Computer Graphics
MTAT.03.015

Raimond Tunnel

Study IT in .ee
The Road So Far...

mtllib triangle.mtl
o Plane
v 1.007839 0.000000 -1.000000
v 1.000000 0.000000 0.978599
v -1.000000 0.000000 -0.588960
usemtl None
s off
f 3 2 1
Procedural Generation

- Generating objects algorithmically
  - Mesh (geometry)
  - Material (texture)
  - Effects (particles)
  - Animation
  - Worlds
  - Characters, weapons, space ships, ...
- More different content, less work for artists
Let's try to generate a tree branch structure.
We start with a trunk.
Tree

- From the trunk, we create two branches for either side.
- We also continue on the forward path.
Tree

- We repeat the same process for all of the new segments.
Tree

- Decrease the length of the segments each time.
Tree

- Repeat again the same process.
Tree

- Introduce randomness.
Tree

- What if we want to store the generated tree structure?
- For example, this smaller tree:
- We should specify the structure and the parameters (length, angle).
Formal Grammar (Chomsky)

- Formal grammar consists of:
  - Set of nonterminal symbols \( N \).
  - Set of terminal symbols \( \Sigma \).
  - Set of production rules.
  - Starting axiom.

- Example:

\[
N = \{ A \} \quad \text{Axiom} = A
\]
\[
\Sigma = \{ a \}
\]
\[
R = \left\{ \begin{align*}
A \rightarrow AA \\
A \rightarrow a
\end{align*} \right\}
\]

Generates words

\[
A \rightarrow a
\]
\[
A \rightarrow AA \rightarrow aA \rightarrow aa
\]
\[
A \rightarrow AA \rightarrow AAA \rightarrow aAA \rightarrow aaA \rightarrow aaa
\]
\[
\ldots
\]
Formal Grammar (Chomsky)

- Used for:
  - Natural language processing
  - Program code processing (compiler, interpreter)

- Hierarchy of types
  - **Type 0: Unrestricted** – $N = \Sigma$
  - **Type 1: Context sensitive** – non-terminal symbol on the left side, can be surrounded by a context
  - **Type 2: Context free** – left side contains only a single non-terminal symbol
  - **Type 3: Regular** – right side is empty, single terminal, or single terminal follower by non-terminal
Lindenmayer System

- Variant of a formal grammar.
- Parallel rewriting system.
- We will look at one, that is:
  - Bracketed system.
  - Stochastic system.
  - Context free (0L-system).
  - Parametric system.

Because of that, does not fall directly under Chomsky's hierarchy.
Lindenmayer System

- **Bracketed system** – we use brackets to indicate branches.

- Using following symbols:

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>Segment</td>
</tr>
<tr>
<td>+</td>
<td>Rotate left 45°</td>
</tr>
<tr>
<td>-</td>
<td>Rotate right 45°</td>
</tr>
<tr>
<td>[</td>
<td>Start of a branch</td>
</tr>
<tr>
<td>]</td>
<td>End of a branch</td>
</tr>
</tbody>
</table>

Can we write our tree using those?
Lindenmayer System

- **Parallel rewriting system** – we can specify rules to generate the tree.
  - Start from the axiom.
  - Rewrite (apply rules) to all symbols at once.

What would be the rules?

Axiom: F


2. iteration:
   - F [+F [+F] [-F] F]
   - [-F [+F] [-F] F]
   - F [+F] [-F] F

This is a trick question.
Lindenmayer System

- **Parametric system** – we can specify parameters for the symbols.
  - The length, the angle etc

\[
F[+(45)F[+(45)F][-(45)F]F] \\
[-(45)F[+(45)F][-(45)F]F] \\
F[+(45)F][-(45)F]F
\]

Every + or - is followed by the angle of rotation.
Lindenmayer System

- We can generate angles with some variance.
- Also specify the lengths of the segments.

If the decrease of lengths is deterministic, we could consider it only, when drawing the tree...

F(1)[+(31.24)F(0.75)][-(47.89)F(0.75)]F(0.75)
Lindenmayer System

- **Stochastic system** – we can have many rules, with the same left-hand side.
- Each rule has a probability.
- The sum of the probabilities of all the rules, with the same left-hand side, has to be 1.

\[
\begin{align*}
A &\rightarrow \frac{1}{3} F[+A]A \\
A &\rightarrow \frac{1}{3} F[-A]A \\
A &\rightarrow \frac{1}{3} F[+A][-A]
\end{align*}
\]
Lindenmayer System

- Rigorous way to specify a mechanism for a **self-similar** structure generation.
- Lot of research and different possibilities.
- Questions?
Particle System

- Used for different effects
  - Fire, fluid, wind, smoke
  - Precipitation (rain, snow)
  - Groups of objects with behaviour (birds, NPC-s)

This you did in the Soft Particle Chopper.
Particle System

- Particles can have a transparency that varies over time.
- Particles can be generated from a pool.
  - If a particle dies, return it to the pool.
- Particle can be 1 pixel in size, or have an image.
- Particle system has an emitter of particles.

Emitter can also be a line, a surface, a volume etc.
Boids Algorithm

- Used to model flocking (of birds).
- Each particle follows a set of rules:
  - **Cohesion** – Move towards the center of mass.
  - **Separation** – Keep distance from other particles.
  - **Alignment** – Follow the average direction.
- There can be other rules.
Particle Systems

- Blender has particle systems

- Example of scar generation via particles: https://www.allegorithmic.com/community/blog/creative-use-particle-brushes
What was new for you today?

What more would you like to know?

Next time: Ray Casting, Ray Tracing, Space Partitioning, BVH