



3d Lindenmayer Systems



since 2. Oct. 2003

Accesses

last Update:
7. Jan. 2006

[Introduction](#) / [File Syntax Notation](#) / [Download](#) / [Examples](#) / [Stochastic L-Systems](#) / [Notation](#) / [Other Applications](#) / [Links](#) / [Books](#)

Introduction

The Lindenmayer System language also known as L-Systems was defined by Aristid Lindenmayer in the book 'The Algorithmic Beauty of Plants, Springer Verlag'.
Text on the back cover: "This book is for * Computer graphics professionals * Biologists, * Theoretical computer scientists and mathematicians who will discover that the mathematical beautiful theory of L-Systems has important practical applications * Individuals interested in fractals, L-Systems and Artificial Life".



Aristid Lindenmayer
(1925-1989)

LSystem File Syntax Notation:

To load and save such definitions from a text file, I use a notation which is an extension to the Virtual Reality Modeling Language (see Extensibility of [VRML 1.0 Specification](#) or [VRML 1.0c Specification](#)).

You can see the syntax extension on the right side.

After the property name definition included by '['...' the default values are defined.

name should have no blanks included.

info is limited to 80 characters.

angle defines how many degrees the 3d turtle should turn.

step defines the step length you go on commands like F, f, G or g.

segments defines the number of cylinder planes which will be draw instead of lines, if the thickness is greater than 1.0.

thickness defines the diameter of the cylinder for one turtle step.

factor defines a multiplication value per deep step for resizing the whole drawing figure.

deep defines the number of recursions until the replacement of the rule characters stops.

colors defines an optional color array.

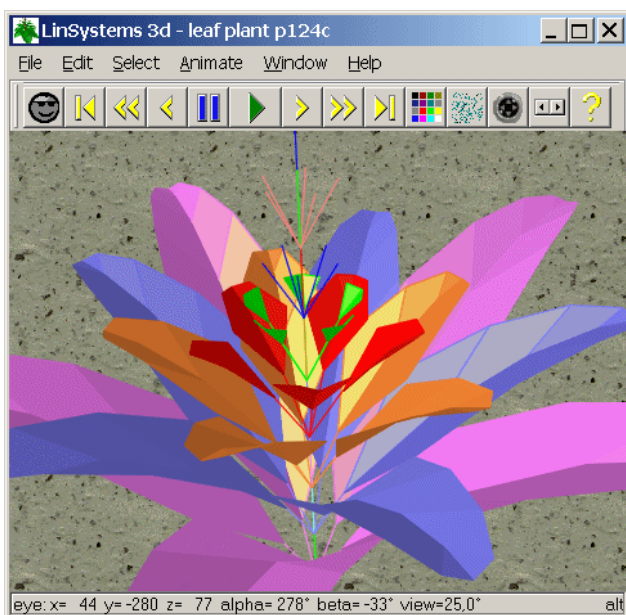
The **axiom** defines the starting values for the LSystem interpreter.

The **rules** defines the replacement statements.

LSystem Definition Syntax and default values:

```
LSystem
{
  fields
  [
    SFFloat name,          SFString info,
    SFFloat angle,         SFLong  segments,
    SFFloat step,          SFFloat thickness,
    SFFloat factor,        SFColor  colors,
    SFLong  deep,          MFString rules
    SFString axiom,
  ]
  name "unknown"          info ""
  angle 30                 segments 6
  step 100.0               thickness 1.0
  factor 0.8
  deep 4                   colors [ ]
  axiom A
  rules [A=AF+]
}
```

Download:



LinSy3d.exe

If you want to create your own Lindenmayer Systems figures or if you only want to play around with the predefined LSystem definitions then **download** my new

Lindenmayer System Interpreter

LinSy3d.exe v1.03



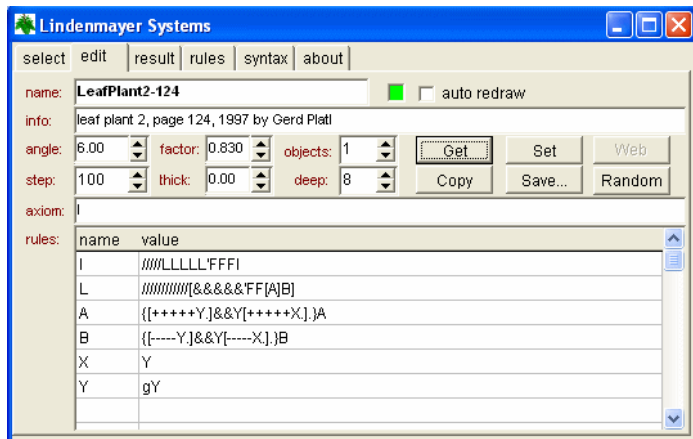
The visualization uses the 3d-Engine **FliSp3d**. Therefore many parameters can be changed interactively, e.g. the background picture or the transparency of lines, planes and sprites.
Get more information about [FliSp](#) at this [link](#).

Note:

Now you can **save** a lindenmayer system as text file and you can **load LSystem files** which may contain a collection of many LSystem definitions.

Statements '(' and ')' for defining values are still not implemented at the moment.

Sorry - but ... I am working on this features.



System selection

To select a predefined LSystem figur press right mouse button on main screen and choose a figure with the menu command 'Select figure / ...'.

How to edit LSystems with LinSy3d

The **dialog for editing LSystems** will be automatically opened at program start.

You can open this dialog also by the menu command 'Edit / Lindenmayer Systems... Ctrl-F8'

Select Tab **Edit** and press button **Get** to get all parameters of the actual displayed LSystem.

If you want see the effect of changed parameters, press the **Set** button.

If you want to see changes immediately, turn on **link drawing**. You can easily copy the whole parameters to e.g. a text file with the Lindenmayer text file (*.lsy) syntax if you press the **Copy** button and paste the copied text there.

The **Save...** button will start the file saving dialog to save the LSystem as a *.lsy text file with a single LSystem definition.

Loading and Saving LSystem collection files


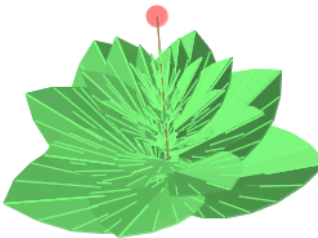

On Tab **Select** you can load and save LSystem text files, which can have one or more LSystem definitions pressing **Load File...** or **Save File ...** button.

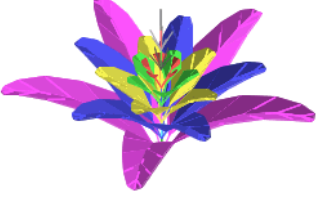
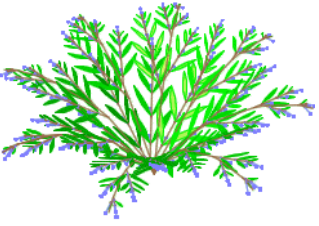


Displayed files can be loaded again easily by pressing button **Load Again** e.g. after file changes or corrections done with a text editor.

After selecting a LSystem item of a predefined or a loaded LSystem collection, you can view it by pressing button **View** or with a double click to it. Pressing button **Edit** will copy all parameters to the edit fields and change to tab Edit for further changes.


Examples:

The following table show youn only some few examples. Many examples are predefined within the LinSys3d application. But the most LSystem definitions you can load from the included Lindenmayer definition files.

	<pre>LSystem { name "GrowingLeaf2Bush-120" info "bush of branches with growing leafs" angle 22.5 factor 0.74 deep 8 axiom A rules ["A=//BY//BYA" "B=[&Y[+L][-L]YBF]" "L=FF' {+X.-X.-X.+ +X.-X.}" "X=gX" "Y=FY"] }</pre>
	<pre>LSystem { name "cordate leaf plant 123" info "plant with cordate leafs of page 123" angle 6.0 factor 0.8 deep 14 axiom "R9'~S" rules ["R:?LLLFR" "L:!15/[EA]" "E:&E" "A:'F[B][C]" "B:/[++B[.]D.}" "C:[--C[.]D.}" "D:gD" "X:FX"] }</pre>
	<pre>LSystem { name "LeafPlant1-124" angle 11 thickness 4 factor 0.8 deep 9 axiom "R4'~S" rules ["R=LLLL!R" "L=////////['&&&&&[A]B]F" "A={ [++++Y.]&Y[++++X.]. }A" "B={ [----Y.]&Y[----X.]. }B" "X=Y" "Y=gY"] }</pre>

	<pre> LSystem { name "LeafPlant2-124" angle 6 factor 0.83 deep 8 axiom "I" rules ["I=/////LLLLL'FFFI" "L=/////////[&&&&'FF[A]B]" "A={ [+++++Y.]&&Y[+++++X.]. }A" "B={ [-----Y.]&&Y[-----X.]. }B" "X=Y" "Y=gY"] } </pre>
	<pre> LSystem { name "BlueBerryBush1" info "Blue berry bush 1, 2003 Gerd Plat1" link "http://www.geocities.com/gplat1/LSystem/LSystem.html" angle 11.25 factor 0.74 deep 10 axiom "9a" rules ["a=4/[^1]" "l=[X^[++B]r]" "r=4X^[--B]l" "B=[b[3+L][3-L]XB]" "L=F'[+Y.-Y.-Y.+ +Y.-Y.]" "X=FX" "Y=gY" "z=4/[^B]" "b=X8'[^4F-S]8`"] } </pre>
	<pre> LSystem { name "Palm1" angle 2 factor 0.88 deep 8 axiom "a'12b" rules ["a=a&~xf" "b=15\[16&cA]" "c=c2&" "A=F[12+B]8&[A][12-B]" "B=F8&B"] } </pre>
	<pre> LSystem { name "Pine3d" angle 5 factor 0.8 axiom "ab'~o" rules ["a=FF" "b=?ci6/b" "c=cF" "e=&&e" "i=?6j" "j=12/[8&eC]" "C=!cF[9+'D!F`]^^[9-'D!F`]C" "D=!F[9+F]^^[9-F]D"] } </pre>

Stochastic L-Systems:

	<p>Normally all L-systems plants generated by the same L-System are identical, while in reality there is no plant in the world growing in the same way.</p> <p>Simulating natural process by using probability is a much simpler approach than underlying natural rules. In my new L-System-Interpreter 1.3 you have the possibility to create stochastic L-Systems. You can define more than one rule for only one symbol. If there are more than one rule the LS-Interpreter will choose one of them randomly.</p> <p>If you generate many virtual plants in a garden, all plants will have different appearance though they are based on the same stochastic L-system definition. As you can see in the example it is a great approximation to nature scene.</p> <pre> LSystem { name "StochasticPlant4" info "Stochastic plant 4 with fruit seeds, 12/2005 Gerd Plat1" step 300 thickness 0 factor 0.53 deep 6 axiom F } </pre>
---	---

```

rules
[ "F=F[+F]F[-F]"
  "F=/F[+FJ]F"
  "F=\F[-FH]F"
  "H=[ '-G[GB]-G[GB]-G[GB]-B]"
  "J=[ ' +G[GB]+G[GB]+G[GB]+GB]"
  "B=[ G4' ~s]"
]

```

Production String Notation:

Basic Commands:

All rules must be defined by using the following commands:

character	default	statement	Befehl (german)
F	step	draw forward	zeichne vorwärts
f	step	jump forward	springe vorwärts
+	angle	turn left	nach links drehen
-	angle	turn right	nach rechts drehen
&	angle	pitch down	nach unten drehen
^	angle	pitch up	nach oben drehen
\	angle	roll left (anticlockwise)	nach links neigen (gegen Uhrzeigersinn)
/	angle	roll right (clockwise)	nach rechts neigen (im Uhrzeigersinn)
 	180°	turn around 180°	um 180° umdrehen
_	180°	roll 180°	um 180° verdrehen (Kopfstand)
G	step	draw forward	zeichne vorwärts in der Fläche ?
g	step	jump forward	springe vorwärts
{		begin plane	neue Fläche
}		end plane	Fläche beenden
.		add plane point	Flächenpunkt hinzufügen
[start branch	beginne neuen Zweig
]		complete branch	beende Zweig
'	+1	increment color index	Farbindex erhöhen
!	-1	decrement diameter	Durchmesser verringern
?	0.8	multiply diameter	Durchmesser multiplizieren
#		start line comment	Beginn Zeilenkommentar
*		jump to origin	zum Koordinatenursprung springen
~		add shape (see predefined shapes)	Figur hinzufügen (siehe Figurentabelle)
>		increment (see increment / decrement)	erhöhen
<		decrement (see increment / decrement)	ernidrigen
\$		extended command (see extended commands)	Zusatzkommando
=		assignment seperator	Zuweisungstrennzeichen
<number>	1	repetition / multiplication factor example '6+' means '++++++'	Wiederholungs- bzw. Multiplikationsfaktor Beispiel: '5?' bedeutet '????'

Predefined Shapes:

characters	default	statement	Befehl
~		add shape	Figur hinzufügen
~c	s = step / 2	add small cube	kleinen Würfel hinzufügen
~C	s = step	add cube	Würfel hinzufügen
~d	d = step / 2	add small disc	Scheibe hinzufügen
~D	d = step	add disc	Scheibe hinzufügen
~e	d = step, h=step/2	add flat ellipsoid	flaches Ellipsoid hinzufügen

~E	d = step/2, h=step	add long ellipsoid	längliches Ellipsoid hinzufügen
~I	d = step/2	add small flower	Blüte hinzufügen
~L	d = step	add flower	Blüte hinzufügen
~o	d = step	add small cone	kleinen Kegel hinzufügen
~O	d = step	add cone	Kegel hinzufügen
~s	d = step / 2	add small sphere	kleine Kugel hinzufügen
~S	d = step	add sphere	Kugel hinzufügen
~x	d = step / 2, h=step	add small cone stump	kleinen Kegelsumpf hinzufügen
~X	d = step, h=step	add cone stump	Kegelsumpf hinzufügen
~y	d = step / 2, h=step/2	add small cylinder	kleinen Zylinder hinzufügen
~Y	d = step	add cylinder	großen Zylinder hinzufügen
~1	d = step	add sprite 1	Sprite Nr.1 hinzufügen
~2	d = step	add sprite 2	Sprite Nr.2 hinzufügen
~3	d = step	add sprite 3	Sprite Nr.3 hinzufügen
~4	d = step	add sprite 4	Billboard-Sprite Nr.4 hinzufügen

Increment / Decrement Commands:

decrement	increment	statement	Befehl
<	>	decrement ... / increment ...	erniedrigen ... / erhöhen...
<a	>a	angle - 1° / angle + 1°	Winkel - 1° / Winkel + 1°
<A	>A	angle * 0.9 / angle * 1.1	Winkel * 0.9 / Winkel * 1.1
<s	>s	step - 1 / step + 1	Schrittweite - 1 / Schrittweite +1
<S	>S	step * 0.9 / step * 1.1	Schrittweite * 0.9 / Schrittweite * 1.1
<t	>t	thickness - 1 / thickness + 1	Dicke - 1 / Dicke + 1
<T	>T	thickness * 0.9 / thickness * 1.1	Dicke * 0.9 / Dicke * 1.1
<c	>c	color index - 1 / color index + 1	Farbindex - 1 / Farbindex + 1
<r	>r	tropism - 1 / tropism + 1	Anziehung - 1 / Anziehung + 1

Extented Commands

characters	default	statement	Befehl
\$F	step / 2	draw forward half step	zeichne halben Schritt nach vor
\$f	step / 2	jump forward half step	halben Schritt nach vor springen
\$G	step / 2	draw forward half step without planepoint	zeichne halben Schritt nach vor ohne einen Flächenpunkt hinzuzufügen
\$g	step / 2	jump forward half step without planepoint	halben Schritt nach vor springen ohne einen Flächenpunkt hinzuzufügen
\$U	thick / 2	jump up thickness radius	um Dickenradius nach unten springen
\$D	thick / 2	jump down thickness radius	um Dickenradius nach oben springen
\$R	thick / 2	jump right thickness radius	um Dickenradius nach rechts springen
\$L	thick / 2	jump left thickness radius	um Dickenradius nach links springen
\$ 	180°	turn around 180° horizontal	um 180° umdrehen
\$_	180°	roll 180°	um 180° verdrehen (Kopfstand)
\$&	v=-90°	pitch down to vertical	vertikal nach unten drehen
\$h	h=0°	roll to horizontal	horizontal ausrichten
\$*		jump to origin	zum Ausgangspunkt zurückspringen

Other Applications:

name	version date	OS / source requirements	author link
BioSchematics	0.4.3 5/2004	Linux / C OpenGL / Guile 1.6	3d L-System interpreter by Xavier Raynaud http://pagesperso.laposte.net/bioschematics/
Lindenmayer Fraktale	2.2 11/2003	Windows / (Delphi4) ---	2d LSystem-Application by Ulrich Schwebinghaus http://www.fraktalwelt.de/myhome/lmayer.htm
		Web-Browser / Java JRE	interactive 2d LSystem-Applet by Ulrich Schwebinghaus http://www.fraktalwelt.de/myhome/lmayer2-g.htm#applet
LParser2	1.0 1/2005	Windows (DOS) / --- Browser VRML-Plugin	3d LSystems Generator for WRL files by Laurens Lapré http://home.wanadoo.nl/laurens.lapre/lparser5.zip
LSYS32	6/1998	Windows / ---	graphical shell for for Laurens Lapre's Lindenmayer system parser http://chris.lichti.org/Lab/LSys32/LSys32.shtml
LSE L-System Explorer	1.1 9/2002	Windows / C++ ---	2d LSystem-Application by James Matthews http://www.generation5.org/content/2002/lse.asp
L-Systems Plant Geometry Generator	?	Macintosh / ? ?	Bryan Horling http://shakti.trincoll.edu/~bhorling/lsvsystems/
L-Systems Applet	1.0 ?	Web-Browser / Java JRE	interactive 2d LSystem-Applet by Robert Jessop http://home.clara.net/niknak/fractal/
L-System 4	4.01 7/2000	Windows / --- ---	3d LSystem Application by Timothy C. Perz http://www.geocities.com/tperz/L4Home.htm
Virtual Plants	1.0 12/1999	Web-Browser / Java JRE 1.2, Java3d	3d LSystem Application by René Gressly http://www.hta-bi.bfh.ch/~swc/DemoJ3D/VirtualPlants/

Links:**2d:**

[An Introduction to Lindenmayer Systems](#)
[Fractal curves](#)
[Fractint L-Systems](#) Tutorial by [William McWorter](#)
[Ulli's Fractal Home - LSystems](#)

3d:

[Algorithmic Botany at University of Calgary](#)
[Virtual Plants at University Hamburg](#)
[greenworks organic software - XFrog](#)
[Lparser: Laurens Lapré's Projects](#)
[L-System 4 by Timothy C. Perz](#)
[How to build LS files Tutorial](#)

Books:

Book: The Algorithmic Beauty of Plants
 by Przemyslaw Prusinkiewicz and Aristid Lindenmayer
 Springer-Verlag in 1990 (ISBN 0-387-97297-8)

You can download the entire book as a PDF file with [high quality](#) (17MB) or [low quality](#) (4 Mb).