CS5643 Final Project: Modeling leaf venation patterns for use in 3D printing

Bryan N. Peele*
Ph.D. Student, Mechanical Engineering, Cornell University

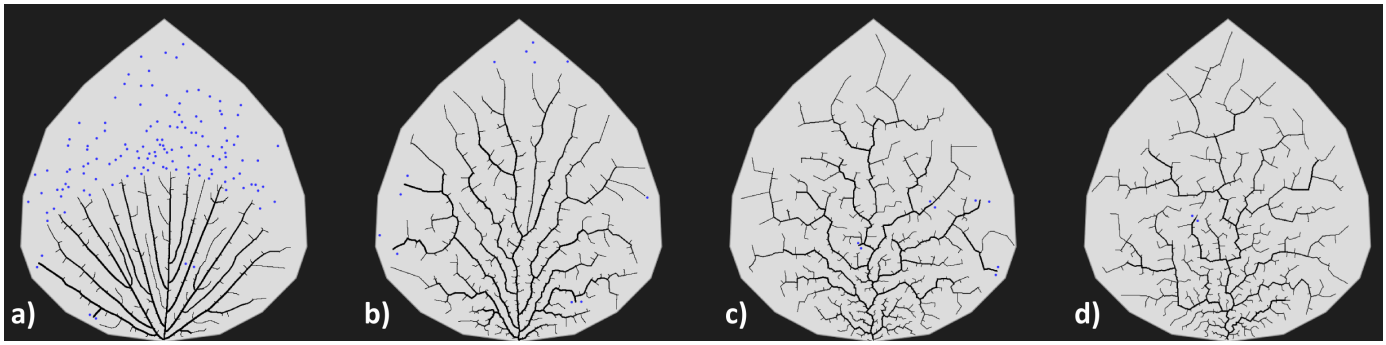

Figure 1: *Venation patterns created by changing the vein growth relative to leaf growth where r = Vein Growth Rate/Leaf Growth Rate. From left to right: (a) r = 0.5, (b) r = 1.5, (c) r = 3.5, and (d) 5 = 5.5*

Abstract

Advances in 3D printing have led to a class of machines that can produce architectures of unprecedented complexity. However, current computer aided design (CAD) tools have left a gap in the ability to effectively design the intricate structures made possible with these technologies. By turning to nature, we can derive algorithms to procedurally generate complex structures through computer simulation. For this project, I apply biologically inspired algorithms to generate 3D architectures that can be 3D printed. Building on the work of Runions et al. [2005], I have simulated leaf venation in order to create visualizations as well as physical models. By adjusting a handful of system parameters, I use the same model to generate a large array of venation patterns. Because the model incorporates a degree of randomness, it is possible to efficiently create an arbitrarily large number of unique structures that are all qualitatively similar. In order to show the viability of physically fabricating these models, I 3D print a sample from this set using stereolithography.

Keywords: morphogenesis, leaf venation, biological growth, 3D printing, stereolithography

1 Introduction

With the onset of fast and affordable 3D printing technologies, the geometric complexity of manufactured objects is no longer limited by fabrication tools. 3D printing offers the ability to create arbitrarily complex geometries using technologies such as fused-deposition modelling, stereolithography and selective laser sintering. However, traditional computer aided design (CAD) tools are built for conventional fabrication tools such as mills and lathes.

3D printing, however, allows entire new classes of geometric primitives. We are no longer restricted to holes that can be drilled or revolved structures that can be turned on a lathe. Instead, it is now possible to create complex lattices and internal voids that could not me made without the use of 3D printing. Moreover, due to reduced material usage, these complex structures can be cheaper and faster to produce.

This broad design space remains largely unexplored, due largely to a lack of CAD tools that allow the user to generate complex structures using high level commands. These new design tools have the potential to allow designs of both aesthetic and pragmatic value. Designs that mimic the complexity of natural systems may be beautiful while also providing lightweight structural lattices.

*e-mail:bnp26@cornell.edu

2 Related Work

Many researchers are now applying the methods of computer graphics and simulation to better understand how geometric complexity arises in nature. Specifically, researchers at the Biological Modeling and Visualization Lab at the University of Calgary have studied diverse systems such as bark patterning in grasstrees [Dale et al. 2014], fruit propagation in apple trees [Costes et al. 2008], and growth of tree branches [Runions et al. 2007].

Recently artists and designers have begun to adapt these biologically inspired algorithms to create complex sculptures that can be 3D printed. Notably Jessica Rosenkrantz and Jesse Louis-Rosenberg founded Nervous System, a design agency that uses computer simulations to generate jewelery and household goods inspired by biological structures [Rosenkrantz 2015].

3 Technical Description

3.1 Simulation

The leaf venation patterns modeled in this project are based on the article *Modeling and visualization of leaf venation patterns* written by Runions et al. [2005]. Simulations are implemented in Java, using the Processing library for simple 2D visualizations. In this work, leaf venation patterns are algorithmically generated by modelling three distinct processes:

1. **Leaf Growth:**Overall leaf growth affects the vein structure and hormone distribution
2. **Vein Growth:** Veins tend to grow towards hormone (auxin) sources along the surface of the leaf
3. **Auxin Placement:** Location and number of hormone sources are time variant and dependent on proximity to existing veins

A map of feedback between these processes is shown in Figure 2. The algortihm for leaf venation simulates each of these three processes in an iterative loop. Leaf blade growth refers to the overall size and shape of the leaf, and is the first process modelled in the loop. For these simulations, it is assumed that the shape and growth patterns of the leaf are already known. I chose to implement marginal growth, in which the edge of the leaf scales outwardly from the edge at a constant rate. The existing vein nodes remain in place and vein edges to not expand as the leaf grows.

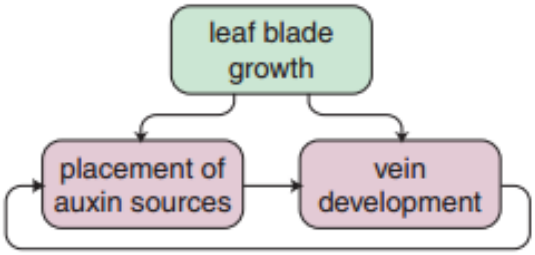

Figure 2: *Biological processes being modeled, figure from Runions et al. [2005]*

Auxin source placement is responsible for setting the targets that the veins will grow toward. Each auxin represents a pool of nutrients that attracts veins, and can be used up when veins come too close. The placement of auxins adds an element of randomness to the simulation and is the primary reason that simulations using the same parameters provide results that are qualitatively similar, but still unique. This placement is handled with a modified *dart-throwing* algorithm [Cook 1986; Mitchell 1987].

Finally vein development is controlled by proximity to auxins. Each source is assumed to influence the closest vein node. Vein nodes with will grow towards neighboring sources by adding a new edge and and node in the average direction of influential sources. The length of these new edges is a system parameter used to control the growth rate of the veins.

3.1.1 Leaf Growth

The initial leaf shape is input by the user a series of coordinates. Each coordinate is stored as a node, and neighboring nodes are connected by edges. The leaf shape is assumed to be a closed 2D polygon, but may contain both convex and concave portions along the perimeter. On each iteration of the simulation loop, all nodes are uniformly scaled by a set factor. With modification, this model could be expanded to include nonuniform growth by defining growth in the horizontal and vertical directions as a function of position.

3.1.2 Auxin Placement

During the initialization, a set number of auxins are placed within the initial shape of the leaf. This placement is accomplished with a modified *dart-throwing* algorithm [Cook 1986; Mitchell 1987]. First a random point is generated, and then tested for compatibility. The first requirement is to determine whether the randomly generated point is contained within the shape of the leaf. This test requires an implementation of the well known *point in polygon* problem. For the point in question, a ray is drawn horizontally to the right. Each edge of the leaf is tested to determine whether it intersects the ray, and the total number of intersections is counted. An odd number correlates to a point inside the leaf and an even number correlates to a point outside the leaf. This method is applicable to both concave and convex polygon structures.

If the point is inside the leaf, the point is tested for proximity to existing auxins and vein nodes. An allowable minimum birth distance is set for both veins and auxins. If the new auxin is both inside the leaf and sufficiently far away from other auxins and vein nodes, it is placed into the system. This testing process is continued until enough suitable auxins are found. The user sets a parameter for the number of auxins to be added with each iteration of the simulation. The density of auxins greatly affects the venation process.

3.1.3 Vein Growth

The vein structure is initialized as a small root system at the bottom of the leaf. This root consists of a vein node at the origin a second node vertically above the origin, offset by one edge length. A vein edge connects these two nodes. On each iteration of the simulation loop, each auxin is assumed to influence the closest vein node. An array of influential of auxins is stored for each vein node. For all nodes with at least one influential auxin, a new vein node and edge are added.

The placement of the new vein node, *v'*, is defined in relation to the position of the existing vein node, *v*, according to

$$v' = v + D\frac{\vec{n}}{\|\vec{n}\|},\ where\ \vec{n} = \sum_{s \in S(v)} \frac{s - v}{\|s - v\|}.$$

Here *D* is a system parameter that is used to set the edge length of the vein. Effectively, this is used to control the rate at which veins grow relative to the leaf. The set of influential auxins for each vein is denoted by *S(v)*, and the position of each auxin in the set is denoted by *s*. This equation ensures that every new vein node is placed a fixed distance away from its parent in the average direction of influential auxins.

At the end of the simulation loop, all auxins are tested to determine their proximity to new vein nodes. If a vein node is found within the set kill distance for auxins, the auxin is considered to be depleted and is removed for the system. Once removed, the auxin no longer influences any of the vein nodes.

As a final step to create a more visually compelling vein structure, variable widths are assigned to each vein edge. The goal is to create a tapered effect, with older veins closer to the root generally thicker, but tapering down in size as growth continues. To accomplish this goal, edge widths are set by using a variation of Murray's law [Murray 1926]. Edges connected terminal nodes are first assigned a minimal width. Moving up to the root from these terminal nodes, edge widths are maintained until a branching structure is reached. At each branching structure, the width (r) of the parent edge is defined by the width of its children as

$$r^n{}_{parent} = \sum_i r^n{}_{child,i}.$$

This is a slight variation from the model used by Runions et al. [2005]. Their model assumed all branches were bifurcations (i.e. only two child edges could originate from a single parent node). Based on the original observations of Murray, the scaling factor is set to *n=3*. However, this factor can be viewed as a system parameter and empirically adjusted to create different visual effects in the venation patterns.

3.2 Fabrication

In order to fabricate physical artifacts using simulation results, the resulting vein graph must be first converted into a suitable 3D CAD format. This CAD model can then be fabricated using a variety of 3D printing techniques. For this project, I used a low cost stereolithography method to quickly generate high resolution objects.

3.2.1 3D Printing Process

I chose to use stereolithography for this project due to its ability to quickly fabricate small components with high resolution. Stereolithography is a process by which high intensity light is used to selectively polymerize a liquid precursor in order to gradually form a solid object. Conventionally, this is accomplished by rastering a UV laser over a 2D surface in a serial process. For this project, I used a variant called Digital Mask Projection Stereolithography (DMP-SL). For DMP-SL, a visible light projector is used in a highly parallel process to selectively cure ˜2 million individual voxels simultaneously. This process is shown schematically in Figure 3.

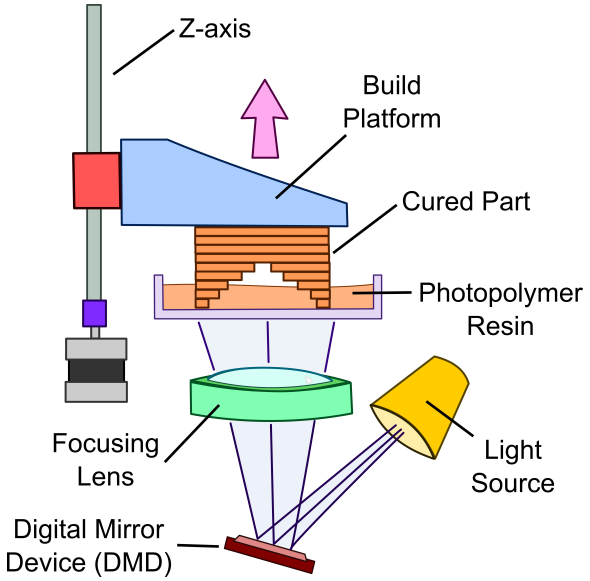

Figure 3: *In the process of Digital Mask Projection Stereolithography (DMP-SL) visible light is used to selectively polymerize liquid precursor to form a solid object, one layer at a time.*

As shown in Figure 3, a Digital Mirror Device (DMD) is used to project light to cure each layer of liquid precursor. After polymerization, the part is raised by one layer height and the next layer is cured. This process continues until all layers have been polymerized. Typical layer height is $25 - 50\mu m$ and typical XY resolution is $35\mu m$. Because an entire layer is cured concurrently, the build time is dependent only on the height of the object. Typical build time is $25mm/hr$, dependent on the properties of the polymer used for printing.

3.2.2 Preparation for 3D Printing

In order to 3D print the vein graph as a physical object, I used two different CAD programs to prepare a 3D mesh. The Java program used to simulate vein growth exports the geometry of the leaf and veins as a Python script designed to run in the open-source CAD program, FreeCAD (Version 0.13). This Python script consists of two sections. The first is used to generate a 3D structure of the vein patterns. Each vein node is represented as a sphere with diameter corresponding to the edge width of its parent. Because of the large overlap between neighboring spheres, the vein edges are not explicitly modelled as part of the 3D structure. Finally the shape of the leaf is added as a prism formed by extruding the 2D geometry of the leaf shape. Initially, I attempted to create a union of all components (˜12,000), but was unable to complete this task using FreeCAD due to memory limitations (16GB of RAM). Instead, I exported the model as 12 separate sections (each consisting of ˜1,000 objects), saving each piece in Standard Tessellation Language (STL) format. The STL format creates a 3D mesh composed of distinct lists of points, lines and triangles. At this stage, the exported STL files contain many intersecting faces and require significant storage (˜1GB). This process is shown in Figure 6a.

In order to reduce file size and create a single watertight mesh, I used Autodesk Meshmixer (Version 10.7.84). The collection of STLs from FreeCAD are combined by taking the union of all meshes and then applying a smoothing operation in order to create a more organic aesthetic. I added a small cylindrical base to allow the model adhere to the build platform more securely during printing. Finally, I scaled the the partto the desired print size. The resulting STL is relatively small (˜5MB). This result of this process is shown in Figure 6b.

As the last step in the pipeline, Envision Labs Creation Workshop is used to prepare the STL for 3D printing by slicing it horizontally into a stack of images. For each layer of the printed object, the corresponding image is sent to the projector. A variety of control parameters are set in Creation Workshop that can affect the quality of the print. These include the duration of light for each layer, the height of each layer and the lift distance to remove each layer from the build substrate. This software is shown in Figure 6c.

4 Results

I was able to successfully implement the leaf venation model described by Runions et al. [2005] for open venation patterns. I used this model to generate animations of the growth process in leaves. I was also able to use this model to generate 3D CAD models for 3D printing.

4.1 Animation

Figure 4 shows an example of several frames from one of the leaf venation simulations. As described in Section 3.1, the outline of the leaf expands uniformly with each iteration of the simulation. At each iteration, auxins are randomly added to suitable locations. These can be seen as blue circles in 4. Every time a new simulation is run with these parameters, the results are qualitatively similar in the type of branching, but unique due to the random placement of auxins. In these images, it can be seen that new veins tend to branch off at approximately right angles. This is a natural behavior that is shown to emerge through a set of very simple rules in this simulation.

By varying the physical parameters used in the model, it is possible to create a large range of qualitatively different venation patterns. This is shown in Figure 1, where each panel is the final result of a different simulation. A compilation video of these simulations is included with the supplemental information (SV1). In each case, all parameters are held constant except for the vein growth rate (as controlled by the vein edge length parameter). As the the vein growth rate is increased, more chaotic vein patterns are formed (d). For lower vein growth rates, a smaller number of dominant, thicker veins tend to form (a).

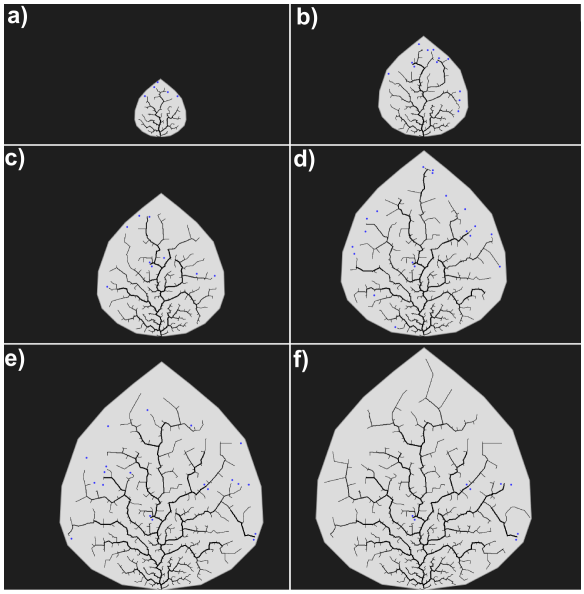

Figure 4: *Growth of a venation pattern during simulation, where f is the frame number: (a) f=150, (b) f=300, (c) f=450, (d) f=600, (e) f=750, and (e) f=900*

4.2 3D Printed Objects

Figure 5 shows an example leaf that was printed using the DMP-SL process described in Section 3.2.1. The leaf is printed using MakerJuice G+ resin, an inexpensive that costs $55 for 1L. Because each leaf has a volume of approximately 1mL, the material cost of each leaf is only $0.06. The build time is determined by the height of the leaf. With a height of 50mm, build time is 2 hours. However, due to their thin nature, many leaves (5) can be stacked together on the build platform. This means that multiple leaves can be produced in the same amount of time as a single structure. A video showing the build process is included with the supplemental information (SV2).

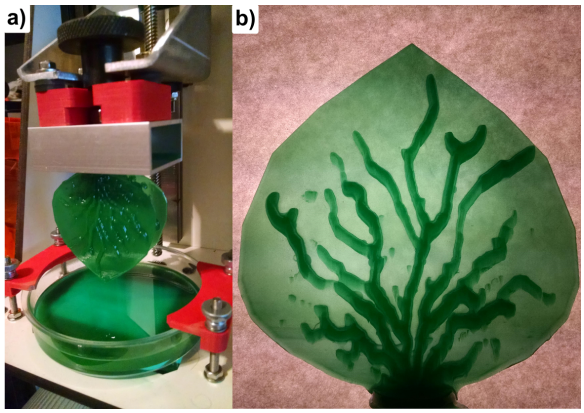

Figure 5: *(a) 3D print of leaf strucutre prior to removal prior to removal from build platform. (b) Backlight photograph of 3D printed leaf structure.*

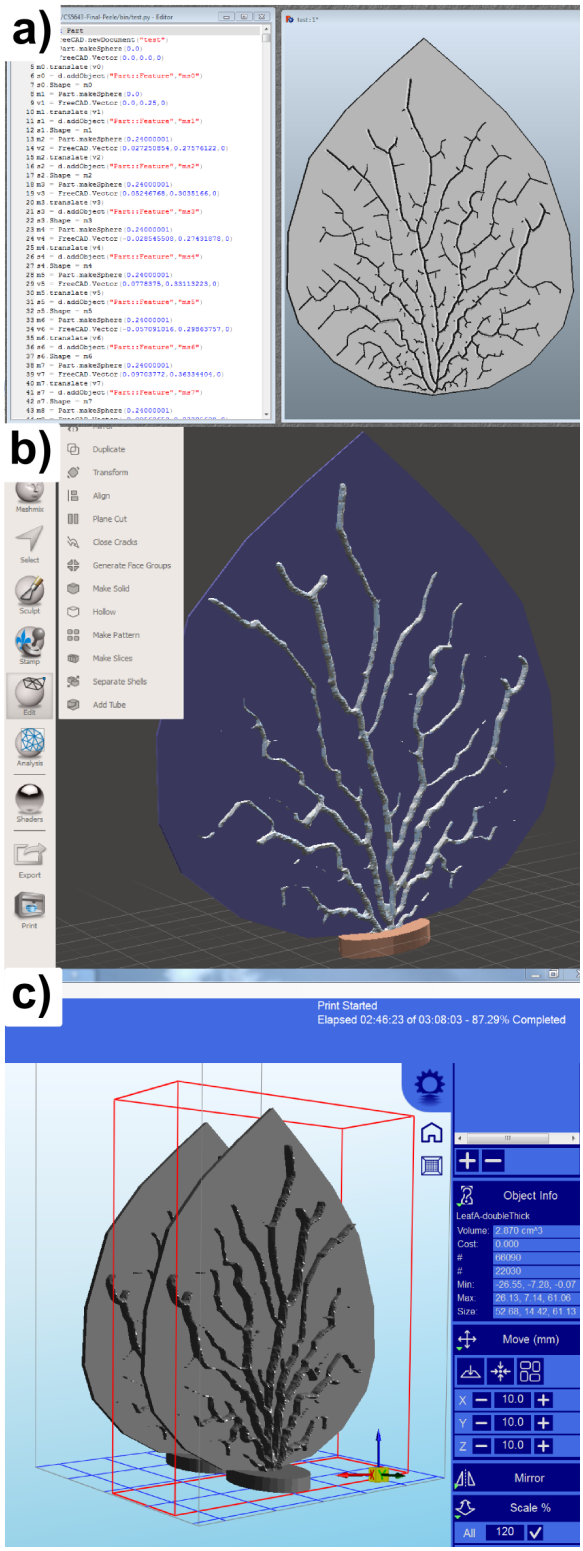

Figure 6: *CAD/CAM Pipeline used to ready vein graph for 3D printing: (a) FreeCAD is used via Python scripting to create a collection of STL files, (b) Meshmixer is used to take the union of STL files, generating a single watertight mesh of reduced size, (c) Creation Workshop is used to convert the STL into horizontal slices and to control the 3D printer*

5 Conclusions

This project shows that biological simulations are a viable way to quickly generate complex 3D structures. By automating these design processes, it becomes affordable to 3D print unique objects. In this example, the pieces created are only of aesthetic value. However, expanding these algorithmic techniques could allow designers to create quickly fabricate practical components (e.g. lightweight support structures that make efficient use of materials). Ultimately these types of algorithmic tools could drastically reduce the cost of designing both functional and aesthetic objects. As these tools gain traction, the role of the designer may shift away from designing individual object and towards developing systems that can be used to design a family of objects.

References

COOK, R. L. 1986. Stochastic sampling in computer graphics. *ACM Trans. Graph. 5*, 1 (Jan.), 51–72.

COSTES, E., SMITH, C., RENTON, M., GUÉDON, Y., PRUSINKIEWICZ, P., AND GODIN, C. 2008. MAppleT: simulation of apple tree development using mixed stochastic and biomechanical models. *Functional Plant Biology 35*, 10, 936.

DALE, H., RUNIONS, A., HOBILL, D., AND PRUSINKIEWICZ, P. 2014. Modelling biomechanics of bark patterning in grasstrees. *Annals of Botany 114*, 4 (aug), 629–641.

MITCHELL, D. P. 1987. Generating antialiased images at low sampling densities. *SIGGRAPH Comput. Graph. 21*, 4 (Aug.), 65–72.

MURRAY, C. D. 1926. The Physiological Principle of Minimum Work: I. The Vascular System and the Cost of Blood Volume. *Proc. Natl. Acad. Sci. U.S.A. 12*, 3 (Mar), 207–214.

ROSENKRANTZ, J., AND LOUIS-ROSSENBERG, J., 2011. Nervous system. `http://n-e-r-v-o-u-s.com/index.php`.

ROSENKRANTZ, J., 2011. Hyphae lamps - an infinite series of lighting designs. `http://n-e-r-v-o-u-s.com/blog/?p=1701`.

RUNIONS, A., FUHRER, M., LANE, B., FEDERL, P., ROLLAND-LAGAN, A.-G., AND PRUSINKIEWICZ, P. 2005. Modeling and visualization of leaf venation patterns. In *ACM SIGGRAPH 2005 Papers*, ACM, New York, NY, USA, SIGGRAPH '05, 702–711.

RUNIONS, A., LANE, B., AND PRUSINKIEWICZ, P. 2007. Modeling Trees with a Space Colonization Algorithm. In *Eurographics Workshop on Natural Phenomena*, The Eurographics Association, D. Ebert and S. Merillou, Eds.