Constructive Path Planning for Natural Phenomena Modeling

Ling Xu and David Mould
{lix272|mould}@cs.usask.ca

Abstract

Path planning is a problem much studied in the context of artificial intelligence, with many applications in robotics, intelligent transport systems, and computer games. In this paper, we introduce the term *constructive path planning* to describe the use of path planning to create geometric models. The basic algorithm involves finding least-cost paths through a randomly weighted regular lattice; the resulting paths have characteristics in common with plants and other natural phenomena, with visible structure imposed on the randomness by the optimization process. This paper explores different arrangements of graph weights and shows the effectiveness of the technique in two detailed examples of procedural models, one for elm trees and one for lightning.

1 Introduction

The virtual worlds of computer games and computer-animated films are filled with objects both familiar and fantastical. Because of the difficulty of creating sufficiently detailed and numerous models, algorithmic (procedural) techniques have been sought, with the aim of easing the burden on digital artists. In particular, procedural techniques to create models of natural phenomena such as trees, mountains, and clouds have been a subject of long-standing interest by the computer graphics community. Numerous algorithms have been devised for procedurally creating such models (Ebert et al., 2003).

In this paper, we describe an approach to natural phenomena modeling which we term "constructive path planning". The technique involves finding least-cost paths through weighted graphs; we introduced the main idea earlier (Xu and Mould, 2007) and here extend and further refine it. The technique involves creating a regular lattice (either a square lattice in 2D or a cubic lattice in 3D) and placing random weights on the edges, then planning least-cost paths through the lattice. By planning paths from a single root node to destination nodes elsewhere in the lattice, a "dendrite" could be created – a sparse, acyclic subset of the original graph. We previously suggested that the algorithm could be used to produce a variety of natural phenomena, and gave examples of coral, lichens, rocks, and lightning.

We make three main contributions in this paper. First, we describe how to make use of the weights in the graph to vary the structure of the output model. Second, we identify the cost value as a useful input to the model and show how cost values can be sensibly used to inform the model appearance. Third, we apply our technique to two specific natural phenomena, elm trees and lightning, and show images and models arising from this application. The lightning we present in this paper is a considerable improvement over previous results.

The remainder of this paper is organized in four parts. First, we discuss some previous work in natural phenomena modeling. Second, we describe the algorithm of constructive path planning and give details on different edge weight distributions that are useful for modeling purposes. Third, we show results, in the form of images and renderings of our models, and discuss the advantages and disadvantages of constructive path planning. Fourth, we conclude with some recommendations for future work.

2 Previous Work

Scientific models for natural phenomena abound (Ball, 2004). In computer graphics, procedural natural phenomena has been studied considerably, with numerous algorithms for procedural terrain, trees, clouds, and other natural objects (Ebert et al., 2003). The most prevalent method for procedural trees and plants is the replacement grammar L-systems (Lindenmayer and Prusinkiewicz, 1990). Diffusion-limited aggregation (Witten and Sander, 1981) is a physically motivated model that is also sometimes used. More recently, image-based systems for tree modeling have appeared (Neubert et al., 2007; Tan et al., 2007).

L-systems possess two parts: a grammar, with rules describing how tokens are transformed into other tokens or sequences of tokens, and a modeling system, describing how to interpret the tokens as geometric shapes or transforms. Research into L-systems has concentrated on the first part, since rich structures represented by strings of tokens can easily be interpreted into meaningful geometric shapes (perhaps the most common in-

terpretation is the "turtle language", where the tokens are commands directing the turtle to move forward, move backward, or turn in different directions). L-systems, augmented by extensions such as stochastic L-systems, open L-systems, and environmentally-sensitive L-systems, have been quite successful at plant modeling and even modeling of entire plant ecosystems (Prusinkiewicz et al., 1994; Mech and Prusinkiewicz, 1996; Deussen et al., 1998). However, design of replacement rules is challenging.

Procedural methods for creating lightning have previously been reported in the literature, notably by Reed and Wyvill (Reed and Wyvill, 1994) and by Kim and Lin (Kim and Lin, 2004). Reed and Wyvill used an ad-hoc particle tracing method to generate a lightning structure, and rendered lightning glow using implicit surfaces. Kim and Lin implemented the dielectric breakdown model for a more physically based approach to lightning; their method produces high quality lightning models and images, but at considerable computational expense.

We previously (Xu and Mould, 2007) presented path planning as a modeling technique, and showed how to build a variety of dendritic structures with it, including lightning. We used Dijkstra's algorithm (Dijkstra, 1959) to obtain least cost paths through a random graph, relying on the property that the graphs had positive weights to avoid cycles.

While we suggested that the edge weights can be used to control the resulting structure, we did not demonstrate the results of attempting to do so. Also, we omitted to exploit one of the chief pieces of information provided by the algorithm: the cost information at each node. (Cost information was used to determine the path, but ignored once the paths had been found.) We show some examples of controlling the model shape by spatially varying the edge weights, and show how path cost can help to shape our models. Our techniques are employed to create two different types of natural phenomena: elm trees and lightning.

3 Algorithm

The base algorithm we employ is that of least-cost paths through a weighted graph (Xu and Mould, 2007), where the paths themselves are the modeling primitives. In this section, we first show how to vary the graph weights spatially to get different effects, and then show how to modify the structures along the path length to produce realistic-looking lightning.

The basic algorithm can be decomposed into the following steps:
1. Create a regular square lattice of nodes: 4-connected in 2D, 6-connected in 3D.
2. Choose edge weights for the graph.
3. Choose a node to be the root of the structure.
4. Apply Dijkstra's algorithm to find path costs to all nodes from the root.
5. Choose path endpoints.
6. Use a greedy algorithm to backtrack from the endpoints to the root, giving the paths.
7. Render the paths.

Figure 1 shows a few different structures obtained by placing the root at the bottom of the image, scattering endpoints through the graph, and planning paths. The differences between the structures come from substantial changes made to the distributions of edge weights.

Typically, edge weights were chosen at random. Here, however, we add structured values to the initial random values. Some randomness is needed so that paths do not appear too regular.

In the upper left structure, edges become more expensive the greater their horizontal distance

from the image centre. The result is a structure where the paths hug the central region as much as possible before striking out towards the endpoints. Conversely, in the upper right structure, costs are larger near the image's centre, making it preferable for the paths to bend outward into the cheaper regions before being forced to return for the endpoints. The lower image pair shows a similar contrast: in the lower left, edges are cheaper the closer to the top they are, while in the lower right, edges are cheaper at the bottom. The result is two contrasting structures, the former more tree-like, the latter more like a shrub.

The examples shown involve simple functions of location modifying edge cost, and were done on flat two-dimensional graphs. Such 2D path planning exercises can be completed quickly (in at most a few seconds on a modern desktop computer with modest hardware) and can serve as prototypes for more elaborate models that will be created in 3D. Some 3D models based on these prototypes are shown later in this paper, in Figure 5.

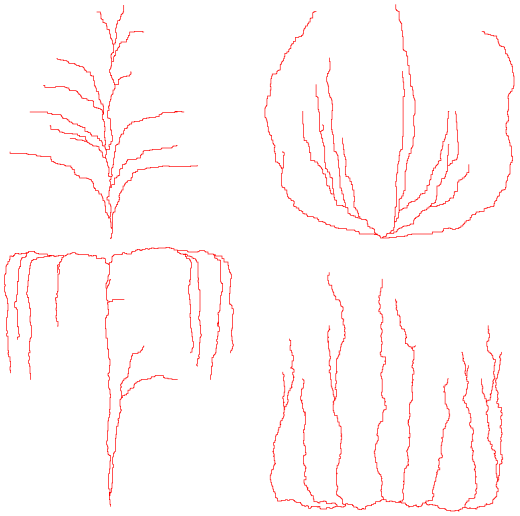

Figure 1: Different structures created by varying the edge weights.

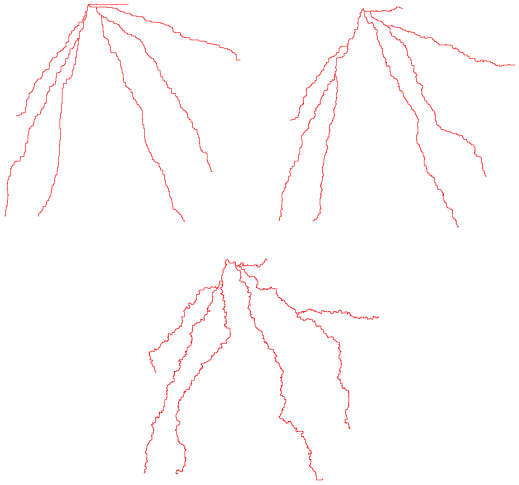

Figure 2: Variations in structure achieved by varying the statistical distribution of edge weights.

The previous examples showed how spatially varying changes to edge weights can affect the structure's overall shape. We now turn to an example of changing the edge weight distribution globally. Previously, we had had edge weights drawn from a uniform distribution (1,max). Here, we suggest that for each edge weight, its value e should be

$$e = R^{\alpha}, \tag{1}$$

where R is a random value drawn uniformly from the range (1,max), and α is a parameter controlling the amount of path variation.

The larger the exponent α, the greater the disparity between the cheapest and most expensive edges, and therefore, the greater the incentive for the path planner to seek paths consisting of cheap edges. It is no longer profitable to seek short cuts through an expensive edge if many cheaper edges could be used instead. For example, for $\alpha > 1$, when $a+d+c+d=e$, $a^{\alpha}+b^{\alpha}+c^{\alpha}+d^{\alpha} < e^{\alpha}$. Thus, increasing α will make it more attractive to take paths with more edges, if those edges are individually cheap. Higher α

will result in structures with longer, more roundabout paths through the graph than the structures made with lower α. The difference is exemplified by the structures in Figure 2, which show results for $\alpha = 0.3$, $\alpha = 1.0$, and $\alpha = 3.0$. The $\alpha = 0.3$ structure has very direct branches, nearly straight lines; the differences between different edge costs have been suppressed. At $\alpha = 1.0$, we have a conventional model, with slight variation in the paths. Finally, with $\alpha = 3.0$, the paths have become more erratic yet, willing to diverge considerably from the Euclidean shortest distance to achieve a better result. Of course, there is no reason that we must stop with $\alpha = 3.0$, and if the desired effect warrants it, still more lengthy and tortuous paths can be obtained with even higher α.

In the remainder of this section, we show how to create plausible-looking lightning. The lightning models use the nonuniform distribution with $\alpha = 3.0$.

The path planning formulation is already well suited to producing the structure of lightning, simply by placing endpoints and planning paths through the graph. Here, we show how the path costs found by the application of Dijkstra's algorithm can be used to achieve the appearance of lightning as well.

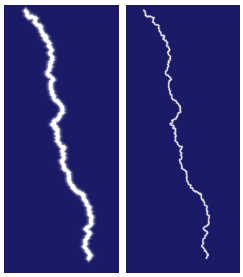

Figure 3: Lightning width adjustment. Left: T=5.0; right: T=1.6.

We can produce a visualization of the lightning stroke by assigning a brightness value to each node (pixel) in the graph, computed based on path distance. Let d represent the path cost found by Dijkstra's algorithm. For a pixel at distance d, and a thickness factor T, we compute a brightness value V as $V = \exp(-(d/T)^2)$. For very small distances, we have V near 1, but the greater the distance, the smaller the V.

Two lightning strokes are shown in Figure 3, a wider one ($T = 5.0$) and a narrower one ($T = 1.6$). Note that the thicker branch has a weak halo around it, obtained without additional computation. The visible halo around light sources is characteristic of participating media, and lightning typically occurs in stormy conditions where participating media (clouds, rain) are also present.

The previous figure showed how to get lightning strokes of constant width. We can straightforwardly obtain tapering strokes by modulating T along the length of the stroke. We use the original path cost value to do this: we set T_i at a point i along the path to

$$T_i = T_s * (1 - (d_i/d_{max})^{\beta}) + T_f * (d_i/d_{max})^{\beta}, \quad (2)$$

for a starting width of T_s and a final width T_f and a path cost value d_i at point i. The parameter β governs the shape of the tapering; for the lightning, we used $\beta = 1$. Recall that d_i is path cost; in the randomly weighted graphs we used, path cost does not correlate perfectly with edge count, and by using path cost we obtain some additional structure (a nonuniformly but monotonically increasing function) without any further effort. The ability of equation 2 to produce tapering is shown in Figure 4.

Having completed our description of the algorithms involved in creating our models, we next show some more elaborate results, beginning with 3D structures and elm trees. Our completed lightning images are at the end of the following section.

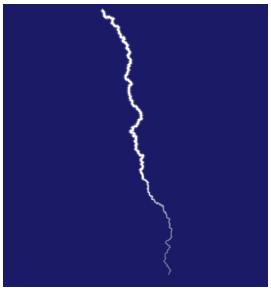

Figure 4: Lightning stroke tapering.

4 Results and Discussion

This section contains the more elaborate results of applying constructive path planning to procedural 3D modeling tasks. We first show some somewhat abstract models obtained simply by extrapolating the 2D models seen in section 3 to 3D; we then describe how to use our system to generate structures resembling elm trees; and we lastly show how to apply the algorithms of section 3.1 to create realistic-looking lightning.

Figure 5 shows different 3D models. These variations were achieved after prototyping the edge weight distribution in 2D, and then performing virtually the same calculation in a 3D space (edge weights that formerly depended on horizontal distance x now depend on horizontal distance $\sqrt{x^2+y^2}$). These shapes were not intended to resemble any particular type of structure, but they are reminiscent of some kinds of plants, or (in the case of the lowermost example) perhaps not only a tree but also some kind of fantastical, Seussian antler.

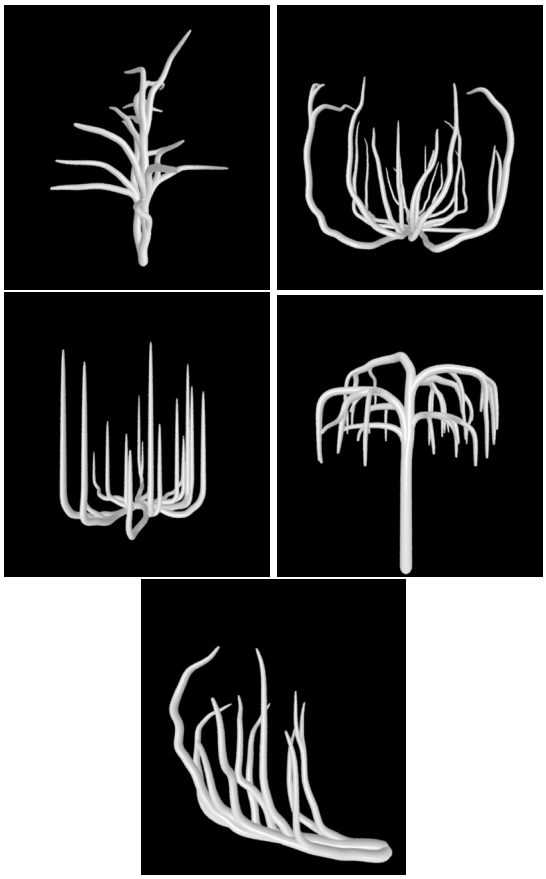

Figure 5: Some 3D objects made with different spatial distributions of edge weights.

We can vary the structure of the objects merely by creating objects with more endpoints. Since the endpoints are randomly placed, and the greedy path traversal is the least expensive part of the algorithm, the more elaborate structures with more endpoints do not actually require more effort to create, either human or computational. Figure 6

Figure 6: Trees with few endpoints (left) or many endpoints (right).

shows the differences between structures as the number of endpoints varies: the structures on the left have 20 randomly placed endpoints, while those on the left have 36 (the same 20 plus 16 more).

The irregular, winding, forking branches of the structures can be made to resemble elm trees with little difficulty. In Figure 7 we show two examples of elm tree models, and in Figure 8 we show a third synthetic tree paired with a photograph of a real tree. The two structures have considerable in common, notably the long, curving individual branches, the lack of a single straight trunk, and the spreading of branches away from the base into a wide space above it.

These trees were created on a relatively low-resolution lattice (80^3), making compute times low; computing the structures required only about 4 seconds each. The trees were rendered in Pixie, with tree geometry created by placing a sphere at each path node. Sphere radii were determined using the tapering formulation of equation 2, with $\beta = 0.3$ used for the larger main branches and $\beta = 0.2$ used for the smaller branches.

Figure 8 shows a comparison between a photograph of an elm tree and one of our models.

Figure 9 shows the creation of a lightning structure. This lightning was made in two parts, using two separate 2D graphs with different random edge weights. The first graph contains the main branch and some additional side branches; the second graph contains only side branches. The two structures are composed to form an overall structure, which has richer detail than could be easily produced in a single graph. Also, the paths in the overall structure cross over, giving the impression of a 3D structure projected onto a plane. This same approach could be used to produce billboarded trees, if desired.

Figure 10 shows a comparison between our path planned lightning and the lightning simulation of Kim and Lin. Although Kim and Lin use a more sophisticated rendering technique to portray their lightning model, we contend that our lightweight system produces equally convincing lightning structure. Note that our system required less than half a minute of computer time to create the model shown, while Kim and Lin report simulation time of several minutes. This means that our system is almost an order of magnitude faster.

Figure 12 shows a comparison with real lightning, where we have taken the lightning structure shown in Figure 9 and composed it with a photograph of a stormy sky. The result is convincing; anecdotally, people shown the photographs side by side were only able to identify the synthetic image after studying the two carefully (the main difference is in the halo around the main branch, largely absent in our rendering, and not present in Kim and Lin's either).

However, the main structural characteristics of lightning are represented in our final image.

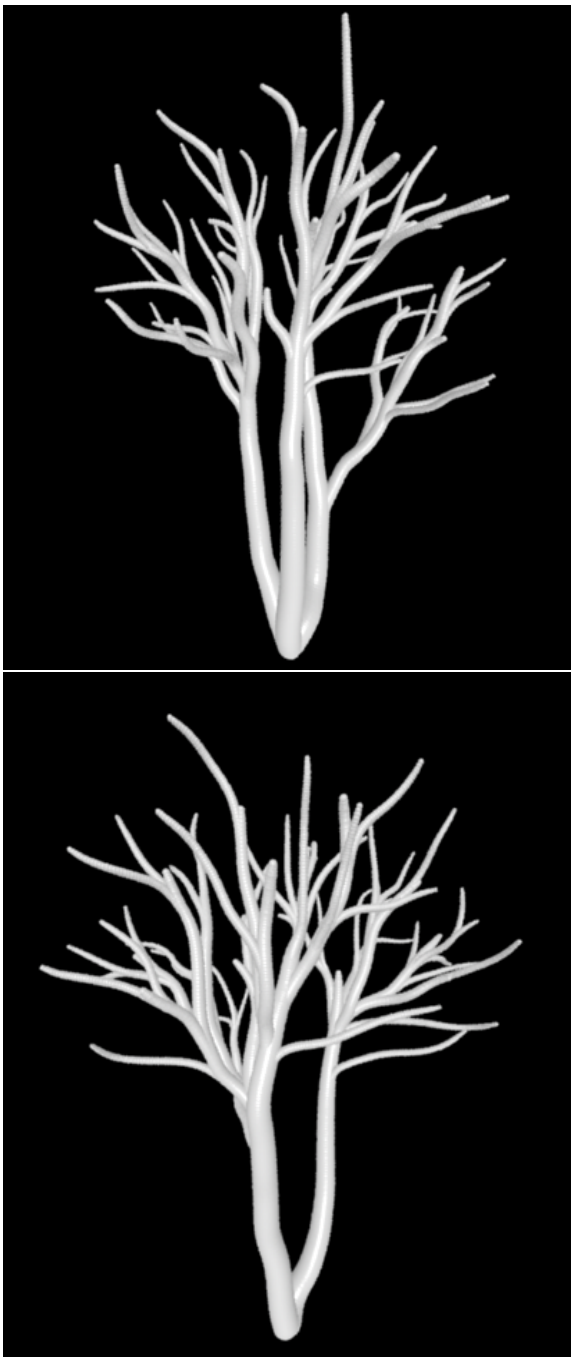

Figure 7: Some synthetic elm trees from different random configurations (endpoint placements and edge weights).

Figure 8: Comparison between synthetic and real elm tree.

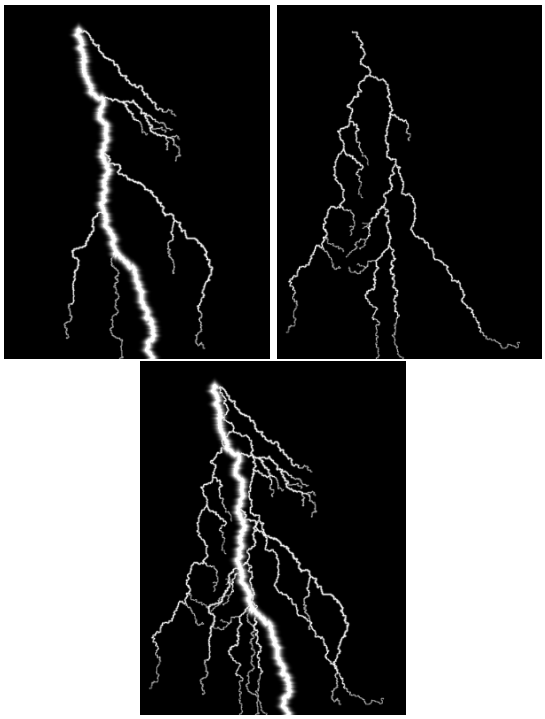

Figure 9: Creation of lightning structure. Above left: the main branch and some forks; Above right: additional minor forks. Below: the completed lightning made by composing the two pieces.

Figure 10: Comparison of lightning models. Left: path planned lightning; right: physically based simulation by Kim and Lin.

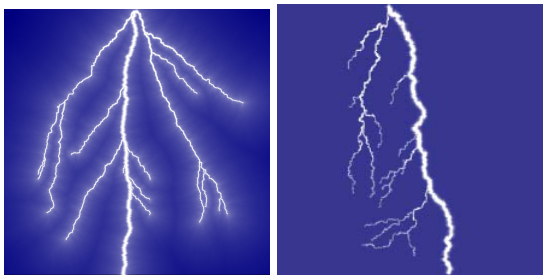

Figure 11: Left: lightning produced by 2007 technique. Right: lightning from our extended technique.

Figure 12: Above: Path-planned lightning composited with a photograph of a stormy sky. Below: photograph of lightning.

There is a single main lightning stroke, with multiple side branches. The main branch is thicker and has a fairly constant width, while the side branches taper visibly. The lightning is forked, both in having side branches leaving the main branch and in having side branches split from one another. Lastly, owing to our use of multiple (as few as two, in the example shown) parallel 2D graphs stacked together, we can present the strong illusion of a 3D effect.

The constructive path planning method has both advantages and disadvantages. The best aspect of the approach is the speed with which coherent structures can be generated, using a minimum (or no) user input. The main disadvantage is the large memory requirement for storing the graph. In this paper, we have shown how spatially organized adjustments to the edge weights can generate various models, and we have shown how to employ constructive path planning to create convincing elm tree models and lightning images. The control afforded by the path planning approach might be useful for some applications; for example, we might want lightning to strike a particular object in the virtual world, and we can place the path planned destination on the desired target.

5 Conclusion

This paper further explores the capabilities of path planning for model creation. We have shown how constructive path planning can be used to create plausible models of natural objects, specifically lightning and elm trees. The main structural characteristics of these objects can be captured in the path planning framework. Further, we showed how spatially varying the edge weights can produce different organic-looking structures, resembling real-world plant structures or fantastical creatures.

Future work can include applying the method to additional natural phenomena, such as cacti, and increasing the level of user control. One approach would be to permit a user to sketch an object; edge weights could be reduced in the vicinity of the sketch lines, guiding paths towards them.

Another possibility for future work would be to combine grammars and path planning. Grammars could be used for endpoint distribution, while path planning could still be used to obtain the branches themselves. Users could still adjust the endpoints if desired, but would not be required to, and a suitable grammar might be a good choice for placing endpoints, rather than placing them according to a statistical distribution as we presently do.

Acknowledgements

Thanks to Jeremy Long and the rest of the IMG group for valuable discussions surrounding path planning for natural phenomena. This work was supported in part by NSERC RGPIN 299070-04.

REFERENCES

Ball, P. (2004). *The Self-Made Tapestry: Pattern Formation in Nature*. Oxford University Press.

Deussen, O., Hanrahan, P., Lintermann, B., Mech, R., Pharr, M., and Prusinkiewicz, P. (1998). Realistic modeling and rendering of plant ecosystems. In *Proceedings of SIGGRAPH 1998*, pages 275–286.

Dijkstra, E. W. (1959). A note on two problems

in connexion with graphs. *Numerische Mathematik*, 1(1):269–271.

Ebert, D. S., Musgrave, F. K., Peachey, D., Perlin, K., and Worley, S. (2003). *Texturing and Modeling: A Procedural Approach*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Kim, T. and Lin, M. C. (2004). Physically based animation and rendering of lightning. In *Pacific Conference on Computer Graphics and Applications 2004*, pages 267–275.

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

Mech, R. and Prusinkiewicz, P. (1996). Visual models of plants interacting with their environment. In *Proceedings of SIGGRAPH 1996*, pages 397–410, New York, NY, USA. ACM Press.

Neubert, B., Franken, T., and Deussen, O. (2007). Approximate image-based tree modeling using particle flows. *ACM Trans. Graph.*, 26(3):88.

Prusinkiewicz, P., James, M., and Mech, R. (1994). Synthetic topiary. In *Proceedings of SIGGRAPH 1994*, volume 28, pages 351–358.

Reed, T. and Wyvill, B. (1994). Visual simulation of lightning. In *Proceedings of SIGGRAPH 1994*, pages 359–364, New York, NY, USA. ACM.

Tan, P., Zeng, G., Wang, J., Kang, S. B., and Quan, L. (2007). Image-based tree modeling. *ACM Trans. Graph.*, 26(3):87.

Witten, T. and Sander, L. (1981). Diffusion-limited aggregation, a kinetic critical phenomenon. *Physical Review Letters*, 47(19):1400–1403.

Xu, L. and Mould, D. (2007). Modeling dendritic shapes - using path planning. In Braz, J., Vázquez, P.-P., and Pereira, J. M., editors, *GRAPP (GM/R)*, pages 29–36. INSTICC - Institute for Systems and Technologies of Information, Control and Communication.