The Visual Computer manuscript No.
(will be inserted by the editor)

Jeremy Long · David Mould

Dendritic Stylization

the date of receipt and acceptance should be inserted later

Abstract Dendritic or branching structures are commonly seen in natural phenomena such as lightning, cracks, and vegetal growth. They are also often used for artistic or decorative purposes. We present a new procedural method for modeling dendritic structures based on a path planning approach. Our method includes the use of a partial non-scalar distance metric that gives us powerful and responsive control over the evolving dendritic structure. We demonstrate the effectiveness of our approach by creating dendritic stylizations of input images. We also show how our approach can be used to model more complex dendritic structures, such as trees; our algorithm allows us to create pareidolia effects, where an image is embedded within the branches of the tree.

Keywords Non-Photorealistic Rendering · Modeling · Dendritic Structures · Tree Modeling · Pareidolia

1 Introduction

A dendritic or branching structure is found in many objects that appear in nature, such as lightning, frost, cracks, and many forms of vegetation. Dendritic structures have also long been used for ornamental and decorative purposes; illuminations are an example of a traditional medium that employs vines and other floral material to create an artistic effect. In addition to decorating letters and ceramics, such branches are also sometimes used to make aesthetic patterns [36] or to convey an image [20,37].

Effectively modeling dendritic structures remains a challenge for computer graphics researchers. The results generated by existing procedural methods, while at times compelling, are notoriously difficult to control and expensive to compute. We present a new procedural technique for modeling dendritic structures using path planning. The use of an alternate distance metric for the path planning algorithm allows for effective control over the evolving structure, and the long winding paths that result are suitable for artistic effects.

We apply our technique to dendritic stylization, where an input image can be embedded into a dendritic structure and then rendered using different sorts of dendritic metaphors. We also show how our modeling method can easily be modified to generate trees whose branches convey the structure of an input image. In addition to the above, we believe our method holds potential for other artistic media where exacting precision is required, such as line art, ceramics, and decorative mazes, to name only a few.

Our method rests on an adaptation of a non-scalar distance metric first proposed by Pai and Reissel [19]. Our variant is faster and less memory-intensive, while still being well suited to generating dendritic structures. We show how our method can create stylized dendrites from input images, where the structure of the image is conveyed by the dendrite: a novel form of non-photorealistic rendering. Furthermore, our method is versatile, and can generate different sorts of dendritic structures, such as vines and lightning, without requiring significant changes to the modeling process. Finally, we show how to use our method to create stylized trees partly shaped by input images, so that both the tree and the image's subject are visible to a viewer. Naturally occurring forms of this kind of illusion are sometimes called pareidolia.

In the next section, we discuss related and previous work in the relevant areas of dendritic structures, path planning, halftoning, and tree modeling. Section three will introduce our method for modeling dendritic structures using path planning, in addition to showing how we can use the partial non-scalar distance metric to achieve superior control characteristics. Section four describes our process for dendritic stylization. We extend

Jeremy Long
Department of Computer Science
University of Victoria
E-mail: jsl@csc.uvic.ca

David Mould
Department of Computer Science
University of Saskatchewan
E-mail: mould@cs.usask.ca

our modeling framework in section five to model non-photorealistic trees such as the ones that are commonly seen in landscape paintings. We discuss conclusions and future work in section 6.

2 Related Work

2.1 Dendritic Structures

Initial work on the procedural modeling of dendritic structures drew much inspiration from research into fractals. Fractals have often been used to model complex natural phenomena, such as trees and snowflakes [6]. Because of their success in this area, many attempts were made to adapt fractal generation methods for use in modeling dendritic structures. This led to the use of Lindenmeyer-Systems (L-Systems) [23, 24].

An L-System can be described as a replacement grammar that evolves initial axioms using a set of production rules over discrete derivation steps in order to create recursive structures [23]. Later, context-sensitive L-Systems [14] were developed, which use the spatial position of the branches to assist in determining the production rule that should be used. However, control by designing production rules remains unintuitive, as the effect of small changes to the production rules can be unpredictable.

The growth of dendritic structures has also been studied within the context of ornamentation. Wong et al. [36] introduced their own grammar for creating ornamentation. Though similar in concept to L-systems in that it involves elements and production rules, this method features the serial processing of rules as opposed to the parallel processing seen in L-systems.

Another direction involves the use of physically based processes to generate dendritic structures [1]. These processes range from fluid flow models to the aggregation of dust particles, and have shown some promise. Several of these techniques have been used in combination with the simulation of natural processes to model natural structures [12], such as the complex formation of ice crystals [10], the growth of lichen [7], and for animating lightning [11]. While at times compelling, these methods tend to suffer from a lack of control. Furthermore, they do not tend to be very flexible, and it is difficult to use them to model objects aside from the ones for which they were specifically designed.

A classic method for procedurally generating dendritic structures is diffusion-limited aggregation (DLA), which simulates the progress of dust particles floating through the air [35, 2, 3]. Particles are released a long distance from the structure and allowed to perform a random walk until they reach the structure, where they attach themselves to it. Output from this method resembles branching structures one might expect to see in the real world. However, DLA is costly to compute and difficult to control.

Path planning is the process of finding a least cost traversal through a weighted graph [34]. It is often seen in the contexts of artificial intelligence and computational geometry [8]. By originating all the paths from a single seed point in the graph and finding paths to an arbitrary array of endpoints, we are able to obtain a dendritic structure. The result is also fast to compute, and more importantly, offers powerful control mechanisms in the forms of the edge costs and the placement of the endpoints.

The distance metric that we use for generating our dendritic structures was first introduced by Pai and Reissell [19] as a means of improving path planning for robots. Their non-scalar distance metric attempts to model the roughness of terrain, under the premise that robots are safer by avoiding peak edge costs, even if it means taking an alternate route with a higher total cost. This results in longer, more winding paths. As we discuss later, this property makes the non-scalar paths better suited for creating artistic effects than those generated by the conventional cumulative distance metric.

2.2 Dendritic Stylization

We demonstrate the effectiveness of our method for modeling dendritic structures by applying it to the task of dendritic stylization, where the growth of a dendrite is guided by an image. There is considerable precedent in using images to guide the growth of various unusual yet artistic structures. Mould used an input image to guide the evolution of crack patterns [16]. Pedersen and Singh [20] and Xu and Kaplan [37] shape labyrinthine structures around a source image. Our approach uses a different method for growing the structure, but we share the goal of representing an arbitrary input image using an unusual set of primitives.

The results of our dendritic stylization are similar in some aspects to artistic halftoning, where the placement of artistic primitives such as stipples [27], brush strokes [28], or pen and ink [26] is intended to convey the input image. We can consider the paths generated by our algorithm as our primitives.

The primary goal of artistic halftoning is to achieve a compelling balance between reproducing the artistic style in question and preserving salient features in the input image. Preserving features such as gradient edges has traditionally proven difficult. Some methods circumvent this complication by allowing users to highlight important features in the input [26]. Streit and Buchanan proposed the notion of a generalized importance map that could be used to place primitives procedurally where they are most needed to convey the essence of the input image [29]. This concept can easily be customized for our stylization approach in order to accurately cap-

ture important image features, including gradient edges, without user interaction.

This is not the first time that path planning has been used to search for gradient edges. Mortensen and Barrett employed path planning to locate features in an image using edge boundaries [15]. Although this is similar to our method, the cumulative distance metric employed in their approach tends to take short cuts over high cost areas in order to decrease the total length of the path, making it more difficult to preserve features that may be important for artistic effects. The non-scalar distance metric does not suffer from this weakness.

2.3 Image-Guided Tree Modeling

The procedural generation of trees has been addressed several times by computer graphics researchers. Most of the approaches that have been taken are intended to model photorealistic trees. Our method for modeling dendritic structures shows promise for growing non-photorealistic trees. One of our main interests is in creating pareidolia effects, where images (usually faces) are seen within a natural structure such as a tree. The framework that we have developed seems effective at creating these sorts of effects.

Several forms of L-Systems have been used to create visually realistic models of plants, including trees [36, 23, 24, 21]. More recent work in this area tends to focus on simulating the interaction between the plants and their environment in order to use information from these natural processes during the growth of the plants, and their placement in a scene [21].

Some graphics researchers have been interested in conveying the appearance of trees without worrying about the natural processes that form them. A few of these approaches have created tree models based on geometric considerations [33] or particle systems [25]. These methods can produce results that resemble real trees reasonably well, at the expense of control and flexibility. Because these methods are designed to model individual types of trees, they require almost completely new sets of instructions in order to create different types of trees or even irregular characteristics within the same type of tree. This lack of control makes these methods unsuitable for containing hidden images.

There has also been some interest in guiding the construction of three dimensional tree models using input images [18, 30]. The resulting models can be quite faithful to the input image. However, these methods are unable to model trees that do not appear in photographs, limiting their flexibility. These methods cannot easily be applied to the task of creating a tree from any arbitrary input image or even from scratch.

Our approach is not intended to model the natural processes involved with vegetal growth, nor is it expected to create photorealistic results. We are more interested in achieving the sorts of artistic effects that can be seen in landscape painting, where highly stylized trees and vegetation are often present.

3 Modeling Dendritic Structures with Path Planning

Modeling dendritic structures for artistic effects requires first and foremost precise control over the evolving structure. There need to be mechanisms in place to guide how the dendrites grow so that the intended artistic effect is not compromised. In an illuminated manuscript, for example, the decorative vines should not obscure any of the text. Furthermore, the control handles need to be responsive enough that satisfactory results can arise from minimal input, such as information inferred from an input image.

The second criterion for producing artistic effects is the flexibility of the approach. Different dendritic structures tend to display different characteristics, as can be seen when comparing crack patterns with tree branches, for example. With this in mind, we are looking for compelling results that are flexible enough to encompass a wide range of dendritic structures through only small changes to the modeling process.

Existing procedural methods for modeling dendritic structures fail in providing sufficient control and flexibility. We demonstrate in the next section that our path planning algorithm does not suffer from these deficiencies, and can be computed quickly.

3.1 Path Planning Algorithm

Path planning [34] is the problem of finding the least-cost path between two nodes in a weighted graph. We can create a dendrite – a branching structure without loops – by computing paths from a single seed point to multiple endpoints distributed through the graph. Our graphs are regular eight-connected lattices in this paper, though it should be noted that this algorithm can be performed on graphs with arbitrary connectivity. An example of a dendrite is shown on the left of figure 1.

Dijkstra's algorithm [34] computes shortest distances from a seed point to the other nodes in the graph. Each node N has an upper bound distance to the seed, written $E(N)$, and an actual distance, written $A(N)$. Dijkstra's algorithm uses a priority queue to store a set of "frontier" nodes: nodes with unknown actual distance, but which lie adjacent to nodes with known actual distances. The priority queue is ordered by increasing upper bound distance, with smallest distances at the top; it is often implemented as a heap. Initially, the frontier contains only the seed node, at upper bound distance zero; the graph is initialized with all nodes' upper bound distances set to infinity. Then, the following is repeatedly performed.

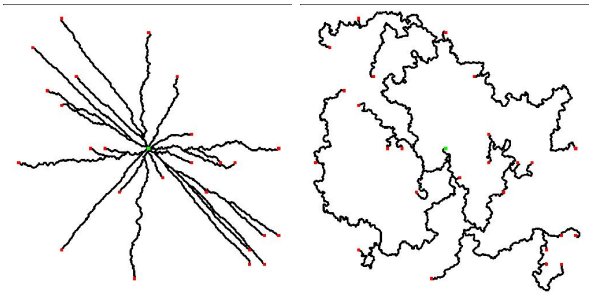


Fig. 1 Modeling a dendritic structure using path planning with the cumulative distance metric (left) and a dendrite modeled using the partial non-scalar distance metric with an identical graph and set of endpoints (right).

First, the top node, say T, is removed from the frontier, and its actual distance set to its upper bound distance. Second, for each node N_i adjacent to T, a new upper bound is computed $E_{new}(N_i)$ as $A(T)$ plus the cost of the edge linking N_i and T; those nodes whose old upper bounds exceeds E_{new} have their upper bound replaced by the new upper bound and are added to the frontier. Dijkstra's algorithm terminates when the frontier is empty, at which point all nodes in the graph have known distance values.

To find the distances to n nodes, Dijkstra's algorithm makes $O(n)$ removal operations. Maintaining the heap has a cost of $O(\log m)$ per interaction, where m is the number of nodes in the frontier; thus, the overall cost is $O(n \log m)$. Typically, $m << n$; that is, only a small fraction of the graph is in the heap at any one time.

Once Dijkstra's algorithm has finished, we can turn to generating the paths. The path planning approach traces backwards from each endpoint towards the single seed point of the structure. We can accelerate this calculation by calculating the paths only until we reach a part of the structure, at which point we already know the shortest cost path back to the seed point since we calculated it when pathing back from a previous endpoint.

Path planning offers favorable control characteristics. We can achieve local control over the paths by manipulating the edge weights, while choosing the endpoints gives us global control. Thus, we can control both where the branches go and how they get there, and we can generate them quickly without requiring any human intervention.

3.2 The Non-Scalar Distance Metric

While path planning offers good control for artistic effects, the cumulative distance metric that is traditionally used to determine the shortest path has some adverse consequences for our applications, as shown in figure 2. In particular, the paths generated using the traditional distance metric will tend to take short cuts over high cost areas in order to avoid long, winding routes. Not only is this bad from a control standpoint, as such short cuts can ruin details that are important for artistic effects, but we also believe that such winding paths more closely resemble ornamental dendrites traditionally used, such as in illumination.

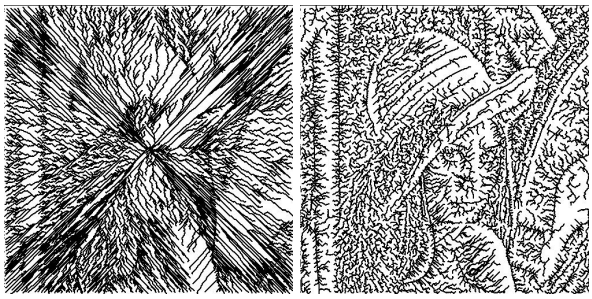


Fig. 2 A dendritic version of Lena (from figure 3) using the cumulative distance metric (left) and the partial non-scalar distance metric (right).

We can obtain long and winding paths using a non-scalar distance metric first introduced in the field of robotics [19]. This non-scalar metric can be implemented by storing the list of edges traversed along a prospective path in the order of decreasing cost. When two prospective paths are being compared, corresponding elements from the beginning of each list of edges are selected for comparison. As soon as one edge value differs from the other, the path with the smaller value is determined to be the shorter of the two. Otherwise, we move on to compare the next element of each list. In essence, this metric can be seen as minimizing the maximum edge cost in the path.

Sorting the edges on a path adds some computational overhead, but the main drawback is the memory requirement. Memory tends to be at a premium during path planning algorithms, and this is no exception. Storing a list of edge costs needed for each potential path is a fairly prohibitive cost. Fortunately, we can achieve results similar to the full non-scalar distance metric using only a partial version.

The partial non-scalar distance metric implemented in this paper keeps track only of the maximum edge cost along the path. Comparisons between paths are first made by comparing these two maximal cost values. In the case of a tie, we then rely on the cumulative distance metric to determine the shorter path. This is shown in the pseudocode of figure 4. Figure 1 demonstrates that even this reduced version of the non-scalar distance metric produces markedly different results from the cumulative distance metric. Using the partial non-scalar distance metric instead of the traditional cumulative dis-

Fig. 3 Input images (from top left) Lena, Eric, a cat, a manor, pottery, a sailboat, some statues and a mandrill.

tance metric has no impact on the complexity of the algorithm.

```
partialNonscalarCompare(Path p1, Path p2): Path
{
 if (p1.maxEdgeCost == p2.maxEdgeCost)
 {
  return the path with the lower cumulative distance
 }
 else
 {
  return the path with the lower maximum edge cost
 }
}
```

Fig. 4 Pseudocode for the comparison of two paths using the partial non-scalar distance metric.

3.3 Controlling the Dendritic Growth

Controlling the growth of the dendritic structure is of great importance. As mentioned previously, we believe that path planning offers responsive control handles. In particular, we wish to be able to control the evolving structure using an input image. Though our technique can be used on arbitrary graphs, including meshes, artistic effects are commonly applied as image filters. Here, we propose to transform an input image into a dendritic version that can simulate some artistic style.

Local control over the evolving dendritic structure can be obtained by setting the edge costs according to some parameters drawn from the same location in the input image. In most cases, a combination of the gradient and intensity values is helpful for preserving the structure of the image, though the exact combination is dependent on the effect being sought. The endpoints will determine the large-scale structure of the dendrite, and should be carefully chosen.

Figure 2 shows an input image and dendritic structures created using the conventional cumulative distance metric and the partial non-scalar distance metric proposed in this paper. The cumulative distance metric tends to discourage longer branches, leading to cases where the branches take short cuts over high cost areas. Meanwhile, the partial non-scalar distance metric tends to lead to longer branches that are better able to fill space and more willing to wind around high cost areas instead of cutting through them.

4 Dendritic Stylization

In this section, we apply our framework to the task of dendritic stylization. Although our process is similar to artistic halftoning in some respects, we do not seek to match the tones in an image. Instead, our dendrites are able to create compelling results by emphasizing intensity edges and contrasts that are present in the input image.

Our dendritic stylization process contains several steps. First, we construct an importance map using the gradient magnitude and intensity from the input image. Next, this importance map is used to select edge weights for our graph and to assist in the placement of endpoints. This fulfills all the preconditions for our path planning

algorithm, and we employ it to create the dendritic structure. Finally, we add importance emphasis to the resulting structure. Once these modeling steps are complete, the resulting structure can be rendered using many different dendritic metaphors.

4.1 The Importance Map

Several procedural halftoning approaches use the notion of an importance map to distribute strokes or primitives throughout the input image [29]. We use a similar approach for our dendritic stylization. We build each pixel of our importance map as a combination of the intensity and the gradient magnitude at the corresponding pixel in the input image.

Areas of the image with a higher importance will be more likely to be visited by the dendritic structure. In order to preserve color contrasts, we ascribe high importance to dark pixels in the input image. We can preserve intensity edges in the input image by giving high importance to areas of high gradient magnitude. In the results we show, we weighted the importance contribution from gradient magnitude three times as heavily as that from intensity.

The importance map provides us with weights for the lattice nodes. The weights in the lattice also include a small constant and a random contribution. The dendritic stylization examples shown in this paper use very small random contributions, which suffice to provide variation without damaging the structure of the image. Edge weights are derived from node weights: the weight of an edge is the average of the weights of the nodes it connects.

We already have enough information to determine that our non-scalar metric responds more appropriately to the control handles inherent in the path planning approach. The weight formulation above was used in our implementation of Dijkstra's algorithm to produce the visualization shown in figure 5. The lighter areas represent nodes that are visited earlier in the algorithm, and can be reached at a lower cost. It is easy to see that the cumulative distance metric is ruled by proximity to the seed point, at the expense of the image structure. On the other hand, the partial non-scalar distance metric is willing to take long, winding paths to reach its destination without crossing high cost areas. This allows it to better preserve the features in the input image.

Feature preservation is further demonstrated by the dendrites themselves. Figure 2 shows an input image and dendritic structures created using the conventional cumulative distance metric and our partial non-scalar distance metric. Because the partial non-scalar distance metric does not penalize long paths, its paths are better able to follow features in the image. As a result, the structure of the image is better captured by our approach.

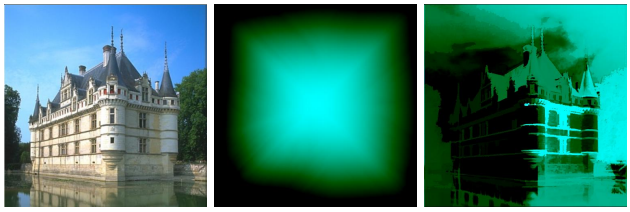


Fig. 5 Source image (left) and contours from Dijkstra's algorithm using the cumulative distance metric (center) and the non-scalar distance metric (right).

4.2 Endpoint Placement

Large-scale control over the evolving dendritic structure can be established by carefully selecting endpoints from the input image. We want to select points so that the resulting dendrite can convey the image; a heuristic for placing the endpoints puts more endpoints in regions of higher interest (guaranteeing that paths pass into or through that region) and places well-spaced endpoints (so that we obtain a distinct path for each endpoint).

Both of these characteristics are also sought for computer-created stipple drawings. Accordingly, we used a recent stippling method for selecting our endpoints [17]. We use our importance map to help us choose the endpoints during this algorithm.

The result of this approach is a set of endpoints that can be used for stippling [27,17], assuming the chosen threshold is low enough. An example of the endpoints chosen is shown in figure 6. As can be seen, the dendrite produced from these endpoints does a reasonable job of capturing the essence of the input image.

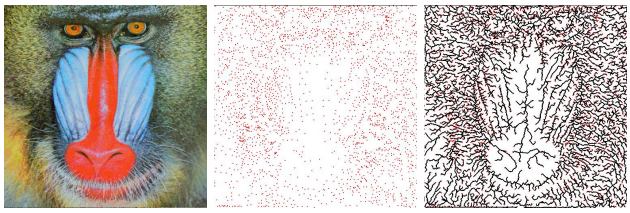


Fig. 6 Source image (left), the set of endpoints (center) and the resulting dendritic structure (right).

4.3 Importance Emphasis

To enhance the effect of our dendritic stylization, we also wish to emphasize certain parts of the image based on their gradient and intensity values. Most procedural artistic halftoning approaches apply some small changes

to their primitives in order to accomplish this, ranging from angling strokes to increasing the size of stipples. Our primitive is a dendritic path, which naturally lends itself to changes in width based on the importance of a region. In this way, we can have thicker branches passing along more critical sections in the input image.

We compute a thickness for each node in the structure using a rolling average of importance along each branch. By taking into account the values of several neighbors on both sides, we are able to achieve a smoother transition between the thickness levels. By allowing this window to overshoot the end of the branches, natural thinning of the dendrites occurs near their tips.

Fig. 7 Source image (left) and a dendritic version (center) and a dendritic version with importance emphasis (right).

Figure 7 shows the results of adding importance emphasis to our dendritic structure. The varying widths of the branches provide visual cues of the structure, highlighting the object's silhouette, in this example. The thickness values can be stored along with information about the dendritic structure for further rendering purposes.

4.4 Stylization Results and Evaluation

Figure 8 shows the results of applying our dendritic stylization process, to the input images shown in figure 3. Our method is effective on a variety of images, able to convey the structure of a human face, as in the Lena and Eric images, or to depict even subtle features, such as the shape of the cat's body (weakly contrasted in the original image).

Similarly, we note that our dendrites will emphasize features that are barely discernible in the input image. Some of these fine details include the cloth folds in the image of Eric in figure 8, or the fabric of the sail. This is because the partial non-scalar distance metric can coerce the paths to avoid locally expensive edges, forcing them to take longer routes along even weak intensity edges. This allows us to easily capture small details, assuming there are not more important features in the area to attract the paths.

Based on the results generated for this paper, we can make several observations insofar as how well different types of images will weather the dendrification process. In general, our process works best when there is high contrast between different areas of the image. It also succeeds in capturing high frequency detail, as seen with the fur in the mandrill image in figure 8 and the feathers on Lena's hat. It does not perform quite as well in clearly representing smooth intensity gradients, or at conveying absolute intensity, as demonstrated in the sky above the sailboat in figure 8. We do not consider this to be a significant drawback, as strict tone matching was not one of our priorities.

Table 1 shows the time our method took to generate some of the results presented in this paper. The number of seconds required to perform our dendritic stylization is largely dependent on the threshold used for selecting endpoints. For the sake of brevity, we show only a subset of our timing results, as all the images tended to fall within the time range of those listed.

4.5 Dendritic Metaphors

We briefly consider different dendritic metaphors that can be achieved through relatively small changes to our modeling process. The left image of figure 9 shows that we can express our dendritic structures as stylized vines that you might find in some forms of ornamentation by smoothing the branches and changing the rendering procedure. The right image of figure 9 shows that putting more weight on upwards and sideways edges can encourage the branches to grow downwards like some form of stylized lightning. We believe that most of the challenge in expanding our set of dendritic metaphors would lie in creating compelling renderings of the results.

Fig. 9 A dendritic structure using a stylized vine metaphor (left) and dendrites rendered using a lightning metaphor (right).

5 Image-Guided Tree Modeling

The preceding section shows that our modeling framework can easily be modified to create different sorts of

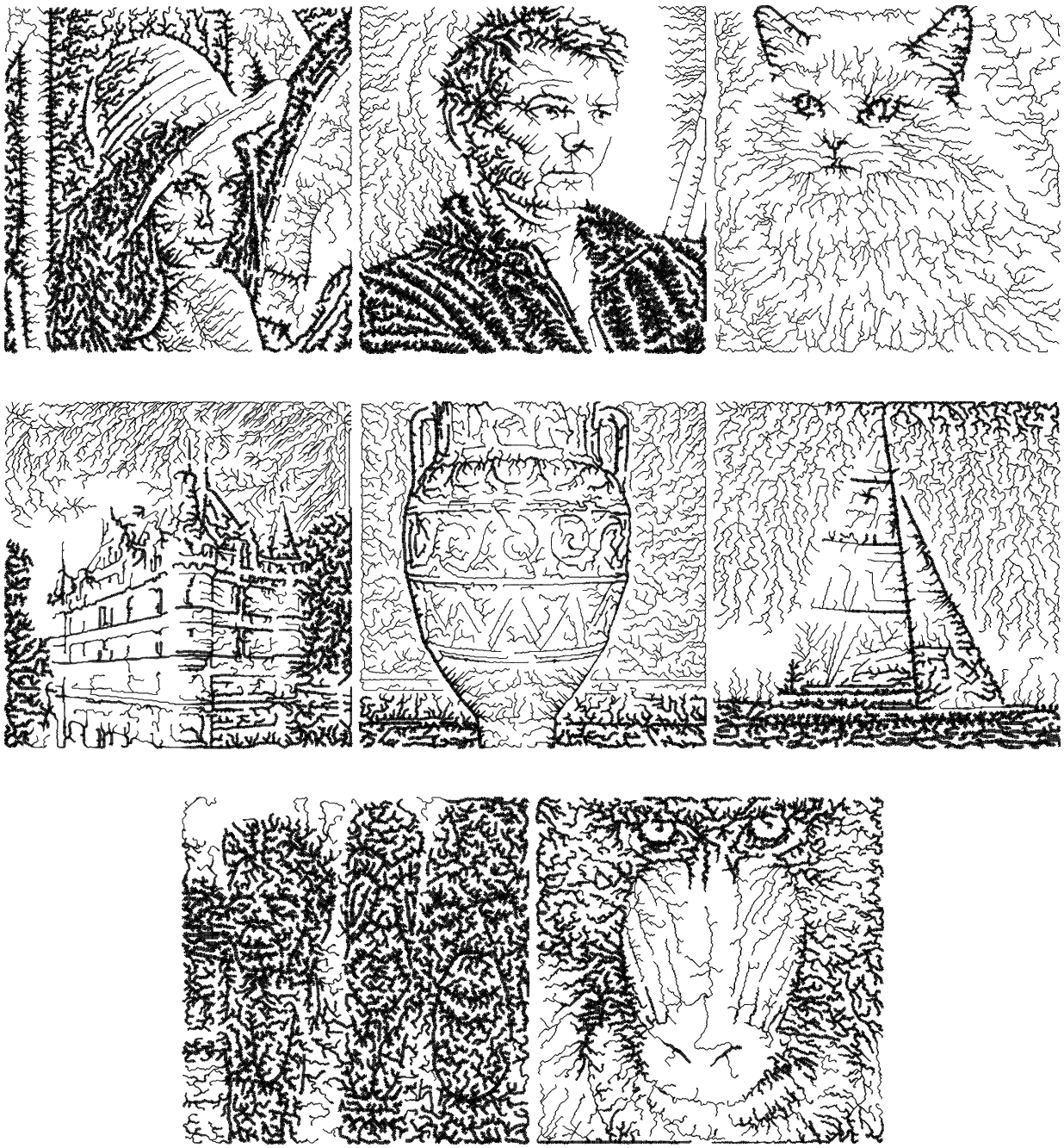


Fig. 8 Dendritic stylization (from top left) Lena, Eric, a cat, a manor, pottery, a sailboat, some statues and a mandrill.

Image	Endpoints	Timing (in secs)
Lena (Figure 8)	3838	530
Pottery (Figure 8)	2992	368
Sail Boat (Figure 8)	3554	546
Cat (Figure 8)	2019	237
Mandrill (Figure 8)	2715	446

Table 1 Timing Analysis

branching objects. Our approach is flexible enough to model more advanced structures, such as trees. This next section describes the changes that need to be made to our modeling framework in order to model non-photorealistic trees.

5.1 Growing Trees

Different kinds of trees have different kinds of branches. We are not at this point attempting to model all possible types of branches, but to generate plausible non-photorealistic trees. Figure 10 shows an example of the kind of painted trees that we are attempting to stylize and model.

Fig. 10 A painted oak tree.

We can observe that the tree in Figure 10 has two classes of structures: the trunk and primary branches tend to be more straight and regular, while the extremities are more gnarled. The former are more consistent with the paths generated using the cumulative distance metric, while the latter display properties more in tune with our partial non-scalar distance metric. We can accommodate both by combining the two metrics into a weighted function and changing the contribution of each metric depending on what type of branch we wish to generate. This requires us to transform the result from the partial non-scalar distance metric into a scalar value, which we can accomplish by taking a weighted sum of the maximum edge cost along the path and the total cost of the path itself. We have found that weighting the maximum edge cost a thousand times more heavily than the total cost of the path tends to produce compelling results.

We use an iterative approach to grow our trees, as shown in figure 11. The first iteration uses the cumulative distance metric, and the seed point is placed at the bottom of the graph. The seed is shown in green in the first image of figure 11. We then generate a sparse distribution of endpoints in order to grow the trunk and the primary branches. In the case shown in figure 11,

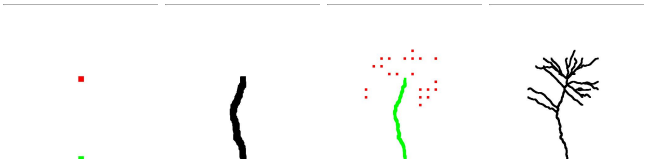


Fig. 11 The growth of one of our dendritic trees over two iterations. The first two images show the first iteration and the second pair show the second iteration.

we use only a single endpoint for the trunk. This point has been placed almost directly above the seed. Once the endpoints have been placed, we perform our path planning algorithm using the cumulative distance metric to obtain the results shown in the second image of figure 11.

Subsequent iterations use the entire existing structure as the seed. This is shown in the third image of figure 11. We confine a larger distribution of endpoints to a region surrounding the already generated structure. We can find the distance from every prospective endpoint to the structure by running the brushfire algorithm again using every point in the structure as a seed. We can then choose endpoints within the radius using rejection sampling. After each iteration, we reduce the radius, so that later iterations will generate shorter branches. Once we have picked all the endpoints, we can use our path planning approach to generate the branches. We can change the contribution from each distance metric between iterations of this algorithm in order to control the characteristics of the branches. The fourth image of figure 11 shows the result of the algorithm after the first two iterations.

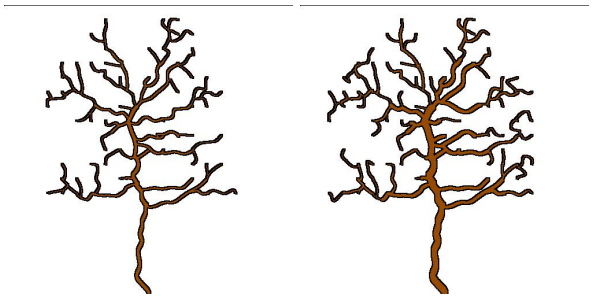


Fig. 12 A dendritic tree structure built using two iterations of our algorithm (left) and one built with three iterations (right).

A tree generated in this fashion is shown in figure 12. The branches were rendered using the same splines that were used in our vegetal stylization. Our structures resemble the desolate trees that sometimes appear in landscape paintings, reminiscent of the proverbial ‘haunted’

tree. We can see that the cumulative distance metric ensures that the trunk of this tree is fairly regular, while the subsidiary branches are more twisted due to the increased contribution of the partial non-scalar distance metric.

Once we have established the general structure of our trees, we can consider adding further details. One important factor for making the structure of trees more evident is the way the thickness of the branches vary. We can achieve this effect by increasing the thickness of the entire structure after each iteration of the algorithm. This means that the trunk and primary branches will remain the thickest, while the periphery will be made up of thin branches.

The right image of figure 12 shows the results of a dendritic tree modeled with iterative branch thickening. There are three thickness levels present in this image. The thickness is highest at the trunk, and decreases for each set of sub-branches.

The representation of trees that we have generated thus far works well in some cases, but our two-dimensional approach does not take self-occlusion into account. Real trees, and drawings of trees, inevitably display self-occlusion. We can imitate occlusion by applying our algorithm several times to the same trunk structure with different noise maps, then stacking the resulting images on top of one another. This gives the results shown in figure 13.

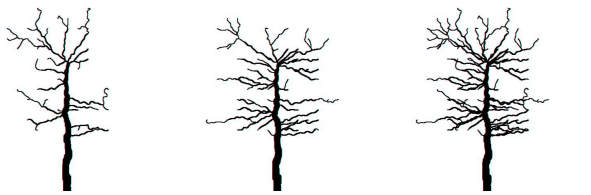


Fig. 13 By stacking together two dendritic trees (left and center), we are able to create a more detailed tree that includes occlusion (right).

5.2 Image-Guided Tree Modeling

Pareidolia can be defined as an effect where a natural structure conveys the illusion of a particular image. This effect has been observed in many sorts of natural phenomena, including clouds, rocks, and trees. A classic example, shown in the left image of figure 14 is the structure on the surface of Mars that resembles a human face. The right image of figure 14 shows that trees and vegetation can sometimes also create pareidolia. In this case, the bent trunk of the tree resembles a bowing person. Humans are adept at recognizing human faces and figures, making them the most common illusion.

Fig. 14 A section of the Mars landscape that resembles a human face (left) and a tree that resembles a bowing person (right).

Artists have shown some interest in creating pareidolia effects within their work. There is also a precedent for using dendritic structures to convey these hidden images. Tree branches have been used by Salvador Dali to form pareidolia [5], and they have also been employed in more recent work such as the poster for the 2007 film *Premonition* [22].

These sorts of images require considerable artistic skill to create artificially, as representing the image without compromising the plausibility of the tree branches used to form it is a daunting task. There has been little research into the possibility of procedurally generating pareidolia, although there is precedent for using input images to guide the modeling of various natural structures [16]. Our framework gives us the control handles we need to work on this problem.

In this case, we will use our principles of dendritic stylization in order to plant hidden images within our tree structures. At the same time, we want our trees to maintain a plausible appearance. This is a difficult line to walk.

We continue to build our dendritic trees over several iterations, as described above. In order to maintain the appearance of a tree, we elect to embed only a certain portion of the image into our tree structure, and allow the rest to be generated primarily in a random manner. The results of this process are shown in figure 15, where Lena's face is embedded within the branches of the tree.

We begin generating image-guided trees in the manner described in section 5.1. We employ a lattice with edge weights drawn from a uniform random distribution to generate the tree trunk during the first iteration of our algorithm. A new lattice is built for subsequent iterations of the algorithm using the gradient magnitude and intensity from an input image to set the edge weights, as described with regard to our dendritic stylization process. We increase the random contribution to the edge weights towards the left side of the image. As a result, the paths will be more inclined to follow the image structure on the right side of the image than the left.

We choose the endpoints during subsequent iterations using the method that we employed for dendritic styliza-

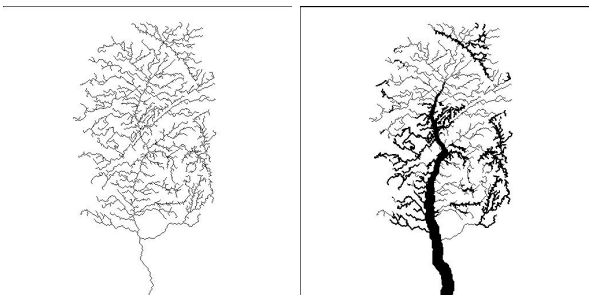


Fig. 15 A dendritic tree with Lena's image embedded within its branches (left) and the same tree with importance thickness (right).

Fig. 16 A dendritic tree containing the image of Eric in its branches with importance thickness (left) and the same tree using branch thickness (right).

tion with two notable modifications. The first difference is that we allow the endpoint selection algorithm to treat every point on the structure as a seed point. The other modification is that we keep track of the number of edges needed to reach each node. This allows us to ensure that branches end within an arbitrary radius of the structure. Endpoints that fall outside the chosen radius are still considered seeds during the endpoint selection algorithm, but they are not treated as endpoints when we generate our paths.

In the examples we show in this paper, we decided that only endpoints on the right side of the structure could be chosen in this manner. We then chose approximately half as many endpoints from the left side of the graph, and their coordinates were determined by random rejection sampling. We then introduced a few random endpoints into the right side of the image in order to add some variation to the tree.

Artistic hidden images also use different features within the trees to convey the shape of the image, including the thickness of branches or the foliage. This is demonstrated in the *Premonition* poster [22]. We would also like to take advantage of this within our implementation. The right image of figure 15 shows an example where the thickness of the branches is based on their importance in the source image, as was done for dendritic stylization. This approach does a good job of highlighting the image, but it detracts from the plausibility of the tree.

An alternate approach to thickness is to base it on the distance from the trunk of the tree. We refer to the results given by this metric as branch thickness, which is shown in the second image of figure 16. This approach is better at preserving the appearance of a tree, although the hidden image is compromised in the process.

5.3 Tree Rendering

With this modeling process in place, we now turn to the rendering of our image-guided trees. We decided to use image analogies [9] to get a reasonable painterly rendering from our tree models. The results from applying image analogies are shown in figure 17.

Fig. 17 Trees containing the images of Lena (left) and Eric (right) in a painterly style provided by image analogies.

5.4 Results and Evaluation

Figure 18 shows one of the trees that we generated composited into a painted image. We chose a winter landscape painting by American impressionist John Twachtman, since we thought it fit with the skeletal appearance of our tree. We decided to place our tree far in the background so that a viewer would not immediately be able to discern that the style of our painted tree is different from the style of the landscape. As shown here, our tree does not look out of place in this background. In fact, some viewers even assumed that this image was showing a real world example of pareidolia.

We use a landscape painting by American portrait painter John Neagle as the background in figure 19. We then composited a tree shaped to resemble Eric's face

Fig. 18 A painted landscape by John Twachtman with one of our trees composited into the background.

Fig. 19 A painted landscape by John Neagle with one of our trees composited into the background.

(from figure 17) into this backdrop. Once again, we have reduced the scale of our tree in order to diminish the differences in painting style between our tree and the landscape.

In general, we have found that our algorithm produces the most compelling results when portraits are used as the input images. The human visual system is adept at detecting human features, and as a result they make a good basis for pareidolia illusions. We also find that our algorithm functions well when the trunk of the tree can be centered near the most prominent features of the input image. In the results we showed, we place the trunk near the image center; most images and photographs are centred around the subject. Finding salient features in an image remains an open problem.

6 Conclusion and Future Work

The main contribution of this paper is the introduction of our procedural method for modeling dendritic structures. Our method uses path planning to achieve superior control and flexibility, and thus avoids the weaknesses exhibited by previous procedural methods. The control characteristics of our approach are further enhanced by the partial non-scalar distance metric that we use, which encourages the branches to obey the control handles more rigorously, and are conducive to creating compelling artistic effects. The effectiveness of our approach was demonstrated when we applied it to dendritic stylization and image-guided tree modeling.

One avenue for future work would be to look into a wider variety of non-photorealistic metaphors for our dendritic structures. We have already done some work on likening artistic dendrites to trees and vines, but we believe that our dendrites could be used to model lightning, frost and other branching structures with only minor changes to the modeling process, demonstrating the versatility of our approach. This was suggested briefly in figure 9. We believe that the main challenge in this regard would be to render the dendritic structures in a manner appropriate to their metaphor.

We also believe that more research could be done with regards to hidden images. Our approach is only intended to deal with the case where the image is being represented within a cluster of tree branches. Artists have hidden images in many other sorts of stylized objects, ranging from the shadows of landscapes to the shapes of clouds. Planting images within these features remains a difficult problem, and dealing with some of these cases might require a completely different approach from the one we used in this paper.

Another area of interest would be in improving the thickness measures we have used for our image-guided trees. We believe that a better balance between importance and branch thickness would allow us to better preserve the features of the input image while maintaining some of the branch structure one would expect in a tree.

Acknowledgements We will add some acknowledgements to the final version of the paper.

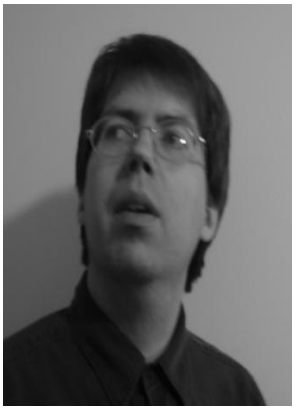
References

1. Ball, P.: The Self-Made Tapestry: Pattern Formation in Nature. Oxford University Press (2001)
2. Brady, R., Ball, R.: Fractal growth of copper electrodeposits. Nature **309**, 225 (1984)
3. Bunde, A., Havlin, S.: Fractals and disordered systems. Springer-Verlag New York, Inc., Secaucus, NJ, USA (1995)
4. Cohen, M.F., Shade, J., Hiller, S., Deussen, O.: Wang tiles for image and texture generation. In: ACM Transactions on Grapics (TOG), vol. 22, pp. 287–294 (2003)
5. Dali, S.: The Salvador Dali Museum Collection (1996)

6. Demko, S., Hodges, L., Naylor, B.: Construction of fractal objects with iterated function systems. In: SIGGRAPH '85: Proceedings of the 12th annual conference on Computer graphics and interactive techniques, pp. 271–278. ACM Press, New York, NY, USA (1985). DOI http://doi.acm.org/10.1145/325334.325245
7. Desbenoit, B., Galin, E., Akkouche, S.: Simulating and modeling lichen growth. In: Computer Graphics Forum 23, vol. 3, pp. 341–350 (2004)
8. Gewali, L., Meng, A., Mitchell, J.S., Ntafos, S.: Path planning in 0/1/ weighted regions with applications. In: SCG '88: Proceedings of the fourth annual symposium on Computational geometry, pp. 266–278. ACM Press, New York, NY, USA (1988). DOI http://doi.acm.org/10.1145/73393.73421
9. Hertzmann, A., Jacobs, C.E., Oliver, N., Curless, B., Salesin, D.H.: Image analogies. In: Proceedings of the 28th annual conference on Computer graphics and Interactive techniques, pp. 327–340 (2001)
10. Kim, T., Lin, M.C.: Visual simulation of ice crystal growth. In: SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation, pp. 86–97. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland (2003)
11. Kim, T., Lin, M.C.: Physically based animation and rendering of lightning. In: PG '04: Proceedings of the Computer Graphics and Applications, 12th Pacific Conference on (PG'04), pp. 267–275. IEEE Computer Society, Washington, DC, USA (2004)
12. Kim, T., Sewall, J., Sud, A., Lin, M.C.: Fast simulation of laplacian growth. IEEE Comput. Graph. Appl. **27**(2), 68–76 (2007). DOI http://dx.doi.org/10.1109/MCG.2007.33
13. Long, J., Mould, D.: Improved image quilting. In: GI '07: Proceedings of Graphics Interface 2007, pp. 257–264. ACM Press, New York, NY, USA (2007). DOI http://doi.acm.org/10.1145/1268517.1268559
14. Mech, R., Prusinkiewicz, P.: Visual models of plants interacting with their environment. In: SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, pp. 397–410. ACM Press, New York, NY, USA (1996). DOI http://doi.acm.org/10.1145/237170.237279
15. Mortensen, E.N., Barrett, W.A.: Intelligent scissors for image composition. In: SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, pp. 191–198. ACM Press, New York, NY, USA (1995). DOI http://doi.acm.org/10.1145/218380.218442
16. Mould, D.: Image-guided fracture. In: GI '05: Proceedings of the 2005 conference on Graphics interface, pp. 219–226. Canadian Human-Computer Communications Society, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada (2005)
17. Mould, D.: Stipple placement using distance in a weighted graph. In: Proceedings of the Computational Aesthetics in Graphics, Visualization and Imaging, pp. 44–52 (2007)
18. Neubert, B., Franken, T., Deussen, O.: Approximate image-based tree-modeling using particle flows. In: SIGGRAPH '07: ACM SIGGRAPH 2007 papers, p. 88. ACM Press, New York, NY, USA (2007). DOI http://doi.acm.org/10.1145/1275808.1276487
19. Pai, D.K., Reissell, L.: Multiresolution rough terrain motion planning. IEEE Transactions on Robotics and Automation **14**(1), 19–33 (1998)
20. Pedersen, H., Singh, K.: Organic labyrinths and mazes. In: NPAR '06: Proceedings of the 4th international symposium on Non-photorealistic animation and rendering, pp. 79–86. ACM Press, New York, NY, USA (2006). DOI http://doi.acm.org/10.1145/1124728.1124742
21. Power, J.L., Brush, A.J.B., Prusinkiewicz, P., Salesin, D.H.: Interactive arrangement of botanical L-system models. In: I3D '99: Proceedings of the 1999 symposium on Interactive 3D graphics, pp. 175–182. ACM Press, New York, NY, USA (1999). DOI http://doi.acm.org/10.1145/300523.300548
22. Premonition: Dir. Menna Yapo. Hyde Park Films (2007)
23. Prusinkiewicz, P., James, M., Mech, R.: Synthetic topiary. In: SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques, pp. 351–358. ACM Press, New York, NY, USA (1994). DOI http://doi.acm.org/10.1145/192161.192254
24. Prusinkiewicz, P., Lindenmayer, A.: The algorithmic beauty of plants. Springer-Verlag New York, Inc., New York, NY, USA (1996)
25. Reeves, W.T., Blau, R.: Approximate and probabilistic algorithms for shading and rendering structured particle systems. In: SIGGRAPH '85: Proceedings of the 12th annual conference on Computer graphics and interactive techniques, pp. 313–322. ACM Press, New York, NY, USA (1985). DOI http://doi.acm.org/10.1145/325334.325250
26. Salisbury, M.P., Anderson, S.E., Barzel, R., Salesin, D.H.: Interactive pen-and-ink illustration. In: SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques, pp. 101–108. ACM Press, New York, NY, USA (1994). DOI http://doi.acm.org/10.1145/192161.192185
27. Secord, A.: Weighted Voronoi stippling. In: NPAR '02: Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering, pp. 37–43. ACM Press, New York, NY, USA (2002). DOI http://doi.acm.org/10.1145/508530.508537
28. Shiraishi, M., Yamaguchi, Y.: An algorithm for automatic painterly rendering based on local source image approximation. In: NPAR '00: Proceedings of the 1st international symposium on Non-photorealistic animation and rendering, pp. 53–58. ACM Press, New York, NY, USA (2000). DOI http://doi.acm.org/10.1145/340916.340923
29. Streit, L., Buchanan, J.: Importance driven halftoning. Computer Graphics Forum **17**(3), 207–217 (1998). URL citeseer.ist.psu.edu/streit98importance.html
30. Tan, P., Zeng, G., Wang, J., Kang, S.B., Quan, L.: Image-based tree modeling. In: SIGGRAPH '07: ACM SIGGRAPH 2007 papers, p. 87. ACM Press, New York, NY, USA (2007). DOI http://doi.acm.org/10.1145/1275808.1276486
31. Wang, H.: Proving theorems by pattern recognition II. Bell Systems Technical Journal pp. 1–42 (1961)
32. Wang, H.: Games, logic and computers. Scientific American pp. 98–106 (1965)
33. Weber, J., Penn, J.: Creation and rendering of realistic trees. In: SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, pp. 119–128. ACM Press, New York, NY, USA (1995). DOI http://doi.acm.org/10.1145/218380.218427
34. Winston, P.H.: Artificial intelligence (3rd ed.). Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (1992)
35. Witten, T., Sander, L.: Diffusion-limited aggregation, a kinetic critical phenomenon. Physical Review Letters **47**(19), 1400–1403 (1981)
36. Wong, M.T., Zongker, D.E., Salesin, D.H.: Computer-generated floral ornament. In: SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques, pp. 423–434. ACM Press, New York, NY, USA (1998). DOI http://doi.acm.org/10.1145/280814.280948

37. Xu, J., Kaplan, C.: Image-guided maze construction. In: SIGGRAPH '07: ACM SIGGRAPH 2007 papers, p. 29. ACM Press, New York, NY, USA (2007). DOI http://doi.acm.org/10.1145/1275808.1276414

Jeremy Long received his B.Sc. and M.Sc. dgrees at the University of Saskatchewan in 2005 and 2007 respectively. He is now pursuing his Ph.D. at the University of Victoria. His current research interests include non-photorealistic rendering, tree modeling, texture synthesis, multispectral imagery, and color constancy.



David Mould received a Ph.D. from the University of Toronto in 2002. He is presently an Assistant Professor at the University of Saskatchewan, where his research includes natural phenomena, non-photorealistic rendering, and procedural modeling and texture synthesis.