

Creating enchanted forests using flocking algorithms

Daniel Stern

Filmakademie Baden Württemberg - Institute of Animation, Visual Effects and Digital Postproduction

1 Motivation

The manual creation of plants and trees using 3D software can be very time consuming. And to create a forest, automation is required. The following procedures explain a novel use, and extension of, standard flocking algorithms used to achieve new growth behaviours for the creation of tangly and connate trees and forests.

2 Basic Concept

This approach does not build each tree by starting with a large stem or trunk with roots at the bottom and branches at the top. It uses a defined number of separate strands which grow from the point of one root to the end of one branch. The stem/trunk is formed through the fusion of all the strands.

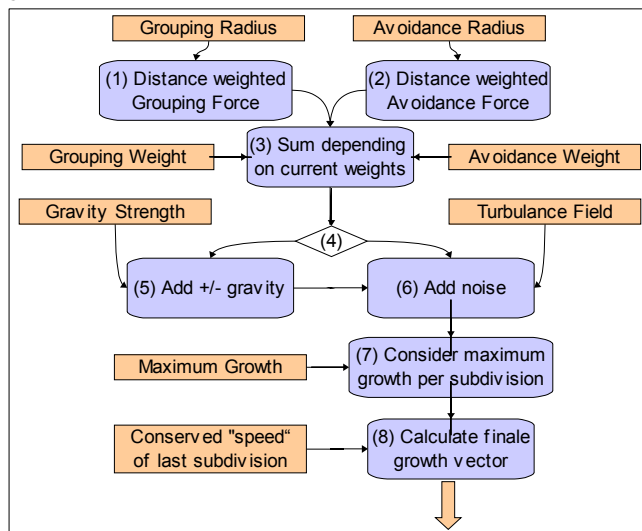
All strands grow up simultaneously and have a defined number of subdivisions, starting position and a number of other parameters. Which specify, amongst other attributes, their inheritance and their maxim growth per subdivision.

While evolving the trees, a list of all of the positions of the strands is kept. Before every subdivision step, the algorithm calculates which strands are in the neighborhood of the one that is currently being processed. By using this information, it determines in which direction to evolve – essentially depending on two rules: Cohesion and Avoidance.

3 Cohesion and Avoidance

To get results that appear like natural trees, the default-settings let the strands try to find others and to build a stronger unit, when beginning their way up. Later they want to build branches with leaves. To get as much of the sun's energy as possible they try to avoid the others and to separate as far as possible from them.

Cohesion and Avoidance are implemented as explained by Craigh Reynolds (see below) by calculating the center of the positions of the surrounding strands. To do that the vector between each neighbour's positions is subtracted from the position of the current strand, and weighted dependent on the length of that vector. The sum of these vectors is divided by the number of surrounding elements and used as a force-vector (1) either pulling (cohesion) or (2) pushing (avoidance) the current strand towards or away from that center and thus changing its direction of growth.



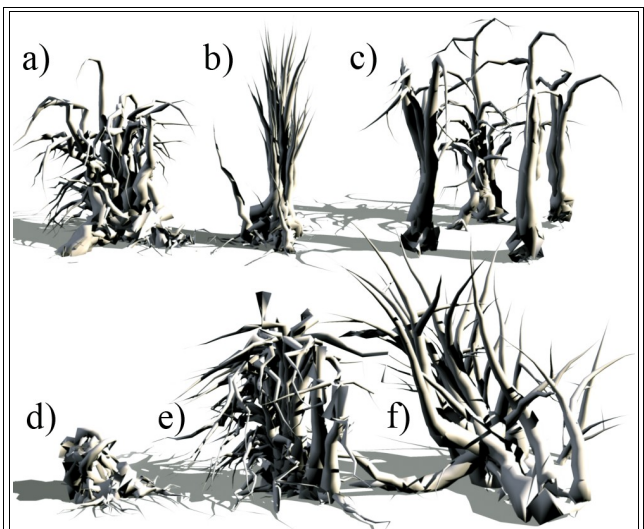
4 Additional Components

To optimize the results (3) of the calculation of the two algorithms, the resulting force vector is modified by several other factors. To achieve a smoother shape, the strands conserve some of their "speed" and don't react to the calculated vector directly (8).

The usage of (negative) gravity (5) allows the user to define where the tree grows up or hangs down. To get the effect of a weeping willow for example, the user might crank up the gravity at the end of the growth.

This approach offers the ability to ignore the gravity in case of a zero avoidance vector (4). In doing so the strands can keep their height, as long as they are seeking others.

To get a more crooked appearance, a noise-field (6) can be used.



a) Strong noise results in wide spread of strands.

b) Strong positive gravity forces the strands to grow upwards.

c) Small group radius results in several weaker units.

d) Zero avoidance leads to a branch-clew.

e)+f) Low speed conservation value produces unsteady strand growth, while a high value leads to a smoother flowing result.

6 Implementation

This approach is implemented as a plugin for Maya. By modifying curves (Maya animation curves) the evolution of the strands is made controllable. While the horizontal axis represents the growth of a single strand from the root to the top (0 to 1), the user can define the variation of several attributes on the vertical axis. There are control-curves for the strength and radius of the algorithms, gravity, noise and the shape of the strands (maximum growth per subdivision, radius and split probability).

7 Conclusion

This approach shows that algorithms which are normally used for the movement of creatures can also achieve interesting results when forming the shapes of plants. This new set of parameters allows the user to create complex and diverse plants.

References: REYNOLDS, CRAIG W., 1987, *Herds and Schools: A Distributed Behavioral Model*, Computer Graphics 21(4)