispiel einer Koch-Kurve

Hier die Visualisierung des zweiten bis vierten Wortes der Sprache, die von der Grammatik mit dem Startwort $F - F - F - F$ und der Regel $F \to F + F - F - FF + F + F - F$ erzeugt wird:



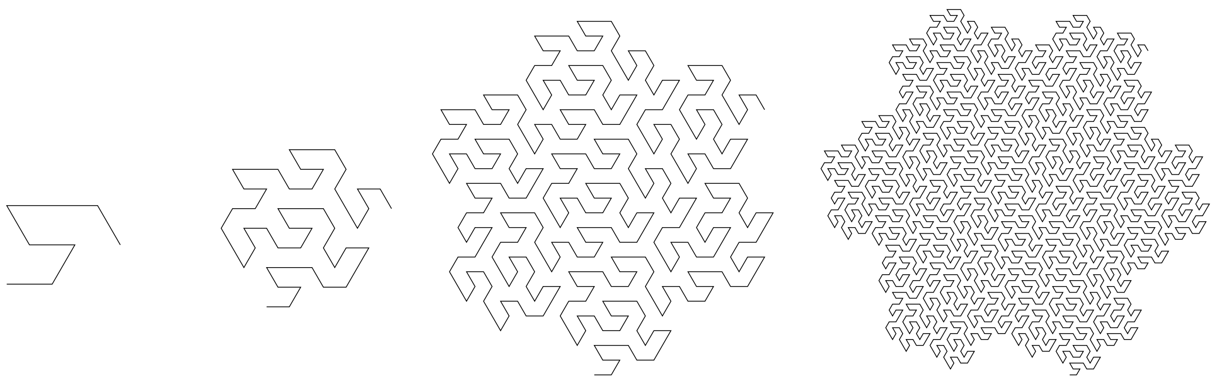
Die Schneeflockenkurve

Hier die Visualisierung des zweiten bis fünften Wortes der Sprache, die von der Grammatik mit dem Startwort $F{+}{+}F{+}{+}F$ und der Regel $F \to F - F + +F - F$ erzeugt wird, mit verändertem Drehwinkel 60° (das Startwort ist ein gleichseitiges Dreieck):



Raumfüllende Kurven

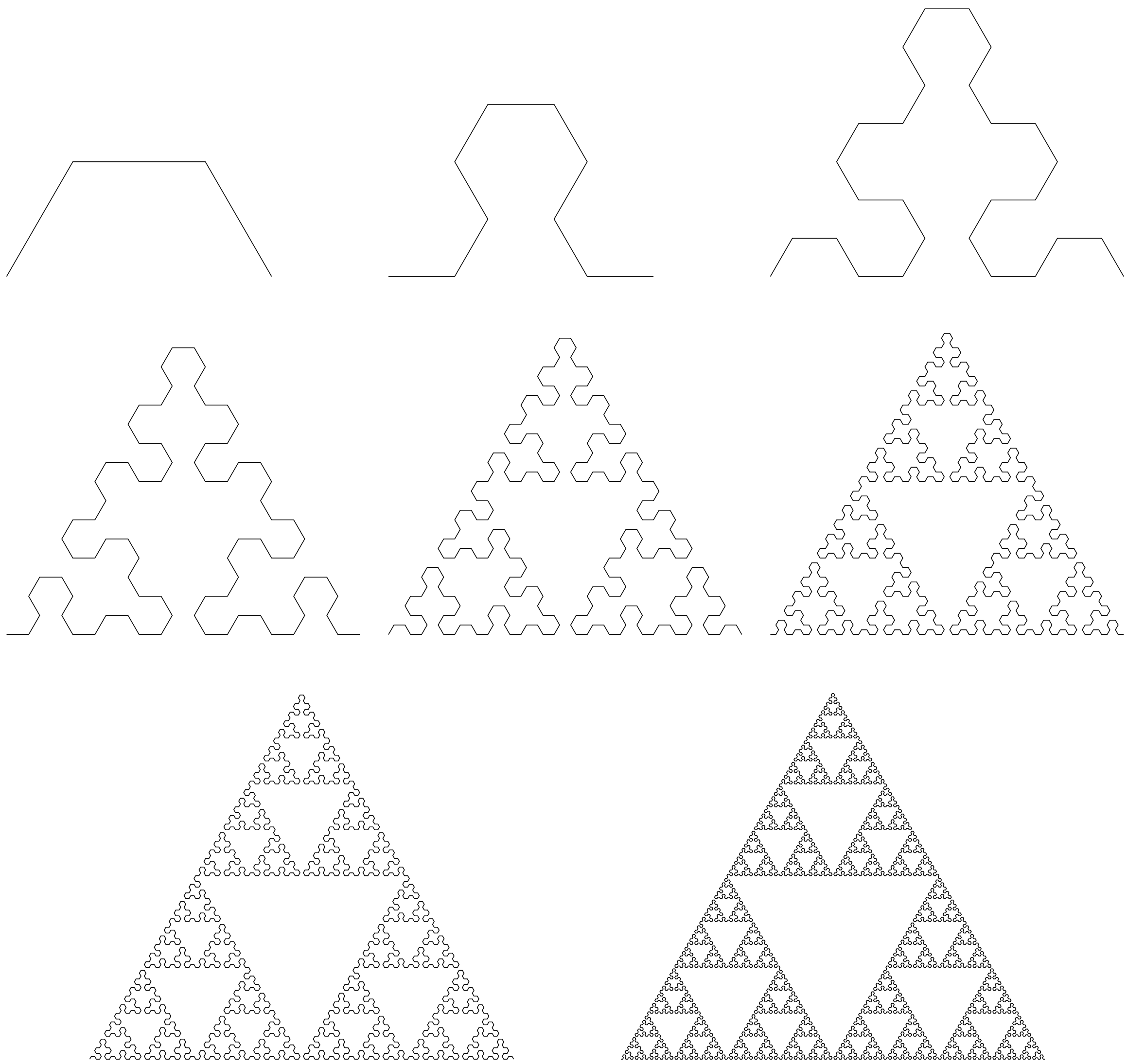
Hier die ersten vier Wörter der Grammatik, die von den etwas komplizierteren Regeln $LF \to LF{+}RF{+}{+}RF{-}LF{-}{-}LFLF{-}RF{+}$ und $RF \to -LF + RFRF + +RF + LF - -LF - RF$ mit dem Drehwinkel $60°$ erzeugt werden, beginnend mit dem Startwort LF und dem Richtungswinkel $0°$:



Ein Beispiel einer Kurve, die – abgesehen von fraktalen Löchern in Form gleichseitiger Dreiecke – ein gleichseitiges Dreieck „füllt“, ist das *Sierpinski-Dreieck* (ins „unendliche“ fortgesetzt ein total unzusammenhängender Raum vom Maß 0), gegeben durch die Grammatik mit den Regeln $LF \to +RF-LF-RF+$ und $RF \to -LF+RF+LF-$.

Die beiden zusätzlichen Variablen L und R geben an, ob bei der nächsten Ableitung ein „Links-“ oder ein „Rechtsbogen“ eingesetzt wird.

Die ersten neun Wörter der Grammatik, die vom Startwort LF mit dem Drehwinkel $60°$ und dem anfänglichen Richtungswinkel $0°$ erzeugt werden, „sehen“ so „aus“:



Verzweigte L-Systeme

Zur Darstellung des Wachstums von Pflanzen reicht das bisherige nicht:

Es werden „Verzweigungen“ gebraucht, d. h. Stellen, an denen mehrere Strichfolgen beginnen. Bei sequentieller Konstruktion der Graphik heißt das, nach der Erzeugung eines „Astes“ zu einer Stelle „zurückzukehren“, um dort einen weiteren Ast abzweigen zu können.

Dabei muß nicht nur der Ort der Abzweigung gefunden werden, sondern auch die ursprüngliche Richtung an dieser Stelle. Damit liegt die Idee nahe, einen *Stapel* einzuführen, auf dem der Zustand bei einer Abzweigung archiviert wird, um ihn nach der Konstruktion eines Zweiges wieder vom Stapel restaurieren zu können.

Die *Grammatik* wird um Symbole für diese beiden Operationen erweitert:

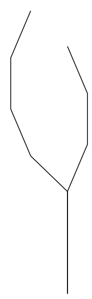
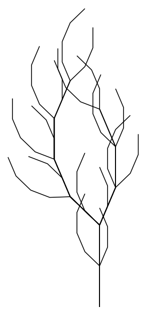
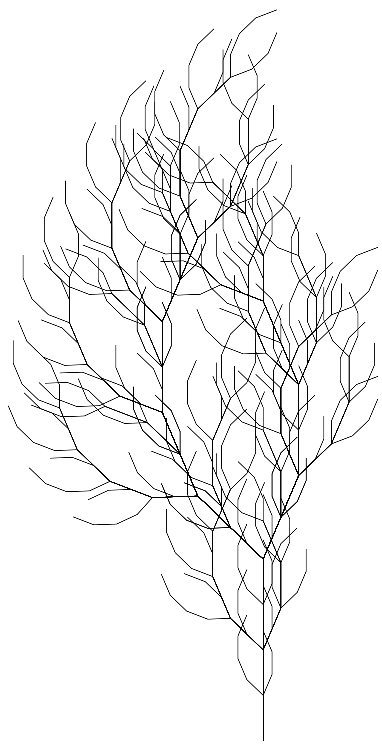
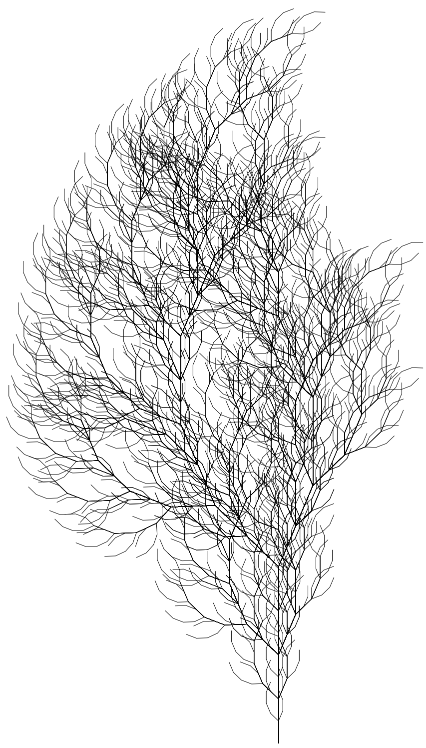
- Zum bisherigen *Alphabet* werden die *eckigen Klammern* [und] hinzugefügt;
- Wörter, die diese Klammern enthalten, werden als *rechte Seiten von Regeln* nur *dann* zugelassen, wenn die Klammern *passend* (d. h. entsprechend den üblichen Klammerregeln; man könnte das mit einer Grammatik definieren) verwendet sind.

Bei der *Visualisierung* werden die Klammern folgt interpretiert:

- [: Ablage des aktuellen Zustands auf dem Stapel und
-] : Aufnahme des zuletzt abgelegten Zustands vom Stapel.

Ein Busch

Startwort F, Regel $F \to FF+[+F-F-F-F]-[-F+F+F]$, Drehwinkel $23°$:

3-dimensionale LSysteme

Zur Konstruktion von Lindenmayer-Systemen im Raum wird die Grammatik angereichert:

- Das *Alphabet* wird um die Symbole _, $\wedge$, \ und / erweitert.

Diese Symbole werden wie folgt als Anweisungen zur Konstruktion von – ggf. unterbrochenen – Polygonzügen aufgefaßt:

- _: eine Neigung um den – bei + gegebenen – Drehwinkel nach vorne,
- $\wedge$: eine Neigung um den Drehwinkel nach hinten,
- \: Kippen um den Drehwinkel nach links,
- /: Kippen um den Drehwinkel nach rechts.

LS-Grammatiken, in denen *keins* dieser Symbole vorkommt, werden als *2D-Grammatiken* bezeichnet, Grammatiken *mit* mindestens einem dieser Symbole als *3D-Grammatiken*.

ANHANG

Das Programm `LS`

Die Grammatiken werden als Texte in Dateien abgelegt, deren Namen das Suffix `.ls` haben müssen.

Diese `ls`-Dateien müssen *zeilenweise* wie folgt aufgebaut sein:

- Startwinkel (in der Form „$g°$“ oder „$g°m$“ für g Grad m Minuten mit $0 \leq g < 360$ und $0 \leq m < 60$),
- Startwort,
- eine oder mehrere Regeln (jede in einer eigenen Zeile), wobei die linken Seiten aus höchstens vier Buchstaben bestehen dürfen und von den rechten Seiten durch „`->`“ (Minus- und Größer-Zeichen) getrennt sein müssen, und – falls vorhanden – Klammern auf den rechten Seiten sinnvoll gesetzt sein müssen,
- Drehwinkel (in der Form wie Startwinkel),
- Anzahl der Iterationen (in Form einer ganzen Zahl ≥ 1).

Leerzeichen werden ignoriert; damit kann die Darstellung der einzelnen Zeilen zur besseren Lesbarkeit strukturiert werden. Aus dem gleichen Grunde bleiben auch Leerzeilen bedeutungslos.

In Zeilen, in denen das *Kommentarzeichen* `#` vorkommt, wird alles ab diesem Zeichen ignoriert, wodurch es – zum besseren Verständnis wie zu Dokumentationszwecken – möglich ist, die Grammatiken mit erläuterndem Text zu versehen.

Die Zeilen mit den Angaben für den Start- oder den Drehwinkel oder die Anzahl der Iterationen dürfen auch fehlen; in diesen Fällen ist der Startwinkel auf 0° und der Drehwinkel auf 90° voreingestellt und die Anzahl der Iterationen beträgt 1.

Visualisierung der Worte von L-Systemen

Mit einem Editor, der reine Textdateien (d. h. ohne Formatanweisungen) erzeugt, wird im Arbeitsverzeichnis `$HOME/.LS` eine Datei mit dem Suffix `ls` editiert, deren Inhalt eine LS-Grammatik ist.

Ein Wort einer Grammatik wird durch den Aufruf des Programms `LS` in eine persistente Folge von Figuren (in der Regel Strecken) in der Ebene bzw. im Raum verwandelt, wenn sie im oben beschriebenen Format als Textdatei abgelegt ist. Die Datei, in der diese Folge abgelegt ist, besteht aus dem Namen der Datei der Grammatik mit dem Suffix `.mug` anstelle von `.ls`.

Bei höheren Anzahlen von Iterationen kann das einen Augenblick dauern.

Der Name des Wortes der Grammatik, das visualisiert werden soll, d. h. der Name der entsprechenden Datei ohne das Suffix „`.ls`“, kann dem Programmaufruf als Parameter mitgegeben werden; sonst muß er nach dem Programmaufruf eingegeben werden. Drücken der Suchtaste `F2` nach dem Programmaufruf liefert ein Menü aller vorhandenen Grammatiken, zwischen deneneinen Augenblick geblättert und aus denen mit der Eingabetaste das zu visualisierende Wort ausgewählt wird.

ANHANG

Das Programm `zeig`

Dreidimensionale Grammatiken bauen Gebilde im Raum auf, die nicht ohne weiteres „flach“ auf dem Bildschirm dargestellt werden können. Vielmehr muß es möglich sein, sie von verschiedenen Seiten zu betrachten, um einen realistischen Eindruck der räumlichen Struktur des visualisierten Wortes zu erhalten.

Dazu ist ein entsprechendes Programm notwendig (das natürlich auch zur Betrachtung dreidimensionaler Strukturen verwendet werden kann, die auf ganz andere Weise als durch Lindenmayer-Grammatiken erzeugt sind). Das entsprechende Programm `zeig` ist Bestandteil des MUlaodrenUniverSums.

Die aktuelle Implementierung von `zeig` basiert auf `openGL`.

Ein Druck auf die Hilfe-Taste (`F1`) erläutert den Gebrauch von `zeig`. Das Programm wird mit der Schlußtaste `Esc` beendet (in Problemfällen kann es jederzeit mit der Kombination `Strg+C` abgebrochen werden).

Mit der Einfügetaste `Einfg` wird die Graphik in einer Datei im *portable pixmap*-Format mit dem Suffix `.ppm` abgelegt, die – z. B. mit den Programmen aus der `netpbm`-Serie – in andere Graphik-Formate gewandelt und/oder – z. B. mit dem `gimp` – weiterverarbeitet werden kann.

Mit der Drucktaste `Druck` wird die Graphik ausgedruckt, wenn die entsprechenden Voraussetzungen erfüllt sind.

Literatur

A. Lindenmayer:
Mathematical Models for Cellular Interaction in Development
J. Theor. Biol. 18 (1968), 230–315

B. B. Mandelbrot:
The Fractal Geometry of Nature
W. H. Freeman 1982

P. Prusinkiewicz:
The Algorithmic Beauty of Plants
Springer-Verlag 1990 (dort umfassende weitere Literaturangaben)

Zentralprojektion

Gegeben sei ein Augpunkt a im $\mathbb{R}^3$ mit lokalem orthonormiertem Koordinatensystem $\{e_1, e_2, e_3\}$ (d. h. $\langle e_1, e_1\rangle = \langle e_2, e_2\rangle = \langle e_3, e_3\rangle = 1$, $\langle e_1, e_2\rangle = \langle e_2, e_3\rangle = \langle e_1, e_3\rangle = 0$ und $e_3 = [e_1, e_2]$).

Der $\mathbb{R}^3$ soll durch Zentralprojektion von a auf die Ebene E durch $z = a + e_1$ mit dem Stellungsvektor e_1 abgebildet werden, d. h. e_1 gibt die Blickrichtung von a aus an; E hat den Abstand 1 von a.

Das Koordinatensystem der Bildebene E sei wie folgt gewählt:

Der Ursprung ist z und die Einheitsvektoren in x- bzw. y-Richtung sind $-e_2$ bzw. e_3; die Koordinaten (x, y) sind dann in der Parameterdarstellung

$$E = \{\, a + e_1 - xe_2 + ye_3 \mid x,\, y \in \mathbb{R} \,\} \tag{1}$$

ablesbar.

Der Bildpunkt b eines Urbildpunktes u errechnet sich wie folgt: Der Ansatz

$$b = a + \lambda(u - a) = a + e_1 - xe_2 + ye_3 \tag{2}$$

liefert $\langle b - a, e_1\rangle = \lambda\langle u - a, e_1\rangle = \langle e_1, e_1\rangle - x\langle e_2, e_1\rangle + y\langle e_3, e_1\rangle = 1 - 0 + 0 = 1$, damit $\lambda = \dfrac{1}{\langle u - a, e_1\rangle}$, also

$$b = a + \frac{1}{\langle u - a, e_1\rangle}(u - a). \tag{3}$$

x und y lassen sich aus (2) berechnen:

$\langle b-a, e_2\rangle = \lambda\langle u-a, e_2\rangle = \langle e_1, e_2\rangle - x\langle e_2, e_2\rangle + y\langle e_3, e_2\rangle = 0 - x + 0 = -x$, analog $\langle b - a, e_3\rangle = y$.

Durch Einsetzen von (3) ergibt sich für die Koordinaten (x, y) des Bildes b eines Urbildpunktes u bei Zentralprojektion von a im Koordinatensystem von E:

$$(x, y) = \left(-\frac{\langle u - a, e_2\rangle}{\langle u - a, e_1\rangle}, \frac{\langle u - a, e_3\rangle}{\langle u - a, e_1\rangle}\right). \tag{4}$$