2. Morphogenese, Turtle-Geometrie, L-Systeme

In der Artificial Life-Forschung werden verschiedene Ansätze zur Untersuchung der Phänomene "Gestaltbildung" bzw. "Musterbildung" und "Wachstum" bzw. "Ontogenese" verfolgt.

"Morphogenese" = Entwicklung von Mustern und Form bei lebenden Organismen.

Gemeinsam ist allen Ansätzen der Grundgedanke, von *lokalen Regeln* auszugehen.

Ziele von Morphogenese-Simulationen (nach Adrian D. Bell):

- Analyse der Natur und Komplexität der Mechanismen, die die Gestaltbildung bestimmen
- ein besseres Verständnis der Form und Entwicklung spezieller Organismen, das man durch die Konstruktion "realitätsnaher" Modelle zu gewinnen versucht
- Analyse des Einflusses einzelner Parameter auf die Gesamtform ("Sensitivitätsanalyse"), dies führt zu besserer Einschätzung der Beziehungen der Parameter untereinander und kann Einsicht in die Richtung evolutionärer Veränderungen liefern
- Zwecke der computergestützten Lehre
- Grafikdesign, Computerkunst, Verwendung in der Landschaftsplanung

Problem:
Quantifizierung der Qualität von Morphogenese-Simulationen
- visueller Vergleich allein ist zwar oft vielsagend, aber nicht präzise
- es fehlt ein formales Maß für die Ähnlichkeit / Distanz zweier Formen

Historische Ursprünge von Morphogenese-Modellen:

2 frühe Ansätze, gegensätzlich

- d'Arcy Thompson (Biologe; "On Growth and Form" 1917): Die Form eines Organismus ergibt sich aus seinen Wachstumsraten in verschiedenen Richtungen; Studium der Form setzt Studium des Wachstums voraus
- Alan Turing ("The chemical basis of morphogenesis" 1952): Formbildung wird gesteuert von Substanzen, die sich in einem nicht-wachsenden Medium (Gewebe) ausbreiten und miteinander chemisch reagieren.

Systematische Einteilung von Morphogenese-Modellen: nach verschiedenen Kriterien möglich:

- zugrundeliegendes Medium **konstant** im Raum oder **wachsend**
- **strukturorientiert** oder **raumorientiert** (im letzteren Fall ist der Raum, der die simulierte Struktur umgibt, mit Gegenstand der Modellierung)
- **kontinuierlich** oder **diskret** (Unterscheidung anwendbar für: die Struktur, Raum, Zeit, Zustände)
- **Topologie:** linear-unverzweigt, Verzweigungs-struktur, Netzwerk, 2D-Oberfläche, solides 3D-Objekt
- **Nachbarschaft** zwischen Modulen der Struktur kann fest oder variabel sein (bewegliche Module, z.B. Blutkörperchen)
- Art der **Kommunikation** zwischen den Modulen einer Struktur:

  - "blinde Muster": Erzeugung und Überleben von Modulen wird nur vom Eltern-Modul gesteuert ("lineage control"), unabh. vom Rest der Struktur und von der Umgebung.
    "nichtsensitive Wachstumsregeln"

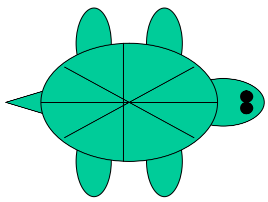
  - "selbstregulierte Muster": über die Topologie der bereits erzeugten Struktur vermittelte Kommunikation steuert die Entwicklung der Struktur ("endogenous control"; kein Umwelteinfluss)
    "kontextsensitive Wachstumsregeln" ("Kontext" bezieht sich auf die Topologie der Struktur)

  - "sehende Muster": Ereignisse im umgebenden Raum können Einfluss auf die Entwicklung der Struktur haben ("exogenous control")
    "global sensitive Wachstumsregeln"

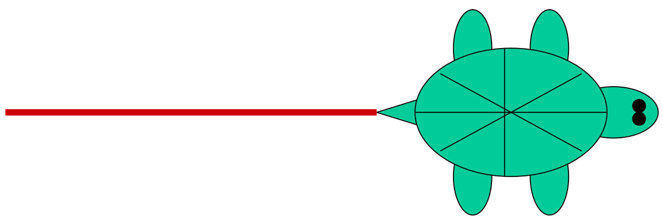
Turtle-Geometrie

- Beschreibung statischer, stückweise linearer geometrischer Strukturen
- imperative Sprache zur Steuerung eines Zeichen-geräts
- relative Längen- und Richtungsinformationen

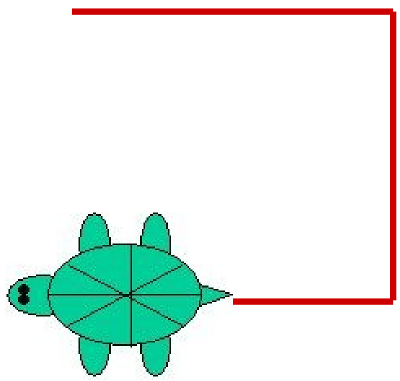
Turtle:
zeichnende Schildkröte, die auf Befehle hört



F0



F0 RU(90) F0 RU(90) LMul(0.5) F0



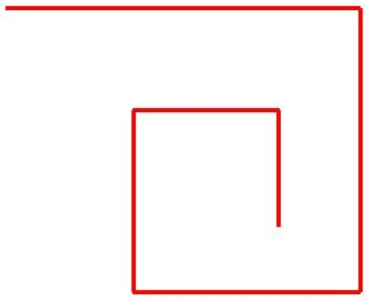
Wiederholung von Abschnitten der Zeichenkette möglich mit "**for**"

z.B. **for ((1:3)) (A B C)**
liefert **A B C A B C A B C**

was ist das Ergebnis der Interpretation von

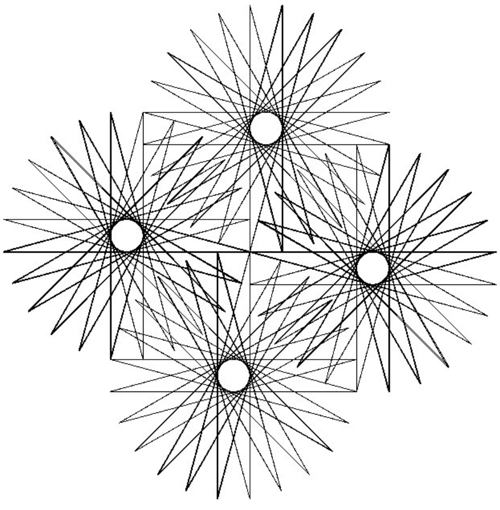
L(10) for ((1:6))
(F0 RU(90) LMul(0.8)) *?*

```
L(10) for ((1:6))
       ( F0 RU(90) LMul(0.8) )
```



anderes Beispiel:

```
for ((1:20)) ( for ((1:36))
               ( F0 RU(165) F0 RU(165) ) RU(270) )
```



Turtle geometry ("Schildkrötengeometrie")

befehlsgesteuertes, lokales Navigieren im 2D- oder 3D-Raum (Abelson & diSessa 1982; vgl. Programmiersprache "LOGO")

"Turtle": Zeichen- oder Konstruktionsgerät (virtuell)

- speichert (grafische und nicht-grafische) Informationen

- mit einem Zustandsspeicher assoziiert (wichtig für Verzweigungen)

- aktueller Zustand der Turtle enthält z.B. Information über aktuelle Liniendicke, Schrittweite, Farbe, weitere Eigenschaften des als nächstes zu konstruierenden Objekts

Befehle (Auswahl):

`F0` "Forward", mit Konstruktion eines Elements (Linienstück, Segment, Gebäudetrakt...), benutzt wird die aktuelle Schrittweite für die Länge (die Null steht für "keine explizite Längenfestlegung")

`M0` forward ohne Konstruktion (*Move*-Befehl)

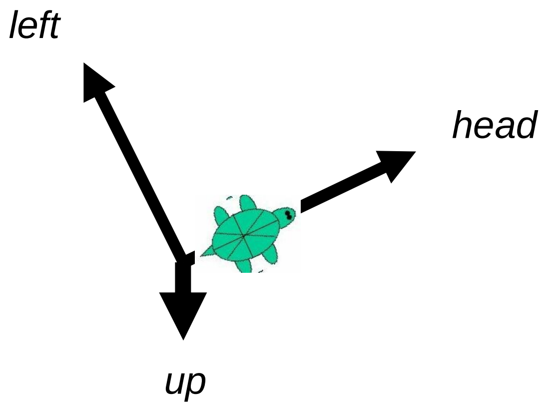
`L(x)` ändere die aktuelle Schrittweite (Länge) zu *x*

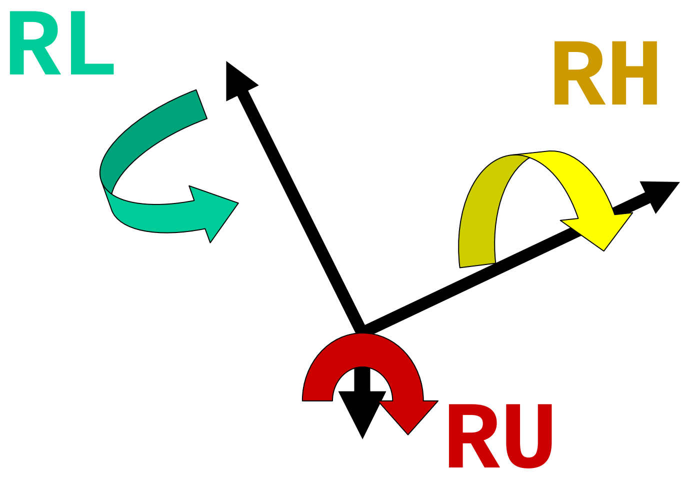
`LAdd(x)` inkrementiere die aktuelle Schrittweite um *x*

`LMul(x)` multipliziere die aktuelle Schrittweite mit *x*

`D(x)`, `DAdd(x)`, `DMul(x)` analog für die aktuelle Dicke

Erweiterung auf 3D-Grafik:
Turtle-Rotationen um 3 Achsen





3D-Befehle:

`RU(45)` Drehung der *turtle* um die "up"-Achse um 45°

`RL(...)`, `RH(...)` analog um "left" und "head"-Achse

up-, *left*- und *head*-Achse bilden ein rechtwinkliges, räumliches Koordinatensystem, das von der *turtle* mitgeführt wird

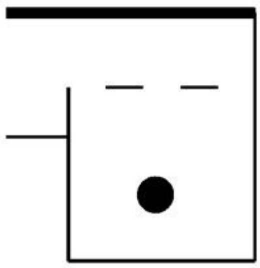
`RV(x)` Rotation "nach unten" mit durch *x* vorgegebener Stärke

`RG` Rotation ganz nach unten (Richtung (0, 0, -1))

Beispiel:

```
L(100) D(3) RU(-90) F(50) RU(90) M0 RU(90) D(10) F0 F0
       D(3) RU(90) F0 F0 RU(90) F(150) RU(90) F(140) RU(90)
       M(30) F(30) M(30) F(30) RU(120) M0 Sphere(15)
```

erzeugt



was ist das Ergebnis der Interpretation der Zeichenkette

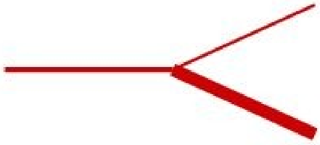
`L(10) F0 RU(45) F0 RU(45) LMul(0.5) F0 M0 F0` *?*

Verzweigungen:
Realisierung mit Speicher-Befehlen

[lege aktuellen Zustand auf Speicher ("Ablage", Stack)

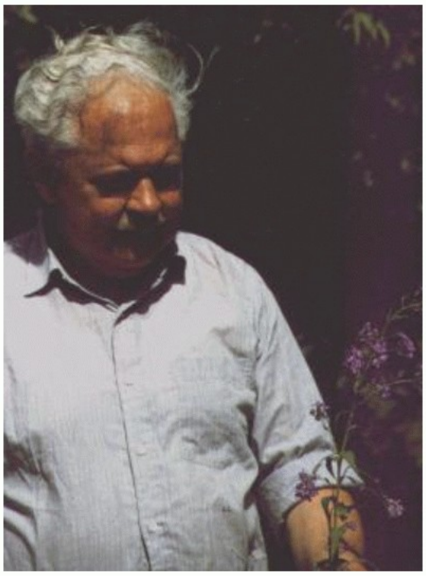
] nimm obersten Zustand von der Ablage und mache diesen zum aktuellen Zustand (damit: Ende der Verzweigung)

`F0 [ RU(-20) F0 ] RU(20) DMul(2) F0`



L-Systeme (Lindenmayer-Systeme)

Modellierung von Pflanzenarchitekturen
als Regelanwendungen: L-Systeme



Aristid Lindenmayer
(1925 – 1989)

Einordnung der L-System-Modelle:

- strukturorientiert
- *dynamisch*
- diskret in Zeit und Struktur (aber kontinuierlich im Raum: Lage, Richtung)
- Topologie linear oder verzweigt
- feste Nachbarschaften

Grundlagen der L-Systeme

- Modellierung der Pflanzenmorphologie aus diskreten Einheiten
- Produktionsregeln bestimmen vom Startzustand iterativ das Pflanzenwachstum
- Selbstähnlichkeit „eingebaut", Fraktale als Grenzfall

mathematische Definition:

Ein L-System ist ein Tripel (Σ, α, *R*); darin ist:

Σ eine Menge von Zeichen, das *Alphabet*,

α eine Zeichenkette mit Zeichen aus Σ, das *Startwort* (auch "Axiom"),

R eine Menge von Regeln der Form

Zeichen $\rightarrow$ Zeichenkette;

darin sind das Zeichen auf der linken Regelseite und die Zeichenkette aus Σ entnommen.

Ein *Ableitungsschritt* (rewriting) einer Zeichenkette besteht aus der Ersetzung aller ihrer Zeichen, die in linken Regelseiten vorkommen, durch die entsprechenden rechten Regelseiten.

Man vereinbart: Zeichen, auf die keine Regeln anwendbar sind, werden unverändert übernommen.

Ergebnis:

Ableitungskette von Zeichenketten, die sich durch wiederholte Anwendung des *rewriting*-Vorgangs aus dem Startwort ergeben.

$$\alpha \rightarrow \sigma_1 \rightarrow \sigma_2 \rightarrow \sigma_3 \rightarrow \ldots.$$

Beispiel:

Alphabet {A, B}, Startwort A

Regelmenge:

$A \rightarrow B$
$B \rightarrow AB$

Ableitungskette:

$A \rightarrow B \rightarrow AB \rightarrow BAB \rightarrow ABBAB \rightarrow BABABBAB$
$\rightarrow ABBABBABABBAB \rightarrow BABABBABABBABBABABBAB$
$\rightarrow \ldots$

wie lang ist die n-te Zeichenkette in dieser Ableitung?

was für die Modellierung von grafischen und biologischen Strukturen noch fehlt:

eine *geometrische Interpretation*

Füge also hinzu:

eine Abbildung, die jeder Zeichenkette
eine Teilmenge des 3-dimensionalen Raumes zuordnet

dann: "interpretierte" L-System-Abarbeitung

$$\begin{array}{ccccccccc} \alpha & \rightarrow & \sigma_1 & \rightarrow & \sigma_2 & \rightarrow & \sigma_3 & \rightarrow & \ldots. \\ & & \downarrow & & \downarrow & & \downarrow & & \\ & & S_1 & & S_2 & & S_3 & & \ldots. \end{array}$$

S_1, S_2, S_3, ... können als Entwicklungsstufen eines Objekts, einer Szene oder eines Organismus interpretiert werden.

Für die Interpretation der Zeichenketten:

Turtle-Geometrie

Der Turtle-Befehlsvorrat wird zu einer Untermenge der Zeichenmenge des L-Systems.

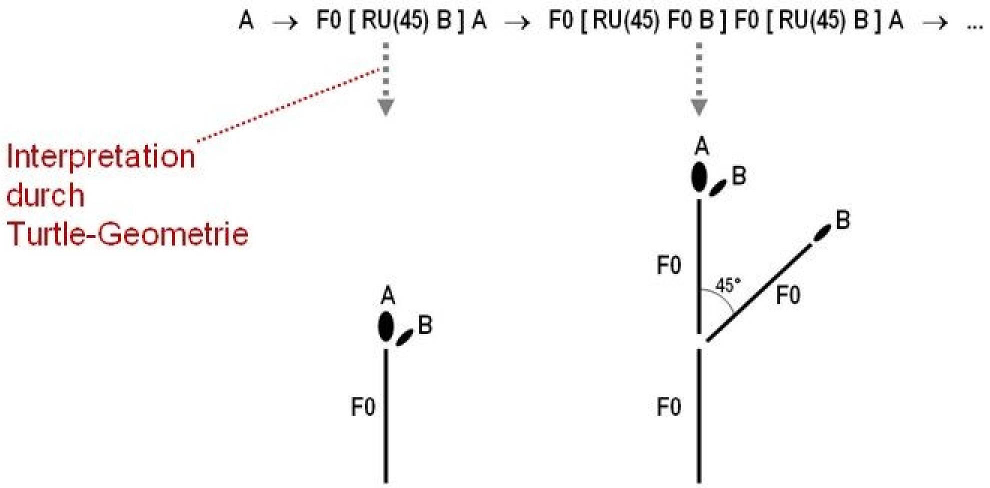
Symbole, die nicht Turtle-Befehle sind, werden von der Turtle ignoriert.

Beispiel:

Regeln

```
A → F0 [ RU(45) B ] A ;
B → F0 B ;
```

Startwort `A`



(`A` und `B` werden normalerweise nicht geometrisch interpretiert.)

was für eine Struktur liefert das L-System

```
A → [ LMul(0.25) RU(-45) F0 ] F0 B;
B → [ LMul(0.25) RU(45) F0 ] F0 A;
```

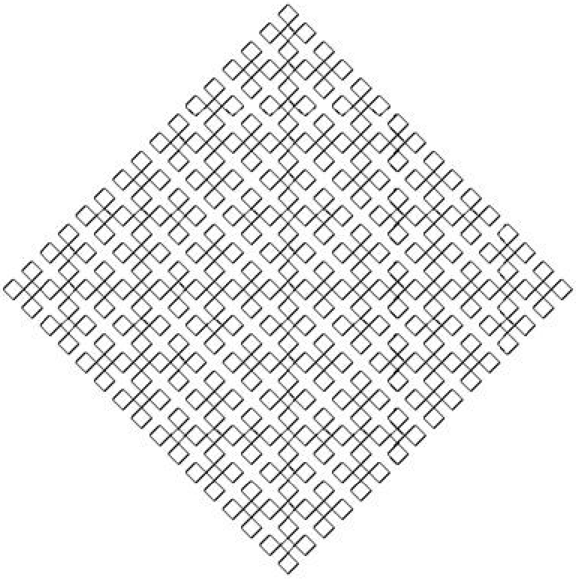
mit Startwort `L(10) A`?

äquivalente Regel:

```
A → [ LMul(0.25) RU(-45) F0 ] F0 RH(180) A;
```

Flächenfüllende Kurve:

```
Start ==> L(10) RU(-45) X RU(-45) F(1) RU(-45) X;
X ==> X F0 X RU(-45) F(1) RU(-45) X F0 X
```



indisches Kolam-Muster
„Anklets of Krishna“

Beispiel für ein Fraktal:

Koch'sche Kurve

```
Start → RU(90) F(10);
F(x) → F(x/3) RU(-60) F(x/3) RU(120) F(x/3) RU(-60) F(x/3)
```