6 Rule-Based Object Production

Interactive Modeling

The previous chapter demonstrated that rule-based procedures are a very powerful method for the production of a variety of plants. Nevertheless, there are many alternative modeling procedures. On one hand, this is due to the wide range of modeling requirements; on the other hand, it is because the controlling mechanisms of the L-systems are not very intuitive. Even for an experienced user, the rendering of a specific plant is a cumbersome process. The recent developments in the L-system methodology seem to somewhat lessen this problem; however, an efficient system practical for all users has not yet been developed.

Procedural methods display the exact opposite characteristics: although usually only a very limited number of plants can be modeled, the handling of the procedures and their parameterization is straightforward and intuitive with such a method.

A combination of the two approaches should combine the intuitiveness of procedural modeling with the power of rule-based methods. In fact, the Xfrog modeling system [40, 41, 121, 122] successfully implemented this specific combinatorial approach on the basis of so-called rule-based object production.

Here a plant is represented by the combination of components. The components generate parts of the plant's geometry, such as leaves, stems or simple geometric primitives, by using procedural methods. Multiplication components multiply the generated geometry of other components, and, in this sense, implement a rule-based system (see below). Another type of component is used for global modeling. The user can trigger the parameters of the individual components using special graphical user interfaces.

For instance, the component for modeling leaves uses a polygonal curve to specify the outline of the leaves. The vertical and longitudinal curvature can be adjusted using sliding controls along the leaf axes. This type of interface was one inspiration for the recently presented interactive editing possibilities for L-systems, which were introduced in Sect. 5.10.

In many cases, parts of a plant must be distributed algorithmically. A classical example is again the arrangement of the seeds of the sunflower according to the

Golden Angle. Such distributions are produced by using multiplication components whose algorithms are parameterized over the number of objects which are to be generated, their distribution characteristics as well as their orientation.

By connecting the component prototypes, the plant is defined as a directional graph (so-called p-graph). The graph represents the rule system; its edges describe production dependences: Once the geometry of a father component is generated, it invokes the production of the geometry for all its children, until the entire description graph is processed. Hereby the components can be freely linked, and recursions are also possible.

p-graph and i-tree →

The p-graph is traversed for the production of the geometry, and the so-called i-tree is built. This is a temporary tree consisting of component instances from which the geometrical data is generated. This intermediate step is necessary since in the p-graph the structure of a plant is represented in two different ways: by the connectivity structure of the edges and by the multiplication components. Despite this intermediate step, the geometry production is sufficiently fast, so that also complex objects can be modeled interactively on the screen. Because of these modeling options, currently several thousand plants have been generated which can be used for various purposes.

Parallel to the intuitive operation of the system by components, the double representation of the structure is the main difference to classical rule systems such as L-systems or the graph-based object instancing paradigm [83]. Prusinkiewicz provides in his “virtual laboratory” for L-systems [166] the possibility of executing external procedures for the production of plant parts. However, in the rule-based object production the algorithms are an integral part of the modeling and thereby allow for the efficient and flexible production of many branching structures. In order to demonstrate this difference clearly, the process of multiplication is described in more detail in the following.

6.1 Algorithmic Multiplication

If a multiplication component is part of a p-graph, then during the production of the i-tree the corresponding subtree is generated as many times as is indicated by the corresponding parameter of the multiplication component. Hereby all copies of the child components are connected with the copy of the multiplication component.

prototype processing →

During production of the i-tree the component prototypes are transformed into instances. The difference between a prototype and an instance lies in the definition of most of the parameters that are passed on to the child components, such as their local coordinate system. In a component prototype these parameters are stored as ranges, while the instances in the i-tree receive fixed and individually computed values during multiplication through interpolation.

If, for example, the sizes of the children are set in the range $[v_0, v_1]$ in a multiplication component, then the ith child of n is assigned the value $v_i = v_0 + (i-1)(v_1 - v_0)/(n-1)$ with $i = 1, ..., n, n \geq 2$ by interpolation. If necessary, before their transmission, a user-defined function can be applied to the interpolation values. In this way trigonometric functions and/or random values can be integrated into the models.

Similar to the procedural multiplications during the production of the i-tree, recursions are treated. In each component the maximum recursion depth is defined as a parameter. If a recursive definition must be expanded, after generating an appropriate number of instances, the expansion is stopped. Referring back to the statements concerning recursive algorithms in Sect. 4.6, it is evident that this form of modeling is not required frequently.

After the i-tree has been constructed, a depth-first traversal starting at the root is performed up to the leaves. The root component generates its geometry and forces all its children to do the same. This is recursively continued, until the entire i-tree has been processed.

$\leftarrow$ example

The following further demonstrates the process of geometry production: Let A be the component prototype of the root of the p-graph, and B be the prototype of a geometry-generating component, which produces a stem. Let C be a multiplication component, which is forced to generate three instances of the successor, but no geometry. Component prototype D generates likewise a small stem. In the example a recursion depth of three is set by the user. The graph is defined so that a recursion is defined on D.

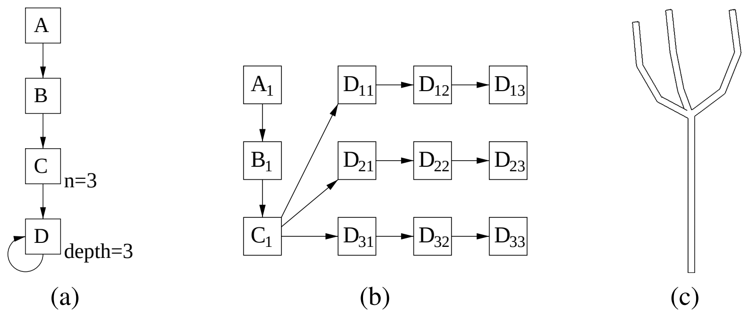

Figure 6.1
Production of geometry: (a) the user defines the p-graph; (b) to generate the geometry, a temporary i-tree is constructed; (c) resulting geometry

Figure 6.1a shows the p-graph, Fig. 6.1b the corresponding i-tree in which the instances of each component prototype X are marked as X_i. The recursion on D generates in the i-tree three instances D_1, D_2 and D_3, which are connected. These subtrees are copied by the multiplication component C three times, where each instance D_{j1} by the interpolation mechanisms is assigned an individual coordinate system, and hence is able to produce branches which grow in different directions. Overall, the expansion produces nine instances of D; each of the three subtrees generates a branch of the geometry in Fig. 6.1c.

6.2 Component Types

The system components can be divided into three classes: geometry production, multiplication, and global modeling. All components have a basic set of parameters, which allows for the production of a geometrical primitive, to position it relative to the predecessor, and to arrange the geometry of successive components relative to its own geometry.

Camera Component

The camera component is the root component of every plant description, thus it must always be the first element in the structure tree. It accommodates all parameters that are needed for viewing the model.

These are, for instance, the position of the camera's coordinate system and the opening angles of the camera (together they determine the perspective projection on the screen), the position and type of light sources for the lighting simulation, and further representation parameters.

Base Component

In the base component only the fundamental parameter set is present, all other components are derived from it and offer additional functionality. The base component belongs to the class of components that are used for geometry production.

geometric primitive →

The component not only is able to produce geometric primitives such as cubes, spheres, cylinders or tori, but also sets of discrete points. These can be defined as open point sets (areas) or closed point sets (tubes), and usually lie on a plane. If a component is attached to another component, which likewise defines a point set, then the two point sets will be triangulated and so form a surface. In the horn component described below, this procedure is applied internally, in order to produce branches and stems from a sequence of triangulated point sets. Figure 6.2 illustrates the algorithm by producing stem-like objects.

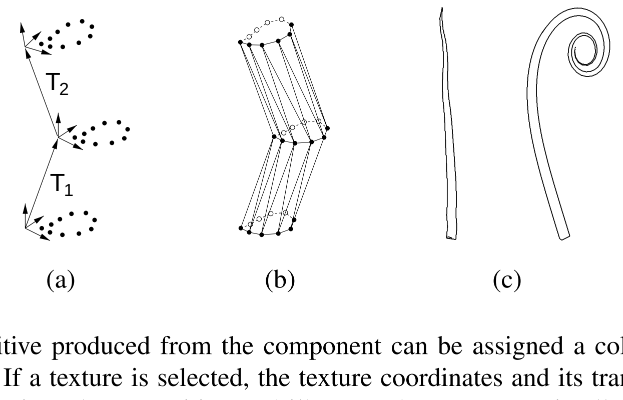

Figure 6.2
Geometric definition of a stem: a) point sets are specified relative to one another; (b) the surface is produced through triangulation; (c) differently formed stalks

The primitive produced from the component can be assigned a color and/or a texture. If a texture is selected, the texture coordinates and its transparency can be set, in order to position and illustrate the content optimally. Textures are generally used for leaves so as to increase their realism and to avoid the definition of complex leaf geometry. Bark is produced similarly.

Within geometry production, two different kinds of transformations are available: the primitive can be modified in position and size, and the modification can be applied to the local coordinate system, also affecting in this way the geometry of all the subsequent components.

An additional parameter is the strength of the phototropism. It determines if, and how strongly the primitive aligns itself with regard to an external light field. This is especially interesting for leaves, whose surfaces can thus be aligned plagiophototropically.

The recursion parameter mentioned earlier determines how often a recursion in the p-graph should be implemented. As an additional important modeling option, the user can select in a field whether the geometry of the component is generated or not if multiplied by a multiplication component. To do so, each multiplication component assigns an individual number to all produced child instances. This number is compared with the values stored in the field. The user in this way is able to prevent several components from generating geometry in a multiplication. This mechanism is used for handling exceptions that occur in each natural plant. Examples are a branch that died off or damaged leaves.

← exceptions

Thus, the definition of exceptions produces a type of context sensitivity in a so-far context-free system, since the appearance of an object is coupled to an environment here, which is defined by the switch and an object number. In Sect. 5.6 this problem was already discussed, and in Sect. 6.5 the process is more clearly explained in the form of examples.

Surface of Revolution Component

This component generates an additional geometrical primitive: a surface of revolution. The user can edit the silhouette as a polygonal curve as well as determine the resolution in the direction of rotation. Since this type of editing needs a special dialogue, the component was separated from the base component.

Leaf Component

Leaf components are needed for all types of leaves and petals. The leaf surface is produced by a sequence of area primitives, which are afterwards triangulated. Different parameters determine the appearance of the leaf. The geometrical complexity is defined by the number of points per area primitive and the number of area-primitives per leaf. A parameter determines to what extent the leaf is to be shaped in the form of a heart (see Fig. 6.3), another parameter specifies its width.

The leaf surface can be bent along or perpendicular to the main axis. The profile of the leaf surface can also be edited in order to produce deformations perpendicular to the main axis (Fig. 6.3e). The outline is defined using a polygonal curve, and the user can further specify jagged or irregularly formed leaves. The phototropism adjustable in the base parameter set can at this point be applied to align the leaf surface independent of the position of the branch relative to the incident light.

In practice it is often better to use simple geometry with only a few triangles[1] and to project a texture obtained from a photograph of a real leaf. The reason lies in the structure of today's graphics hardware in which the frame rate is determined by the total number of triangles produced, regardless of weather textures are applied or not. As a consequence, it is better to use fewer triangles with complex textures instead of many triangles without textures.

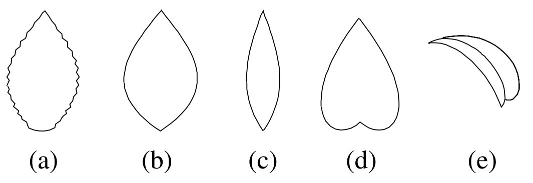

Figure 6.3
Definition of various leaf geometries: (a) jagged edges; (b) standard form; (c) narrow leaf; (d) heart shaped; (e) profiled and bent

Horn Component

The geometry produced with the horn component is used as the basis for all types of stems, branches or trunks, and it can additionally be used for the rendering of other organic objects (see [150, 219]). Occasionally, it is termed a generalized cylinder and is developed from the triangulation of tube primitives, as shown in Fig. 6.2. Aside from tubes also other primitives can be used and arranged along a curve using the same mechanism.

The horn component additionally permits us to multiply the geometry of successive components. In this case, the local coordinate systems of the child components are attached to the spots along the curve, which together with the tube primitives define the cylinder. Hence, the horn component produces geometry, and at the same time functions as a multiplying component. The same holds true for the tree component in the following section.

Tree Component

Just like the horn component, the tree component also produces a generalized cylinder.[2] The difference to the horn component is simply the way successive components are multiplied and over which parameters the form of the cylinder can be affected.

Usually a tree is constructed as a cascade of tree components or is defined over a recursion. In the latter case, however, the parameters cannot be set individually for each branching level, which often is a disadvantage.

[1]In the extreme case just one triangle can be used; however, that means that the leaf cannot be bent. This is a disadvantage, since it often affects the visual appearance of a plant.

[2]In this case, only tubes are permitted as primitives.

The parameters of the tree component are divided into three groups. One group determines the positioning of the branches. Here the branching angle (see Fig. 6.4b) as well as the branching characteristic can be adjusted. The branching characteristic defines how many branches per unit length along the trunk are produced (Fig. 6.4c). The adjustment is handled by the user via a density curve.

Also the deviation angle between neighboring branches can be modified. By default, here the Fibonacci Angle, i.e., the Golden Section cut, is applied (see Sect. 3.4).

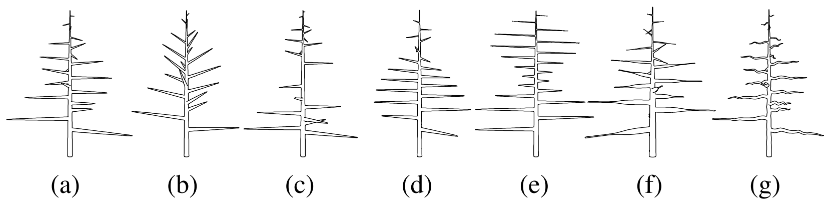

Figure 6.4
Parameter variance in a tree component: (a) standard form; (b) vertical angle; (c) branching density; (d) horizontal angle between two branches; (e) size; (f) thickness along the shoot axis; (g) gnarled look

Further parameters determine the size of the branches (Fig. 6.4e), their thickness along the branch axis (f), the gnarliness (g) as well as the directional change of the trunk at a branching. Additionally, the form of the trunk can also be defined directly over a spline function.

← tropism

Similar to the leaves, phototropism and gravitropism can also act on the generalized cylinders of the trunk and the branches. For this reason, the produced tube primitives are turned locally towards the light field, which altogether produces a bending of the geometry. In addition, the appearance of many trees is significantly determined by these parameters, though the effect can also be used to model deformations caused by wind influences.

In tree modeling another crucial difference between classic rule-based systems and rule-based object production comes to light. While with rule-based systems the appearance of a plant is controlled via the modification of local parameters, here global aspects are modeled. An example is the size of the forking branches, which is indicated by a function over the length of the trunk. Modeling with L-systems is at this point rather difficult, since the smallest local changes over several recursion stages can cause great differences in the appearance. As already mentioned, this problem is lessened through the use of editable functions, as demonstrated in Sect. 5.10.

Hydra and Wreath Components

The hydra component multiplies all components attached to the p-graph and places them in a star-shaped arrangement. With the hydra component, the user can define the number and size of successors, the opening angles of the star, and the turning of the successors with respect to the direction of the centerline. In Fig. 6.5a the centerlines of the father component and the multiplied components are shown.

The wreath component arranges its successors on a ring. Input parameters are its diameter as well as the number of successors.

Phiball Component

If components are multiplied by a Phiball component, then the geometries are generated according to the rule of the Golden Section on a spherical surface. The user enters the number and size of the components which can be multiplied as well as the opening angle of the sphere. This opening angle permits us to arrange the geometry also on a spherical section which makes the component usable in many applications. If only the cap of the sphere is needed, and if a large radius is entered, circular arrangements can be produced.

This technique is used, for example, with the sunflower seeds, in order to render these in their typical arrangement. A lateral cut through the sphere, as shown in Fig. 6.5c, can be applied for producing petals.

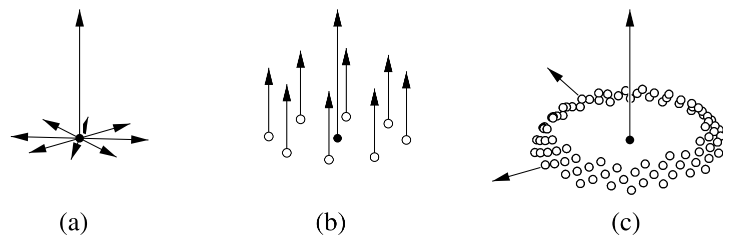

Figure 6.5
Positions and main axes of multiplied components: (a) multiplication over Hydra component; (b) Wreath component; (c) Phiball component

FFD and Hyperpatch Components

Both components belong to the group of components that influence the global shape of a plant. With both, a freeform deformation (FFD) can be defined that works either on the geometric data or only on the underlying shoot axes.

Two types of deformation definitions are utilized. While the FFD component functions $D_x(x)$, $D_y(y)$ and $D_z(z)$ must be entered by the user, the hyperpatch component works with a three-dimensional Bézier function of degree one to three. This in turn defines a cube with a sequence of control points that can be moved by the user. An adequate graphical interface allows for the direct manipulative interaction.

Furthermore, the components can also be utilized as a switch to interrupt the influences of a preceding deformation. In this case, no deformation is specified, but rather a special parameter is defined to switch off the process. This way, an FFD can be defined in the p-graph at one place, while the effect on the child components can be voided at another place. In Sect. 6.5, this is explained in more depth and demonstrated with an example showing the difference between the deformation of the geometry and the shoot axes.

World Component

The positioning of all geometries, and especially the positioning of leaf and tree components, can be influenced by photo- and gravitropism. These work relative to the light and to the gravitational field. The light field by default comes from the vertical direction, the gravitational field by default points in the opposite direction.

Using the component, this field can be arbitrarily redefined, in that for the x-, y- and z-components of the directional vector a spatial function is specified. The range of possible applications for these fields is numerous, as will be shown in the following examples.

6.3 Combination of Components

For the production of plants, the components are linked by the user to form the p-graph. The user then defines the respective parameters of the components. To construct a p-graph, the components are selected from a graphical toolbox and linked to the already produced components. Three types of links are available:

- **Child link:** This is a standard link. The component's geometry is placed relative to the preceding component. The p-graph displays these links as thin lines (Fig. 6.6a).
- **Branch link:** The component is multiplied as a branch of a tree component or a horn component. In all other cases it is interpreted as a child link. These links are displayed as bold lines (Fig. 6.6b).
- **Leaf link:** If the father component is part of a recursion, the geometry of the child component is created only once after the recursion terminates via the recursion parameter. Thus, in the structural sense, the component is a leaf of the i-tree. These links appear as dashed lines.

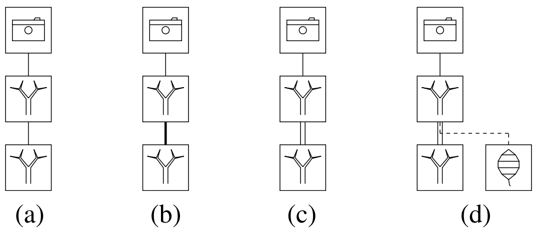

Figure 6.6
Types of links of components: (a) child link; (b) branch link; (c) recursive combination; (d) recursive combination with leaf link

In the figures following in this chapter, the directions of the lines are omitted in the p-graph, since these always point away from the root component. To display recursions, double lines are used. Additionally, in this case the first component is displayed twice in order to denote the starting and end points of the recursion in the graph.

These modifications allow for the display of the p-graph as a tree, which avoids many problems, and is visually clearer. The disadvantage of the illustration is that nested recursions cannot be displayed. However, since, for other reasons, they also cannot be handled by the system, they are not permitted in the p-graph.[3]

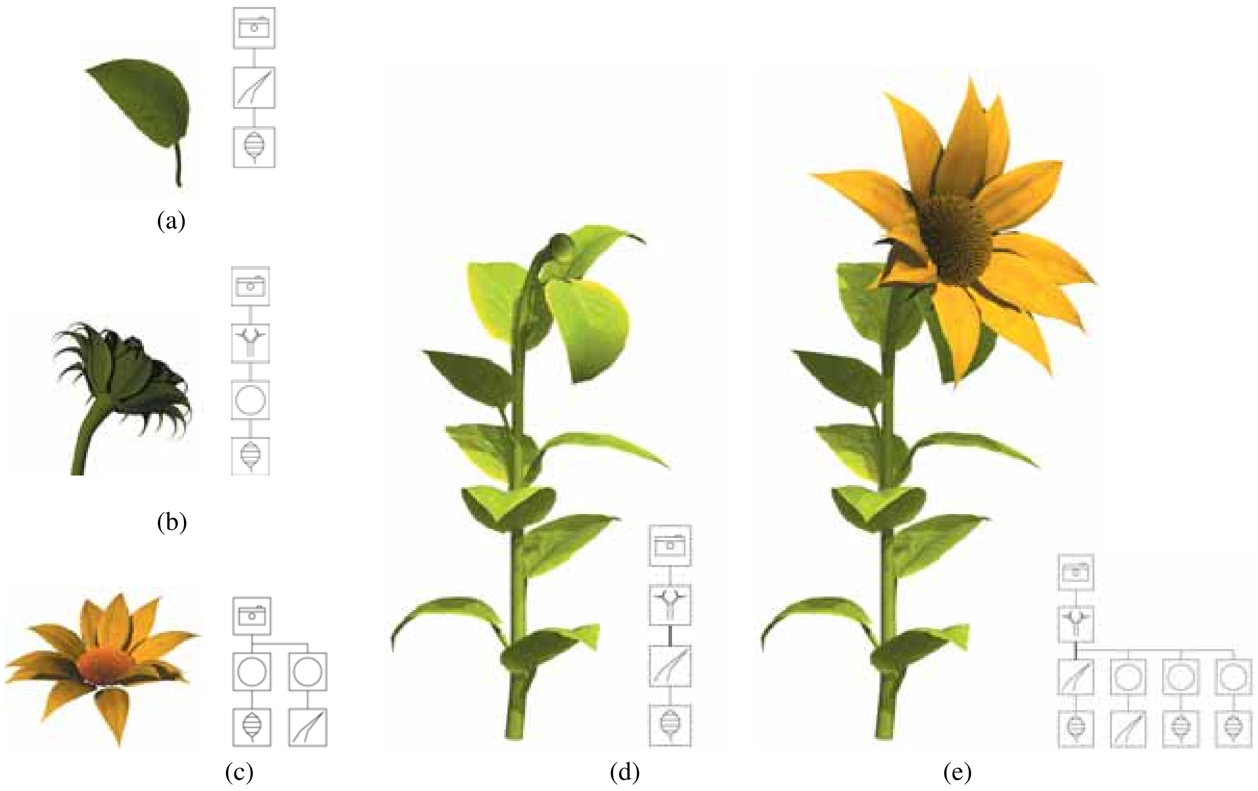

Figure 6.7
Parts of a sunflower with corresponding p-graphs: (a) leaf with stalk; (b) flower head; (c) flower with seeds; (d) stalk; (e) completed plant

6.4 Examples

After we have described the components and the options of their combination to the p-graph, three examples are used to show how plants are modeled. A sunflower, a rhododendron, and a chestnut tree exemplify the potential of the numerous combination possibilities of components as well as the power of the overall approach.

To model a sunflower, photographs or scanned textures of natural leaves and petal surfaces are needed. With a photo editing program, these are projected onto a transparent background. Here, the so-called alpha-channel, found in standard photo editors, is used, which defines the transparency as an additional

[3]For the in appendix described PC-based variant of the modeling system recursions are not permitted.

color channel. Usually, for the textures of leaves, only the values zero (completely transparent) or one (completely opaque) are required.

The picture of the leaf is applied as a texture to the leaf component's geometry and a small stalk is attached. Figure 6.7a shows the corresponding p-graph that consists of the icons of the camera, horn and leaf component. Using deformation values and transformations, the components are parameterized until a natural appearance of the leaves is reached.

In the next step, the leaves are used as branches of a tree component (Fig. 6.7d). The upper part of the stalk is opened, in order to connect it to the head of the sunflower.

The modeling of the blossom from the seeds and the blossom's petals are shown in Fig. 6.7b and c. Blossom petals are multiplied with a Phiball component, whereby the parameterization is selected so that it arranges the petals on a small ring. In a similar way, tiny sticks are multiplied by another Phiball component to produce the seeds of the flower. Both subtrees are attached using child links to the stalk of the sunflower. After further adjusting the parameters, the result is the complete flower.

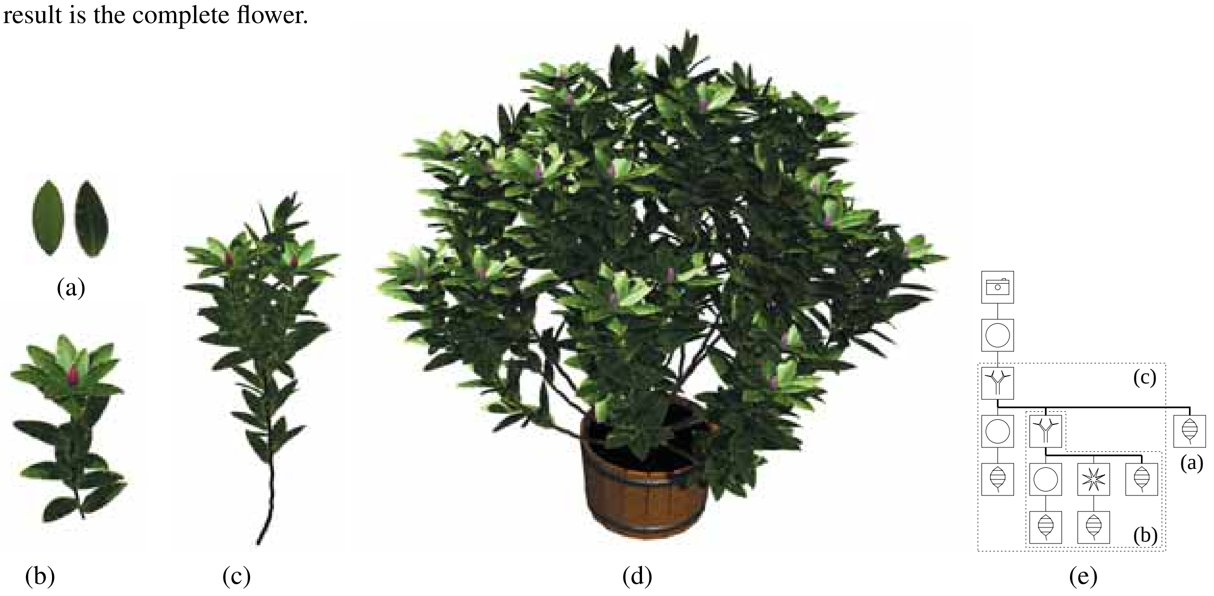

Figure 6.8
Modeling of a rhododendron: (a) leaf with texture; (b) small twig with blossom; (c) twig with branching leaves and two small twigs; (d) whole shrub; e) p-graph with marked subgraphs

A rhododendron is a good example for modeling a medium-sized shrub. In comparison to the sunflower, the geometric complexity is larger, with approximately 30,000 triangles. Here the focus is on the branching structure rather than on geometric properties of the blossoms and leaves.

Again, we will first scan a natural leaf and use it as texture in the leaf component (see Fig. 6.8a); its p-graph is shown as the upper part of the complete p-graph in Fig. 6.8e. Now a small twig is created. The tip of the twig is built separately, since the leaves of the Rhododendron are set in a special way around the blossom. A phiball component multiplies the leaves and a hydra component places the blossom leaves inside in the appropriate order (Fig. 6.8b). The

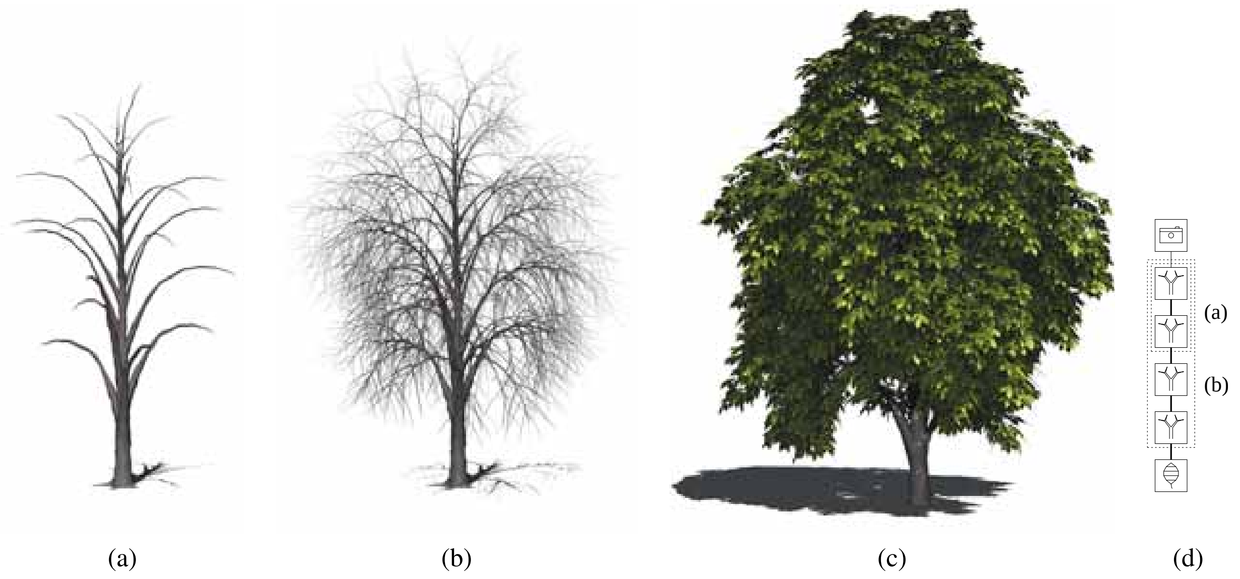

Figure 6.9
A tree is produced using a sequence of tree elements: (a) two components after parameterizing; (b) four components; (c) the leaves are attached to the last component; (d) p-graph with marked subgraphs

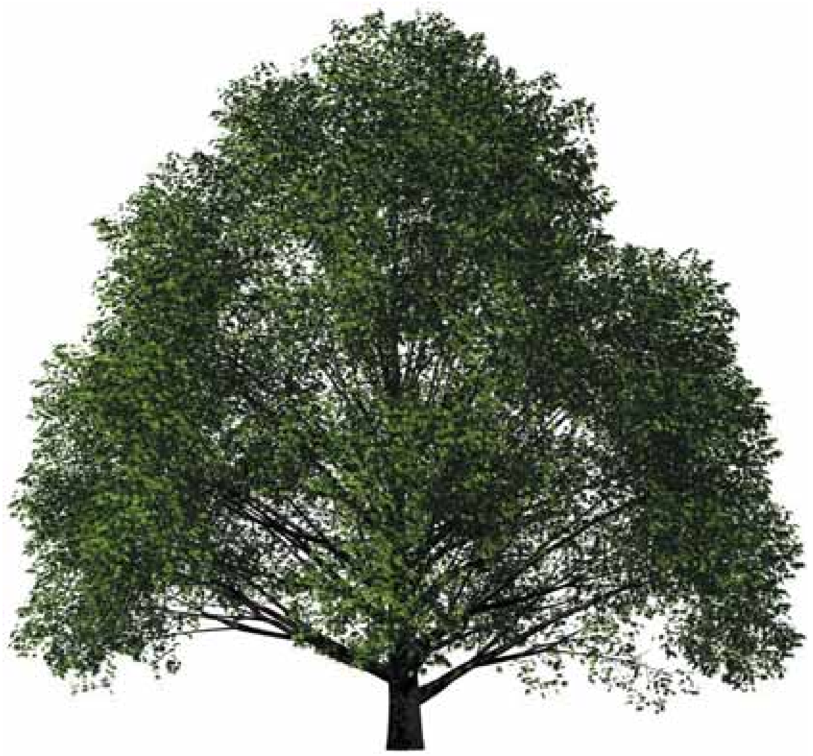

Figure 6.10
Maple tree with a total of six branching levels

p-graph now consists of two more components: the phiball to multiply the main leaves and the hydra component to multiply the blossom leaves. Both are joined by the child link with the tree component that will produce the twig.

The production of an entire branch is slightly more complicated: the stems should branch into leaves as well as small twigs. However, to achieve a more realistic shape of a rhododendron shrub, the leaves should not be generated at the position of a twig or vice versa. This is an exception that will be dealt with using the field previously described in the base component. Here some of the branches for leaves are switched off, and instead additional components for the production of small twigs are attached. In the leaf component, only for those branches for which the leaf multiplication was switched off the corresponding switch is turned on. This way, the twig can branch into two different ways (see Fig. 6.8c).

Finally, in the last step, 20 twigs are multiplied using the phiball component. The component is placed somewhat below the ground and thereby phototropism is added to the twigs, which slightly bends them upwards (see Fig. 6.8d).

As a third example, a tree will be created. As already mentioned, the general structure of the p-graph is similar in most trees. A cascade of tree components represents the branching structure, while the leaves, blossoms or needles are attached to the lowest level of tree components.

To generate a chestnut tree, as an example of a middle-sized tree, again textures are produced via scanning of a real leaf and bark and then are incorporated into the geometry of a leaf and tree component. If two components are joined and the parameters are set appropriately, a tree as seen in Fig. 6.9a is the final result. For the chestnut tree, a large branching angle in combination with a high gravitropism was configured to produce the main branches. Two additional branching levels are attached and parameterized appropriately.

Although it sounds simple, the definition of the parameters especially for the small twigs is still not trivial. The parameters are not independent of each other and cause to some extent a complex interplay. But since the model description can be reused in a very simple way, similar trees can be generated very quickly from an existing model.

6.5 Shape Modeling

Often not only just a species has to be modeled, but rather an individual plant with a particular shape. Examples here are trees growing along a wall or plants deformed under the influence of wind. In both instances, the overall shape of the plant has to be influenced in such a way as to yield a characteristic shape.

A total of five mechanisms have been implemented for the modeling of shape. One of the mechanisms is the definition of exceptions within the base component introduced in Sect. 6.2. The four others are functional modeling, tropisms, freeform deformations, and pruning. These mechanisms will be discussed next.

As already mentioned in the introduction, while building the temporary i-tree, component prototypes are transformed into instances, whereby the parameters are computed individually for each instance, particularly those of the multiplication components which were stored as parameter ranges. The individual assignment is performed by interpolation, which also permits us to apply a function to the result.

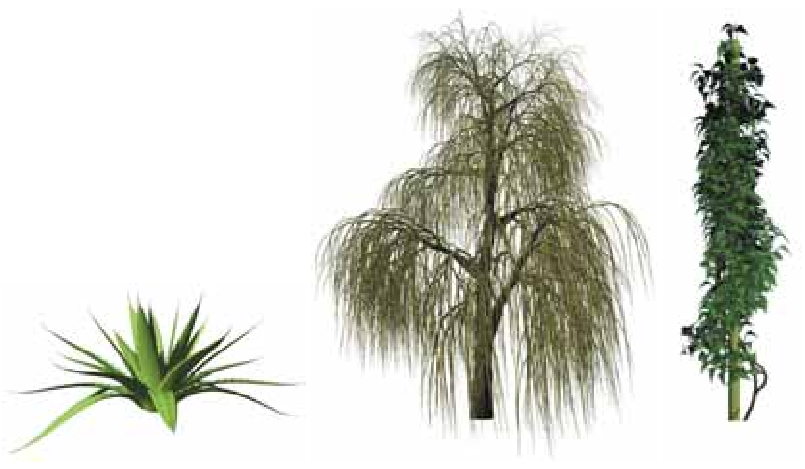

Figure 6.11
Modeling of shapes: (a) functional modeling: the bending of the leaves is computed by the iteration number and a random factor; (b) a weeping willow is fundamentally formed by its strong gravitropism; (c) a philodendron is led by a cylindrical field around a stick

At this point, random functions as well as other functionally specified parameters can be introduced into the system. For example, variations are generated by adding a small random value for the curvature or for the size; for the functional specifications all essential mathematical functions are available.

To parameterize this function meaningfully, a number of variables can be applied in combination with the active value interpolated for the respective instance. In this way, positions and orientation, and also recursion depth and the actual iteration number of the instance can be applied.

In Fig. 6.11a, an agave plant was modeled using these techniques. Each leaf is individually curved, where the iteration number and a random value determines the curvature. The first generated leaves with a small iteration number are created by the Phiball component at the top of the sphere and thereby receive a small curvature. For the subsequently generated leaves the curvature is increased continuously.

tropisms →

Tropisms were already discussed in connection with base, leaf, tree, and world components. The modeling with tropisms allows for generating a number of different effects. For the production of a weeping willow, a downward-pointing gravitational field as shown in Fig. 6.11b was applied. If the field is defined in such a way that the directional vectors point towards a center line, and within a cylinder around this line point outwards, then the branching structure can be forced to grow onto a cylindrical shape. This technique was applied to render the philodendron in Fig. 6.11c. More examples are wind simulations, in which a lateral field is used, and obstructions, which are realized with fields that block the entrance to a space.

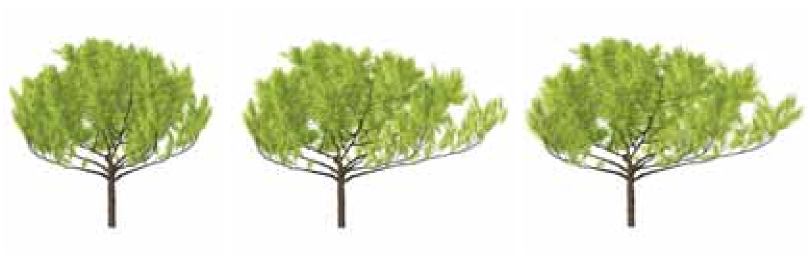

Figure 6.12
Freeform deformation applied to a pine tree: (a) initial model; (b) deformation of branches, the needles remain unchanged; (c) deformation of the entire tree

Figure 6.12 shows an example of the use of freeform deformations. While in Fig. 6.12c the entire tree is deformed, in Fig. 6.12b only the branches were distorted, but not the needles. In particular in the strongly deformed right part of the tree, the needles are more natural, since they were not subjected to a deformation.

← pruning

Many plant shapes are created through effects that do not occur in nature. Aside from hedges, this applies to most trees, i.e., if they were growing free-standing, they would grow as shrubs, and only as a result of pruning the lower branches, do they receive a tree-like shape.

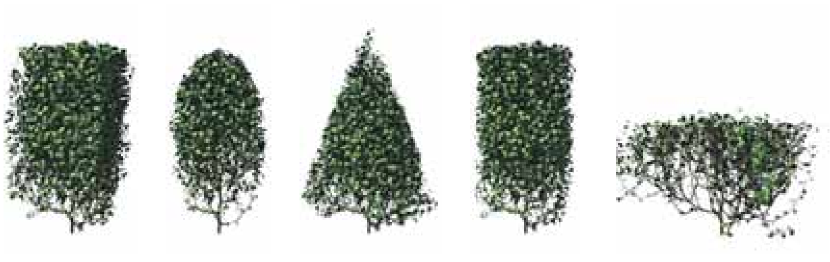

Figure 6.13
Pruning of a plant: (a) cube shape; (b) ellipsoidal shape; (c) cone shape; (d) cylindrical shape; (e) espalier shape

Pruning is realized by a parameter that controls the growth in the tree component. At the same time, base components are defined, whose geometrical primitives serve as a restriction volume. During the geometry production, plant geometry is then restricted by these volumes. In Fig. 6.13, a shrub was pruned in different ways. Such models can be used for creating simple hedges up to complex Renaissance gardens.

Depending upon the model used, the plant is to appear cut off or appears to be naturally grown within its restriction volume. Therefore two kinds of interaction with the restriction volume were implemented: either branches are cut off or a branch only grows up to the defined boundary. Then the lengths of the branches are defined, and in a second run their thickness, and in particular their taper ratios are adapted to these lengths. Using this technique, the plant appears natural within the restraining volume.

6.6 Animation

In the two foregoing chapters we occasionally addressed the animation of plants. A problem in this area is always the interplay between geometrical and topological changes in the growth of the plant. Prusinkiewicz works with differential L-systems, in which model changes are described by conventional L-systems; the intermediate growth is generated by differential equations with relatively complex boundary conditions. Some of the procedural approaches likewise permit growth of the produced trees, which is due to the limited model palette (mostly simple trees) easier to implement.

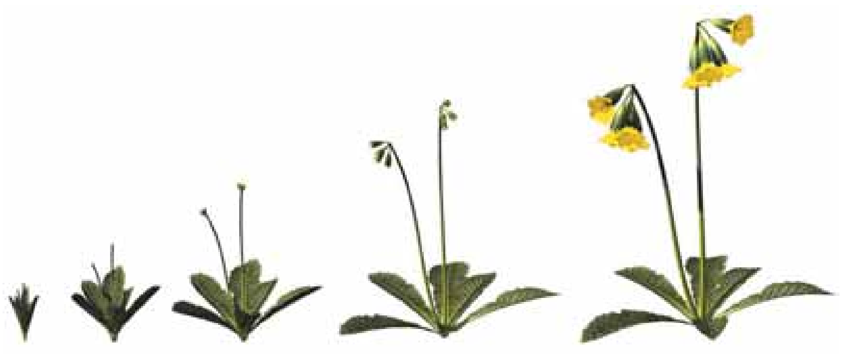
Figure 6.14
Growth of a primrose through changing parameter values and parameterized keyframing

With the rule-based object production, we take advantage of the fact that most of the model topology – for instance, how many objects are generated with a multiplication component – is not stored in the topology of the p-graph but as parameters of the components. The topology of the p-graph can therefore be accepted being constant during the process of an animation without too much restriction.

In this case, the parameters which are specified within the components, are understood as parameter sets for a specific time. If different parameter sets are defined for different times, then the values of the parameters for intermediate times can be determined through interpolation. Thus, we are concerned here with a keyframing concept based on parameter values.

Using cascaded keyframing, a local time is implemented. Here one keyframing sequence can be incorporated into another one. If a sequence is produced by a multiplication component during the course of an animation, the local time starts with zero and runs parallel to the global time. In this way, processes such as the opening of petals in a blossom have to be specified only once, and can then be run again at different times and locations independent from each other.

For the cascading of a keyframing sequence, an individual component is used that stores a complete sequence of model descriptions and parameter values at different times. This component is joined just like a normal component to the p-graph of the plant.

A great variety of plants has so far been modeled with this system. The generated trees again consist of a cascade of tree components and have between four and seven branching levels. The geometric complexity ranges between several thousand and, as in the case of the maple, several million polygons per plant (see Fig. 6.10).

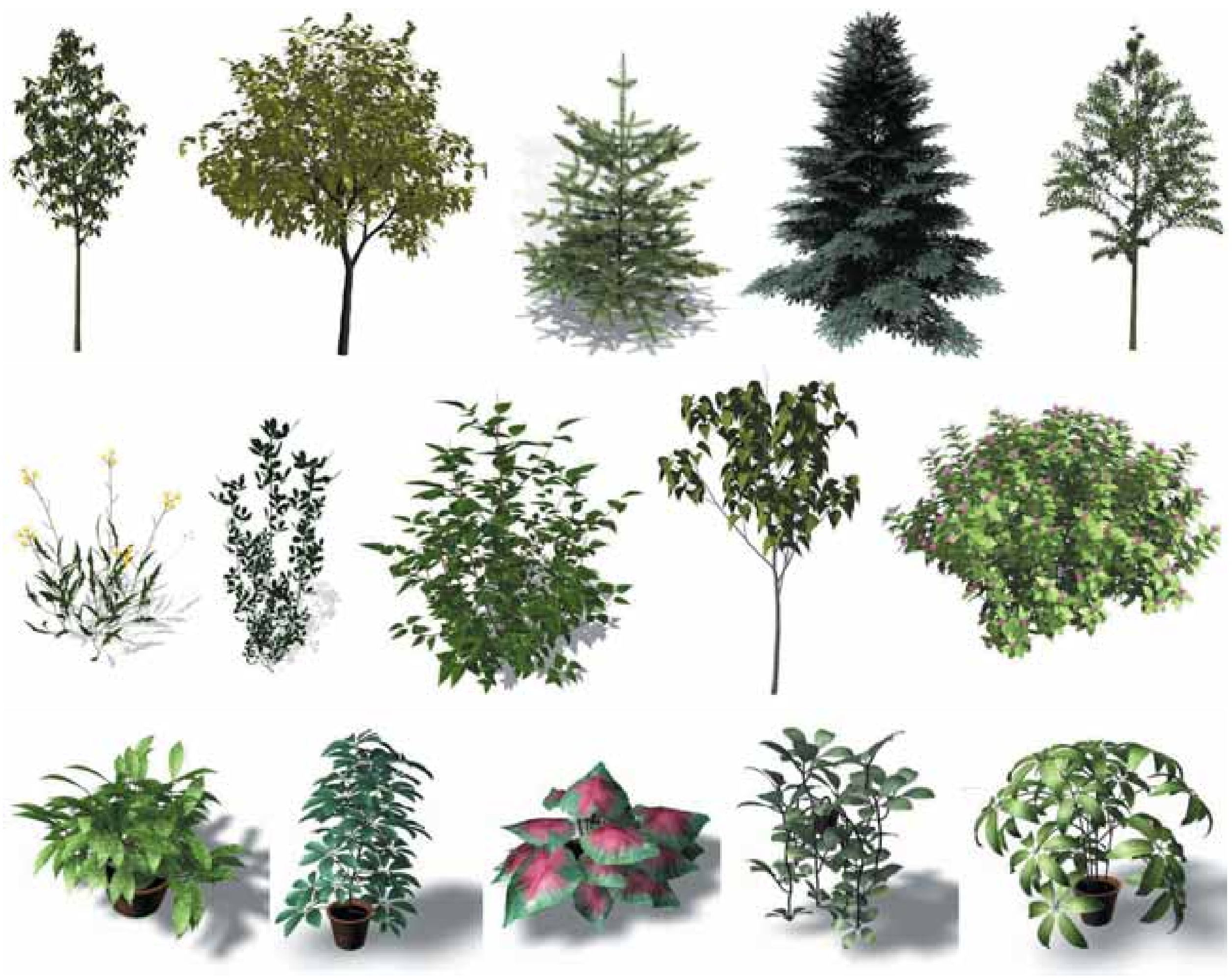

Figure 6.15
Several models created with the system.

The amounts of time needed for the production of the trees were between one hour and one day for complex trees. Generally modeling expenditure increases with the number of branching levels. Thus, if three or four levels are still producible within an acceptable time frame, a greater expenditure of time must be estimated for large trees not only because of the larger complexity of the data but because of the number of branching levels.

While modeling, the geometrical complexity of the representation can be limited, however, using the exception mechanism from Sect. 6.2 (base component). For example, all branches except for one can be switched off. The remaining branch is then modeled, and only at the end the others again are activated.

In Fig. 6.15 additional models are shown that were generated with these methods. The tree models show the spectrum of different branching structures, and the shrubs are additional examples of average complex models. In the lowest row of Fig. 6.15 various house plants are illustrated, which were likewise created with this system. These models were transferred after their creation into a professional animation system, which then created the renderings.

6.7 Resume

Contrary to the many methods presented so far, rule-based object production permits a fast and intuitive modeling of plants. In order to support this statement, a small user study with 18 persons was conducted.

After a 10-minute briefing about the system, 10 people had to model the head of a sunflower within 30 minutes, and eight people were to complete a tulip within a 45-minute time span. The modeling task of the first group required more intensive work with the multiplication components, while the second group was confronted with geometrical modeling problems. In addition, during the test period, the participants were allowed to consult with the project leader, and both groups were given photographs of the objects to be modeled.

Figure 6.16 shows the results of the questioning following the experiment. All values are indicated as 90%-confidence intervals for a normal distribution. Despite the relatively large confidence intervals, the results prove that the system was judged to produce an intuitive, understandable, and easily edited representation of a plant.

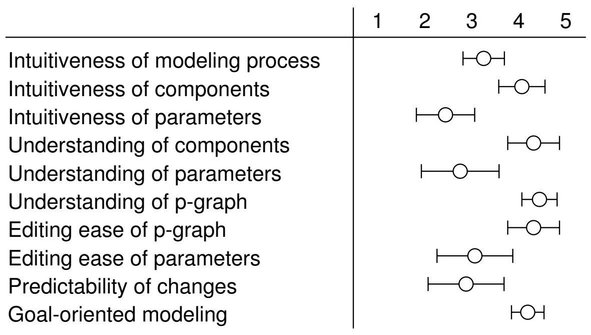

Figure 6.16
Evaluation of untrained users
(1=bad/insufficient, 5=very good/high)

The p-graph is likewise very intuitive in its structure. However, the number and the comprehensibility of the parameters is sometimes a problem. Here the prominent problem of all modeling systems comes to light: the modeling power is nearly always accompanied with too many parameters.

Due to the varying approaches of the different plant modeling methods discussed in the last three chapters, it is relatively difficult to draw a clear comparison between the individual procedures. The techniques are evaluated therefore on the basis of individual criteria that proved to be important in the practical work.

In the introduction we already addressed the various motives for the reproduction of plant morphology. Botanists are interested in the basic rule mechanisms, since the geometries of the models serve only for validating the production mechanisms or as basic data for the development of extended parameters such as the maximum utilization of light. For computer graphics the shape is the center of attention: "everything goes as long as it looks good". This phrase allows us even to consciously generate botanically incorrect shapes, as long as the desired visual effect is obtained. In this sense both the botanical accuracy, and with it the question to what extent botanical effects were replicated, and the topological and geometrical quality of the representation are to be evaluated.

← modeling aspect

Furthermore, in computer graphics the modeling aspect plays an essential role. It is crucial how quickly the user arrives at a desired shape. An important point of evaluation is thus the modeling effectiveness and in addition the modeling power that indicates what spectrum of plants can be produced. In Fig. 6.17 the evaluations are summarized.

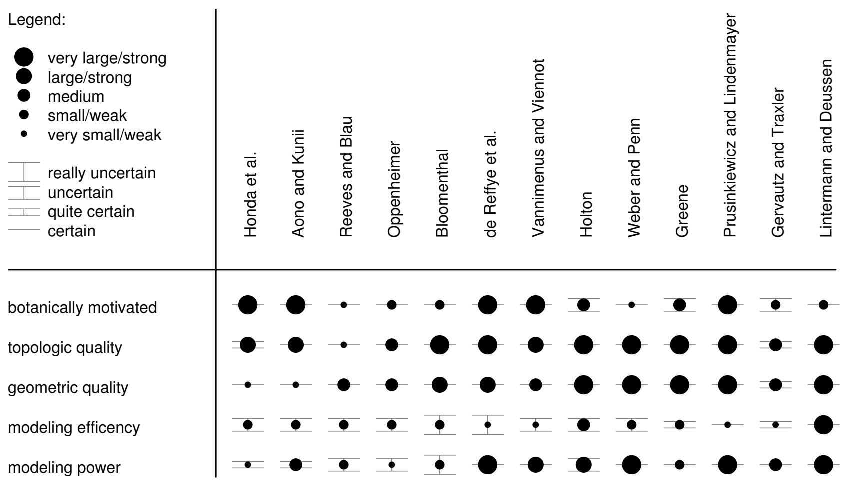

Figure 6.17
Evaluation of methods for plant modeling. The values and correlating intervals regarding the accuracy of each evaluation are indicated

Since the material for the individual procedures was presented in varying quality and quantity, additionally the accuracy of the evaluation was specified for each evaluation, i.e. the modeling effectivities of the procedures are indicated as relatively large uncertainties in nearly all cases, since only papers were used, and experimenting with the respective approaches was not possible. Unfortunately, the authors nearly never got involved in the indication of the modeling times.

The overall results, for all intents and purposes, reflects different strengths and weaknesses in the individual methods, which leads to the conclusion that these methods were designed for different purposes, and the focus shifted over time.

The early procedures of Honda et al. as well as those of Aono and Kunii adhere to a purely botanic motivation; they attach importance to the modeling of topology and neglect the geometrical aspects. In contrast, the subsequent work of Reeves and Blau, Oppenheimer and Bloomenthal place botany in the background, and the realistic appearance of the plants becomes the actual goal. De Reffye seems to have found a good compromise between a botanically motivated procedure and the necessary power of the approach.

Although the visual results of Vannimenus and Viennot are not really high quality, their combinatorial procedure is methodologically interesting and thus counterbalances the weaknesses in the representation. Holton pursues an interesting principle and moreover is also able to supply realistic pictures.

Weber and Penn consciously distance themselves from botanic regularities and model shapes of plants. The results are impressive, in particular since there also different degrees of detail can automatically be produced.

Certainly the most prominent method in connection with the modeling of plants is L-systems. With described extensions the group of Przemyslaw Prusinkiewicz at the University of Calgary has succeeded in extending L-systems step by step, so that these are able to replicate almost all important processes of plant growth. The only weakness remaining is the less intuitive structure of the systems, at least insofar as structure and interaction of subsystems is concerned. It demands profound experience in order to be able to use L-systems efficiently.

Rule-based object production seems to be a good compromise between the necessary modeling power and intuitivity of a practical approach. Though there is nothing that cannot be modeled using parametric L-Systems, the components of rule-based object production provide the typical user with an intuitive modeling strategy. Global and local manipulation methods enable the user to influence the shape of a plant efficiently. With a small number of components most plants on earth can be modeled – and much more. The examples on the following pages show some of the works that were done using the system.

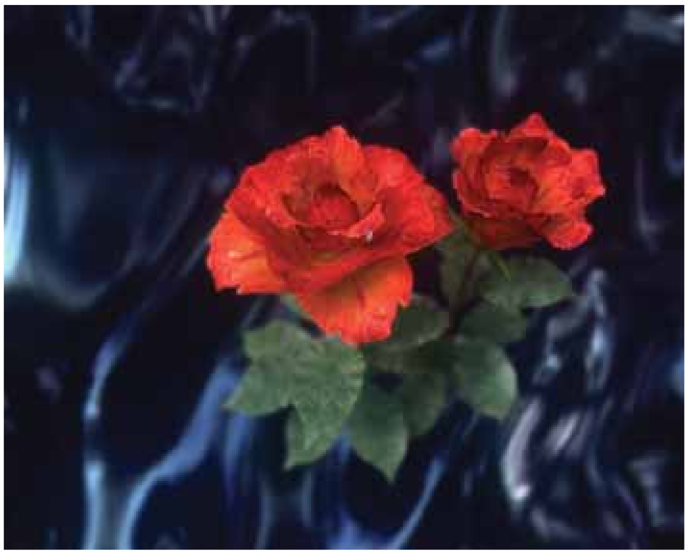

Figure 6.18
“Valentine” by Jan-Walter Schliep, using an Xfrog model.

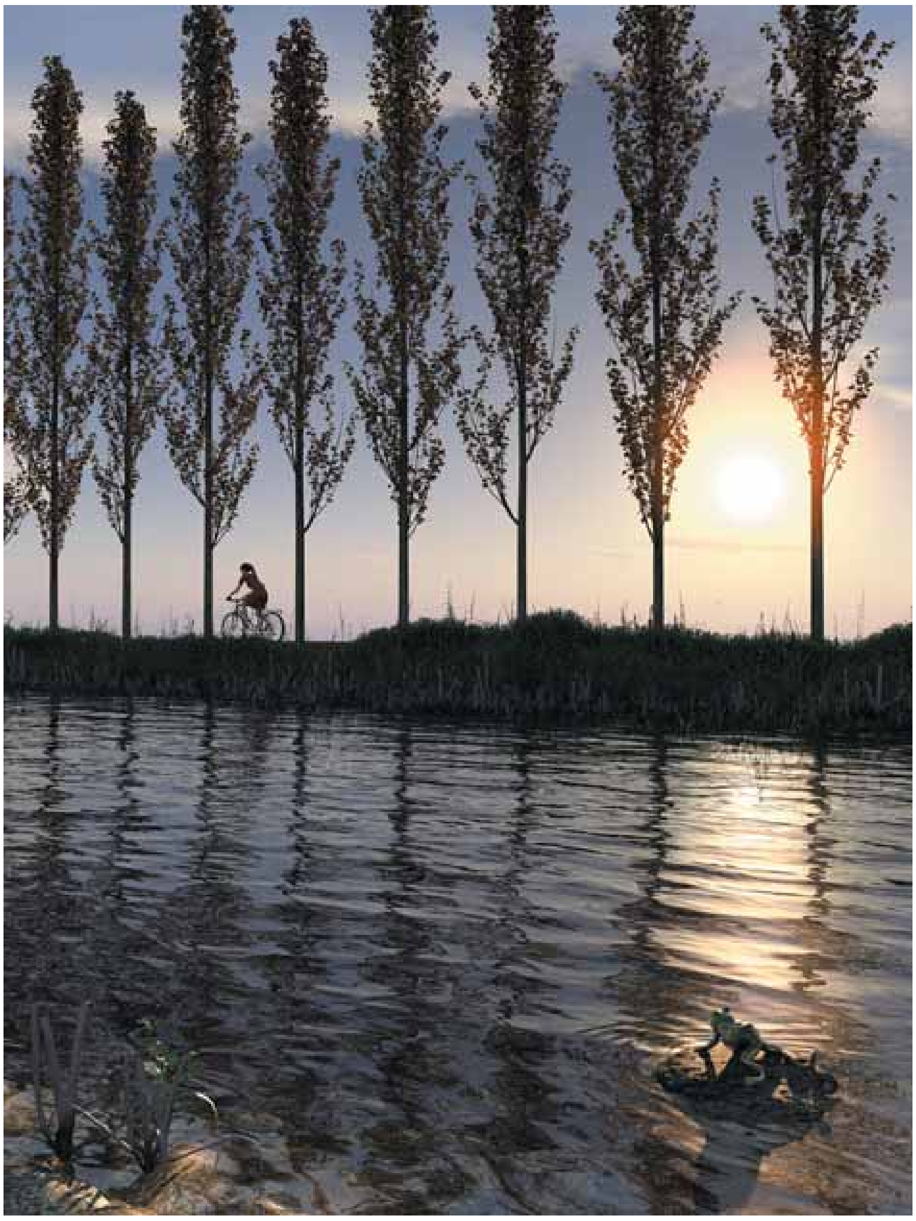

Figure 6.19
"Poplars" by Gilles Tran, generated with POVray using Xfrog plants

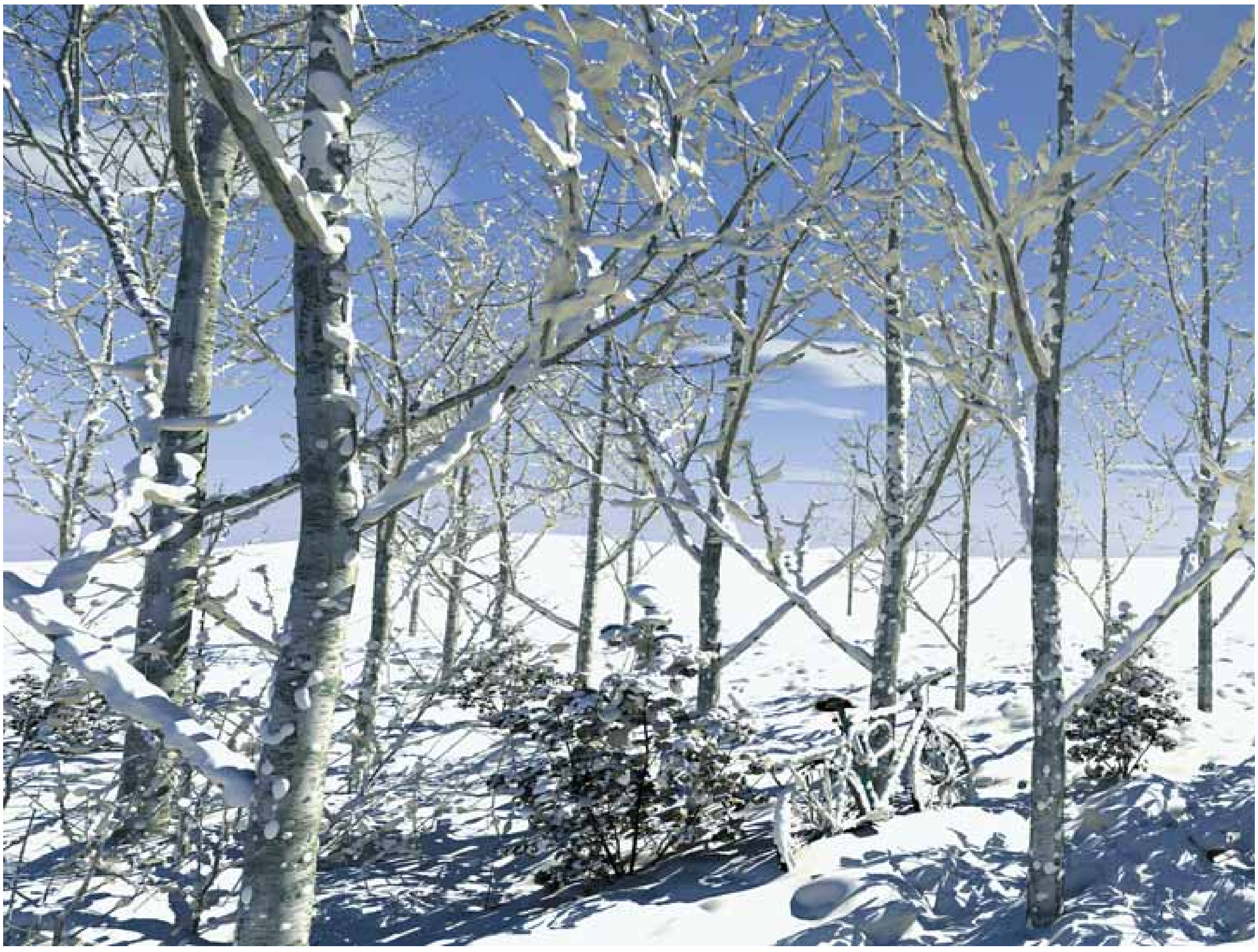

Figure 6.20
“Evergreen” by Gilles Tran

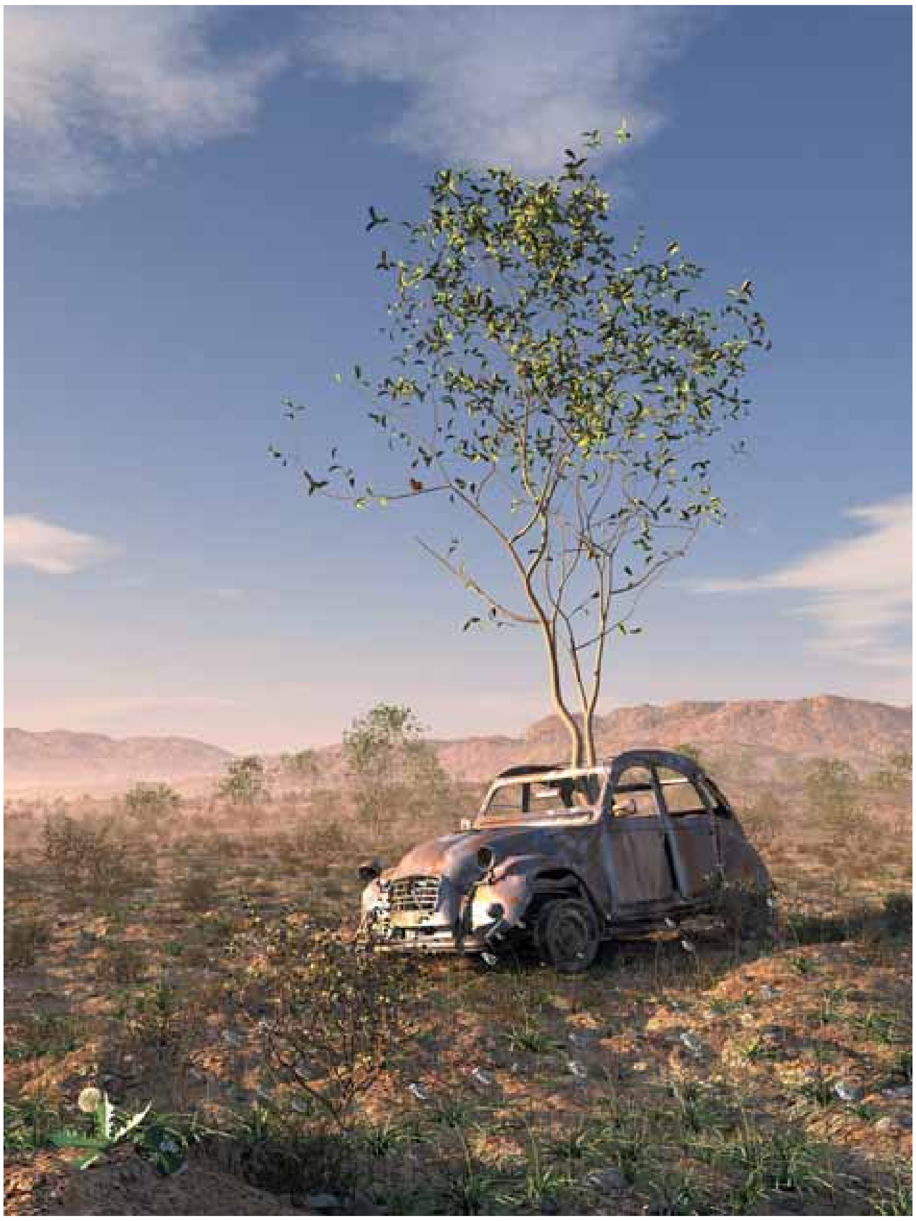

Figure 6.21
"2CV" by Gilles Tran, see also http://www.oyonale.com