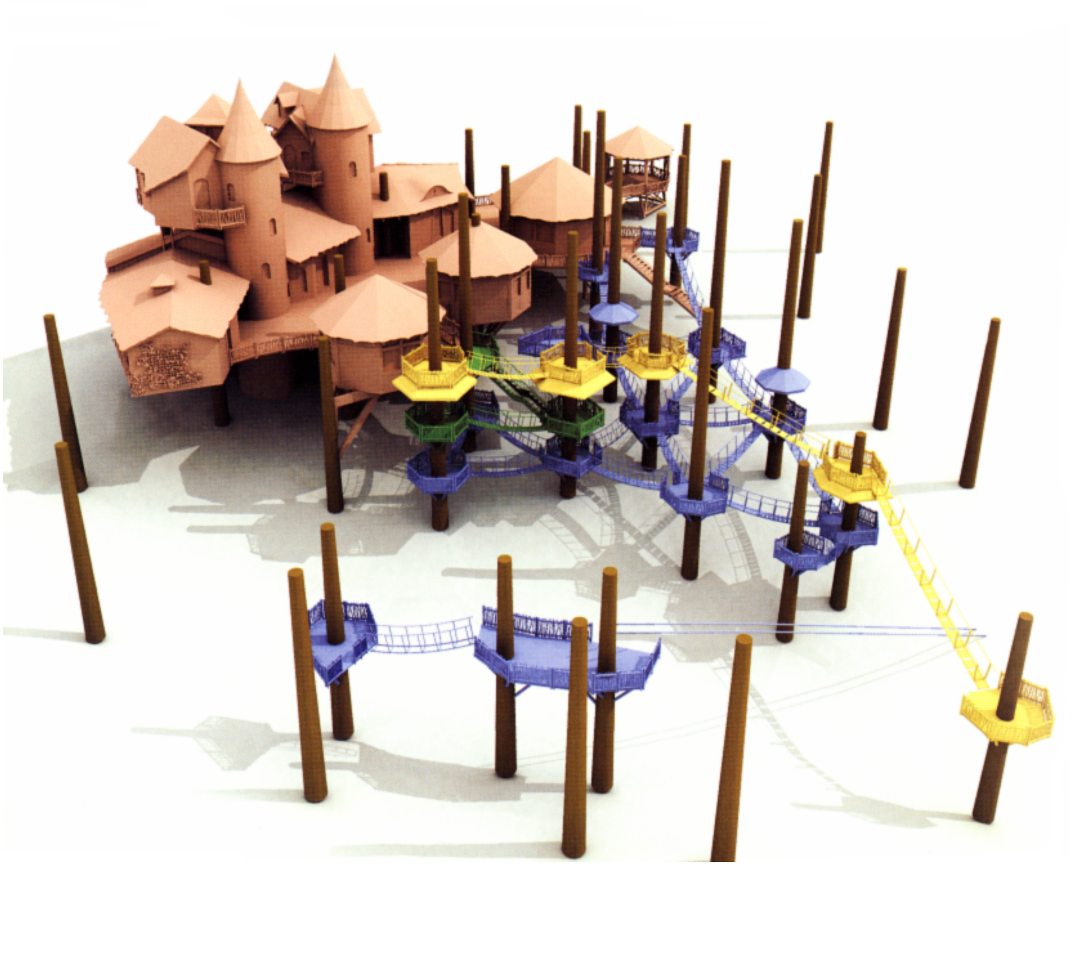
Universiteit Utrecht

Planning Procedural Architecture in Tree-like Geometry

non-realistic procedural game worlds

by Ruud op den Kelder



Abstract

Contents

Chapter 1

Schematic Outlay Generation for Elements

I am interested in the structure of trees and the possibilities and restrictions it poses for placement of architectural shapes. For the purpose of this thesis the generated trees do not have to be visually convincing, however the basic shape should still be identified as a tree. The geometric properties of a tree model that are to interest of us are those that have effect on the possibilities with respect to the incorporation of architectural man-made structures. The tree geometry functions as the support structure for the building blocks I define in detail in section **??**.

In this section we will discuss a method that generates a graph representing the structure of elements in a forest environment.

1.1 Scattering Techniques for Tree Positions

1.2 Ecosystem Modeling

1.3 Multilayer TreeNode Generation

For our method we have to be able to strategically place geometric architectural elements within tree structures. Finding the positions at which these architectural elements can be placed could be performed as a postprocess type process by analyzing the generated geometry, however we have the oppertunity to incorporate positional semantic information to the trees during the generation phase which simplifies the problem.

L-system tree generation methods are mainly focused at producing convincing visual representations of trees [**?**]. However, we do believe that these methods can be extended quite easily to enable the addition of structural semantics. This thesis does not pursue the introduction of such an extension. For our problem we mainly have to deal with structure of a forest which we will abstract to a set of nodes, representing trees, for the moment. Finding useful connections between nodes and incorporation of structual elements within these tree nodes is what we are interested in.

We propose a multilayer graphbased approach for the generation of a schematic outlay for elements in a forest structure. The nodes in this structure function as empty slots which define semantics that determine the type of elements this node can contain. See figure **??** for an example element graph for a single tree.

The element properties of a node are used to query the architectural element database. Listing 1.1 shows an example of element properties for a node.

Listing 1.1: Example of node element properties

```
Element_node_14
{
        single_node_platform
        {
                available = true;
                max_diameter = 30.0;
        }

        bridge
        {
                available = false;
                type = suspension;
        }

        multi_node_platform
        {
                available = true;
                other_nodes = array{34, 23, 78};
                type = undefined;

         if (used)
                        bridge.available = true;
         else
                        bridge.available = false;

        }

        building
        {
                available = false;
        }

        stairs
        {       //staircase connection to parent node
                available = true;
                type= revolving;
        }
}
```

I will cover node element properties and quering of the element database in the next chapter in more detail.

Our schematic outlay generation method uses the following parameters:

1. Area A
2. Density $D (0.0 < D < 1.0)$
3. Layers $L (L > 0)$
4. Tree parameters defining min bounds T_{min}
5. Tree parameters defining max bounds T_{max}
6. LocalTreeDeviation (controls deviation of tree properties within an area of size ClusterArea)
7. ClusterArea

With these parameters in place I can start discussing the algorithm. The rootnodes of our tree element graphs are contained by the base layer. The algorithm start by assigning 2d coordinates to the root nodes of each element graph using the scattering algorithm.

The scattering algorithm requires the $Area$, $Density$ parameters and the bounding $Treeparameters$ which determine the actionradius for the generated trees. For each root node the algorithm sets a semi-random parameterset P_t ($T_{min} < P_t < T_{max}$). The second step of the algorithm creates the upper $L - 1$ layers.

A layer is in fact a container for element nodes of all trees that allow connections through bridging. Nodes within a single layer roughly have the same height to the baselayer. Whether two nodes within a layer can be connected with a direct bridge depends on the element element properties of these nodes, the distance between them, and obstructions caused by other nodes.

With the layers in place we iterate through the root node list in the base layer. For each root node a corresponding element graph is build. The method that builds an element graph depends on the global parameters of the forest but also on the structure of local neighbours. The method performs local statistical analysis on local element graphs within an area of size $ClusterArea$, and the result, combined with the $LocalTreeDeviation$ parameter, is used to generate a deviating tree element graph. Again note that the nodes in the element graph provide information about allowed architectural elements at this node, and the edges of an element graph represent connections by means of stairs or bridges. The element graph is in fact a subgraph of the final tree-structure graph that I will discuss in section **??**.

Growing an element graph for a tree means iterating through the layers. For each layer, n ($0 >= n > MaxNodes$) nodes with element properties are generated (a pointer to each node is stored in the local tree element graph and the corresponding layer). Note that when the element graph has no nodes at a specific layer this does not mean that the graph has stopped growing. Instead it means that the final tree structure prohibits elements to be placed at this position. When a new node is added to a layer the method tries to connect it to local nodes, provided that the node allows bridge connections. Figure **??** shows a configuration of element graphs.

When all element nodes have been constructed and placed into corresponding tree element graphs and layer element graphs the method checks wether all nodes are reachable from any other nodes. Single nodes or sets of connected nodes that are not reachable from the the main node graph will be deleted. We now have the schematic structure of a forest which we need as input for the architecture planning method discussed in the next section.

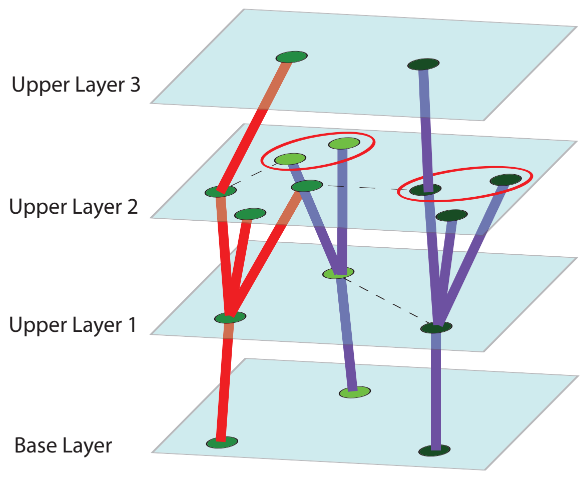


Figure 1.1: example of multilayer element graphs for a small tree set

1.3.1 Simple Tree Structure

Procedurally generating trees by means of l-systems has had a great amount of succes since the original proposal by Lindenmayer. The L-system formalism is widely applied in academic work and is also succesfully used within commercial applications. This thesis does not focus on the l-system formalism in particular, since it is already a well established theory **(author?)** [1]. I describe L-system in short in the following section as it is the most used fomalism for the contruction of trees.

1.3.2 Creating Tree Geometry

Bibliography

[1] A. L. Przemyslaw Prusinkiewicz. *The Algorithmic Beauty of Plants*. Springer Verlag, 1990.