Experimenting with Grammars to Generate L-Systems
November 30, 2009

Prof. Susan Rodger
Computer Science Dept.
L-Systems

- Model biological systems and create fractals
- Similar to Chomsky grammars, except all variables are replaced in each step, not just one!
- Successive strings are interpreted as strings of render commands and displayed graphically
English Grammar

- `<sentence> → <subject> <verb> <direct obj>`
- `<subject> → <noun> | <article> <noun>`
- `<verb> → hit | ran | ate`
- `<direct obj> → <article><noun> | <noun>`
- `<noun> → Fritz | ball`
- `<article> → the | an | a`

- Variables (shown in `< >`) are replaced by right side of arrow
Example: Derive a sentence

- `<sentence>→ <subject> <verb> <direct obj>`
  - `<noun> <verb> <direct obj>`
  - Fritz `<verb> <direct obj>`
  - Fritz hit `<direct obj>`
  - Fritz hit the `<noun>`
  - Fritz hit the ball
Parts of an L-System
(a type of grammar)

• Defined over an alphabet

• Three parts
  – Axiom (starting place)
  – Replacement rules (replaces all variables at once)
  – Geometric rules (for drawing)
    • g  means move forward one unit with pen down
    • f  means move forward one unit with pen up
    • +  means turn right by the default angle
    • -  means turn left by the default angle
Example – lsys-samp1

• Axiom

• Replacement Rules

• Geometric Rules

NOTE: Must use spaces as separator between symbols
Example – lsys-samp1 (cont)

• Derivation of strings

\[
\begin{align*}
X & \rightarrow gggX+Y \\
ggggggX + Y + g & \\
gggggggggX+Y+g+g & \\
ggggggggggggX+Y+g+g+g & \\
\end{align*}
\]

Note: replace both X and Y each time
More Geometric rules

• % change direction 180 degrees
• ~ decrement the width of the next lines
• [ save in stack current state info
• ] recover from stack state info
• { start filled in polygon
• } end filled in polygon
Example – lsys-samp2
Example – lsys-samp2 (cont)

\[ g[\sim + Y g] g X \]

\[ g[\sim ++ Y g] g g[\sim + Y g] g X \]

\[ g[\sim +++ Y g] g g[\sim + Y g] g g[\sim + Y g] g X \]

…
Example - tree

```
<table>
<thead>
<tr>
<th>Axiom:</th>
<th>B</th>
<th>[ \sim # # T L - B + + B ]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>L</td>
<td>[{- g + + g % - - g}]</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>@@@ R</td>
</tr>
<tr>
<td></td>
<td>T</td>
<td>T g</td>
</tr>
</tbody>
</table>
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>brown</td>
</tr>
<tr>
<td>polygonColor</td>
<td>forestGreen</td>
</tr>
</tbody>
</table>
Example – tree rendered
Stochastic Tree

- Add a rule T -> T
- Now there is a choice for T, draw a line or don’t
Same Stochastic L-System

- Rendered 3 times, each at 8\textsuperscript{th} derivation
JFLAP

- JFLAP is available for free:
  www.jflap.org
- JFLAP was developed by many Duke undergraduates over many years, has many other parts to it for studying theoretical computer science concepts
- JFLAP is downloaded in over 160 countries.
- Duke School of Environment uses L-systems to model pine needles in Duke Forest
Exercise 1

• Write an L-system for the picture below.
• Symbols needed are: g, + and one variable
• Distance of the line is 100, rendering at 1 draws the first line, each additional render draws another line.
Exercise 2

• Write an L-system for the picture below.
• Symbols may need: g and +
• Distance is set to 10, angle to 90, first rendering draws smallest square, additional render draws next larger square
Exercise 3

• Write an L-system for the picture below.
• Symbols may need: g, %, +
• Distance set to 15, angle set to 45, side of square is length 30, first diagonal line is 60
• 1st, 2nd and 6th renderings shown
Exercise 4

- Write an L-system for the picture below (this is a sample tree to focus on branching, don’t look at the tree from before).
- Symbols may need: g, +, -, [ ]
- Angle set to 30, distance set to 20
- 3rd rendering shown
Exercise 5

• Write an L-system for the picture below.
• Symbols may need: g, +, -, [ ]
• Angle set to 90, distance set to 15
• Shows 1st, 2nd and 3rd renderings