Realistic Plant Modeling from Images based on Analysis-by-Synthesis

Jérôme Guénard, Géraldine Morin, Frédéric Boudon, Vincent Charvillat

Realistic Plant Modeling from Images Based on Analysis-by-Synthesis

Jérôme Guénard[1], Géraldine Morin[1],
Frédéric Boudon[2], and Vincent Charvillat[1]

[1] IRIT - VORTEX - University of Toulouse
[2] INRIA - Cirad - Montpellier

Abstract. Plants are essential elements of virtual worlds to get pleasant and realistic 3D environments. Even if mature computer vision techniques allow the reconstruction of challenging 3D objects from images, due to high complexity of plant topology, dedicated methods for generating 3D plant models must be devised. We propose an analysis-by-synthesis method which generates 3D models of a plant from both images and a priori knowledge of the plant species.

Our method is based on a skeletonisation algorithm which allows to generate a possible skeleton from a foliage segmentation. Then, we build a 3D generative model, based on a parametric model of branching systems that takes into account botanical knowledge. This method extends previous works by constraining the resulting skeleton to follow a natural branching structure. A first instance of a 3D model is generated. A reprojection of this model is compared with the original image. Then, we show that selecting the model from multiple proposals for the main branching structure of the plant and for the foliage improves the quality of the generated 3D model. Varying parameter values of the generative model, we produce a series of candidate models. A criterion based on comparing 3D virtual plant reprojection with the original image selects the best model. Finally, results on different species of plants illustrate the performance of the proposed method.

1 Introduction

Procedural methods to generate plant models can build a complex plant architecture from few simple rules [1]. In his pioneering work [2], Lindenmayer proposes the formalism of L-systems as a general framework. By carefully parameterising these rules, it is possible to achieve a large variety of realistic plant shapes [3,4]. However, a strict recursive application of rules leads to self-similar structures and thus, to enhance realism, irregularities may be generated through probabilistic approaches [5,1]. Adjusting stochastic parameters to achieve realistic models requires intensive botanical knowledge [6]. Another approach consists in modeling plant irregularities as a result of the competition for space between the different organs of the plants [7]. In this case, the volume of a plant is specified by the user and a generative process grows a branching structure with branches competing between each other. Competition can be biased to favor certain types of

structures. However, automatic control of competition parameters to achieve a given shape is still complicated.

All these first works are derived from computer graphics community. Other approaches use information provided by images to increase the degree of realism. A couple of research directions should be investigated. Clearly, a plant should follow the biological property of its species and also ressemble a picture of an existing instance. That is typically the subject of our work. Our idea is not to exactly reconstruct the plant from an image, including its hidden parts (which seems impracticable) but rather to drive the instantiation of the plant 3D model by minimising the difference between its reprojection and the original plant in the image.

Fig. 1. On the left, an original image of a vine plant before and after a metric rectification. In the middle, a possible architecture of the branching extracted with our skeletonisation method. At the right, a corresponding 3D model of the plant.

Unlike existing methods detailed in section 2, ours must be able to get a 3D model of a plant without any human interaction from images with possibly no visible branches. By integrating biological knowledge of the plant species, we propose a simple fully-automatic process to extract the structure of a plant from the shape of its foliage. The picture can be taken in arbitrary conditions and may be of poor quality (for example, in the vine case, the image is degraded after a metric rectification due to the assumption that all the principal branches are in a plane as shown at the left on the Fig. 1). We start by presenting a new skeletonisation algorithm in section 3 in the vine case (as we can see in the middle of the Fig. 1) and we explain a possible extension to our skeletonisation method for other kinds of plants with 3D branching architecture in section 3.5. Then a 3D model is generated thanks to our 3D generative model (section 4). Finally, an analysis-by-synthesis scheme allows to improve this reconstruction insuring that the foliage model reprojection matches closely the original foliage like explained in section 5 (right on the Fig. 1). The last section shows results and validation comparing with data provided by experts.

2 State of the Art: Generating Plants from Images

Realistic plants are challenging objects to model and recent advances in automatic modeling can be explained by the convergence of computer graphics and

computer vision [8]. We start this state of the art with the first method of plant modeling from images. Then, we continue with the ones starting by reconstructing clouds of 3D points. After, we talk about other methods using several images to finish with approaches using a single image as ours.

A pioneering work on the reconstruction of trees from images was made by Shlyakhter *et al.* [9] who reconstruct the visual hull of the tree from silhouettes deduced from the images. A skeleton is computed from the hull using a Medial Axis Transform (MAT) and is used as main branches. Branchlets and leaves are then generated with an L-system. The skeleton determined from the MAT does not necessarily look like a realistic branching system. Also, the density of the original tree is not taken into account.

Quan *et al.* [10,11] and Tan *et al.* [12] also use multiple images to reconstruct a 3D model of trees or plants. In order to avoid searching features correspondances in different images, they use views close to each other (more than 20 images for any plants). Thus, they obtain a quasi-dense cloud of points by structure from motion. For simple plants, a parametric model is first fitted on each set of points representing a leaf. They then generate branches based on information given by the user. For trees, they start by reconstructing visible branches to create branch pattern that they combine in a fractal way until reaching leaves. Reche-Martinez *et al.* [13] propose another reconstruction from multiple images, based on billboards. Neubert *et al.* [14] construct a volume encompassing the plant in the form of voxels using image processing techniques and fill it with particles. Particles paths toward the ground and a user given general skeleton define the branching system.

Wang *et al.* [15] model different species of trees using images of tree samples from the real world which are analysed to extract similar elements. A stochastic model to assemble these elements is also derived and parameterised from the image. The resulting model can generate many similar trees. The goal in this case is not necessarily to reconstruct a specific tree instance corresponding to an image. Similarly, Li *et al.* [16] propose a probabilistic approach to reconstruct a tree parameterized from videos. For these methods, the only source of information is the given images leading to template branching patterns. If the set of patterns is rich enough, it will produce aesthetically pleasing results, but without guarantee to be representative of its species. Additionally, a user must specify a draft of the structure on the image to avoid segmentation. Talton *et al.*, in [17], propose to fit a grammar-based procedural methods using Markov Chain Monte Carlo technique to model objects from a 2D or 3D binary shape. Their results are aesthetically very convincing but their optimisation routine requires long computation time.

Other approaches explore the use of a single image [18,19]. In [18], from a manual segmentation of the plant in the image, they extract a skeleton. A 3D representation of the branches is derived from visible parts, then the leaves are added. Here, an user sketching step is required. In [19], a graph topology is first extracted from a single image of a branching system (a tree without foliage). Then the 3D tree model is reconstructed by rotating the branches.

In general, methods of the literature, such as [12] and [18] require visible branches to learn about the structure of the skeleton. In our case, branches are directly derived from the foliage structure. Fig. 16 (top row) shows branching structures devised manually by experts from image: we see that the branches are deduced on one hand from the knowledge of a space filled by a branch and its attached leaves and on the other hand from the silhouette of the foliage. We propose a generalised recursive skeletonisation algorithm together with an analysis-by-synthesis mechanism to determine the branches and their attached foliage that is the 3D model. Our approach is fully-automatic, that is, does not require any user interaction.

3   Skeletonisation

3.1   General Field Skeletonisation Method

Skeletonisation is a classical topic in image processing. We have followed and completed the analysis of different approaches as proposed in [20] and as illustrated in Tab. 1. This table summarises the different properties that skeletons respect like the thinness or the robustness and compares them to our requirements shown in the left column. All these properties are detailed in [20].

'Homotopic' and 'connected' are topology preserving properties. As the foliage may have holes, we prefer not to be attached to its topology. For the same reason, reconstruction is not very relevant. On the opposite robust is an important property, since the foliage may not be stable (for example, the foliage may change in case of wind). Being reliable means that any point in the foliage should be visible from a point on the skeleton. For our setting, it is not necessary to request a branch in each bump of silhouette. The transformation invariance property (in particular affine invariance) is not important in our setting as we do depend on orientation. For example, vine branches grow vertically. In our case, the smooth property is important to get realistic branches but we do not need a centred skeleton. Finally, we shall see that for plants more complex than vines, the hierarchic nature of the skeleton is used (section 3.5). For these reasons, we choose to adapt the general field method, and in particular the work of Cornea [21], since their approach fits the best our needs.

Cornea *et al.* original method [21] consists in computing the skeleton (Fig. 2 (c)) from a vector field (Fig. 2 (b)). For each interior pixel $\mathbf{p_i}$ of the binary shape $\mathcal{B}$, a force vector $\overrightarrow{\mathbf{f_i}}$ is computed as a weighted average of unit vectors to the boundary pixels:

$$\overrightarrow{\mathbf{f}_i} = \sum_{\mathbf{m_j} \in \Omega} \frac{1}{||\overrightarrow{\mathbf{m_j p_i}}||^2} \frac{\overrightarrow{\mathbf{m_j p_i}}}{||\overrightarrow{\mathbf{m_j p_i}}||}$$

where Ω contains the contour pixels $\mathbf{m_j}$ of $\mathcal{B}$ (Fig. 2 (a)). Then, points where the magnitude of the force vector vanishes, so-called *critical points* (Fig. 2 (b)), are connected by following the force direction pixel by pixel. The results of this method can be seen in Fig. 2. The problem here is that this method is not robust

Table 1. Summary of properties achievable by different skeletonisation methods. In green, the characteristics compliant with our needs.

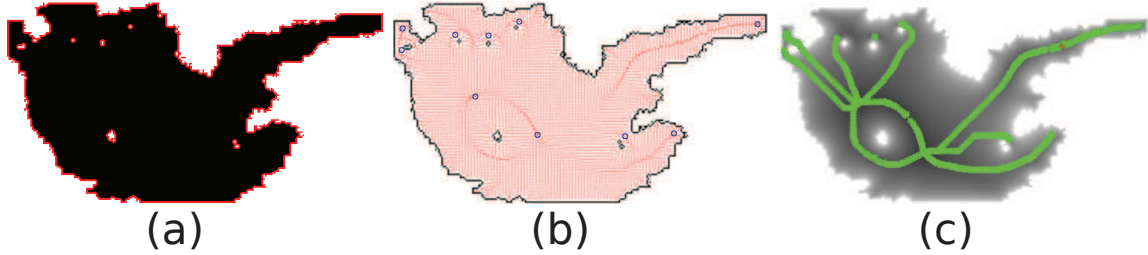|	Medial Axis	Thinning		Distance Field	Geometric		General Field		Our needs
G: general		G	[22]	G	G	[23]	G	[21]	
Homotopic	Y	Y	Y		Y	Y	N	N	**N**
Connected	Y	Y	Y			Y		N	**Y**
Reconstruction	Y	N	N		N	N	N	N	**N**
Robust	N	N	N	N	N	Y	Y	Y	**Y**
Reliable	Y		Y			N		N	**N**
Transf. Invariance	Y		N	Y		Y	Y	Y	**N**
Centred	Y		Y			Y		N	**N**
Smooth	N		N			N	Y	Y	**Y**
Hierarchic	Y	N	N			Y	Y	Y	**Y**

Fig. 2. Cornea *et al.* original method. (a) shape $\mathcal{B}$ with contours pixels $\in \Omega$ represented in red. (b) vector field with critical points in blue. (c) extracted skeleton in green.

to the holes in the binary shape. Furthermore, sometimes only one branch grows when two or more are required.

3.2 A New Computation of the Vector Field

By redefining the set of contour points, we manage to use Cornea's vector field method to get a realistic skeleton in 2D.

Based on botanical expertise, we assume that different branches of relatively similar size coexist and share the space of the plant crown. A large convex silhouette is usually explained by more than one branch. For the skeleton to reflect this hierarchy of branches, we propose a strategy to partition the foliage surface into subsets by positioning artificial *contour points* in the shape. Fig. 3 shows that by adding *contour points* within the shape, we define an appropriate branch set.

We compute a probability map $\mathcal{P}$ on $\mathcal{B}$ containing, for each interior point $\mathbf{p_i}$, the probability to be considered as a *contour point*. The new force vector $\overrightarrow{\mathbf{f_i}}$

Fig. 3. At the left, the skeleton (in green) extracted with Cornea's original method. A spatial partition can be generated (red lines) to constraint the skeleton to have more branches. At the right, the skeleton computed by Cornea's method when all red points are considered as contour points.

does not depend only on the points $\mathbf{m_j} \in \Omega$ but on all the points of $\mathcal{B}$. The new formula to compute the vector field is:

$$\overrightarrow{\mathbf{f_i}} = \sum_{\mathbf{m_j} \in \Omega} \frac{1}{||\overrightarrow{\mathbf{m_j p_i}}||^2} \frac{\overrightarrow{\mathbf{m_j p_i}}}{||\overrightarrow{\mathbf{m_j p_i}}||} + \sum_{\substack{\mathbf{p_j} \in \mathcal{B} \setminus \Omega \\ j \neq i}} \frac{\mathcal{P}_j}{||\overrightarrow{\mathbf{p_j p_i}}||^2} \frac{\overrightarrow{\mathbf{p_j p_i}}}{||\overrightarrow{\mathbf{p_j p_i}}||} \tag{1}$$

We can see that if $\mathcal{P}_j = 0$ for all the interior points, equation 1 is equivalent to Cornea's original computation whereas if some interior points have high probability they act as a repulsive force on the positioning of the branches.

3.3 Definition of the Probability Map

We assume here that n the number of branches is given. We compute the *probability map* $\mathcal{P}$ with an iterative algorithm. The first step is the choice of cuts in $\mathcal{B}$. The cuts are segments with one *starting point* and one *ending point* and represent the possible positions of the separations between the n branches in the shape. Assuming that the shoots grow vertically from the cane, we propose to place trivially the *ending points* $\mathbf{e_i}, i = 1..n-1$ of the cuts uniformly in the bottom of $\mathcal{B}$ (Fig. 4). Then, the *starting points* are computed one by one. To do that, we compute the DCE (*Discrete Curve Evolution*) of Ω as in [24]. It provides a simplified polygonal boundary composed of N vertices $(\mathbf{s_l})_{,l=1..N}$ like shown in Fig. 5. Usually, we choose $N = 2n$. An angle α_l can be associated with each vertex, representing clockwise angle between the 2 segments around the vertex. A set of points $(\mathbf{c_k})_{,k=1..K}$ uniformly discretises the polygon.

Then a new probability ρ_k to be a *starting point* is computed for each point $\mathbf{c_k}$ taking into account two values:

– the proximity to an inward angle:

$$\rho_k^1 \sim \sum_{l=1}^{N} \frac{1}{d(\mathbf{c_k}, \mathbf{s_l})} (1 - \frac{\alpha_l}{2\pi})$$
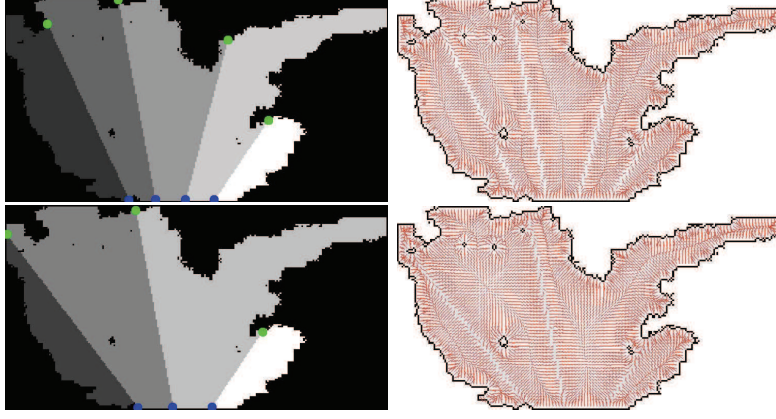
Fig. 4. At the left, examples of cuts with $n = 4$ and $n = 5$ branches. Ending points are represented in blue and starting points in green. At the right new vector fields.

Fig. 5. DCE algorithm examples. At the left, the original image with the contour points around the foliage. The two other images show the DCE algorithm with a 24-point polygon in the middle and a 8-point polygon on the right. The inward angles are represented in blue.

  – the distance along a boundary to the set $\mathcal{H}$ of already chosen *starting points*:

$$\rho_k^2 \sim \min_{\mathbf{c} \in \mathcal{H}} d(\mathbf{c_k}, \mathbf{c}).$$

The mix probability ρ_k is proportional to $\phi(\rho_k^1, 1, \sigma) + \phi(\rho_k^2, 1, \sigma)$ where $\phi(., 1, \sigma)$ represents the gaussian function with a mean equals to 1 and a standard deviation equals to σ (here, $\sigma = 0.4$).

A *starting point* $\{\mathbf{c_k}\}$ is selected according to the probability ρ_k. Then it is associated with an *ending point* $\mathbf{e_i}$ and accepted if:

$$\left| \frac{\#pixels \in \mathcal{B} \ on \ the \ left \ of \ (\mathbf{e_i c_k})}{\#pixels \in \mathcal{B}} - \frac{i}{n} \right| \leq \tau \qquad (2)$$

where τ is a parameter allowing the created partitions of the binary shape to have the same size or not. In the vines case, $\tau = 15\%$.

Finally, when all the cuts have been accepted, the new vector field is computed like shown at the right part of Fig. 4.

3.4   Adjusting First Order Branches

We now have a vector field coherent with the n branches assumption. We want to extract branches from this vector field. For each row i of the image and each

area p of the partition that we can see at the left part of Fig. 4, we extract the attracting point $\mathbf{a_i^P}$ which is the one with the smallest vector norm. Each branch $\mathbf{b_p}$ is a Catmull-Rom curve adjusted on the attracting points $\mathbf{a_i^P}$, using a least square criterion.

An example of skeleton can be seen in Fig. 6.

Fig. 6. A final skeleton with 5 branches

The following algorithm summarises the proposed approach:

ALGORITHM:
input: binary shape $\mathcal{B}$ and number of branches n.
output: a skeleton model of the plant consisting in n branches.

---

* Compute the DCE of $\mathcal{B}$ with $N = 2n$ vertices and extract candidate $\{\mathbf{c_k}\}$ uniformly along the polygon.
* Initialise $\mathcal{H} = \emptyset$.
* Place the *ending points* $\{\mathbf{e_i}, \mathbf{i} = \mathbf{1}..\mathbf{n} - \mathbf{1}\}$ in the bottom of $\mathcal{B}$.
* For i from 1 to $n - 1$
  · Compute ρ_k for each $(\mathbf{c_k})$.
  · while (!equation (2))
    ∘ Randomly select a *starting point* $\mathbf{c}$ among $\{\mathbf{c_k}\}\backslash\mathcal{H}$ in function of ρ_k and an *ending point* among $\{\mathbf{e_i}\}$.
  · $\mathcal{H} = \mathcal{H} \cup \mathbf{c}$.
* Compute the probability map $\mathcal{P}$.
* Compute the vector field using equation (1).
* Extract the points $\mathbf{a}_i^p$ for each row i of each area p of the partition.
* Adjust a branch in each area p of the partition.
* For each node of each branch, extract a foliage width.

3.5 Higher Order Branches and Depth Information

In the case of the vines, all the main branches are assumed in a same plane and there are only first order branches. We adapt this planar setting to a more recursive structure and to a 3D setting where rotational symmetry is assumed, like in typical monopodial plants (i.e. plants organised around a main trunk).

Iterative Skeletonisation Algorithm
An automatic colour based segmentation computes a 2D binary shape from the foliage. Then, the branches are extracted using a modified version of our algorithm presented above. The *ending points* are placed on the vertical line passing through the trunk of the *monopodial tree* and the condition (2) is replaced by a condition checking that the angle between the cut and the trunk is coherent (i.e. around $\frac{\pi}{2}$ in the bottom of the tree, $\frac{\pi}{6}$ in the top and with an angle computed linearly between these two values for an intermediate cut). This algorithm is applied recursively to get second order branches for each partition. We can see an example of cuts with a *Liquidambar tree* in Fig. 7.

Fig. 7. An example of skeleton for the Liquidambar tree

Depth of Branches
To generate 3D information, we drew inspiration from Zeng *et al.* [19] and Okabe *et al.* [25]. The goal is to deduce depth information for the branches in the 2D skeleton to make a realistic plant from other views, preserving the appearance from the original viewpoint as it is shown in Fig. 8. First, we compute the convex hull of our 2D skeleton. Then, revolving this convex hull around the line passing through the trunk, we obtain a encompassing volume of the plant. Considering an orthographic projection onto the ground, for each branch which does not touch the 2D convex hull, we change depth information for that the end of this branch touches the boundary of the bounding volume. We have two possibilities, at the front or at the back. We choose the one which maximises the angles between the projections of all the branches to the ground, adding the branch one by one.

4 3D Generative Model

Now we have a possible structure of the plant, the next step is to generate a 3D model of this plant. To do that, we need to build a 3D generative model thanks to all the *a priori* knowledge of the plant.
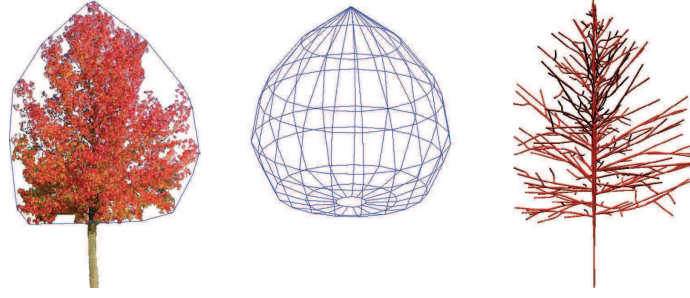
Fig. 8. At the left, we can see the 2D convex hull of the foliage, in the middle, the-bounding volume and at the right, the final 3D skeleton of a Liquidambar tree

In our work, we combine procedural methods to generate a plant and image based approaches. Procedural methods makes it possible to take into account botanical constraints such as possible regular arrangements of organs (for instance leaves). The procedural model uses stochastic parameters in the positionning of the branchlets and the leaves.

We choose to generate the constrained model with L-systems, using the L-Py modeller [26]. An L-system [1] is a formal grammar, most commonly used to model the growth processes of plant development. The main idea of L-systems is to rewrite a string of modules representing the structure of the plant. Rewriting rules express the creation and change of state of the various modules of the plant over time. Our model include a deterministic part which is controlled by the understanding of plant images and a stochastic part to allow a more realistic result. The model is deduced by both learning from a large number of plants and also knowledge given by specialists.

We choose to generate our model in two stages: the branching system model and the foliage.

Branching System Model
Each branch is set by a number of 3D nodes which are the control nodes of the branch. From each of these nodes one or more lateral branches of the same nature may grow. Then, to model the 3D structure, each branch is a generalised cylinder along a curve passing through all nodes. A B-Spline curve is built with a local interpolation scheme of degree 3 [27]. A radius is assigned to each of these nodes to determine the radius of the generalised cylinder in these nodes. This radius is linearly interpolated between two nodes. Textures taken from real images are then applied to branches.

Foliage Model
We begin by extracting leaf textures from real images. At each node defining the branches structure is also assigned a value R which is the radius of the cylinder encompassing the leaves. Thus, to model the foliage, *branchlets* are generated randomly along the branches. Stems are placed along main branches and *branchlets*. Their density and their length is a random variable distributed normally with mean R and standard deviation $\frac{R}{4}$. On each of these stems a leaf modeled by a Bezier surface is placed on which a randomly chosen texture is mapped (Fig. 9b.).
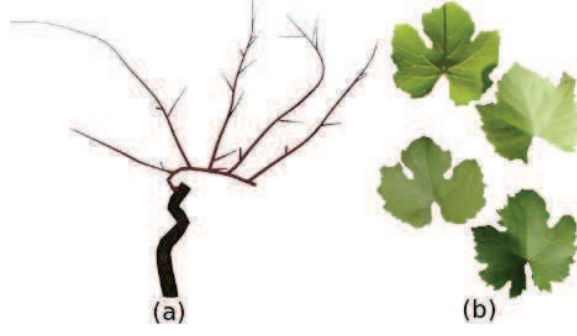
Fig. 9. Example with the case of vine. (a) Branch structure modelisation. (b) Textures of vine leaves.

5 Reprojection Criterion

The last step goals to evaluate the quality of the 3D reconstructed 3D model. We reproject the 3D model in an image with the same viewpoint of the original image. We obtain a binary shape $\mathcal{I}_i$ (1 if foliage, 0 elsewhere). In the same way, the original segmented foliage forms a second binary image $\mathcal{B}$. The error reprojection is computed as:

$$error_i = \frac{\#((\mathcal{I}_i - \mathcal{B})^2 == 1)}{\#pixels(\mathcal{B})} \tag{3}$$

Fig. 10 illustrates the comparison between the projections and the original image.
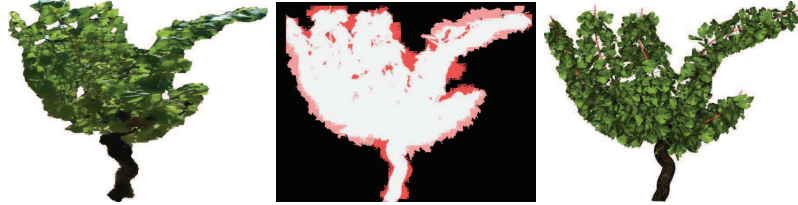

Fig. 10. At the left, the original image. At the right, the reprojected model. In the middle, the projection errors map. White pixels correspond to pixels where the original image and the reprojected one are superposed and red pixels are *wrong* pixels.

The idea now is to improve the proposed 3D model using an analysis-by-synthesis strategy which allows to merge information from the *a priori* botanical knowledge and from the image. An increasing number of authors propose to use *external* knowledge for easing reconstruction from images. Indeed providing knowledge about the scene to be reconstructed simplifies the image processing steps. In [28], Tu *et al.* define generative models for faces, text, and generic regions which are activated by bottom-up proposals learnt using probabilistic methods. These proposals are then accepted or rejected using a stochastic criterion. Yuille *et al.* [29] claim that this approach, which allows to deal with

the complexity of natural images, has intriguing similarities to the brain. They present a method where low level features are used to make bottom-up proposals, finally validated by high-level models. In a similar analysis-by-synthesis method where *a priori* knowledge consists in geometric and mechanical properties, Gupta *et al.* [30] iteratively make proposals for interpreting parts (blocks) of the image. We use a similar approach, but not iterative.

In our case, we give more freedom to the generative model. For example, we do not impose the number of branches of the plant that we do not *a priori* know. Thanks to all the knowledge of the plant, we can model each parameter (like the position of the cuts, the leaves densities or the number of branches) by a random variable, and thus generate numerous models.

We select the best candidate proposed by the generative model using the following formula:

$$\mathcal{M}_{i_0} = \underset{\mathcal{M}_i}{\operatorname{argmax}}\, p(\mathcal{M}_i|\mathcal{I}_i) = \underset{\mathcal{M}_i}{\operatorname{argmax}}\, p(\mathcal{M}_i)p(\mathcal{I}_i|\mathcal{M}_i) \tag{4}$$

We choose $p(\mathcal{I}_i|\mathcal{M}_i) = 1 - error_i = 1 - \frac{\#((\mathcal{I}_i - \mathcal{B})^2 == 1)}{\#pixels(\mathcal{B})}$ and $p(\mathcal{M}_i)$ is a product of terms which are probabilities function of all the knowledge of the plant. For the vine case example, one of the term of $p(\mathcal{M}_i)$ is a gaussian representing the probability of the number of shoots. Fig. 11 shows different error maps. White pixels correspond to pixels where the reprojected model and the original image are superposed.
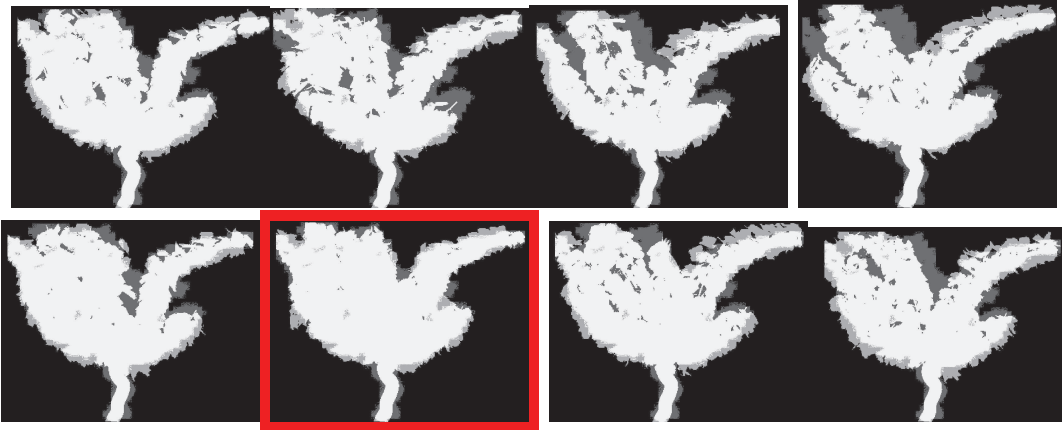

Fig. 11. Different errors maps with different numbers of branches, different distributions of leaves and different densities. The map outlined in red is the error map of the selected 3D model because there is the smallest percentage of gray pixels.

6 Results and Validation

Our method has been tested on a large number of images and videos. Some results are shown in Fig. 15, 14 and 12. To validate our model, we use the error criterion of equation (3). The average error for the case of vines is 6.9%, 7.2% for

Fig. 12. Vine plants modelisation. At the top the original images. At the bottom, rendering of automatically generated vine models using our approach.
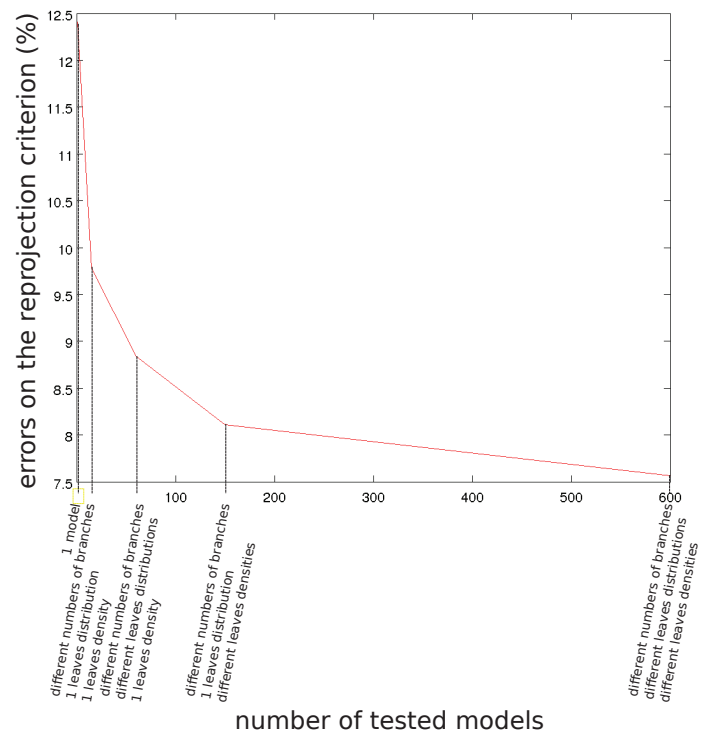

Fig. 13. Reprojection errors according to the number of tested models. Different variations are tested as the number of branches or the leaves density. For example, if we test different 3D models with only one leaves distribution and 1 leaves density, the error is almost 10%. Then, this error decreases when we add different leaves distributions or different leaves density.

Fig. 14. Walnut. At the left, the original image. At the right, the 3D model with the same viewpoint.
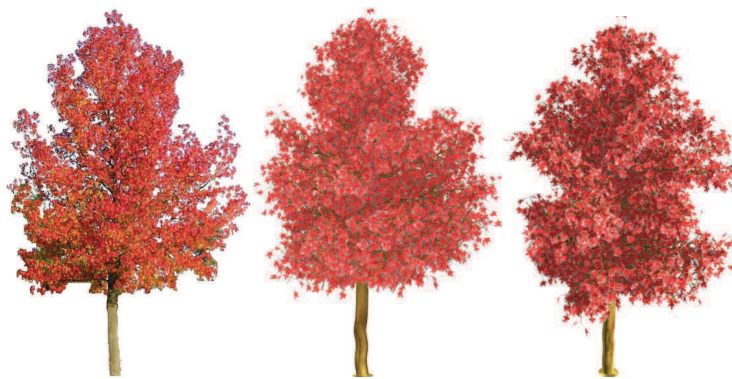

Fig. 15. Liquidambar. At the left, the original image. In the middle, the 3D model with the same viewpoint. At the right, the 3D model with an other viewpoint.

Fig. 16. Expert skeletons. At the top, a viticulture expert has drawn skeletons on vine images (in red). At the bottom, the projections of the skeletons of our method (in yellow).

Table 2. The first row represents the number of the vine image. The second and the third rows represent the numbers of shoots drawn by the experts. The last row represent the number of shoots of the 3D models generated with our method from these images.

Image	1	2	3	4	5	6	7	8	9
First expert estimation	3	6	5	5	6	5	3	7	7
Second expert estimation	2	6	4	5	4	4	2	5	4
Our method	2	5	4	5	6	6	3	5	6

the Walnut and 8.5% for the Liquidambar. In Fig. 13, we show the reprojection error according to the number of tested models for an example of vine (the third example of Fig. 12). It is interesting to note that the greater is the number of tested models, the lower is the reprojection error. The curve decreases very quickly between 1 and 15, fairly quickly until 100. This proves the effectiveness of our skeletonisation method which restricts significantly the search space. This curve illustrates the importance to test several models but of course, the final selected model is not necessarily the last one.

A second validation is to compare our solution to the one provided by viticulture experts (Fig. 16). It seems difficult to find a significant measure by comparing the ground truth to our skeletons. Furthermore, for us, the most important is the final appearance with leaves. Indeed, for the first example in Fig. 16, our algorithm found a very similar skeleton to the expert one at the left but it is not the one which has been validated by our method at the right. So, we used our algorithm on the drawn ground truth skeletons with different leaves distributions and different leaves densities to find the best 3D model. The improvement of the reprojection criterion in comparison to our automatically generated skeleton is only 0.2% in average. This small difference proves the performance of our method which does not require human intervention.

In Tab. 2, we can see the number of shoots drawn by two differents experts from vines images. The last row shows the number of shoots of the 3D models generated with our method from the same images. We almost reconstruct a 3D model with the same number of shoots drawn by one of the two experts.

For the 3D case, we have shown that our method can reconstruct realistic 3D trees from a single image. However, the branching system is mostly based on branches.

7 Conclusion

Combining analysis and synthesis, we have proposed a new fully-automatic method of plant modeling from a low resolution image without any branching pattern unlike [12,19].

The leading contribution of this paper is a new skeletonisation algorithm able to extract the structure of a plant from an image of its foliage. Then, we built 3D parametric generative models for different plants using the knowledge about the species. A final analysis-by-synthesis step improves the quality of

the final 3D model by comparing the original image with a large number of 3D models generating varying different parameters of the 3D generative model. The skeletons are used to make proposals to the 3D generative model. The reprojection criterion insures the similarity between the proposed 3D model and the original image. We further validated our proposed model by comparing it to ground truth given by experts in the case of vine.

In future work, we are first interested in extending our setting to non monopodial plant. The current algorithm can be applied to non monopodial plants, like the Walnut, but the result is still not very satisfying. Moreover, we could investigate the automatisation of the use of *a priori*. Indeed, the construction of the generative model could be done by learning from a large data set avoiding the necessity prior knowledge on the branching structure of the plant species. The process would be evolving in loop, where the analysis could give feedback to the generative models.

References

1. Prusinkiewicz, P., Lindenmayer, A.: The algorithmic beauty of plants. Springer (1990)
2. Lindenmayer, A.: Mathematical models for cellular interaction in development: Parts i and ii. Journal of Theoretical Biology 18 (1968)
3. Weber, J., Penn, J.: Creation and rendering of realistic trees. In: Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1995, pp. 119–128. ACM, New York (1995)
4. Deussen, O., Lintermann, B.: Digital Design of Nature: Computer Generated Plants and Organics. Springer (2005)
5. de Reffye, P., Edelin, C., Françon, J., Jaeger, M., Puech, C.: Plant models faithful to botanical structure and development. In: Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1988, pp. 151–158. ACM, New York (1988)
6. Markov and semi-markov switching linear mixed models used to identify forest tree growth components. Biometrics (2009)
7. Palubicki, W., Horel, K., Longay, S., Runions, A., Lane, B., Měch, R., Prusinkiewicz, P.: Self-organizing tree models for image synthesis. SIGGRAPH, 1–10 (2009)
8. Quan, L.: Image-based Plant Modeling. Springer (2010)
9. Shlyakhter, I., Rozenoer, M., Dorsey, J., Teller, S.: Reconstructing 3d tree models from instrumented photographs. IEEE Comput. Graph. Appl., 53–61 (2001)
10. Quan, L., Tan, P., Zeng, G., Yuan, L., Wang, J., Kang, S.B.: Image-based plant modeling. ACM TOG, 599–604 (2006)
11. Quan, L., Wang, J., Tan, P., Yuan, L.: Image-based modeling by joint segmentation. IJCV, 135–150 (2007)
12. Tan, P., Zeng, G., Wang, J., Kang, S.B., Quan, L.: Image-based tree modeling. ACM TOG 87 (2007)
13. Reche-Martinez, A., Martin, I., Drettakis, G.: Volumetric reconstruction and interactive rendering of trees from photographs. ACM TOG, 720–727 (2004)
14. Neubert, B., Franken, T., Deussen, O.: Approximate image-based tree-modeling using particle flows. ACM TOG (Proc. of SIGGRAPH) (2007)

15. Wang, R., Hua, W., Dong, Z., Peng, Q., Bao, H.: Synthesizing trees by plantons. Vis. Comput. 22(4), 238–248 (2006)
16. Li, C., Deussen, O., Song, Y.Z., Willis, P., Hall, P.: Modeling and generating moving trees from video. ACM Trans. Graph. 30(6), 127:1–127:12 (2011)
17. Talton, J.O., Lou, Y., Lesser, S., Duke, J., Měch, R., Koltun, V.: Metropolis procedural modeling. ACM Trans. Graph. 30(2), 11:1–11:14 (2011)
18. Tan, P., Fang, T., Xiao, J., Zhao, P., Quan, L.: Single image tree modeling. ACM SIGGRAPH, 1–7 (2008)
19. Zeng, J., Zhang, Y., Zhan, S.: 3d tree models reconstruction from a single image. In: ISDA, pp. 445–450 (2006)
20. Cornea, N.D., Silver, D., Min, P.: Curve-skeleton properties, applications, and algorithms. TVCG, 530–548 (May 2007)
21. Cornea, N.D., Silver, D., Yuan, X., Balasubramanian, R.: Computing hierarchical curve-skeletons of 3d objects. The Visual Computer (2005)
22. Yang, X., Bai, X., Yang, X., Liu, W.: An efficient quick thinning algorithm. In: Proceedings of the 2008 Congress on Image and Signal Processing, CISP, vol. 3, pp. 475–478. IEEE Computer Society, Washington, DC (2008)
23. Bai, X., Latecki, L.J., Society, I.C., Yu Liu, W.: Skeleton pruning by contour partitioning with discrete curve evolution. IEEE Trans. Pattern Anal. Mach. Intell. 29, 449–462 (2007)
24. Latecki, L.J., Lakämper, R.: Shape similarity measure based on correspondence of visual parts. PAMI, 1185–1190 (2000)
25. Okabe, M., Owada, S., Igarashi, T.: Interactive design of botanical trees using freehand sketches and example-based editing. Computer Graphics Forum 24(3), 487–496 (2005)
26. Boudon, F., Pradal, C., Cokelaer, T., Prusinkiewicz, P., Godin, C.: L-py: an l-system simulation framework for modeling plant development based on a dynamic language. Frontiers in Plant Science 3(76) (2012)
27. Piegl, L., Tiller, W.: The NURBS book, 2nd edn. Springer-Verlag New York, Inc. (1997)
28. Tu, Z., Chen, X., Yuille, A.L., Zhu, S.C.: Image parsing: Unifying segmentation, detection, and recognition. In: Ponce, J., Hebert, M., Schmid, C., Zisserman, A. (eds.) Toward Category-Level Object Recognition. LNCS, vol. 4170, pp. 545–576. Springer, Heidelberg (2006)
29. Yuille, A., Kersten, D.: Vision as bayesian inference: Analysis by synthesis? introduction: Perception as inference. Psychology, 1–15 (2006)
30. Gupta, A., Efros, A.A., Hebert, M.: Blocks world revisited: Image understanding using qualitative geometry and mechanics. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part IV. LNCS, vol. 6314, pp. 482–496. Springer, Heidelberg (2010)