Natural Structures through the Convergence of Particles and Shapes

Prof. Rama C. Hoetzlein, Ph.D
Digital Media Design in the Bower School of Music & The Arts,
Florida Gulf Coast University, Fort Myers, Florida
www.ramakarl.com
e-mail: rhoetzlein@fgcu.edu

Figure 1. Tree made with particle-shapes

Abstract

This work reports on the development of a generative simulation for natural structures based on a convergence of particle systems and shape grammars. A review of procedural methods for natural vegetation leads to an integrative approach resulting in a new primitive: *particle-shapes*. Artistic aspects will be discussed as they relate to the abstraction of nature. Results emulate a wide range of complex natural structures.

1. Background & Motivation

One of the first processes to convincingly represent natural structures is the L-system, which expresses branching forms using *replacement* rules that progressively increase the detail of a shape in a hierarchical fashion, and has been used extensively to model plants and trees [1]. Subsequent work focused on modifications that allow for greater variety and irregularity. Oppenheimer introduced twisting and randomness over a generated hierarchy [2]. Weber introduced explicit shapes such as cones placed on a branching structure [3]. These works all share the notion of a rule-based fractal grammar.

Particle systems, on the other hand, were first applied to model soft objects such as fire and water [4]. Reeves also developed an early particle system for plants, although this effort focused primarily on dense forest rendering rather than elaborating on plant structure [5].

We explore the relationship between particles, grammars and shapes to develop a conceptual framework and a simple integrated model for the growth of natural structures based on merging these two representations.

	L-System	**Shape Grammar**
Definition	G = (V, w, p)	SG = (V_T, V_m, R, I)
Objects	V = alphabet of symbols	V_T, V_m = finite set of shapes
Initiator	w = axiom or initiator symbols	I = initial shape
Rules	p = production rule pair (s,t)	R = rules as ordered pairs (u,v)

Table 1. Similarity between L-systems and Shape grammars, both inspired by Chomsky production grammars. Definitions from Lindenmayer [1] and Gips [6].

2. Conceptual Framework

L-systems and fractals are related to the more complex *shape grammars* which have been applied extensively to architecture and design [6]. Indeed, the definition of both L-systems and shape grammars refer to Chomsky grammars as their underlying principle. Their similarity is observed in Table 1. By assuming the L-system is used to generate a geometric form, the symbols in V can be mapped to V_T. The L-system production rules, p, can be mapped to shape rules R. Whereas L-systems were classically a string rewrite language shape grammars are more powerful since the elements express arbitrarily complex geometric shapes in 2D or 3D.

While shape grammars are extremely expressive they come at the cost of increased complexity. One must keep track not only of hierarchical relationships, but must also represent 2D or 3D shapes, and perform complex geometric operations represented by the rules. The author previously developed a shape grammar language for procedural modelling [7], yet the program complexity continually limited software scalability.

Particle systems, on the other hand, are significantly simpler to design and implement than shape grammars. Particles typically follow physically-based *simulation rules* for motion. Rodkaew models plants using particle systems but oddly these particles start at the leaves and are attracted toward the trunk [8].

Conceptually we can understand L-systems, shape grammars and particles as systems which vary in terms of their *structure* versus *behaviour,* see Figure 2. Classic L-systems have a discrete hierarchy with no continuous growth. Newer models such as FL-systems, functional L-systems [9], and parametric shape grammars can be thought of as extensions that increase the behavioural aspect of these system. Significant work on shape grammars has focused on making them procedural, stochastic and more functionally flexible [10].

The present work explores the idea of extending particle systems to allow for an inherently behaviour-oriented particle object to have more structural properties.

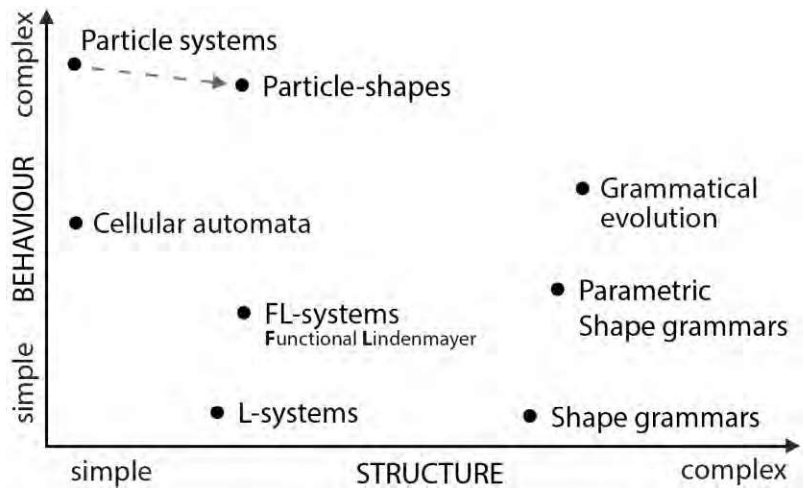

Figure 2. Structure and behaviour of several systems for the expression of form. Particle-shapes are inspired by extending the properties of particle systems to include structural qualities (top left).

3. Particle-Shapes

The notion of a *particle-shape* is introduced as an atomic element that has properties of both particles and shapes. These properties are listed in Table 2. Particle-shapes have position, velocity and orientation like their classical counterparts. Yet they also have mass and volume defined by a rectilinear solid space. Importantly, particle-shapes define child and next references allowing them to form chains and branches.

No explicit grammar is needed (but is implicit in the chains). The branching structure of trees is modelled with the notion of *spawning* new particles at regular, random intervals. Instead of replacing branches with fractally smaller pieces, these particles *grow* new branches in a natural way from previous points outward. While plant cells operate on a microscopic scale, the idea of growth based on cellular splitting and propagation is well supported by nature as in Figure 3.

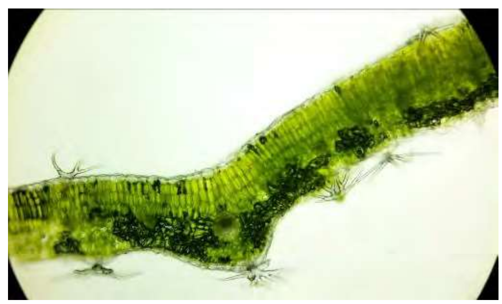

Figure 3. Cells split and grow to form the solid parts of a plant in Alyssum alyssoides. Image © Stefan Lefnaer

Def. ParticleShape:

vec3	position
vec3	velocity
vec3	direction
vec3	scale
quaternion	rotation
matrix4x4	transform
int	level
int	child
int	next

Table 2. Properties of a particle-shape.

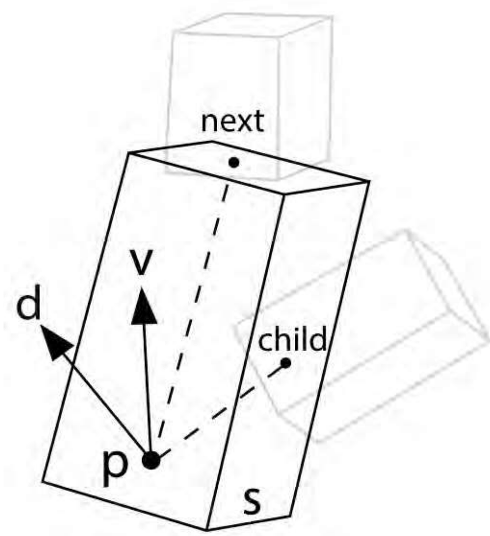

Figure 4. Design of a particle-shape: p (position), v (velocity), d (target direction), s (scale), child and next shapes.

The motion of particle-shapes is determined by Eulerian simulation techniques whereby force and velocity increment position, with additional terms for more complex behaviour.

$$d_{t+1} = d_t + \textit{rand}[-1,1]\ \textit{wander}\ dt$$

$$v_{t+1} = [v_t + \textit{spread}\ (d_{t+1} - v_t)]\ Q(\textit{twist}, d_t)$$

$$p_{t+1} = p_t + v_{t+1}\ dt$$

Velocity (v) expresses the current motion of the particle while target direction (d) expresses where it will or should move, allowing for complex motions such as twisting and bending of branches by *wandering* randomly in [-1,1]. The *spread* determines how strongly the velocity is attracted to the target direction. Q(twist,d) is a quaternion rotation of the velocity around the direction vector for twisting.

Tree forms evolve naturally as particles follow their trajectory and periodically spawn new branches, Figure 5. A set of parameters control the timing, distribution and orientation of new particle-shapes as they are spawned to form branches. The depth of recursion is controlled by maintaining the branch level with each particle and passing this onto its children. Limits to the age, length and level of particle-shapes terminate growth.

Figure 5. Trunk form from Figure 1 generated by the behaviour model described here.

4. System & Rendering

The present framework for particle-shape modelling, simulation and rendering are all developed as custom software written in C++/OpenGL including a custom rendering option using OptiX for high quality raytracing. The author builds on previous efforts in procedural modelling, Luna [7], to create the present framework, titled SHAPES.

Figure 6. Stages of rendering: a) Growth and spawning of particle-shapes, left, result in complex branching, b) The entire set of shapes is skinned, middle, to create a smooth appearance, and c) additional shapes are generated for smaller branches and leaves, right.

Direct rendering of particle-shapes is possible both in real-time and via raytracing by using the technique of *geometry instancing* whereby the same shape – in this case a rectilinear prism – is repeatedly rendered at different locations (Figure 6a).

Rendering of a smooth tree trunk and branches is desirable. Rather than replace individual shapes with geometry this is achieved by lofting the entire set of particles with a single skinning primitive (Figure 6b), similar to Subramanian [11] and Obradovic [12]. The resulting surface closely resembles a modelled tree to which color and texturing could be applied.

Additional particle-shapes may be generated at any time to fill in smaller branches and leaves (Figure 6c).

5. Results

The model presented provides a continuous parameter space for the generation of a wide range of natural structures. Trees and bushes generated using these techniques are shown in Figures 6, 7 and 8. These were created solely by modifying the parameters of the system as there are no discrete generation rules.

Particle-shapes are able to represent a wide variety of forms that are difficult to achieve with L-systems or shape grammars. Twisting vines and trees are represented by increasing the rotational force (Figure 7). Meanwhile, grasses are expressed by having many initial particle-shapes which then grow and fall due to gravity (Figure 8).

Figure 7. Tree generated with increased twisting force

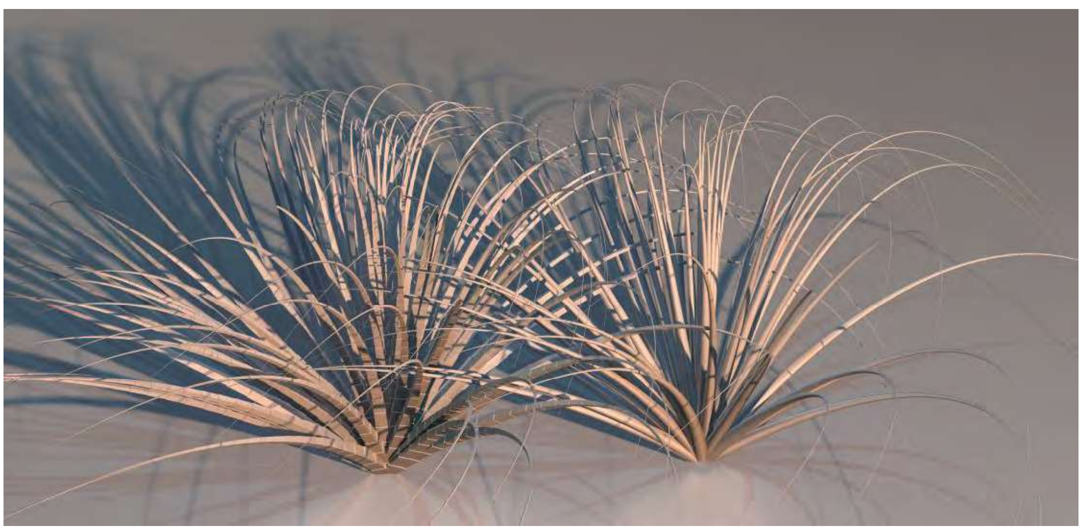

Figure 8. Grasses generated in the same framework by increasing gravitational force

Figure 9. Detail of branch structures. Notice that each branch is not a rigid or substituted shape in a hierarchy but flows from a chain of particles. The tree hierarchy is implicit.

Behavioural changes in scale result in either fat, stubby branches or long skinny ones. The branching points are not decided a priori or deterministically. This approach supports many types of vegetation – from trees, to bushes, to grasses.

6. Aesthetic Discussion

Many interesting questions are raised by this work. Does nature operate more like a rule-based grammar or a particle-based simulation? There are arguments to be made for both. The typographic nature of DNA suggests a grammatic relationship between genetic code, genotype, and the observed forms of phenotypes [13]. Yet cellular splitting, growth and motility suggest the important role of physical forces in natural systems. Certainly, both play a role, yet our understanding of how complete biological structures derive from genetics is still limited.

Also fascinating is the degree to which abstraction is present in all simulations of nature. There is no current system capable of modelling complete plants down to the cellular level. Thus the present particle-shape model, while inspired by cellular growth, does not operate on that scale. Particles are a mathematical abstraction of behaviour similar to abstractions of shape in architecture [14]. Any simulation of nature raises the question of whether a complete model is possible, or what this would even mean, since nature is intimately interconnected with its environment. Nonetheless, our ability to build digital models of natural systems will continue to improve. Can these models eventually achieve structural parity with nature or will they always retain an element of abstraction?

Creatively, the distinction between artist and scientist has witnessed an increasing overlap. The role of the artist is admittedly no longer to only mimic the outward appearance of forms but to reflect on their structure and substance. Yet there is an inherent element of play and fantasy present in generative art since these forms are not a true reflection of reality but a synthetic creation – unable to capture many natural forms, yet also capable of expressing forms well outside of nature. One difference between the artist and scientist, therefore, is that whereas the scientist proceeds continually toward an accurate reflection of nature (by testing theories), the artist is not limited thus and may explore generative worlds to understand new principles not bound to reality, or may simple explore beauty in form for its own sake.

7. Future Work

Regarding the present model there are several interesting directions for this work. Particle-shapes are currently much more like particles than shapes. The most natural extension is to replace these prisms with more complex geometries. This would allow for intricate leaf profiles, buds and flowers. Such an approach moves away from the purity of particles by introducing substitution rules – thus bringing them closer to shape grammars (Figure 2). Yet the author believes that a final convergence of behaviour-based particles and rule-based shapes is inevitable in this field. Nature does both at differing scales: it *behaves* (macroscopically according to physics) while also following *rules* (microscopically according to DNA, etc.) to create structures.

Other limitations are also apparent. This work has focused on the realism of branches yet much more could be done to generate other plant parts. Coloring and texturing in this work were intentionally avoided to focus on the modelling outcomes where it is understood that greater realism would be found by adding these. These and other extensions are left for the future.

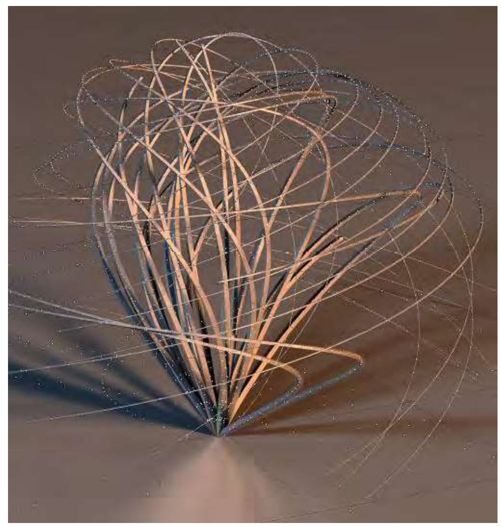

Figure 10. Unexpected generated forms

While procedural methods can already achieve visual realism with vegetation using existing techniques, the future of generative natural forms is far from complete. Generative modelling and simulation deserve new attention beyond visual appearance. We are approaching a point where hardware will be sufficient to model entire plants at the cellular level. This will likely bring a whole spectrum of novel insights while providing new tools for advances in biology, genetics and ecology. Aesthetic visions found in this future are still unfolding and have yet to be discovered.

References

[1] Prusinkiewicz, P. and Lindenmayer, A. (1990). The Algorithmic Beauty of Plants, *New York: Springer-Verlag*, 1990

[2] Oppenheimer, P. (1986). Real Time Design and Animation of Fractal Plants and Trees. *ACM Transactions on Graphics*, Vol 20, No 4, 1986

[3] Weber J. and Penn, J. (1995). Creation and Rendering of Realistic Trees, *Proceedings of the 22nd annual conference of ACM SIGGRAPH '95*, Sept. 1995, p. 119-128

[4] Reeves, W. (1983). Particle Systems – a Technique for Modeling a Class of Fuzzy Objects. *ACM Transactions on Graphics*, Vol 2, No 2, April 1983, pp. 91-108.

[5] Reeves, W. (1985). Approximate and Probabilistic Algorithms for Shading and Rendering Structured Particle Systems. *ACM Transactions on Graphics*, Vol 19, No 3,1985

[6] Stiny, G and Gips, J. (1972). Shape Grammars and the Generative Specification of Painting and Sculpture, in CV Frieman, *Information Processing 71*, *Proceedings of the IFIP Congress*, North-Holland, Amsterdam, pp. 1460-1465.

[7] Hoetzlein, R. (2010). Luna: A Puzzle-Based Metaphor for Procedural Modelling. *In Dissertation*, University of California Santa Barbara. December, 2010

[8] Rodkaew, Y., Chongstitvatana, P., Siripant, S., Lursinsap, C. (2003). Particle Systems for Plant Modeling, In *Plant Growth Modeling and Applications*, p. 210-217

[9] Marvie J., Perret, J. and Bouatouch K. (2005). FL-system: A Functional L-system for procedural geometric modelling, *The Visual Computer*, Vol 21, pp. 329-339, May 2005

[10] Roncoroni, U. and Crousse V. (2016). Programming generative grammars. *Proceedings of the XIX Generative Art Conference*, GA2016.

[11] Subramanian, S., Eng, M., Krishnamurthy, V. and Ergun, A. (2019). Delaunay Lofts: A Biologically Inspired Approach for Modeling Space Filling Modular Structures. *ACM SIGGRAPH 2019 Posters*, Article 81, July 2019.

[12] Obradovic M. (2017). Creating 3D shapes by time extrusion of moving objects, *Proceedings of the XX Generative Art Conference*, GA2017

[13] Achim C., Caldwell D., Good, T., Olster, D. (2017), Understanding the Rules of Life: Predicting Phenotype, *National Science Foundation*, 10 Big Ideas, NSF News Release, Aug 8, 2017

[14] Leyton, M. (2001). A Generative Theory of Shape. *New York: Springer-Verlag*, 2001