EUROGRAPHICS 2009 / P. Dutré and M. Stamminger
(Guest Editors)

Volume 28 (2009), Number 2

Structure from silhouettes: a new paradigm for fast sketch-based design of trees

J. Wither [1] and F. Boudon [2] and M.-P. Cani [1] and C. Godin [2]

[1]EVASION, LJK / Grenoble Universités, INRIA, France.
[2]Virtual Plants, UMR DAP, CIRAD INRIA, Montpellier, France.

Abstract

Modeling natural elements such as trees in a plausible way, while offering simple and rapid user control, is a challenge. This paper presents a method based on a new structure from silhouettes *paradigm. We claim that sketching the silhouettes of foliage at multiple scales is quicker and more intuitive for a user than having to sketch each branch of a tree. This choice allows us to incorporate botanical knowledge, enabling us to infer branches that connect in a plausible way to their parent branch and have a correct distribution in 3D. We illustrate these ideas by presenting a seamless sketch-based interface, used for sketching foliage silhouettes from the scale of an entire tree to the scale of a leaf. Each sketch serves for inferring both the branches at that level and construction lines to serve as support for sub-silhouette refinement. When the user finally zooms out, the style inferred for the branching systems he has refined (in terms of branch density, angle, length distribution and shape) is duplicated to the unspecified branching systems at the same level. Meanwhile, knowledge from botany is again used for extending the branch distribution to 3D, resulting in a full, plausible 3D tree that fits the user-sketched contours. As our results show, this system can be of interest to both experts and novice users. While experts can fully specify all parts of a tree and over-sketch specific branches if required, any user can design a basic 3D tree in one or two minutes, as easily as sketching it with paper and pen.*

Categories and Subject Descriptors (according to ACM CCS): Computer Graphics [I.3.3]: Methodology and Techniques—

1. Introduction and related work

Providing users with easy ways to create and control digital content is particularly challenging when natural scenes are concerned. Such scenes are full of self-similar details which follow specific laws and distributions. Although many elements look more or less the same, two of them are never exactly identical, so re-using instances of the same shape is immediately noticeable. Creating these multiple elements using standard interactive techniques is a burden. Moreover, they require specific knowledge from the user when realism is required. In such cases procedural models which embed a priori knowledge of a phenomenon ensure the design of plausible solutions. However procedural models can restrict user interaction to indirect parameter tuning, making the overall shape very difficult to control.

Tree modeling is a very good example of such complex design. Among dedicated modeling techniques for trees, the most typical ones are based on growth rules, which are carefully parameterized for given species and whose stochastic shape parameters are set to obey specific, measured distributions. At the other end of the spectrum, image-based reconstruction of trees is another way to use some a priori knowledge while constraining the result [TZW*07, NFD07, TFX*08]. This approach however requires images of an existing, isolated tree usually from multiple viewpoints. A major drawback of both approaches is the lack of user control. None of these methods allow a user to design the shape of a tree in a global way, before refining it and adding specific elements where needed, as they would do with paper and pen (see Fig. 2). This lack of control is a major practical problem for production artists, where realistic looking generated or reconstructed trees are not very likely to fit the directors requirement. Since there does not yet exist any sensor which can capture the structure of large trees, this is also an increasingly important issue for plant scientists (ecologists,

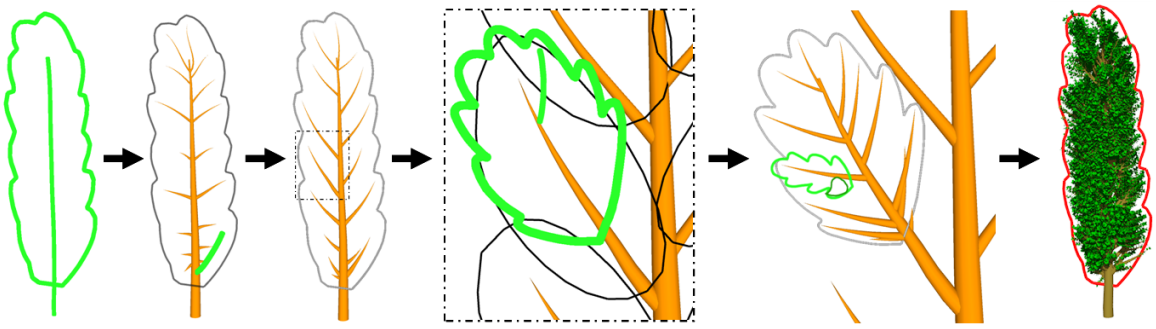


Figure 1: *Creation of a 3D poplar tree in less than 2 minutes, by sketching successive silhouettes of foliage at different zoom factors, from the full tree to a leaf. Plausible branches and construction lines for the smaller-scale silhouettes are inferred from each sketch. When the user zooms out, each branching system transmits its style to the other systems at the same level and 3D is inferred, resulting in a full 3D tree. The latter can still be edited by over-sketching from arbitrary viewpoints.*

agronomists, botanists and foresters). They want to use for their studies accurate 3D reconstructions of actual trees that were observed and sketched in the field.

This paper presents an effective method for interactive tree design, based on a new paradigm, which we call *Structure from Silhouettes*. The user sketches multiple 2D silhouettes of the tree foliage, from coarse to fine scale. Prior knowledge on the nature of trees is used to infer the corresponding branching structures and their 3D distributions. The style of the unrefined branching systems can be inferred from the styles of their neighbors or of their parent in the hierarchy, based on the self-similar property of trees. So the user only needs to refine his sketch locally, leading to the generation of a full 3D tree that fits the sketched contour in minutes.

Procedural modeling of trees: Due to the way plant grow, their geometry obeys specific rules, which for instance make most trees self-similar at different scales. Procedural modeling techniques generate geometry using this knowledge. They are thus particularly well adapted for trees. Previous models includes for instance fractals [Opp86], and L-systems [PL90], which has been the most successful approach up to now. The major drawback of these methods is the indirect nature of shape control they provide. Since geometry emerges from the rules and their user-specified parameters, modeling a specific tree one has in mind is difficult, even using some recent advances such as the geometric parameters in [WP95] or the XFrog system [LD99]. The idea of providing the user with some control over the overall shape of a tree while still relying on procedural laws was first introduced by [PMKL01] and further developed into a full 3D multiscale modeling system illustrated on the design of complex Bonsai trees [BPF*03]. However this system could still take several hours to model a given tree. Our work re-uses this idea of coarse-to-fine modeling, offering user control at different scales and enabling style copying. However, we eliminate tedious user input by **inferring structure from silhouettes**, the latter being quickly specified through sketching.

Tree and plant modeling from sketches: Sketching interfaces are an attractive way to enhance user control. A first work dedicated to the sketch-based modeling of trees was proposed by [OOI05], leading to nice examples constructed in less than 10 minutes. The user draws a 2D view of the tree branching system and then defines examples of organs that are copied all over the structures, while the branching system is automatically extended to 3D. In contrast with our system, crown shapes are difficult to control unless many branches are drawn, which can be time consuming. Moreover, the degree of realism depends on the user's skill. Nothing ensures that the user will follow the well known rules on the variation of branching angles along a parent branch, nor that the styles of the different sub-branches he sketched will look compatible. Anastacio *et al.* [APS08] used conceptual sketches based on construction lines to define arrangements of plant organs (the system doesn't support trees). Similarly the silhouettes we sketch and refine for trees can be seen as construction lines. However, in our case they are controlling the shape, not the 3D arrangement of foliage, and lower scale construction lines are generated automatically from the parent silhouette, to serve as a support for user refinement. Chen *et al.* [CNX*08] use Markov random fields to determine 3D orientation of sketched branches. We perform similar stochastic optimisation but taking into account botanical factors such as phyllotaxis.

2. General Methodology

Our structure from silhouettes approach builds on the drawing methodologies of both botanists and artists.

Botanists create 2D drawings of trees using pen and paper. These drawings aim to capture the essential architec-

tural features of trees and can be used to study the way trees grow [BC07]. They are time consuming to produce. For big trees, a botanist will only draw details in certain sections of the tree, using 'zoom boxes'.

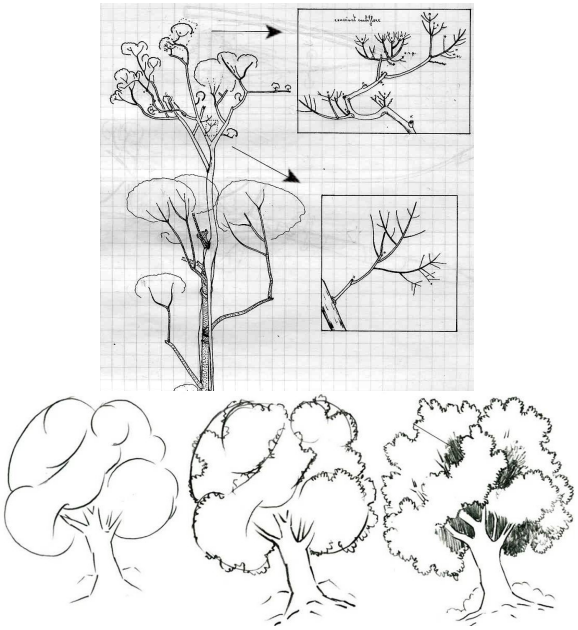


Figure 2: *(top) Illustration of how botanists work, courtesy of Y. Caraglio (bottom) Illustration of how artists work, from [Pow98].*

Traditional artists create drawings incrementally (see Fig. 2), placing early construction lines used to guide later more precise refinements [Pow98]. Detail is added after general outline, and fine detail is often represented only locally. Computer artists cannot use this incremental approach for creating tree models, since they currently rely on procedural, local-to-global, tree generators.

Our work addresses the needs of both disciplines by introducing the idea of inferring tree structure from multiple silhouettes. The user is only required to sketch crown silhouettes at different scales, and only needs to refine a subpart of the tree, the style of the branches being automatically copied elsewhere. Compared with sketching the branching system directly, we believe that this methodology both improves shape control and makes the design of complex trees much faster. The user starts at the largest level of scale - the trunk and the crown silhouette of the whole tree (level zero (L0)). Child branches (L1) are automatically inferred from the silhouette shape using botanical knowledge of trees and guidelines for sub-silhouettes (L1) are generated. After possibly making alterations through oversketching, the user selects an area to zoom into and progressively draws finer silhouettes until the level of leaves. Once the finest level of detail is reached, the last branch is placed in 3D using one of our botanical-based distribution laws and the style is copied to the siblings at that level; when the user decides (when he zooms out for instance), 3D distributions and styles are copied until a full 3D tree has been created. Note that advanced users tend to choose different 3D distribution laws at the different scales, to better represent real tree species. The generated tree can be modified further once it has been embedded into 3D, as the camera is free to move at any stage and all operations work from any viewpoint.

Some challenging problems need to be solved to implement the methodology we just presented. First a method for inferring plausible tree structures from user-sketched silhouettes needs to be defined. We build such a structure based on botanical knowledge of trees, as presented in §3. Our method for recursively refining the tree shape and copying the style to the non-refined parts is given in §4. Placing the 2D structure in 3D and generating extra branches is discussed in §5.

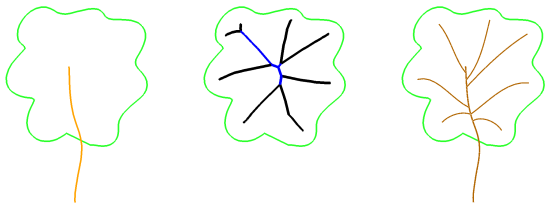
3. Inferring 2D structure from silhouettes

Although structure from silhouettes was in some sense used in all free form sketching systems, based on inflation along a geometric skeleton [IMT99, BPCB08], adapting this idea to the case of trees is not trivial. As one can see in Figure 3 the geometric skeleton of a typical tree silhouette is very different from the geometry of the branches which create that silhouette (as noted in work by [SRDT01]). Given a silhouette of a tree crown, there are an infinite number of 2D branch and leaf arrangements that could provide that silhouette. We aim to provide one 'reasonable' solution from among the alternatives, where reasonable means a solution that attempts to meet the expectation of the user and that bears some resemblance to common tree architectures.

3.1. Silhouette segmentation and major branches

Our first assumption about the major branches producing the silhouette of a tree is that they start from a common trunk and they extend into the 'bumps' observed on the silhouette. These bumps can be seen as indication of the crowns of the associated sub-branching systems. These assumptions confirmed by a botanist (§6) match the natural segmentation humans perceive when regarding tree silhouettes, thus helping to meet the user expectation.

We first isolate the bumps on the silhouette and determine a central point for each bump. This is achieved by calculating the geometric skeleton of the silhouette and using only the terminal points and tangents of this geometric skeleton. In our implementation we use a modified version of the Chordal Axis Transform technique of [Pra97] to calculate a geometric skeleton and maintain a correspondence between silhouette edges and the skeleton. Alternatively an image based skeleton approach could have been used. We assume that major branches will end on these central points,



(a) (b) (c)

Figure 3: *The branching system of a tree is very different from a geometric skeleton. (a) Sketched trunk and silhouette (b) Geometric skeleton (c) The tree structure we infer by combining botanical knowledge with some information from the geometric skeleton.*

with branch end tangents matching those from the skeleton. It now remains to infer the shape and starting point of each major branch. We do this using supplemental information from the field of botany. As in [PMKL01] we observe that the gradient of lateral branching angle along the trunk is very important to the designer, since it strongly influences the general posture of the structure. Thus we combine the user supplied shape information (bump centroid and tangent) extracted from the geometric skeleton, with botanical insights on the variation of this branching angle. This method, described next, leads to the generation of simple, natural looking branching structures which the user can easily alter through over-sketching if desired.

3.2. Inferring branches

The gradual variation in lateral branching angle of child branches along a parent trunk is modeled in our system by a function we name the *branching angle function* (BAF). The BAF maps positions $t \in [0,1]$ along the trunk ($t \in [0,1]$) to angles between the tangent of the trunk and the tangent of a child branch at their junction point. The values of the BAF are chosen to match the botanical observation that branches at the bottom of a trunk tend to grow horizontally (plagiotropy), but become more vertical towards the top of the tree (orthotropy). Thus, the BAF is initially defined by two angle values (at the bottom and top of the trunk), linear interpolation being used in-between (the BAF is depicted using the black segments along the trunk on Fig. 4). If the user redraws a branch (see §3.3), this will modify or add a value in the function.

For the shape and position of the inferred branches we already have their end-point E and the associated tangent to the geometric skeleton (see Fig. 3(b)). The goal is now to find a starting point B on the trunk for these branches, the direction of the branch at this starting point being given by the BAF. We use a 3rd order Hermite curve to guess the initial branch shape, since they are defined from their endpoints and the associated tangent vectors.

We use a simple, incremental method for finding the best B for a given E. Using a regular sampling along the trunk, we approximately compute the length of each Hermite curve joining B_i to E with the tangent vectors mentioned above (the length of these vectors being set to half the distance between B_i and E). Then we simply select the shortest of these branches. Examples of branches resulting from different tangents at endpoints are depicted on Fig. 4 and a full result is shown on Fig. 3(c).

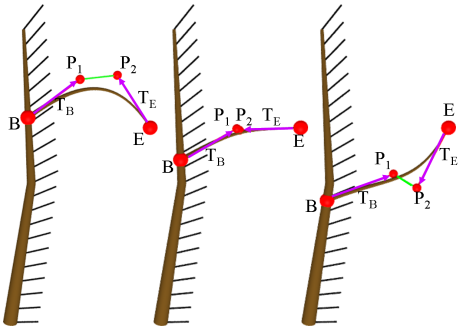
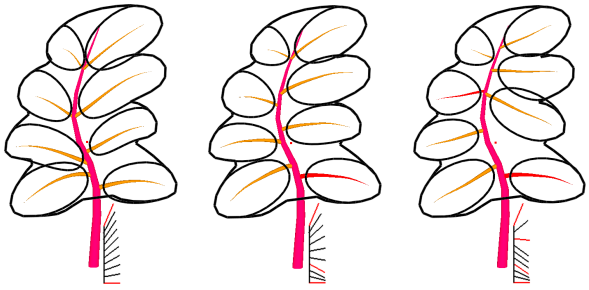


Figure 4: *Three examples of automatic branch shape determination starting from the same endpoint E but with different end tangents T_E. The Branching Angle Function (BAF) is shown using black vectors along the trunk.*

3.3. Redrawing branches and transferring their shapes

The user modify the resulting structure by adding, deleting or changing the shape of a branch to a more complex (B-Spline) curve through over-sketching. The new (source) branch S is drawn close to the branch to be edited. The target branch is selected using a cost function based on proximity between base and end points.



(a) (b) (c)

Figure 5: *(a) The automatically inferred sub-branches. Inset - the branch angle function (BAF). (b) The red branch has been redrawn by the user, and the shape change automatically copied to the sibling shapes. Inset - the altered BAF. (c) The user redraws a second branch (left side, second from top) and the shape change is copied.*

The angle and position of the modified branch on the parent trunk are used to add a sample point to the parent BAF (see Fig. 5). The position of the bases of the sibling branches are automatically changed based on this new BAF. Sibling branches are then automatically re-shaped: a spline curve is fitted to the new branch S and control points are re-expressed in a local frame based on the span vector of S. These local coordinates are converted to a frame based on the span vector of the target branch, and scaled according to the ratio of span lengths. This defines a new spline curve, giving the new shape for the target branch. Note that the default sub-silhouettes of sibling shapes are modified accordingly to the new branch starting point.

4. Recursive local refinement and style transfer

4.1. Inferring sub-silhouette shapes

As mentioned earlier, the bumps we detected on the sketched silhouette indicate the crowns of the sub-branching systems associated with the major branches. Rather than leaving the user to redraw these sub-silhouettes from scratch while zooming in, our system extracts a simple version of them from the original sketch. This has several benefits: (a) the inferred sub-silhouettes will be used to generate more branches if the coverage of the region inside the tree is not good; (b) they will serve as construction lines for the user to locally refine the sketch; (c) they will be used as guides during style copying from one branch to another.

An inferred sub-silhouette should share some of its edges with a bump on the parent silhouette, and then deviate smoothly to connect near to the base of the associated branch. We first select as shared edges the edges of the parent segmented bump that lay inside a cone defined around the segment $[B,E]$ of the branch. As shown on Fig. 6, the angle α_c of this cone should depend, for a given branch length, on the radius of curvature at the end point of the branch. Large, smooth features on the silhouette should generate large sub-crowns, while sharp features should generate narrow ones. More precisely, we use the ratio r/d between the radius r at the extremity and the span length d of the branch to select an adequate value for α_c, using linear interpolation between two pre-set extreme angle values (we currently use 20° and 110°, values that we measured on pictures of real trees).

Once the section of the silhouette inside the cone has been selected, we complete the sub-silhouette by joining it to the base of the new branch using two Hermite curves. We choose the tangents so that the shape is G1 continuous (see Fig. 6).

Ensuring crown coverage: The last automatic step before the user can refine the sketch is to check whether the detected sub-silhouettes give full coverage of the parent crown. If the user drew a very smooth silhouette for the tree, the coverage will typically be low (due to the lack of high curvature features) if this is the case, we generate more branches and

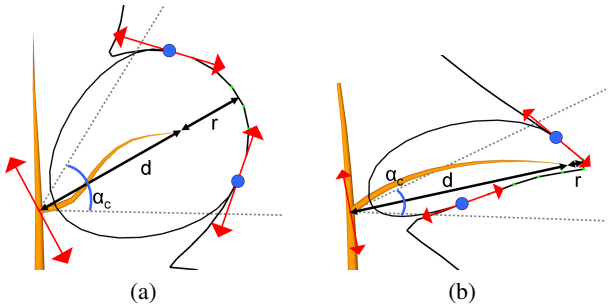


Figure 6: *Inferring sub-silhouette shapes. Features on the parent silhouette with (a) large and (b) small radii of curvature. The cones (dashed lines) used to select how much of the parent silhouette should be used to form a sub-silhouette. The red vectors are the tangents at the ends of the two Hermite curves and the ends of the shared edge section. The blue dots are where the shared edge meets the curves. α_c, d and r are described in the text.*

sub-silhouettes. The coverage test is based on the inferred sub-silhouettes. We 'fill-in' any remaining space by proceeding iteratively from the base to the top of the parent trunk. At each sample position along each side of the trunk, we copy the nearest sibling branch and crown, re-orient it according to the BAF and scale it to fit within the parent silhouette. It is then tested for overlap with the set of sibling crowns. If the area of overlap is above a threshold (we use 20% of current crown area) then the branch copy is rejected and the next position is tested. Otherwise the new branch is added to the set of siblings and the selection continues. In this way, the density of created branches is related to the shape of the silhouette.

4.2. Refining sub-silhouettes and copying style

We use a simple metric based on overlap to automatically detect which of the sub-silhouettes the user redrew. The branch which has the largest absolute length covered by the new sub-silhouette is selected. We now discuss the way we copy the style (branch shapes and distribution) of a sub-crown to the neighbouring sub-crowns.

From botanical studies [GBYC01], we know that if a section along the beginning of the branch bears no substructure, then it is likely that a similar lack of growth is observed on other sibling branches. So we assign the branch two ranges, an unbranched range at the beginning followed by a ramified (branching) range. The unbranched range is measured as a relative or absolute length which is preserved on other branches. The ramified range is parameterized by a mean and standard deviation (S.D.) of the absolute distances between branching points. A normal distribution is used to generate new branching points on the ramified ranges of sibling branches using these details. The BAF of the source branch

is also copied to the target. Branch shapes at new branching points are copied from nearby source child branches, chosen by comparing the normalized t positions along the trunks. The copied child shape is re-oriented according to the BAF and scaled to fit the target silhouette. The different steps of this process are depicted on Fig. 7.

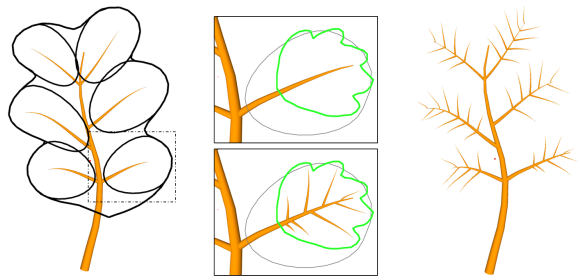


Figure 7: *The process of copying the style of a branch. (a) The initial structure with no L2 branches. (b) The bottom right branch silhouette is redefined by the user and the internal branch structure is automatically inferred. (c) The branch distribution from the example is used to automatically generate all L2 sub-branches. The refinement process terminates when the user draws a silhouette representing a leaf and then presses a button to indicate he has finished.*

5. Positioning organs in 3D

Sketched branches or leaves have a defined position on their parent branch but their 3D azimuthal orientation around their parent branch is not specified in the drawing as we only perceive a projection of the branch in the screen plane. Reconstructing a full 3D branching system thus requires the inference of three types of information from the drawing: a) the azimuthal orientation of the sketched branches b) the existence of extra non-sketched branches c) the shape, length, position and azimuthal orientation of these additional branches.

To solve this problem, two types of constraints have been taken into account. First, the reconstructed tree must fit within the sketched silhouettes to make the design controllable. Second, the solution must adhere to some botanical laws, depending on the degree of realism the user wants. This section describes how both constraints are specified and used to infer the final 3D tree.

5.1. Apparent phyllotaxis

Plants axes are built by small embryogenic tissues at the tip of branches, called meristems. During plant development, meristems create lateral organs, leaves, lateral meristems or flowers, that are arranged in well organized patterns along the axes. This organ patterning is called *phyllotaxis*. Botanists have identified various types of phyllotaxis: spiral, decussate, alternate-decussate, whorled, *etc.* Different phyllotaxies may exhibit on different type of axis within a plant [BC07].

To explain these patterns, one of the most widely accepted theories relies on the assumption that young lateral organs created by meristems generate an inhibitory field in their neighborhood that prevents any new organ appearing in a region around the inhibitory organ. The size of the region depends on the intensity of the inhibitory field and has been shown to control the type of observed phyllotactic pattern [DC96, SKP06]. For young plants, the phyllotactic organization of lateral organs can be readily observed, but for trees the original organization is often blurred by the death of lateral organs due to internal competition for resources (e.g. water, light).

To model the apparent phyllotaxis of plants, we designed a mixed model in which we combine an inhibitory-field approach with a global optimization process. Each branch generates an inhibitory field in its neighborhood which deters other branches from being placed nearby. Given a particular distribution of branches and constraints on the positions of these branches, we find a distribution that minimizes the energy of branch distribution in their own global inhibitory field.

5.2. Position and orientation of branches

To represent the position of organs along the trunk, we use a representation that is frequently used for the analysis of phyllotactic patterns and which consists of unfolding the branch cylinder along its carrier curve (see Fig. 8). In this system, the insertion point p of each branch on its parent branch is characterized by a set of coordinates (θ, u), where θ is the azimuthal orientation of the branch and u is the curvilinear abscissa of the branch along the carrier curve ($\theta, u \in [0,360] \times [0,1]$). To account for the cyclic nature of the stem cylinder, we assume that the domains of insertion points are cyclic. We also associate with (θ, u) the normalized coordinates (θ, u) such that $\tilde{u} = u/L$ and $\tilde{\theta} = \theta/360$, where L is the length of the parent axial axis.

Constraints may be imposed on the points by defining a positioning domain $D_i = (\left[u_i^{min}, u_i^{max}\right], \left[\theta_i^{min}, \theta_i^{max}\right])$ for each point p_i. Valid branch positioning solutions are thus such that $p_i \in D_i$ for all i.

5.3. Inference rules and constraints

Addition of new branches: For each sketched silhouette, a first series of branches are inferred (see §4.3). As remarked by [OOI05], *"people tend to draw branches that extend sideways and omit branches extending toward or away from the screen"*. This was confirmed by the botanist who tested our system. Therefore, the density of branches we get on the

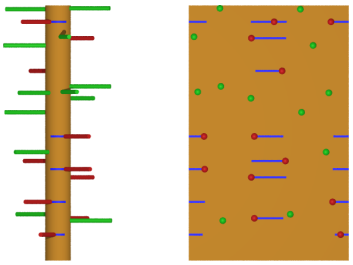


Figure 8: *Cylindrical (folded) and planar (unfolded) views of the positions of a set of lateral branches. Height corresponds to position along the trunk, width to angle. Red points correspond to positions of sketched branches. Blue segments represent their possible ranges. Green points correspond to positions of added branches. Added branches may be positioned anywhere in the whole domain.*

main stem is generally enough for 2D coverage, but is underestimated for 3D. Consequently, we provide a mechanism for creating additional branches to fill up the 3D space around a parent branch. In our system, density can be incrementally increased using a key.

Constraints on the positioning of sketched branches: If we assume with [OOI05] that the user sketches branches that are contained in planes close to parallel with the screen plane, we must respect this by imposing constraints on the position of these branches on the plant. For this, we assume that the coordinates θ of their insertion points are close to either $\theta_{ref} = 0^\circ$ or 180° on the left or right sides of the tree respectively. Thus we set their positioning domains such that $D = u_0 \pm 0, \theta \pm 45^\circ$. This default definition can be modulated according to the branch spans (see §5.6). For the additional branches, all u and θ values are available.

5.4. Optimization of branch positioning

The 3D arrangement of branches along a parent stem is controlled by a stochastic point process, known as the *Gibbs process* [Dig83]. In such a process, a pairwise interaction function $f(i,j)$ represents the cost for two insertion points p_i and p_j of being at a relative distance d(i,j) from each other. It can be seen as modeling the interaction/competition between branches. A realization of this process corresponds to minimizing a global cost function defined as the sum of the costs for each pair of points: $F = \sum_{i \neq j} f(i,j)$ where f is a function that decreases according to the distance between the two points. The Gibbs process simulates situations of dynamic equilibrium. We first define d (a normalized distance) as $d(i,j) = \left\| (\Delta\tilde{u}_{ij}, \Delta\tilde{\theta}_{ij}) \right\|^2$ with $\Delta\tilde{u}_{ij} = \tilde{u}_i - \tilde{u}_j$ and $\Delta\tilde{\theta}_{ij} = \tilde{\theta}_i - \tilde{\theta}_j$. Note that $\Delta\tilde{\theta}_{ij}$ and $\Delta\tilde{u}_{ij}$ have to be adjusted to take into account the cyclic nature of the domain. We then define the following closeness penalty function f as the cost function of the system. $f(i,j) = \frac{1}{1+\alpha * d(i,j)}$ where α is a parameter used to weight the contribution of the cost function $d(i,j)$ (set by default to 20). f is thus maximum when i and j are at the same position and decrease with distance.

For some specific tree species, the above cost function must be modified. Additional terms are added to promote specific position angles between branches, for example to stress a particular phyllotactic pattern. A branch arrangement is controlled by a phyllotactic angle ϕ, defining the angle between organs at two successive nodes. To account for whorls we allow a node to bear several organs. The angle between successive organs of the same whorl composed of n elements is generally $\gamma = \frac{360}{n}$. To determine a theoretical angle between two given organs, we also need to know how many nodes are in between. For this, we use an average node length l that can simply be approximated by Ln/N, where N is the total number of branches borne by the stem.

To make regular branching patterns emerge from the cost function, we will assume that a branch i at a given position p_i induces *privileged positions* for the other branches. The cost function between two branches is thus made proportional to the distance to the closest privileged relative position. The new cost function g is:

$$d_{node}(i,j) = \lfloor (\frac{\Delta u_{ij}}{l}) \rfloor \quad (1)$$

$$d_{ang}(i,j) = (\Delta\theta_{ij} - d_{node}(i,j) * \phi) \bmod \gamma \quad (2)$$

$$d_{pattern}(i,j) = \frac{\min(d_{ang}(i,j), \gamma - d_{ang}(i,j))}{\gamma} \quad (3)$$

$$g(i,j) = w_1 * f(i,j) + w_2 * d_{pattern}(i,j) \quad (4)$$

where w_1 weights the contribution to the cost due to closeness and w_2 weights the contribution due to the distance, $d_{pattern}$, to the nearest privileged position. To compute this distance, we use d_{node} to define the number of nodes between i and j and d_{ang} to define the variation of angle to the branch position in the pattern at this node distance. In practice to enforce non-overlapping of organ positions, w_1 and w_2 are set to 2 and 1 respectively. This second cost function has thus the same shape than f but with regular local minima and maxima if the relative position of i and j corresponds or is far from a given phyllotactic pattern respectively.

To simulate the Gibbs process, we use the iterative algorithm of *depletion–replacement* [Rip79]. Let P be an initial set of insertion points p_i and F the associated global cost function value. Starting from P, a point is selected at random and its position is changed randomly. The old configuration is replaced by the new one only if the new one has a smaller global cost value F. The process is initiated with a random distribution and iterated until the change in the cost value between consecutive steps remains under a prescribed threshold. During iteration, we observe that F decreases quickly during early iterations and then tends slowly to a local min-

imum (the optimality of the solution is not of major importance here). In our implementation, we use a maximum of 3000 iterations which is interactive and produces realistic solutions.

5.5. Resulting distributions

Figure 11 shows different branch arrangements produced by our stochastic process. Red lateral branches are deduced from the sketch and green ones are added to give more volume to the tree.

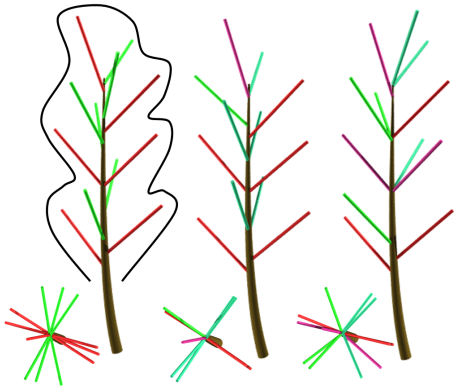


Figure 11: *Illustration of different 3D branch arrangements. Note that the top branch, which is almost aligned with the trunk, looks much shorter than it really is when seen from above.*

The first one uses the cost function f. Branches are evenly distributed along the trunk length in terms of both u and θ. The second distribution, called *opposite-decussate*, uses the cost function g and assumes whorls of two branches where each whorl is rotated at 90° from the previous one. We observe an alternating series of two opposed branches. The final distribution is a *whorl* distribution where the whorls are composed of three lateral branches with a phyllotactic angle of 60° between whorl branches. Branch colors have been slightly modified to aid whorl identification. Other arrangements can be formed easily by choosing a number of organs per whorl and a phyllotactic angle. In our system, additional arrangements, such as spiral ($\phi = 144^\circ$ and $n = 1$ for whorl), decussate ($\phi = 0^\circ$ and $n = 0$) and horizontal decussate, are available (see Fig. 9) or can be set by the user. Once the 2D structure has been drawn, the user can explore the various arrangements the system provides and choose the one he prefers (by keyboard selection).

5.6. Shape and dimensions of new 3D branches

New 3D branches, oriented using the BAF, are scaled to keep their 2D projection coherent with the sketch. Their shapes are determined by interpolating the shapes of neighboring branches inferred from the sketch. Their length is set to meet the pseudo-surface of revolution obtained by rotating the two halves of their parent silhouette (left and right) while interpolating between them.

6. Implementation and results

Although non-optimized (written in python using C++ modules), our prototype implementation runs at interactive rates, enabling fluid user interaction. Figures 1 and 9 give an idea of the variety of trees we can design in a few strokes. Table 1 shows the number of strokes required and the resulting complexity. Note that sketching the branching system (as in previous work) instead of hierarchical silhouettes would have taken much longer, and might have been difficult to achieve for our dense tree examples. The minimum input to our system, as depicted at top left of Fig. 9, is one stroke per level of hierarchy. Designing four levels of detail and building the whole hierarchy while zooming out took about two minutes. The teaser example, a poplar which required some branch editing, took less than two minutes. The eucalyptus at the bottom left, modeled on a photo, involved more redrawing to express the variety of branch shapes, and took about ten minutes to design.

Tree example	Stroke count	Time (mins)	Element count
poplar (teaser)	7	2	53,000
pine	7	2	32,000
christmas tree	41	3	40,000
palm	29	3	3800
willow	43	4	35000
eucalyptus	35	10	1294000
fern forest	280	30	90000
tree stand	220	30	440000

Table 1: *The number of strokes required by the user, the modeling time and the resulting complexity (leaf and branch count) of each tree model.*

In out implementation the tree is stored as a hierarchy of branching systems with associated 2D drawing planes [IOI06]. Each branching system stores two strokes, representing the trunk axis and the silhouette. This gives a complete tree in 3D with each individual branch being planar. This planar restriction allows the tree to be easily assembled from billboards at render time, thus providing a natural way to rapidly visualize scenes with many drawn trees. Alternatively branches could be automatically given 3D spiral shapes by assuming constant curvature, as was done in [OOI05].

We invited an artist and a botanist to use our system. They found that the combination of sketching silhouettes and copying styles to be powerful. With little input, testers achieved results which seemed convincing to them. They sometimes felt that it was more natural to sketch branches directly at larger scales and to sketch silhouettes at smaller

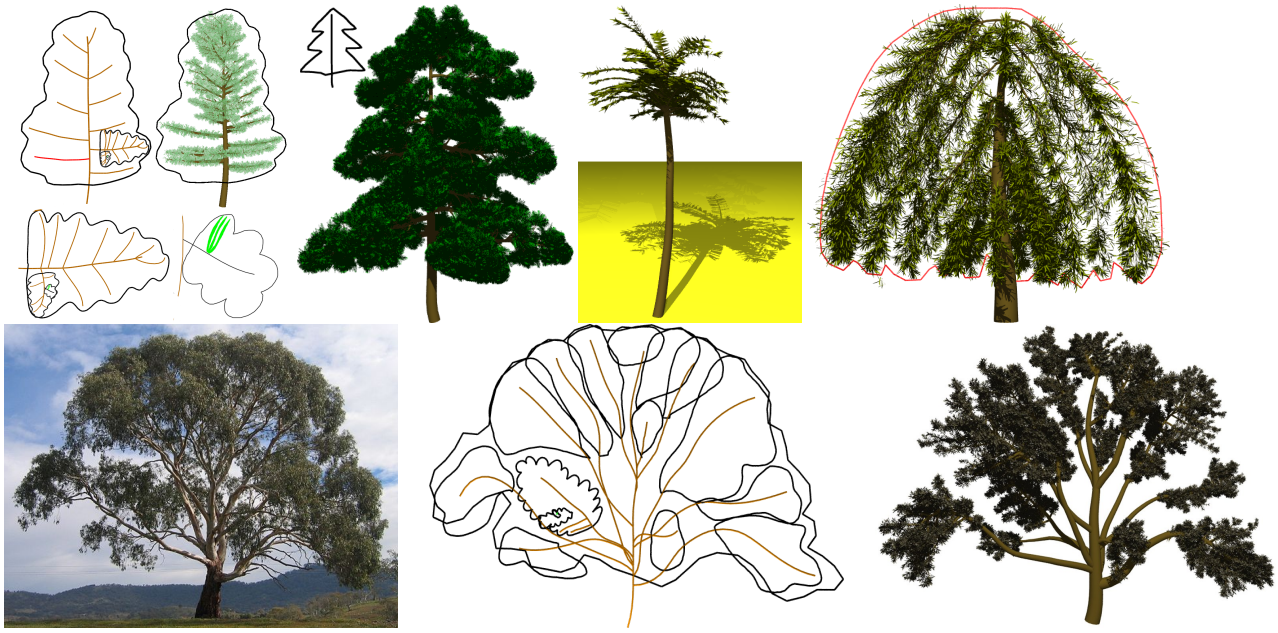


Figure 9: *Results showing the variety of complex trees easily achieved with our system. (Top row) a young pine tree of low density, a dense Christmas tree (from a simple, child-like sketch of the silhouette), a palm, a willow. (Bottom row) This complex eucalyptus example illustrates the way a biologist can use our system for expressing their observations on real trees.*

Figure 10: *(top) Application of our method to quickly sketch a fern forest. (bottom) A stand of trees. Both scenes were sketched over photographs (right) and took thirty minutes each to model.*

scales. Giving the outline first took some getting used to, but lead to better proportions and control over the sketch.

We identified several limitations. First, the correspondence between the sketched silhouettes and final output is usually good, but results may be a bit deceiving when a low density is chosen, such as in the pine example on top left of Fig. 9: the empty spaces left may give the impression that the desired silhouette is badly fitted. The user can always increase density if this is a problem. Another point is that crown shapes, trunks and leaves are only specified using a single 2D stroke. Enabling the use of more drawing planes for the same crown would enable to create more general results. For the moment, the user can always make the tree look different from the sketched silhouette when seen from another viewing angle by redrawing some of the branches. Lastly, style is currently copied to all siblings in the current implementation. Simple operations for choosing source and target branching systems would make style transfers more flexible.

7. Conclusion and future work

This paper presented the first sketch-based method for trees based on a structure from silhouettes paradigm. Our results have shown that it saves user time and improves shape control when compared to only sketching structure directly. Automatic inference of the branch shapes, their 2D distribution and their 3D arrangement around a parent rely on strong a priori knowledge from the field of botany. This enables the creation of plausible trees while reducing the need for user expertise. Our approach thus bridges the gap between the direct sketch-based methods such as [OOI05] and procedural systems with little shape control.

An extension would be to extract automatically silhouette parameters from photograph such as [TFX*08]. For the future, one could imagine using our approach of seamless, coarse to fine sketching from any viewpoint at the scale of a full landscape. For quickly shaping a coarse forest on a hill in the background, the user could just sketch one silhouette per tree, which could be recursively duplicated at a smaller scale in a fractal style, as a first guess for finer details. Then, the user could add detail to the important trees in the foreground to ensure that the design would meet the art director's requirements for the different scenes. We believe that such a system, providing a quick first guess but still enabling simple manual editing where needed, would be of a great help in production pipelines.

References

[APS08] ANASTACIO F., PRUSINKIEWICZ P., SOUSA M. C.: Sketch-based parameterization of l-systems using illustration-inspired construction lines. In *EG SBIM Workshop* (2008).

[BC07] BARTHÉLÉMY D., CARAGLIO Y.: Plant architecture: a dynamic, multilevel and comprehensive approach to plant form, structure and ontogeny. *Annals of Botany 99*, 3 (2007), 375–407.

[BPCB08] BERNHARDT A., PIHUIT A., CANI M.-P., BARTHE L.: Matisse: Painting 2D regions for modeling free-form shapes. In *EG SBIM Workshop* (2008).

[BPF*03] BOUDON F., PRUSINKIEWICZ P., FEDERL P., GODIN C., KARWOWSKI R.: Interactive design of bonsai tree models. *Computer Graphics Forum 22*, 3 (2003), 591–599.

[CNX*08] CHEN X., NEUBERT B., XU Y.-Q., DEUSSEN O., KANG S. B.: Sketch-based tree modeling using markov random field. In *ACM Trans. Graph. (SIGGRAPH Asia 2008)* (2008).

[DC96] DOUADY S., COUDER Y.: Phyllotaxis as a dynamical self organizing process. part 1: The spiral modes resulting from time-periodic iterations. *Journal of Theoretical Biology 178* (1996), 255–274.

[Dig83] DIGGLE P.: *Statistical analysis of spatial point patterns*. Academic Press, London, UK, 1983.

[GBYC01] GUÉDON Y., BARTHÉLÉMY D., Y. C., COSTES E.: Pattern analysis in branching and axillary flowering sequences. *Journal of Theoretical Biology 212*, 4 (2001), 481–520.

[IMT99] IGARASHI T., MATSUOKA S., TANAKA H.: Teddy: a sketching interface for 3d freeform design. In *Siggraph 1999* (1999), pp. 409–416.

[IOI06] IJIRI T., OWADA S., IGARASHI T.: Seamless integration of initial sketching and subsequent detail editing in flower modeling. *Computer Graphics Forum 25*, 3 (2006), 617–624.

[LD99] LINTERMANN B., DEUSSEN O.: Interactive modeling of plants. *IEEE Comp. Graph. Appli. 19*, 1 (1999), 56–65.

[NFD07] NEUBERT B., FRANKEN T., DEUSSEN O.: Approximate image-based tree-modeling using particle flows. *ACM Trans. Graph. 26*, 3 (2007), 88+.

[OOI05] OKABE M., OWADA S., IGARASH T.: Interactive design of botanical trees using freehand sketches and example-based editing. *Computer Graphics Forum 24*, 3 (2005), 487–496.

[Opp86] OPPENHEIMER P. E.: Real-time design and animation of fractal plants and trees. *Siggraph 1986 20*, 4 (1986), 55–64.

[PL90] PRUSINKIEWICZ P., LINDENMAYER A.: *The algorithmic beauty of plants*. Springer Verlag, 1990.

[PMKL01] PRUSINKIEWICZ P., MÜNDERMANN L., KARWOWSKI R., LANE B.: The use of positional information in the modeling of plants. *SIGGRAPH 2001 22*, 4 (2001), 289–300.

[Pow98] POWELL W. F.: *Drawing Trees*. Walter Foster Publishing, Laguna Hill, CA, 1998.

[Pra97] PRASAD L.: Morphological analysis of shapes. *CNLS Newsletter*, 139 (1997).

[Rip79] RIPPLEY B.: Simulating spatial patterns: dependent samples from a multivariate density. *Applied Statistic 28* (1979), 109–112.

[SKP06] SMITH R. S., KUHLEMEIER C., , PRUSINKIEWICZ P.: Inhibition fields for phyllotactic pattern formation: a simulation study. *Canadian Journal of Botany 84* (2006), 1635–1649.

[SRDT01] SHLYAKHTER I., ROZENOER M., DORSEY J., TELLER J.: Reconstructing 3d tree models from instrumented photographs. *IEEE Comp. Graph. Appli. 21*, 3 (2001), 53–61.

[TFX*08] TAN P., FANG T., XIAO J., ZHAO P., QUAN L.: Single image tree modeling. In *ACM Trans. Graph. (SIGGRAPH Asia 2008)* (2008).

[TZW*07] TAN P., ZENG G., WANG J., KANG S. B., QUAN L.: Image-based tree modeling. *ACM Trans. Graph. 26*, 3 (2007), 87+.

[WP95] WEBER J., PENN J.: Creation and rendering of realistic trees. *Siggraph 1995* (1995), 119–128.