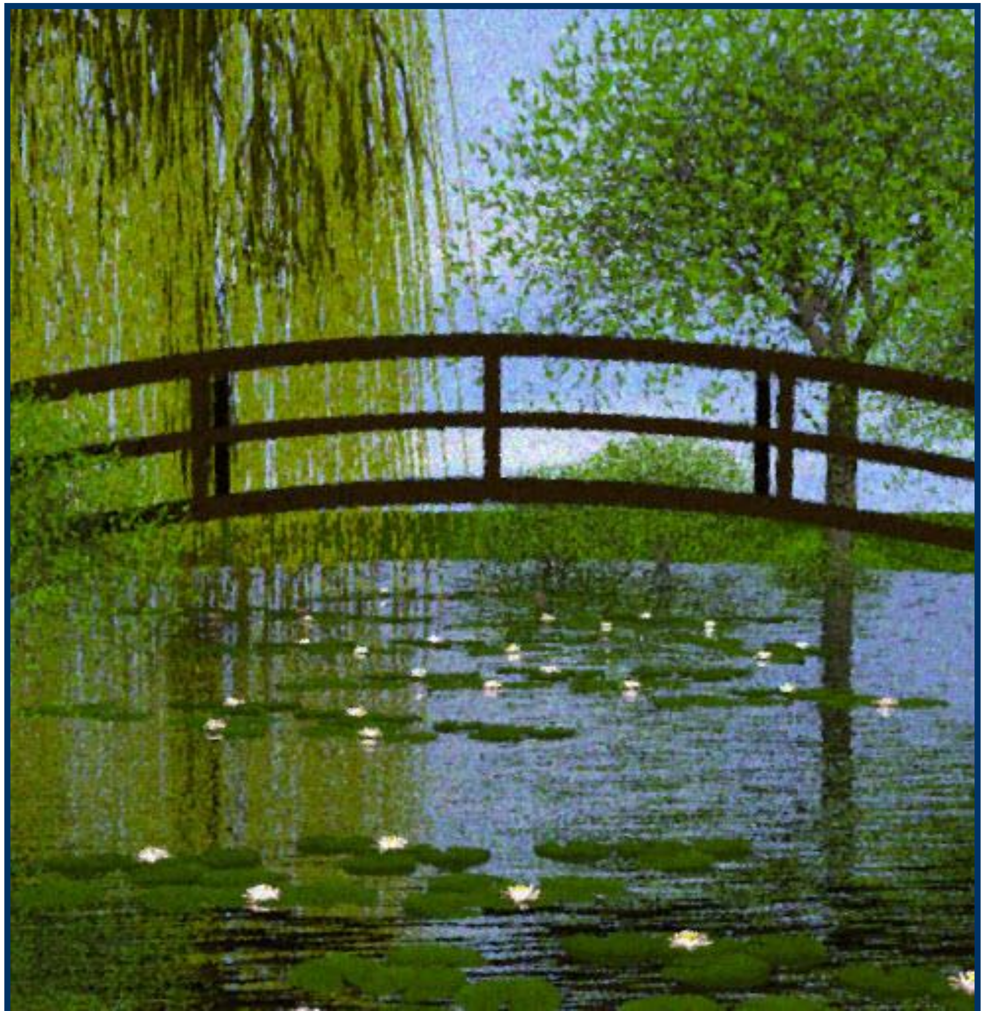
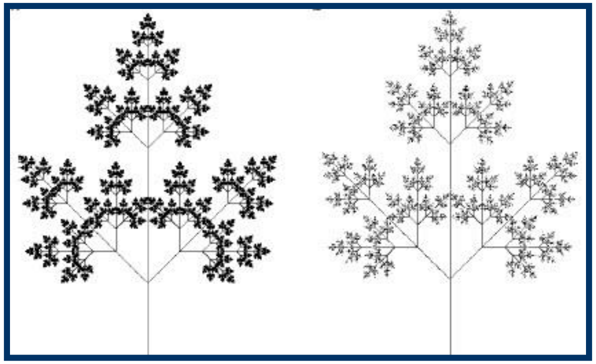
12.2 Plants

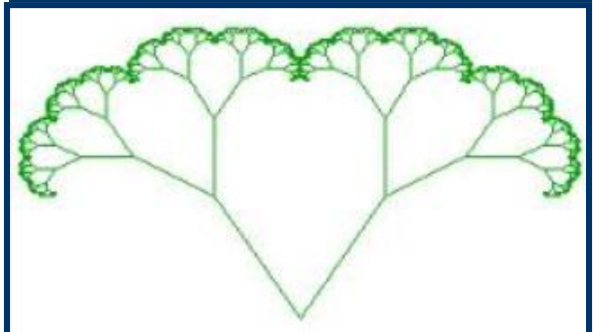
- modeling and animation of plants represents an interesting and challenging area
- exhibit *arbitrary complexity* while possessing a *constrained branching structure*
- grow from a single source point, developing a branching structure over time while the individual structural elements elongate
- the topology of a plant is characterized by a *recursive branching structure*
- have been modeled using *particle systems*, *fractals*, and *L-systems*
- focus on the *growth process* of plants

Fractals – a never ending pattern

- a mathematical set that typically displays self-similar patterns, which means it is "the same from near as from far"



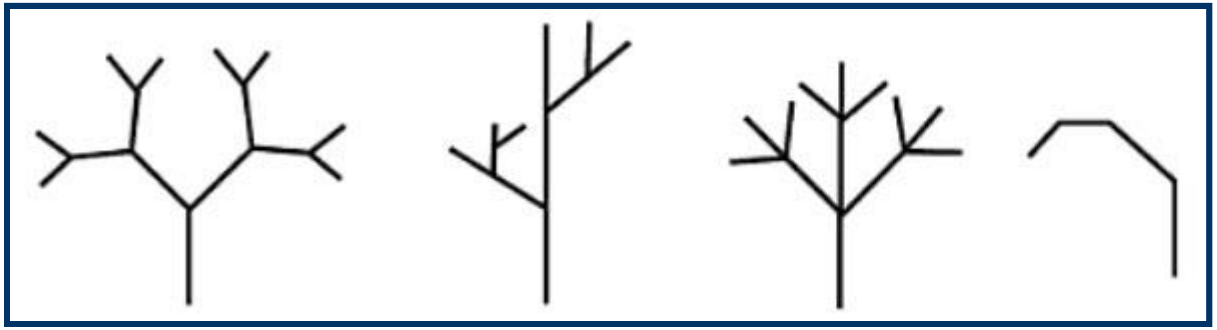




L-Systems – rewriting

- defining complex objects by successively replacing parts of a simple initial object using a set of *rewriting rules* or *productions*

Basic branching structures



Structures resulting from repeated application of a single branching scheme

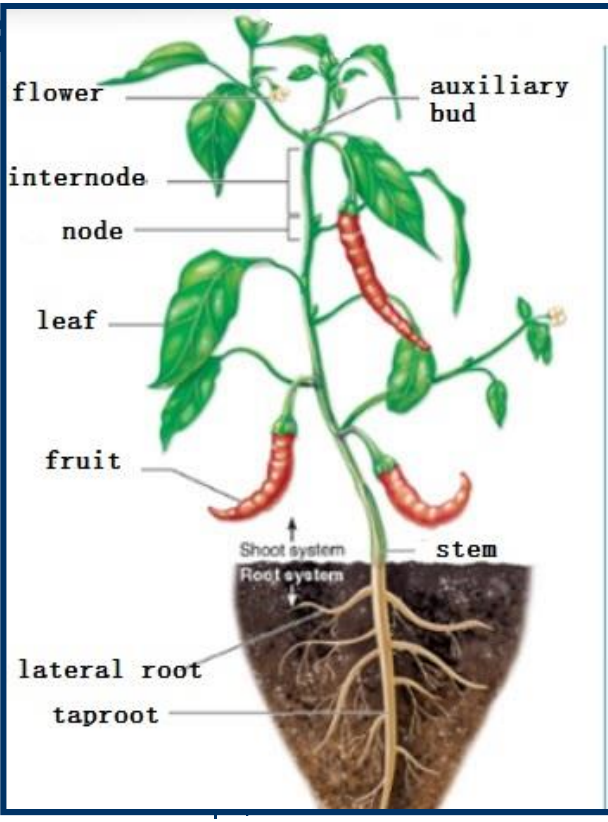
L-Systems – rewriting

- defining complex objects by successively replacing parts of a simple initial object using a set of *rewriting rules or productions*

Johan Knutzen

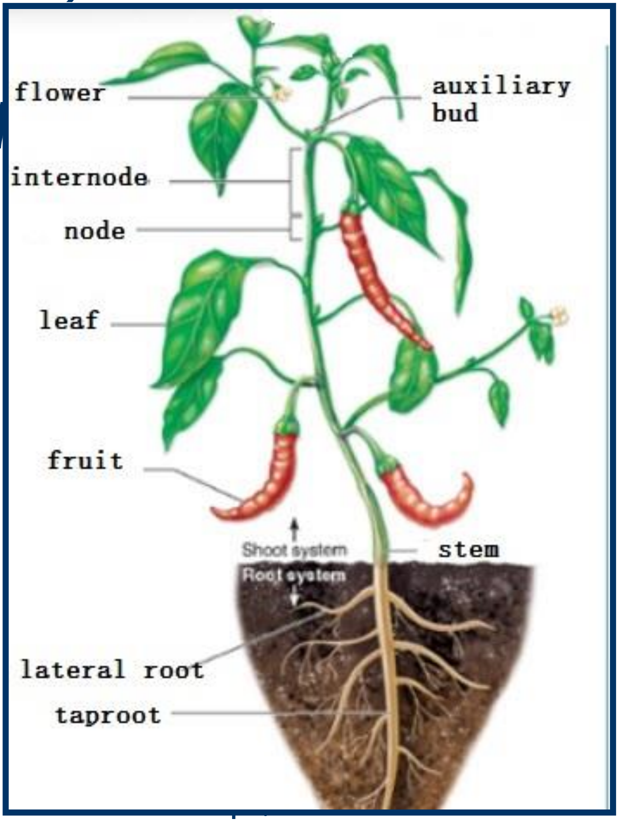
A Little Bit of Botany

- We are only interested in the visual characteristics of the plant
- Structural components of plants: ***stems, roots, buds, leaves, flowers***
- Plants with a definite branching structure: ***herbaceous, woody***
- Woody plants: ***larger, heavier branches, more structurally independent, branches tend to interfere and compete with one another, more subject to effects of wind, gravity and sunlight.***



A Little Bit of Botany

- Herbaceous plants: ***smaller, lighter, such as ferns and mosses, more regular branching patterns, less subject to environmental effects***
- Stems: ***above ground, grow upward, bear leaves. Leaves are attached at nodes. Portions between nodes are called internodes. Branching is the production of subordinate stems from a main stem. Branching can be formed in several ways (see slide 3)***
- Buds: ***embryonic state of stems, leaves, and flowers; classified as vegetative, and flower buds, or terminal bud, or lateral bud.***



A Little Bit of Botany

- Leaves: ***grow from buds. Arranged on a stem in three ways: alternate, opposite, whorled***
- Cell growth ***has four main influences***

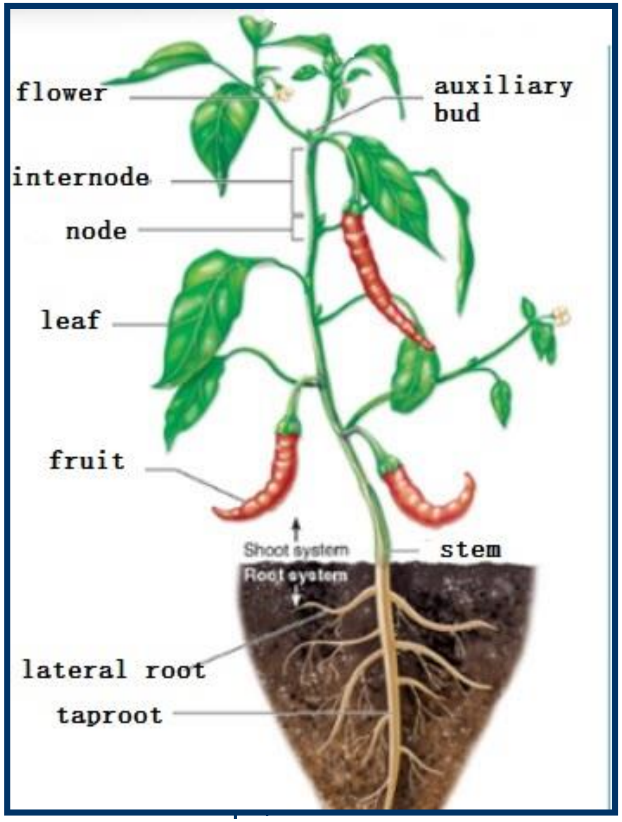
  lineage: growth controlled by the age of the cell

  cellular descent: passing of nutrients and hormones from adjacent cells

  tropisms: phototropism – bending of a stem toward light

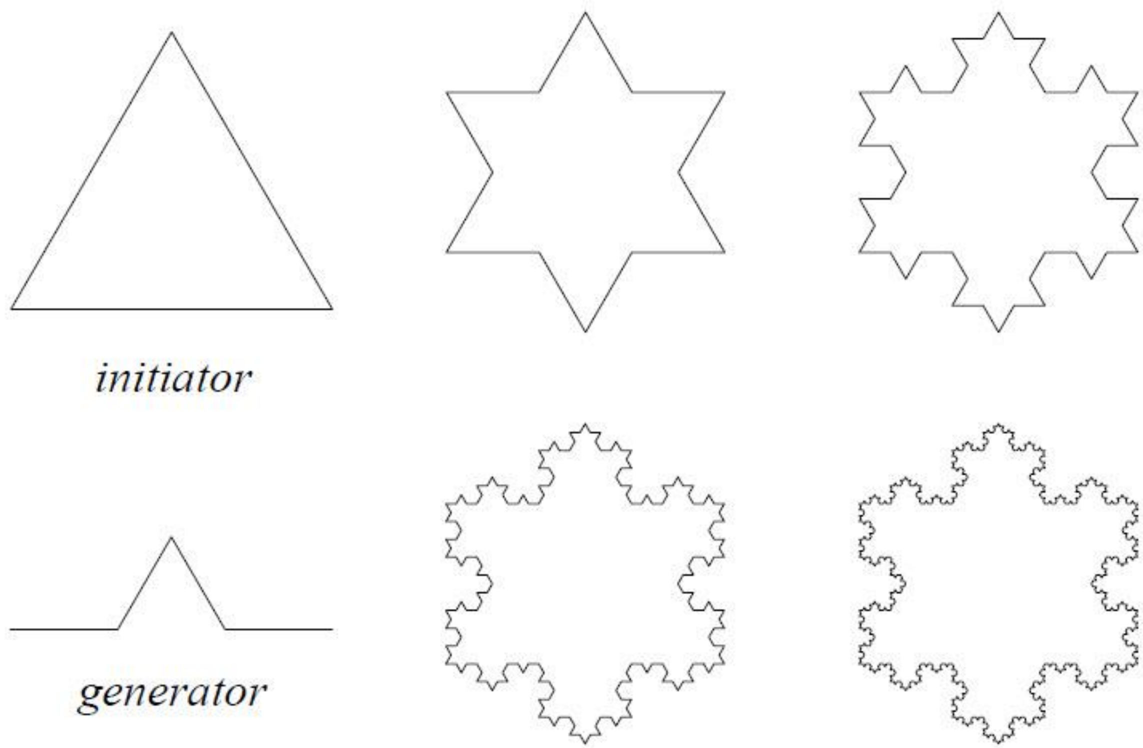
  geotropism – response of a stem or root to gravity

  obstacles: collision detection and response can be calculated for temporary changes in shape; per-manent changes can occur with longer obstacle existence



L-systems

- Central concept: ***rewriting***
- A classical example: ***Koch construction of snowflake curve***



L-systems – a brief history

- The most extensively studied & best understood rewriting systems operate on ***character strings***.
- The ***1st*** formal definition of such a system was given at the beginning of this century by Thue, but a wide interest in string rewriting was spawned in the late 1950s by ***Chomsky's*** work on ***formal grammars***. He applied the concept of rewriting to describe the syntactic features of natural languages.
- A few years later ***Backus/Naur*** introduced a rewriting-based notation in order to provide a formal definition of the programming language ***ALGOL-60***.

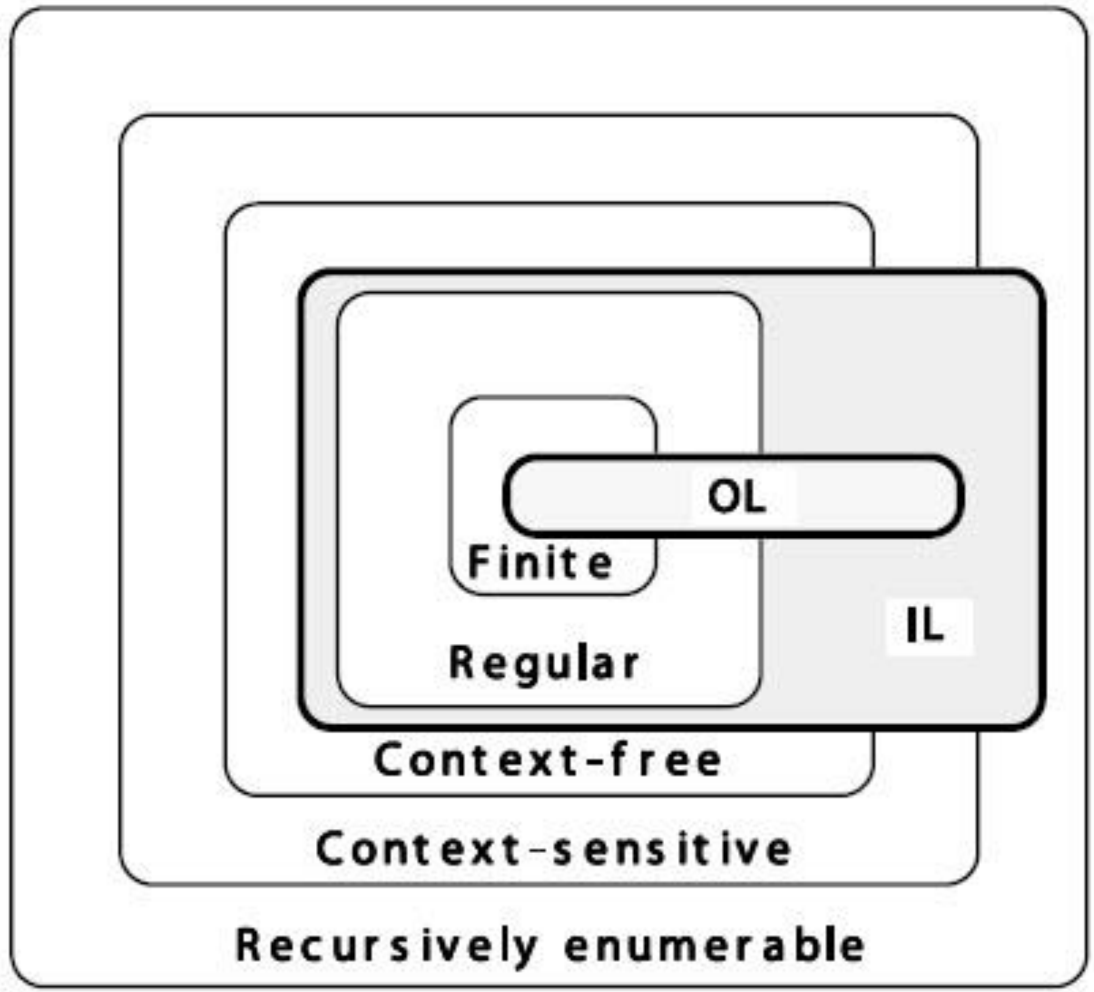
L-systems – a brief history

- The equivalence of the ***Backus-Naur form (BNF)*** and the ***context-free class*** of ***Chomsky grammars*** was soon recognized [***Ginsburg/Rice, 1962***], and a period of fascination with syntax, grammars and their application to computer science began.
- At the center of attention were sets of strings — called ***formal languages*** — and the methods for generating, recognizing and transforming them.
- In 1968 a biologist, ***A. Lindenmayer***, introduced a new type of string-rewriting mechanism, subsequently termed ***L-systems***.
- The difference between ***Chomsky grammars*** and ***L-systems*** lies in the method of ***applying productions.***

L-systems – a brief history

- In Chomsky grammars productions are applied ***sequentially***, whereas in L-systems they are applied ***in parallel*** and ***simultaneously replace all letters in a given word***.
- This difference reflects the biological motivation of L-systems. Productions are intended to capture cell divisions in multicellular organisms, where ***many divisions may occur at the same time***.
- ***Parallel production application*** has an essential impact on the formal properties of rewriting systems. For example, there are languages which can be generated by ***context-free L-systems*** (called ***OL-systems***) but not by ***context-free Chomsky grammars***.

L-systems – a brief history



OL: language classes generated by context-free L-systems

IL: language classes generated by context-sensitive L-systems

Relations between Chomsky classes of languages and language classes generated by L-systems.

L-systems : D0L-systems

- parallel rewriting systems
- D0L-system : ***deterministic &*** 0***-context*** (or, ***context-free***)***,*** simplest class of L-system
- a set of ***production rules*** of the form

$$\alpha_i \longrightarrow \beta_i$$

α_i : ***predecessor,*** a single symbol

β_i : ***successor***, a sequence of symbols

- ***Axiom***: a sequence of one or more symbols is given as the ***initial string***
- The production rules are ***iteratively applied (in parallel)*** until no occurrences of a lefthand side of a production rule occur in the string

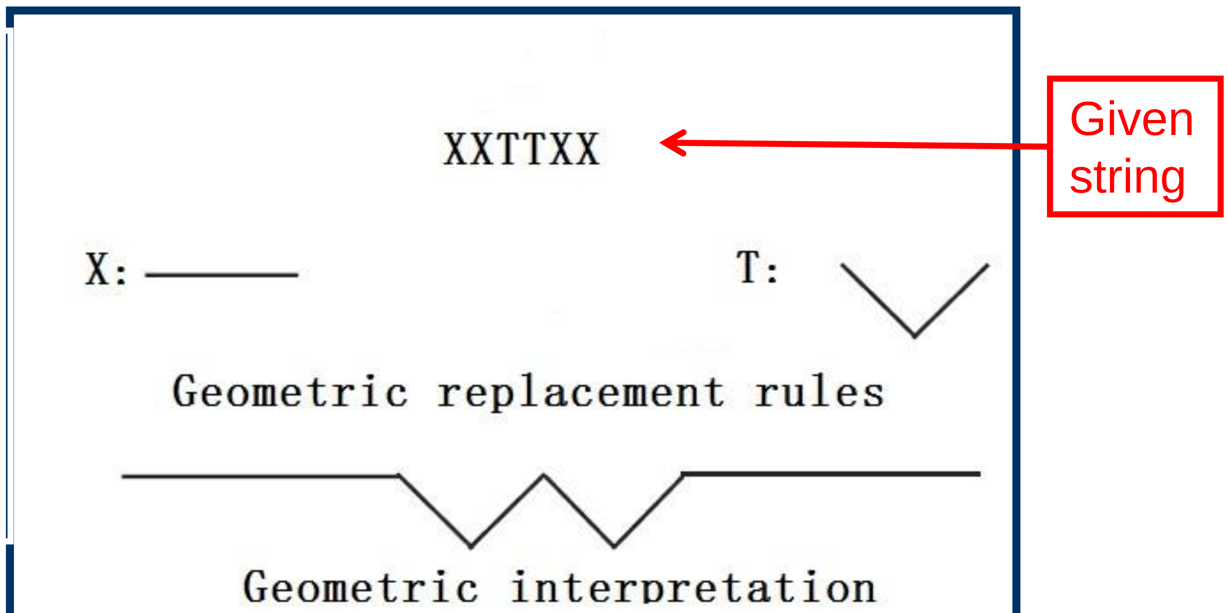
L-systems : D0L-systems

Production rules	String sequence
S -> ABA	S ← axiom
A -> XX	ABA
B -> TT	XXTTXX

Geometric Interpretation of L-Systems:

two ways: ***geometric replacement***, ***turtle graphics***

Geometric Replacement:
each symbol is replaced by a geometric element



L-systems : D0L-systems

Turtle graphics: interpret the symbols as drawing commands given to a simple cursor called a ***turtle***

Symbol	Turtle Graphic Interpretation
F	Move forward a distance d while drawing a line. Its state will change from (x, y, α) to $(x + d \cdot \cos\alpha, y + d \cdot \sin\alpha, \alpha)$.
f	Move forward a distance d without drawing a line. Its state will change as above.
+	Turn left by an angle δ. Its state will change from (x, y, α) to $(x, y, \alpha + \delta)$.
–	Turn right by an angle δ. Its state will change from (x, y, α) to $(x, y, \alpha - \delta)$.

$(\underline{x, y}, \underline{\alpha})$

Location of cursor

Direction of cursor relative to a given reference direction

L-systems : L-systems

Production rules	Sequence of strings produced from the axiom
S -> ABA	S ← axiom
A -> FF	ABA
B -> TT	FFTTFF
T -> -F++F-	FF-F++F- -F++F-FF

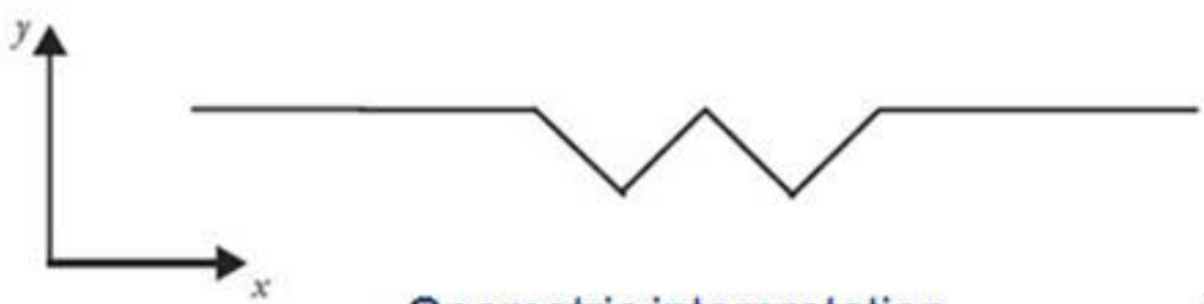
d =

$\delta = 45°$

reference direction : →

initial state : (10, 10, 0)

(initial conditions)



Geometric interpretation

L-systems : Bracketed L-systems

- In ***bracketed L-systems***, brackets are used to mark the beginning and the end of additional offshoots from the main lineage
- The turtle graphics interpretation of the brackets is given below

Symbol	Turtle Graphic Interpretation
[	Push the current state of the turtle onto the stack
]	Pop the top of the state stack and make it the current state

Production Rules:

$$S \Rightarrow FAF$$
$$A \Rightarrow [+FBF]$$
$$A \Rightarrow F$$
$$B \Rightarrow [-FBF]$$
$$B \Rightarrow F$$

Nondeterministic, context-free production rules

L-systems : Bracketed L-systems

Production Rules:

$$S \Rightarrow FAF$$
$$A \Rightarrow [+FBF]$$
$$A \Rightarrow F$$
$$B \Rightarrow [-FBF]$$
$$B \Rightarrow F$$

Nondeterministic, context-free production rules

- The production rules are ***context-free*** and ***nondeterministic***
- *S* is the start symbol, and *A* and *B* represent a location of possible branching
- *A* branches to the ***left*** and *B* to the ***right***
- The production stops when all symbols have changed into ones that have a ***turtle graphic interpretation***

L-systems : Bracketed L-systems

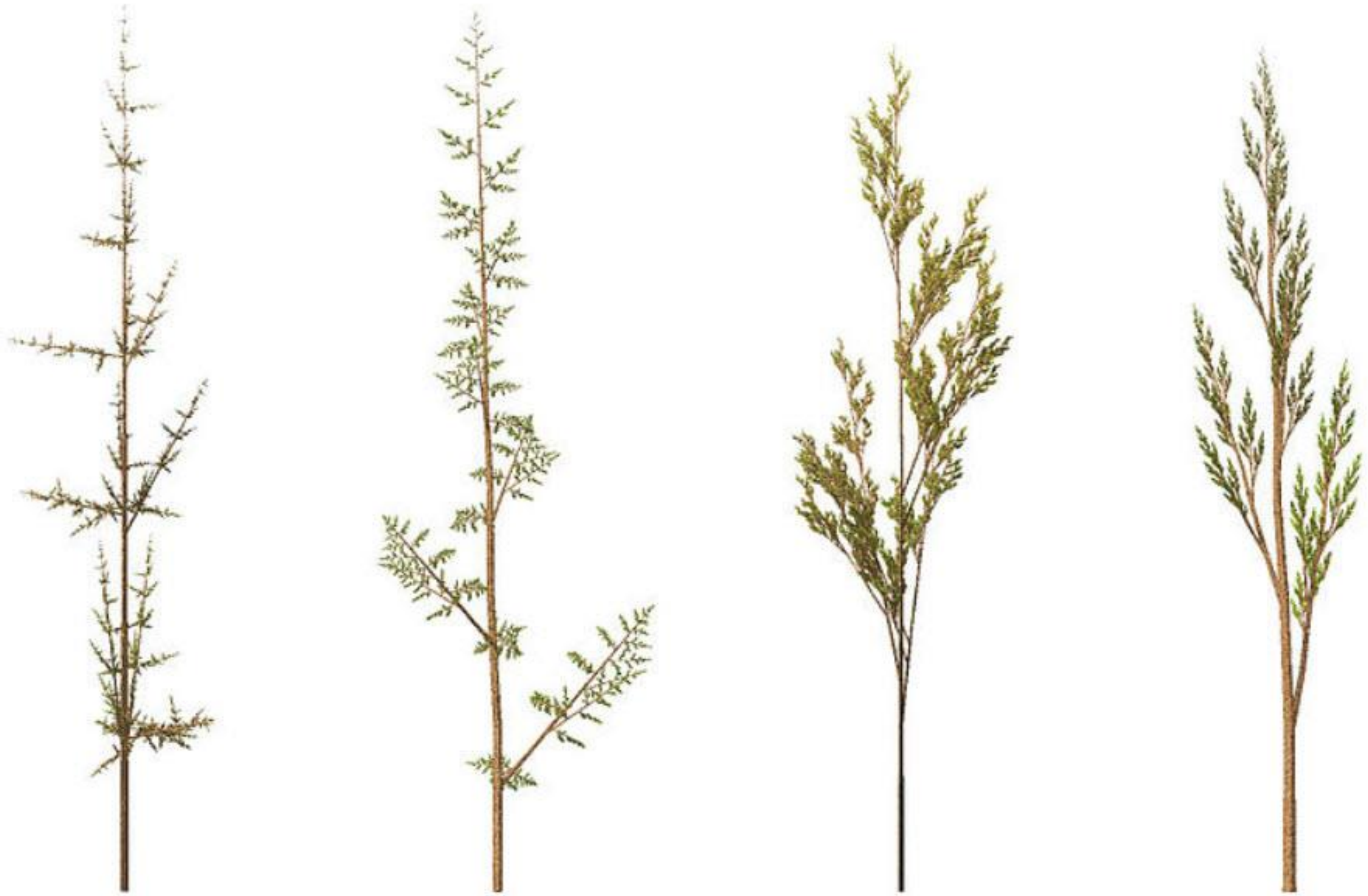
Production Rules:

$$S \Rightarrow FAF$$
$$A \Rightarrow [+FBF]$$
$$A \Rightarrow F$$
$$B \Rightarrow [-FBF]$$
$$B \Rightarrow F$$

Nondeterministic, context-free production rules

- Some possible ***terminal strings*** & the ***corresponding graphics*** produced by the turtle interpretation

L-systems : Bracketed L-systems



Weeds, generated using an L-system in 3D.

L-system notations:

L-systems are now commonly known as *parametric* L-systems, defined as a tuple

$$\mathbf{G} = (V, \omega, P),$$

where

- **V** (the *alphabet*): a set of symbols containing elements that can be replaced (*variables*)
- **ω** (*start*, *axiom* or *initiator*): a string of symbols from **V** defining the initial state of the system
- **P**: a set of *production rules* or *productions* defining the way variables can be replaced with combinations of constants and other variables. For any symbol A in V which does not appear on the left hand side of a production in P, the identity production A → A is assumed; these symbols are called *constants* or *terminals*.

L-system: revisit

- An L-system is *context-free* if each production rule refers only to an individual symbol and not to its neighbors.

depends only on a single symbol

- Context-free L-systems are thus specified by either a prefix grammar, or a regular grammar. If a rule depends not only on a single symbol but also on its neighbours, it is termed a *context-sensitive* L-system.

see next two slides

- If there is exactly one production for each symbol, then the L-system is said to be *deterministic*

- A deterministic context-free L-system is popularly called a *D0L system*.

Prefix grammar

$$G = (\Sigma, S, P)$$

$\Sigma:$ finite alphabet

$S:$ finite set of base strings over Σ

$P:$ set of production rules of the form $u \to v$

where u and v are strings over Σ

For strings *x*, *y*, we write $x \to_G y$ (and say: *G* can derive *y* from *x* in one step) if there are strings *u*, *v*, *w* such that $x = vu$, $y = wu$, and $v \to w$ is in *P*.

The *language* of *G*, denoted *L(G)*, is the set of strings derivable from *S* in zero or more steps

Prefix grammar: example

The prefix grammar

- $\Sigma = \{0, 1\}$
- $S = \{01, 10\}$
- $P = \{0 \to 010, 10 \to 100\}$

describes the language defined by the regular expression

$$01(01)^* \cup 100^*$$

E.g.,

$$01 \to 0101 \to 010101 \to 01010101$$

$$10 \to 100 \to 1000 \to 10000$$

L-system: example 1 – algae

Lindenmayer's original L-system for modelling the growth of algae.

variables : A, B
constants : none
start : A
rules : (A → AB), (B → A)

which produces:

$n = 0$: A
$n = 1$: AB
$n = 2$: ABA
$n = 3$: ABAAB
$n = 4$: ABAABABA
$n = 5$: ABAABABAABAAB
$n = 6$: ABAABABAABAABABAABABA
$n = 7$: ABAABABAABAABABAABABAABAABABAABAAB

L-system: example 1 – algae, explained

```
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
```

If we count the length of each string, we obtain the famous Fibonacci sequence of numbers:

1 2 3 5 8 13 21 34 55 89 ...

L-system: example 2

variables : 0, 1
constants: [,]
axiom : 0
rules : (1 → 11), (0 → 1[0]0)

The shape is built by recursively feeding the axiom through the production rules. Applying this to the axiom of '0', we get:

axiom: 0
1st recursion: 1[0]0
2nd recursion: 11[1[0]0]1[0]0
3rd recursion: 1111[11[1[0]0]1[0]0]11[1[0]0]1[0]0

L-system: example 2

Turtle graphics:

0 : draw a line segment ending in a leaf

1 : draw a line segment

[: push position and angle, turn left 45 degrees

] : pop position and angle, turn right 45 degrees

L-system: example 2

Turtle graphics:

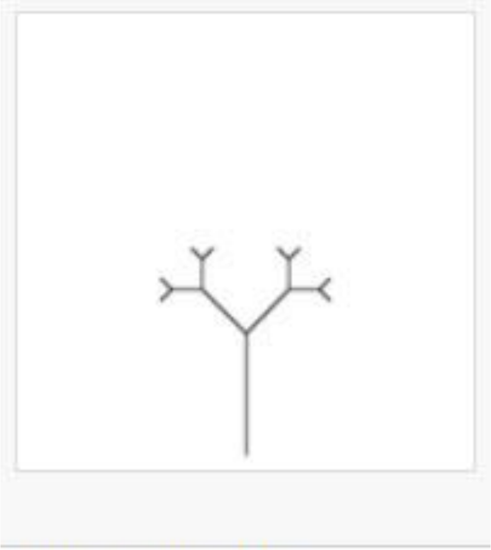
0 : draw a line segment ending in a leaf

1 : draw a line segment

[: push position and angle, turn left 45 degrees

] : pop position and angle, turn right 45 degrees

1111[11[1[0]0]1[0]0]11[1[0]0]1[0]0



Third recursion

L-system: example 3 – Koch Curve

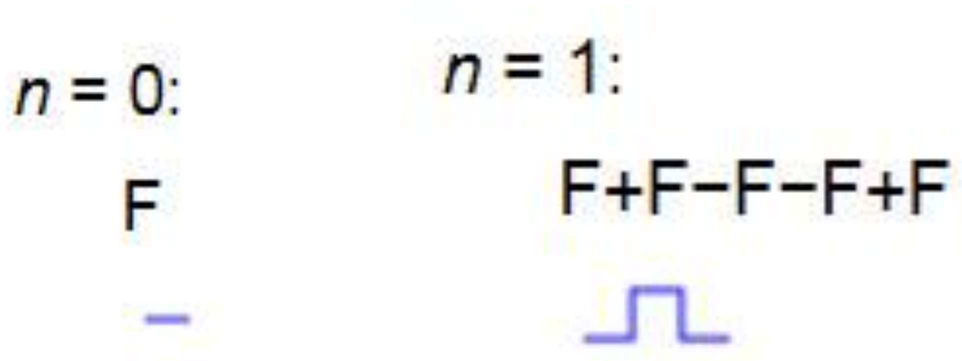
variables : F

constants : + −

start : F

rules : (F → F+F−F−F+F)

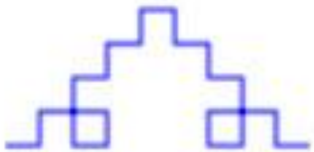
Here, F means "draw forward", + means "turn left 90°", and − means "turn right 90°".



L-system: example 3 – Koch Curve

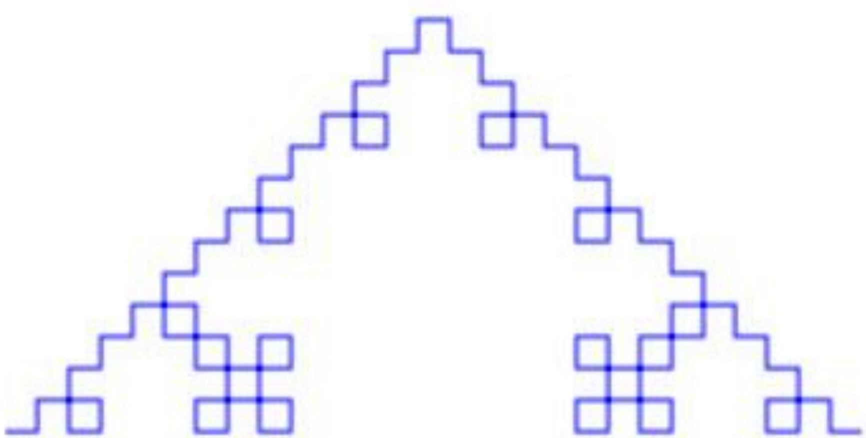
n = 2:

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



n = 3:

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



L-system: example 4 – Sierpinski triangle

variables : A, B

constants : +, −

start : A

rules : (A → B−A−B), (B → A+B+A)

angle : 60°

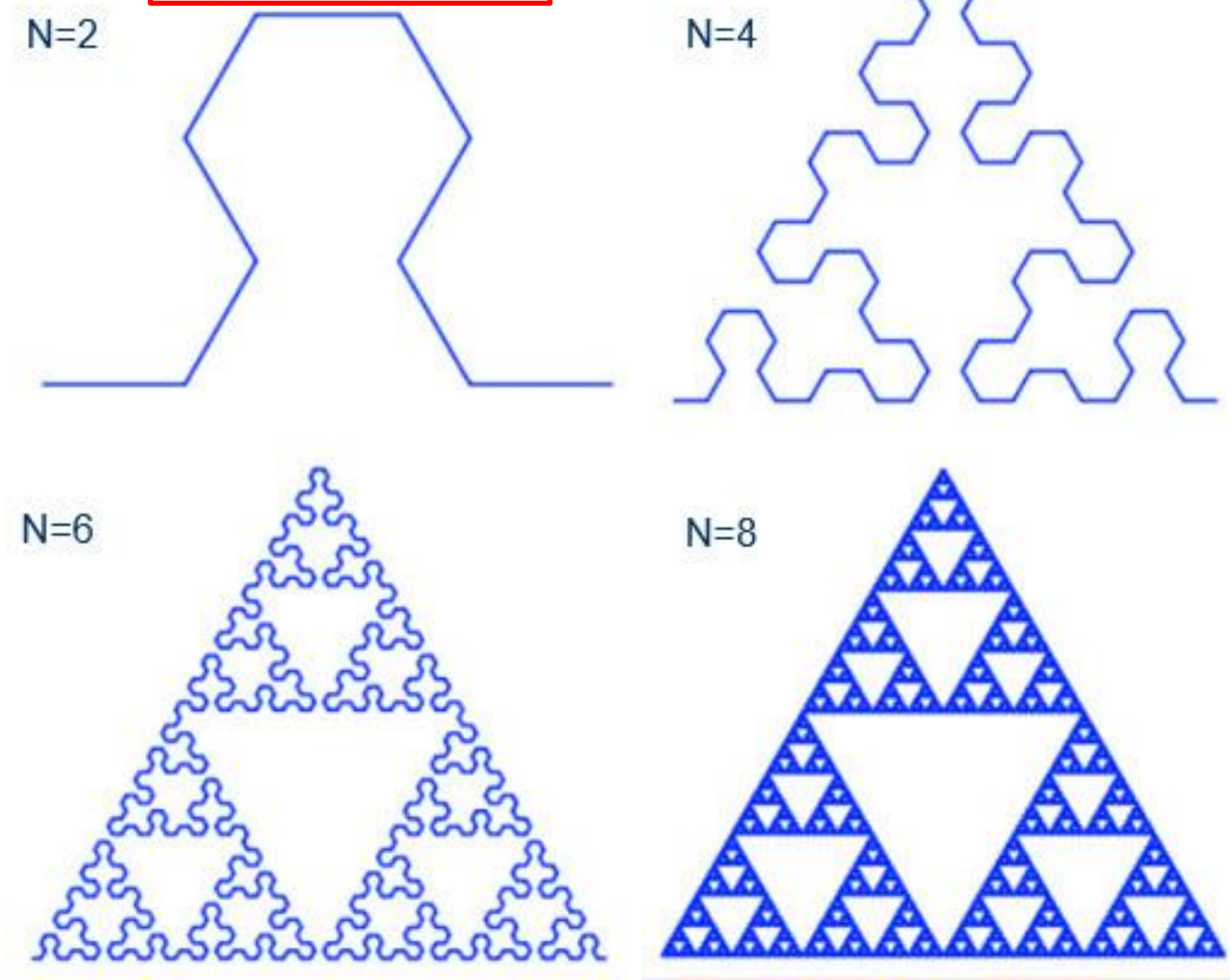
Here, A and B both mean "draw forward", + means "turn left by angle", and − means "turn right by angle" (see turtle graphics).

The angle changes sign at each iteration so that the base of the triangular shapes are always in the bottom (otherwise the bases would alternate between top and bottom).

L-system: example 4 – Sierpinski triangle

$A \rightarrow B - A - B$

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



L-system: example 5 – Fractal plant

variables : X F

constants : + − [ ]

start : X

rules : (X → F-[[X]+X]+F[+FX]-X), (F → FF)

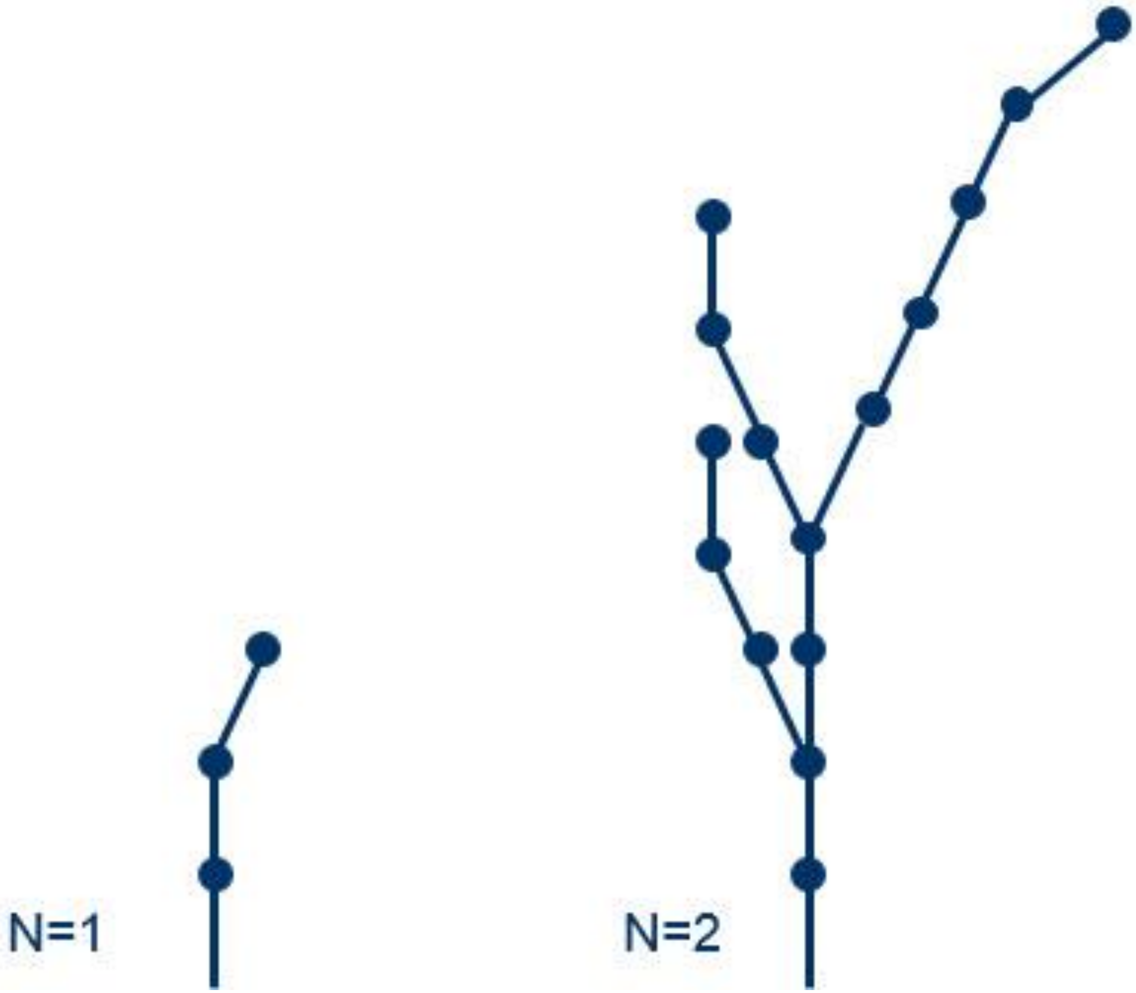
angle : 25°

Here, F means "draw forward", - means "turn left 25°", and + means "turn right 25°".

X does not correspond to any drawing action and is used to control the evolution of the curve.

[corresponds to saving the current values for position and angle, which are restored when the corresponding **]** is executed.

L-system: example 5 – Fractal plant



L-system: example 5 – Fractal plant

N=7

L-systems : Stochastic L-systems

- The previous section introduced ***nondeterminism*** into the concept of L-systems, but the method used to select the possible applicable productions for a given symbol was not addressed
- ***Stochastic L-systems*** assign a user-specified probability to each production. These probabilities indicate how likely it is that the production will be applied to the symbol on a symbol-by-symbol basis
- With stochastic (nondeterministic) L-systems, one can set up an L-system that produces a wide variety of branching structures that still exhibit some ***family-like similarity***

L-systems : Stochastic L-systems

Production Rules:

$$\begin{aligned}
\omega &: F \\
p_1 &: F \rightarrow F[+F]F[-F]F \\
p_2 &: F \rightarrow F[+F]F \\
p_3 &: F \rightarrow F[-F]F
\end{aligned}$$

Nondeterministic, context-free production rules

Production Rules with assigned probabilities:

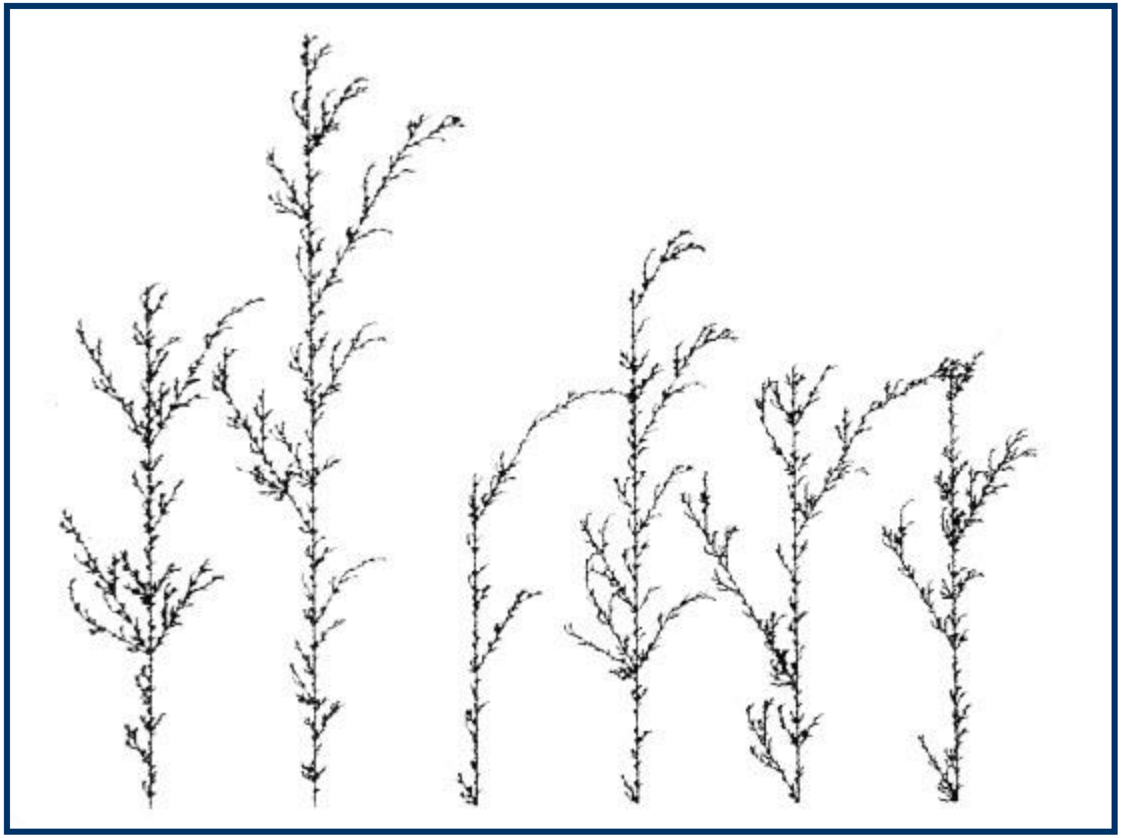
$$\begin{aligned}
\omega &: F \\
p_1 &: F \xrightarrow{.33} F[+F]F[-F]F \\
p_2 &: F \xrightarrow{.33} F[+F]F \\
p_3 &: F \xrightarrow{.34} F[-F]F
\end{aligned}$$

Stochastic L-system

When used in an evolutionary context, it is advisable to incorporate a random seed into the genotype, so that the stochastic properties of the image remain constant between generations.

L-systems : Stochastic L-systems

Examples of branching structures generated by this L-system with derivations of length 5 are shown below. Note that these structures look like different specimens of the same (albeit fictitious) plant species.



Example was copied from: “The Algorithmic Beauty of Plants” by P. Prusinkiewicz and A. Lindenmayer

L-systems : Stochastic L-systems

A more complex example is shown below. The field consists of four rows and four columns of plants. All plants are generated by a stochastic modification of the L-system used to generate the figure shown on the next page.

Example was copied from: “The Algorithmic Beauty of Plants” by P. Prusinkiewicz and A. Lindenmayer

L-systems : Stochastic L-systems

n=5, *δ=18°*

*ω : **plant***

*p1 : **plant** → **internode + [plant + flower] − − //***
*[− − **leaf] internode [+ + leaf] −***
[**plant flower] + + plant flower**

*p2 : **internode** →*
F seg [// & & leaf] [// ∧ ∧ leaf] F seg

*p3 : **seg** → **seg F seg***

*p4 : **leaf** →*
[' { +f−ff−f+ | +f−ff−f }]

*p5 : **flower** →*
[& & & pedicel ' / wedge //// wedge ////
wedge ///// wedge ///// wedge]

*p6 : **pedicel** → **FF***

*p7 : **wedge** →*
[' ∧ F] [{ & & & & −f+f | −f+f }]

L-systems : Context-sensitive L-systems

- add the ability ***to specify a context***, in which the left-hand side (the predecessor symbol) must appear in order for the production rule to be applicable
- e.g.

Production rules	String sequence
$S \Rightarrow FAT$	
$A > T \Rightarrow [+FBF]$	S
$A > F \Rightarrow F$	FAT
$B \Rightarrow [-FAF]$	F[+FBF]F
$T \Rightarrow F$	F[+F[–FAF]F]F

- Can be extended to *n* left-side context symbols and *m* right-side context symbols in the productions, *called* ***(n, m)L-systems***

L-systems : Context-sensitive L-systems

- e.g.

Production rules	String sequence
$S \Rightarrow FAT$	
$A > T \Rightarrow [+FBF]$	S
$A > F \Rightarrow F$	FAT
$B \Rightarrow [-FAF]$	F[+FBF]F
$T \Rightarrow F$	F[+F[-FAF]F]F

- Compatible with nondeterministic L-systems
- In ***(n, m)L-systems***, productions with fewer than *n* context symbols on the left and *m* on the right are allowable
- Productions with ***shorter contexts*** are usually given ***precedence*** over productions with ***longer contexts*** when they are both applicable to the same symbol

L-systems : animating plant growth

- three types of animation in plants:
  - ***flexible movement of an otherwise static structure***
  - ***changes in topology that occur during growth***
  - ***elongation of existing structures***

- ***Topological changes*** (captured by the L-systems already described) occur as ***discrete events in time*** and are modeled by the application of a ***production that encapsulates a branching structure***, as in $A \Rightarrow F\ [+F\ ]B$
- ***Elongation*** can be modeled by ***productions of the form $F \Rightarrow FF$***

L-systems : animating plant growth

- ***Elongation*** can be modeled by ***productions of the form F ⇒ FF***
- Problem: growth is chunked into units equal to the length of the drawing primitive represented by *F*. If *F* represents the smallest unit of growth, then an ***internode segment*** can be made to grow arbitrarily long. But ***the production rule F ⇒ FF lacks termination criteria for the growth process***
- ***Additional drawing symbols*** can be introduced to represent successive steps in the elongation process, resulting in a series of productions ***F0 ⇒ F1, F1 ⇒ F2, F2 ⇒ F3, F3 ⇒ F4,*** and so on. Each symbol would represent a ***drawing operation of a different length***

L-systems : parametric L-systems

- Providing a solution to the ***proliferation of symbols and productions*** in the elongation process if the time steps is large
- symbols can have one or more ***parameters*** associated with them
- parameters can be set and modified by ***productions of the L-system***
- optional ***conditional terms (***in terms of ***parametric values***) can be associated with productions
- production is applicable only if its ***associated condition is met***

L-systems : parametric L-systems

$$
\begin{array}{lll}
S & \Rightarrow & A(0) \\
A(t) & \Rightarrow & A(t+0.01) \\
A(t) : t>=1.0 & \Rightarrow & F
\end{array}
$$

Simple parametric L-system

- ***Context-sensitive productions*** can be combined with ***parametric systems*** to model the passing of information along a system of symbols

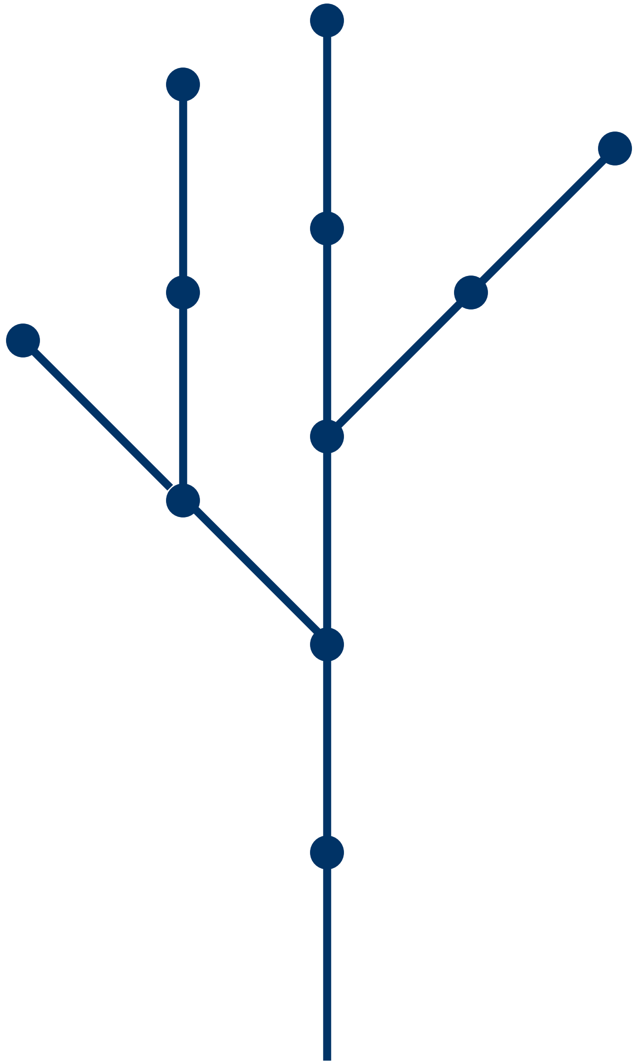
$$A(t0)<A(t1)>A(t2):\ t2>t1\ \&\ t1>t0 \Rightarrow A(t1+0.01)$$

- single ***context symbol*** on both sides of the left-hand symbol that is to be changed. These productions allow for the relatively easy representation of such processes as ***passing nutrients*** along the stem of a plant

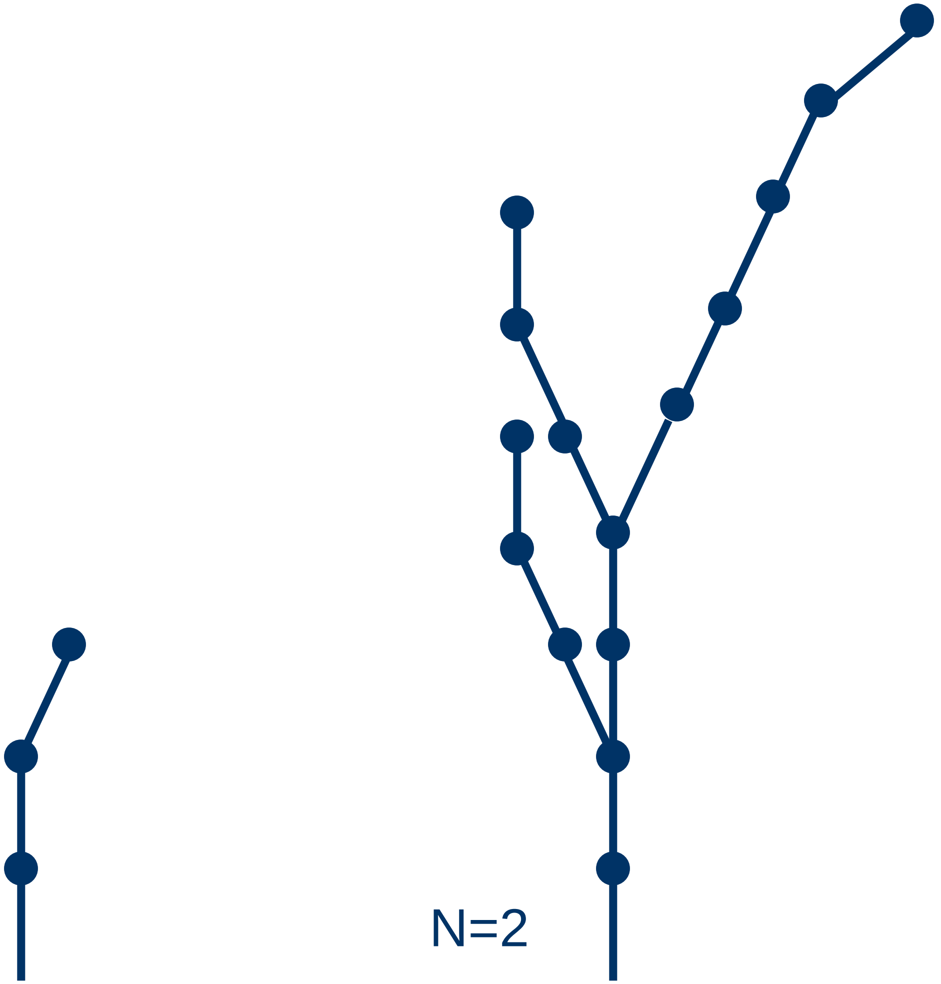
End of Special Models for Animation II

L-systems : timed L-systems

- Add two more concepts to L-systems:
  - global time variable
  - local age value



For the system shown on slide 34:



N=1

N=2