Floral diagrams and inflorescence: interactive flower modeling using botanical structural constraints

花式図・花序: 生物学の知識を利用した 花のモデリングインタフェース

by

Takashi Ijiri
井尻 敬

A Master Thesis (Abstract)
修士論文

Submitted to
The Graduate School of
The University of Tokyo
In Partial Fulfillment of the Requirements
for the Degree of Master of Information
Science and Technology in Computer Science

Thesis Supervisor: Takeo Igarashi 五十嵐健夫
Professor of Computer Science

ABSTRACT

We present a system for modeling flowers in three dimensions quickly and easily while preserving correct botanical structures. We use floral diagrams and inflorescences, which were developed by botanists to concisely describe structural information of flowers. Floral diagrams represent the layout of floral components on a single flower, while inflorescences are arrangements of multiple flowers. Based on these notions, we created a simple user interface that is specially tailored to flower editing, while retaining a maximum variety of generable models. We also provide sketching interfaces to define the geometries of floral components. Separation of structural editing and editing of geometry makes the authoring process more flexible and efficient. We found that even novice users could easily design various flower models using our technique. Our system is an example of application-customized sketching, illustrating the potential power of a sketching interface that is carefully designed for a specific application.

論文要旨

植物として正しい構造を持った花の 3D モデルを, 迅速に簡単にデザインするためのシステムを提案する．提案システムでは，花の分岐構造を定義するのに，生物学で利用されている花式図・花序を利用する．これらは，花の構造を表現するのに最適な汎用性を持ち，単純化された図表現のため，これらを利用すると花の持つ複雑な分枝構造の制御を単純な GUI 操作で行える．また，ジオメトリをデザインするために，スケッチインタフェースも提案する．分枝構造とジオメトリのモデリングを明確に分けることにより，個々のモデリングプロセスが単純で柔軟なものになる．最後にユーザテストを行い，提案システムが初心者にも扱え，様々な花を生成できることを示した．本システムは、特定のアプリケーションのために注意深くデザインされたスケッチインタフェースの高い可能性を示すものである。

Acknowledgements

I would like to thank you to Takeo Igarashi, my supervisor. He gave me advice not only for my work but also about how to work during my master course. Members at Takeo Igarashi laboratory helped me very much too. They and I discussed our researches, and I've enjoyed it and learned a lot from it. Also, I would like to thank Prof. John F. Hughes, Prof. Etsuya Shibayama, Dr. Shin Takahashi, Prof. Katsuhiko Kakei, and Prof. Ikuo Takeuchi for their comments and advice based on their experience and deep insights.

This work was funded in part by grants from the Japanese Information-Technology Promotion Agency (IPA).

Contents

Figure Contents

Chapter 1. Introduction

Flowers pose an interesting and important challenge for three-dimensional (3D) computer graphics modeling. They have a great number of components, such as petals, stems, and pistils, which take on highly varied 3D shapes and which are connected with intricate structures. To create a flower, users must design each component as a freeform surface and lay them all out in 3D space. The geometric and structural complexity makes this a difficult and time-consuming task even for experienced users; for novice users, creation of beautiful and biologically plausible flowers using traditional tools is almost impossible.

Various botanic modeling systems have been created to support the design of plants. These can be classified into two groups according to their purposes. The first group concentrates mainly on visual plausibility rather than botanical correctness [Deussen and Lintermann 1999]. This type of modeler tends to offer a simple user interface, but its underlying method is to use a predefined library, and it is therefore difficult to design models that are not in the library. The second group tries to build a theoretical framework based on biological knowledge. For example, the L-System, one of the best known plant modeling systems, defines plant structures using a set of rewriting rules [Prusinkiewicz and Lindenmayer 1990]. However, it is very difficult to encode and decipher the behavior of real-world plants in such a simple form, and users must also have specific biological knowledge about plants. Furthermore, while an L-system encodes various characteristics of the gross structure of a plant, the actual geometry of the individual components; leaves, petals, stems, etc. remains to be determined by the user.

Our goal is to strike a balance between these two approaches to modeling, that is, to provide an easy-to-use interface, while allowing users to model a wide variety of biologically plausible flower models. When guiding the modeling process, we incorporate *floral diagrams* and *inflorescences* as general and compact frameworks to describe most real-world flowers. A floral diagram is an iconic description of a flower's structural characteristics (figure 1-1 : *Lily model designed by author using our system* (a)); we use it to design individual flowers. An inflorescence is a branch with multiple flowers and its branching pattern is represented in a pictorial form; we use it to design models that consist of many flowers, such as bostryx, lavender, and lilies (figure 1-1

(b)). These two frameworks define the structure of a flower model; it is also necessary to specify the geometry of each component, such as the floral receptacle, pistil, stamen, petal, and sepal. To make geometric modeling intuitive and efficient, we use a customized freeform sketching interface.

This paper describes the user interface of our prototype flower modeling system based on these ideas. The structure editor consists of two sub-systems: one is for individual flowers, driven by floral diagrams, the other for arrangements of multiple flowers, based on inflorescences. The geometry editor also has two sub-systems: one for floral elements, the other for inflorescences. We believe that this separation of structure editing from geometry editing is applicable to general modelers, simplifies the modeling process, and achieves high configurability and reusability. Without this separation, it is very difficult to change a basic structure after details have been completed. Using our system, once a whole model has been created, it is possible to apply the model to different geometry to create a new model with the same or a similar structure.

Note that our contribution is in simplifying the process of flower modeling, not in improving the final results. The resulting flower models can be replicated by existing modeling systems, but the process is different. With customized and well-designed high-level editors for particular classes of objects, the modeling process becomes much more intuitive and efficient.

We describe the user interface of these editors in the following sections after discussing related work and the basic background. Our results show that users can design interesting flower models, such as the one shown in figure 1-1, with little training. A user spent only 30 minutes creating this model from scratch.

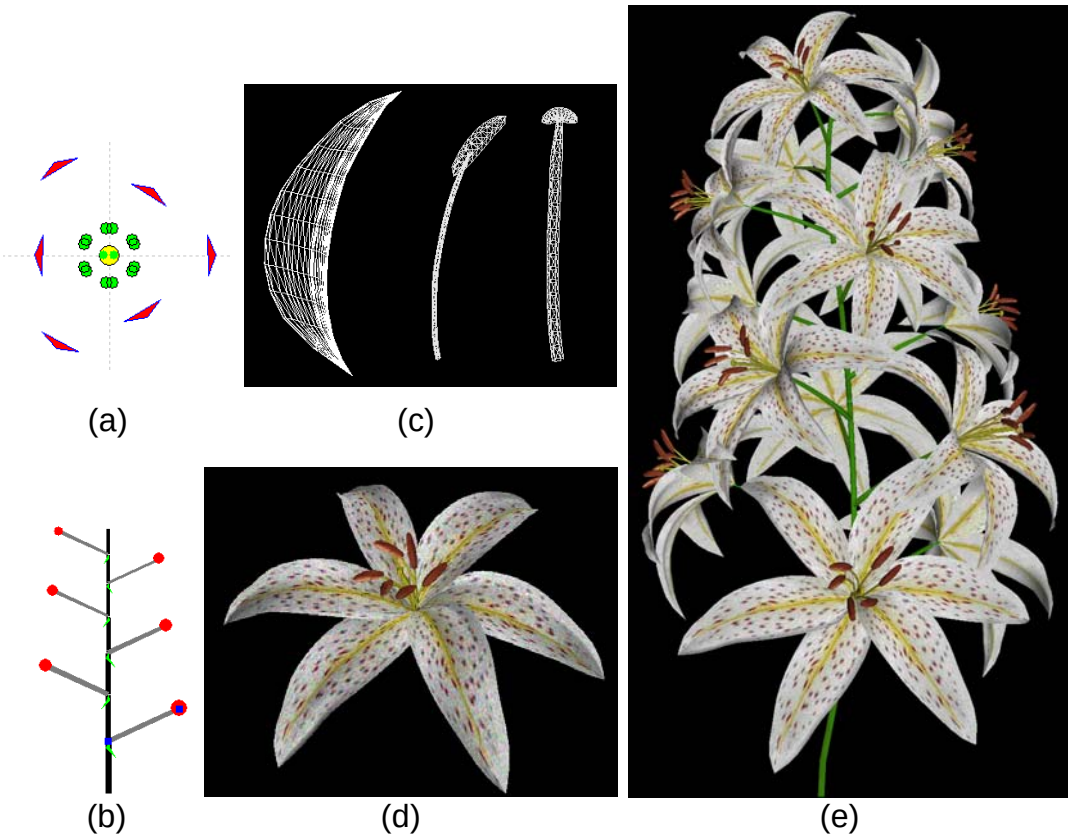


figure 1-1 : *Lily model designed by author using our system*

The structural information is given as a floral diagram (a) and an inflorescence (b). The floral diagram consists of one pistil, six stamens, and six petals. The inflorescence pattern is raceme. The geometry models are designed in the sketch-based editor (c). The user creates a flower (d) and the entire model (e) of a lily combining the structural information and the geometries.

Chapter 2. Related Work

2.1. Modeling of Plants

Lindenmayer [1968] formulated the L-System and Prusinkiewicz and Lindenmayer [1990] later introduced it to the computer graphics community. The L-System has been extended to simulate a wide variety of interactions between plants and their environments [Měch and Prusinkiewicz 1996; Prusinkiewicz et al. 1994; 1996]. Prusinkiewicz et al. [2001] also proposed using positional information to control parameters along a plant axis. Boudon et al. [2003] proposed an L-system-based process for designing Bonsai tree models; it uses decomposition graphs to make it easier to manipulate various parameters.

Some other articles introduced usages of growth simulation for creating variation of plant's structure. The benefits of simulation based modeling system are their capabilities of dealing with global parameters such as phototropism or gravitropism, while rule based systems only handle local parameters. Streit et al. presented a method to generate plant variations from an original model [Streit 2005]. The system makes variation of single stalk by running plants growth simulation that is based on a feed back control system by which a plant naturally responds to environmental factors. Runions et al. provided biologically-motivated method that simulates a growth of a leaf to generate venation patterns [Runions 2005]. They employ a *canalization hypothesis* for establishing the simulation.

Deussen and Lintermann [1997; 1999; Lintermann and Deussen 1996] developed the Xfrog system, which combines the power of a rule-based approach and intuitive user interfaces using a graph representation. Users design a graph representing the branching structures of a plant with 11 node types. This system offers an intuitive user interface and the resulting models are highly realistic, but the graph is designed heuristically and is too general for flower modeling (i.e., the graph can create structures other than plants). Furthermore, the graph representation includes geometric components such as FFD, so it is not possible to separate structural and geometric definitions completely.

2.2. Sketch-based 3D Modeling

Over the past decade, sketch-based modeling has become popular; instead of creating precise, large-scale objects, a sketching interface provides an easy way to create a rough model that quickly conveys a user's intentions. The main focus is on inferring 3D shapes from two-dimensional (2D) sketches. Previous work has reconstructed rectilinear models covered by planar faces by solving constraints [Pugh 1992; Eggli et al. 1997] or by using optimization-based algorithms [Lipson and Shpitalni 1996]. The SKETCH system [Zeleznik et al. 1996] allows users to design 3D scenes consisting of simple primitives, while the Teddy system allows users to design freeform models [Igarashi et al. 1999]. Generating 3D curves through sketching is also a rich research domain; Pentland and Kuo [1989] generated a 3D curve from its 2D projection using energy minimization, while Tanaka et al. [1989] used symmetric relations. Another strategy for defining a 3D curve is to draw strokes twice, for example, a screen projection of a curve and its shadow [Cohen et al. 1999; Tobita and Rekimoto 2003].

In these years, researchers published papers that employ sketch-based interface for plants modeling. Okabe et al. introduced the tree modeling system [Okabe 2005] that allows the user to design large variation of trees from drawing sketches. This system also introduced example based manipulation to create complicated branching structure easily. Ijiri et al. proposed sketch based flower modeling system [Ijiri 2004] that allows the user to model 3D flower components such as petals, leaves and stems by drawing a few strokes.

2.3. Floral diagrams and inflorescences

Floral diagrams and *inflorescences* are technical representations used in the study of plant morphology, which uses plant structure to explore their evolution, ecology, and systematics [Hara 1994; Shimizu 2001; Bell 1991].

A *floral diagram* pictorially represents the layout of four kinds of floral elements on a receptacle (the base of a flower): pistils, stamens, petals, and sepals (figure 2-2). A floral diagram also describes additional information, such as the stem cross-section, number of ovules,

and whether petals are connate. However, it does not describe the 3D geometry of floral components or their relative sizes. There is no universal definition of a floral diagram, and various forms of floral diagram exist.

An *inflorescence* represents a branch bearing multiple flowers. In an inflorescence, flowers are generally arranged in one of a fixed number of patterns specific to their species. There are three inflorescence groups: indeterminate, determinate, and compound. In indeterminate inflorescences, lower flowers bloom first and higher flowers follow. In determinate inflorescences, top or central flowers bloom first and lower or lateral flowers follow. Compound inflorescences are a mixture of the other two patterns. Simple 2D figures can be used to represent all branching patterns (figure 2-2). Here, black lines represent the central axis and its branches, red circles represent flowers, and green crescents represent bracts. Larger circles indicate older flowers.

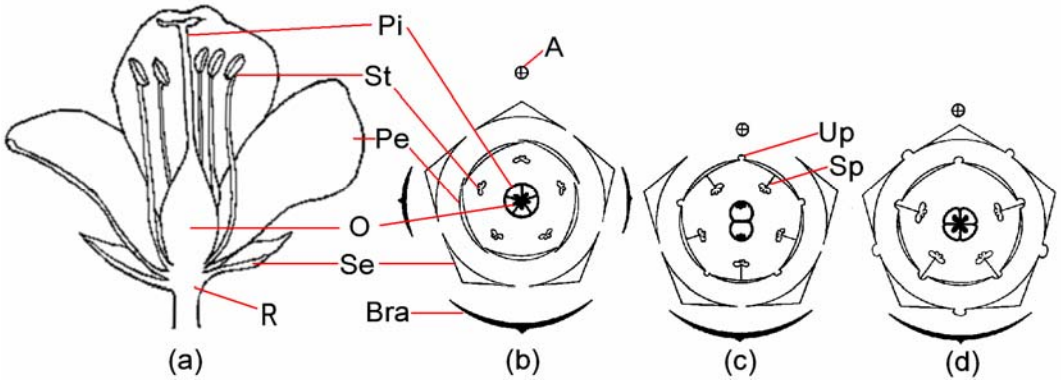


figure 2-1 : *Examples of floral diagrams*

A: axis, Bra: bract, O: ovary, Pe: petal, Se: sepal, St: stamen, Sp: sepal adnate to stamen, R: floral receptacle, Up: petal connate to petal.

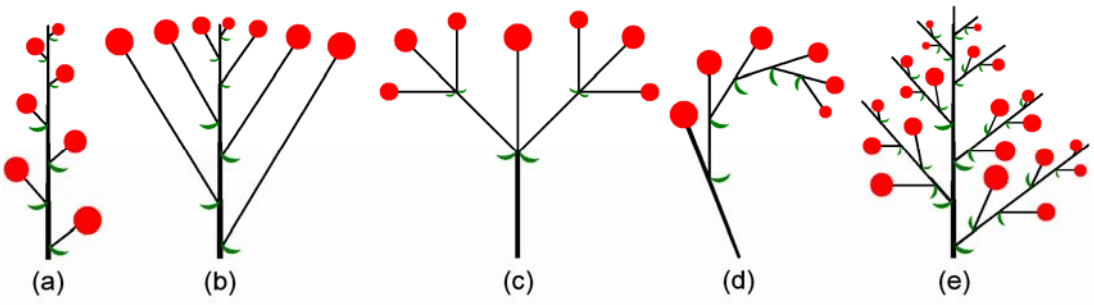


figure 2-2 : *Examples of inflorescence patterns*

The two on the left are indeterminate inflorescences: *raceme*(a) and *corymb*(b). The next two are determinate inflorescences: *dichasium*(c) and *drepanium*(d). The last is a compound inflorescence: *compound-raceme*(e).

Chapter 3. Overview of the modeling process

Our system consists of a set of independent editors, which can be basically categorized into two groups: structure editors and geometry editors. The structure editor consists of a floral diagram editor and an inflorescence editor. Users can alternate between these two editors. A typical scenario is as follows (figure 3-1).

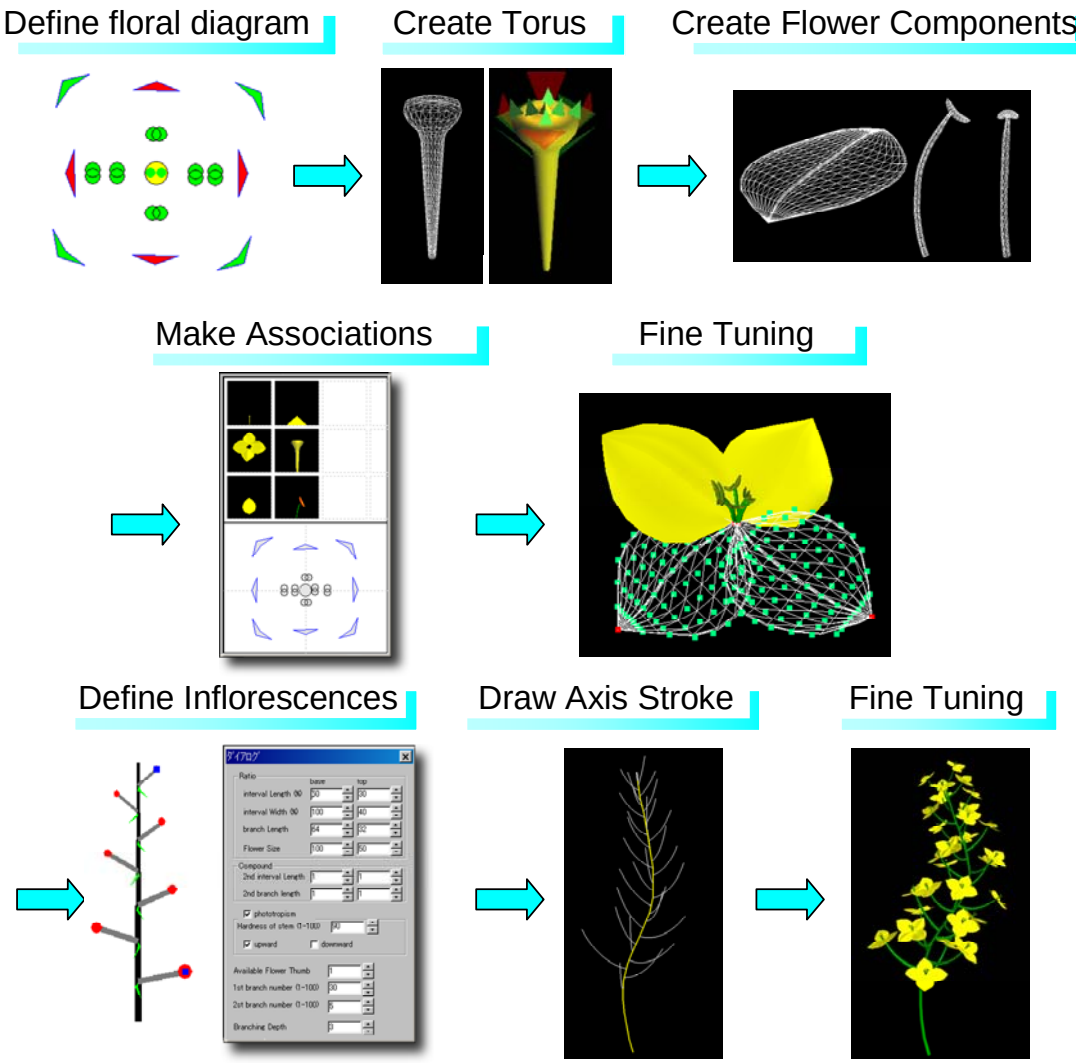


figure 3-1 : *Overview of the modeling process*

The user first defines the flower's structure in the floral diagram editor by editing the layout of the floral components. The user then models the shapes of the floral receptacle and floral components using the sketching interface in the geometry editor. The resulting receptacle model appears at the bottom of the floral diagram editor and the component thumbnails are listed on the right side of the window. Next, the user associates geometries of floral components with corresponding elements in the floral diagram using drag-and-drop operations. The system automatically places geometric objects on the receptacle model. The user can interactively adjust the angle of attachment, size, and shape of the components in the geometry editor. The user can also modify layout using the floral diagram editor.

After designing individual flowers, the user models the inflorescence. The user first defines the structure in the inflorescence editor, choosing one pre-defined inflorescence pattern from the list and making basic adjustments to various parameters. Then the user defines the central axis geometry by drawing a freeform stroke in the geometry editor. The system creates a three-dimensional inflorescence along the axis. The user adjusts the angles of flower and branch attachment using the geometry editor and can adjust parameters such as branching angle, branch length, etc. using the inflorescence editor.

Chapter 4. Structure editors

4.1. Floral diagrams editor

A standard floral diagram represents not only the structure of a flower but also some geometric information. However, our floral diagram editor focuses on the layout of floral components, and geometries are modeled separately in the geometry editor. Floral components (pistil, stamen, petal, and sepal) are represented as icons (Figure 4-1 (b)). Users first specify the number of parts by typing the number, then specify layout by dragging and moving icons in the diagram.

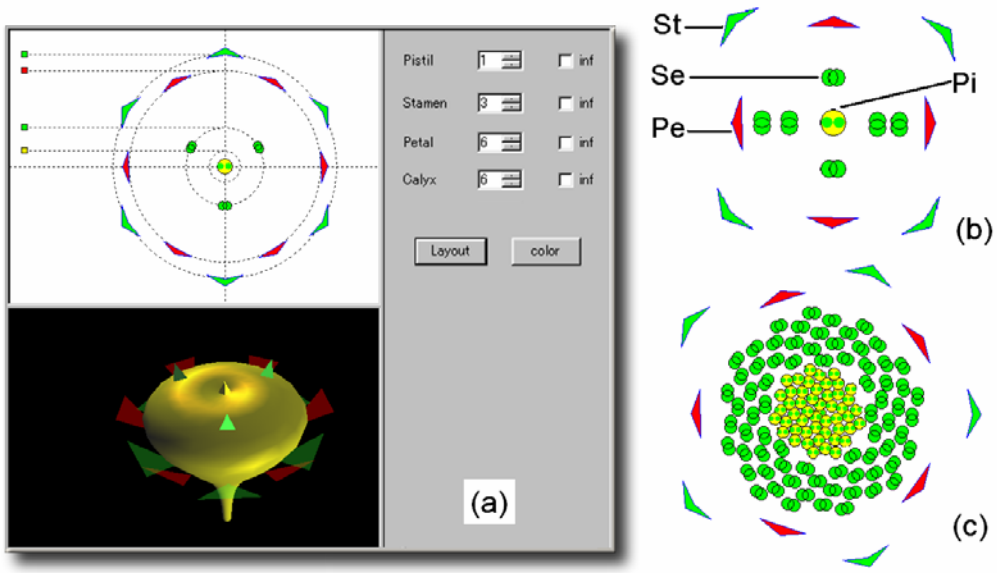


figure 4-1 : *floral diagram editor*

A snapshot of the floral diagram editor (a) and examples of floral diagrams: Brassica rapa (b) and Ranunculus acris (c). Pi: pistil, St: stamen, Pe: petal, Se: sepal.

Floral components are often arranged in radial symmetry, so our editor provides a function to arrange them in radial symmetry. There are four circular regions in the diagram

editor and users can modify their size by dragging borders. If users press the "layout" button, the system distributes the parts uniformly in each region. Some species (*e.g.*, Ranunculus acris) have an indefinite number of components. In this case, a specific region of the flower is filled by as many corresponding components as possible. In our system, if users check the "indefinite" box, the corresponding region is filled by as many icons as possible (Figure 4-1 (c)). We use a filling algorithm introduced by Prusinkiewicz et al. [2001].

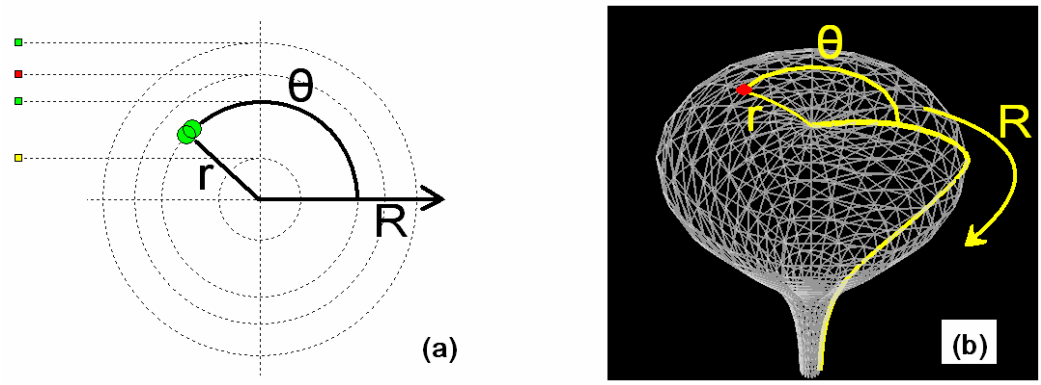


figure 4-2 : *Mapping the 2D diagram onto the 3D receptacle.*

One stamen object on a floral diagram in 2D (a) and its corresponding position in 3D (b).

A floral diagram is a 2D representation of a layout, so in order to construct the final 3D flower, the system has to convert the 2D layout into a 3D composition of geometric objects. A floral receptacle is represented as a surface of revolution, the outline of which is drawn by the user. The system uses a polar coordinate system on this surface, shown in figure 4-2 (b). In our implementation, the receptacle's 3D view is located underneath the floral diagram view (figure 4-1 (a)). A change using the floral diagram editor is immediately reflected in the 3D view. We currently do not allow users to use the 3D view to directly manipulate the layout; this remains for future work.

4.2. Inflorescences editor

In the inflorescence editor, users select a branching pattern from the list and modify parameters by dragging handles in the visual pattern display (figures 4-3 (b), (c)). We have implemented 8 of 22 patterns reported in the literature [Bell 1991]. The variety of adjustable parameters depends on the pattern selected. Figure 7-1 shows all patterns and their parameters. Using a raceme as an example, branch angle, branch length, and flower size at the top and bottom of the axis can be modified using the handles (Figure 4-3 (c)). Values between the top and bottom are linearly interpolated. Parameters such as the existence of tropism or stem hardness are specified in dialog boxes, since these parameters are difficult to represent in a 2D illustration. In future research, we plan to allow for more flexible positional control [Prusinkiewicz et al. 2001].

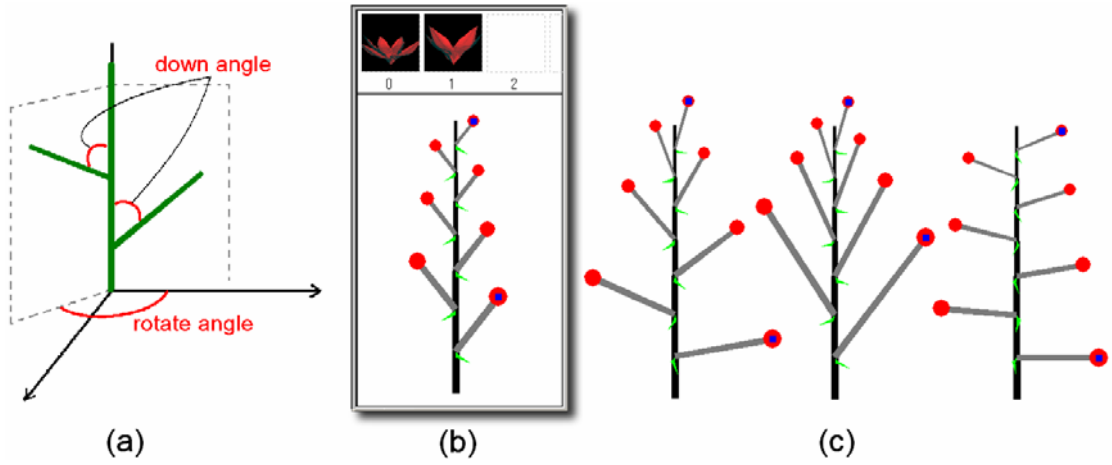


figure 4-3 : *Inflorescences editor*

(a) Down angle and Rotation angle. (b) Inflorescence editor. (c) Inflorescence pattern of a raceme with various parameters.

To determine each branch's 3D direction, the system must compute branch angle to the stem; we call this the rotation angle (Figure 4-3 (a)). In certain inflorescences, branches have

one rotation angle value, which can be described as follows:

$$angle = \frac{F_n}{F_{n+2}} \times 360 \qquad n = 0, 1, 2, 3 \cdots$$

$$F_n : fibonacci\ sequence$$

This formula produces the following values: 180, 120, 144, 135, 138.45, 137.14, and 137.65, covering almost all species [Bell 1991]. These values are listed, and users can simply choose the desired value. Users can also specify an arbitrary angle when necessary.

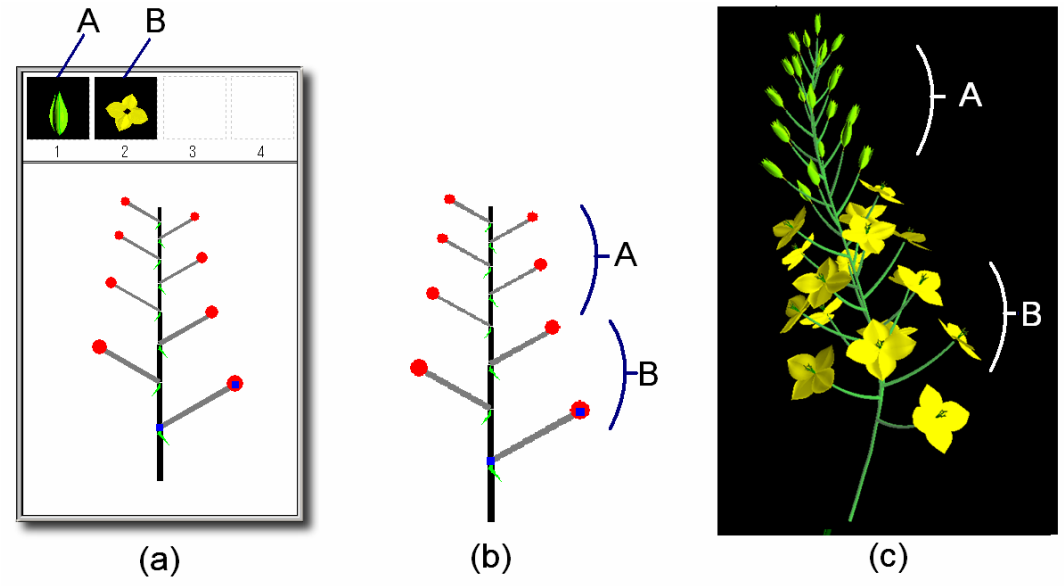


figure 4-4 : Example of placing two flowers on an inflorescences

(a) Bud and blooming flower models (A and B) are specified. (b) (c) Buds are placed on the higher (younger) half of the branches. Blooming flowers are placed on the lower (older) half of the branches.

Users associate flower models (created in the floral diagram editor) with inflorescence branch terminals. Aging of a flower is represented simply by multiple flower models; as shown in figure 4-4 (a), users import multiple models of different ages into the inflorescence editor top row in ascending order of age. The age is also linearly interpolated depending on the pattern (see section 2.1). For instance, when two flower models are provided for an indeterminate

inflorescence pattern, the lower half is associated with the old flower model and the upper half is associated with the young flower model (figure4-4 (b), (c)).

After adjusting parameters, users add geometric information to the inflorescence in the geometry editor. If desired, users can return to the structure editor and adjust parameters. The system provides immediate visual feedback to the 3D inflorescence model during the parameter adjustment process.

There are special inflorescence patterns called *head* and *spadix*. A head is a pattern in which small flowers cover a base called a disc, *e.g*. sunflowers. A spadix is a pattern in which many flowers are densely arranged on a thick stalk, *e.g.* Lysichiton camtschatcense. These inflorescence patterns can be compactly represented in floral diagrams, so we work with them in the floral diagram editor, allowing users to arrange flowers on the receptacle as well as arranging standard floral components.

Chapter 5. Geometry editors

Flower model components are 3D freeform shapes. We use a sketch-based interface to allow quick and intuitive modeling. Sketch-based modeling systems [Zeleznik et al. 1996; Igarashi et al. 1999] allow users to design interesting 3D geometry by drawing strokes on the screen; by contrast, traditional modeling systems require users to work with menus and many control points. A key aspect of sketch-based systems is that they make strong assumptions in interpreting user input to maintain a simple user interface. Our system simplifies the interface by providing a customized modeling interface for each floral component. Traditional modeling interfaces are generally suitable for careful editing by expert users; sketching interfaces are suitable for quick exploration by novices or casual users.

5.1. Floral receptacle and floral components

In the geometry editor, users can create the geometries of the floral receptacle, pistil, stamen, petal, and sepal.

A floral receptacle is defined as a surface of revolution, the profile of which is given by a user as a freeform stroke. A pistil is modeled using an inflation algorithm similar to “extrusion” in the Teddy system [Igarashi et al. 1999]. A stamen is defined as the sweep surface of a circle along a central axis drawn by the user. The user then draws another stroke to describe the axis of the stamen’s *anther* and the system creates a mesh by warping an ellipsoid along this stroke.

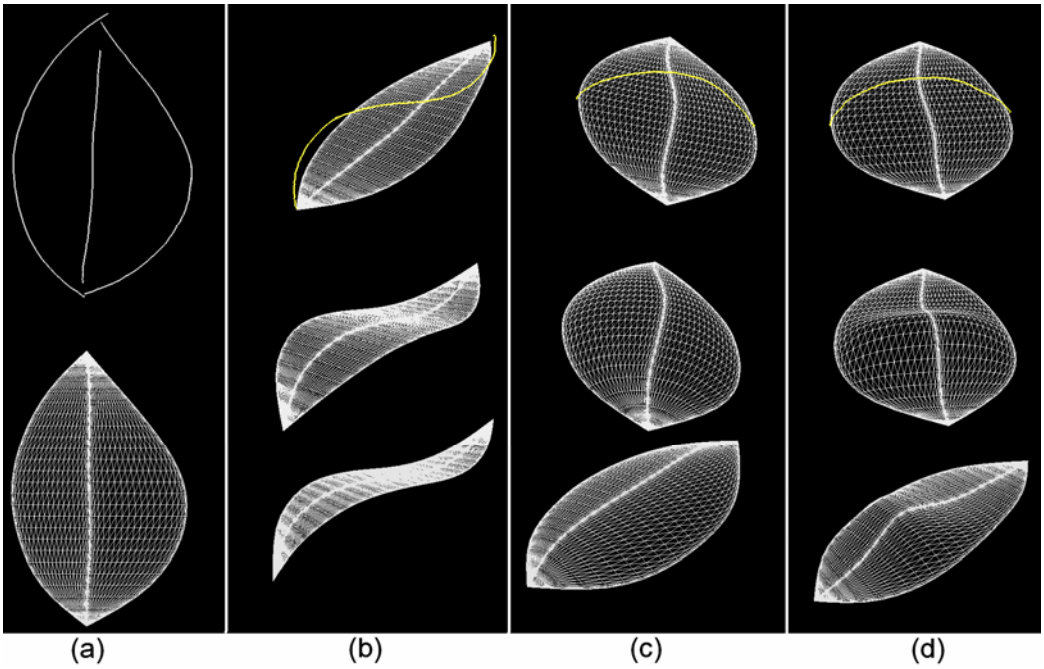


figure 5-1 : *Petal modeling 1*

(a) Initial creation. (b) Transforming an object along the center vein. (c) Transforming an object in global mode and (d) in local mode.

The petal and sepal share a common user interface (figure 5-1). A user first draws three

strokes to represent the outline and central vein of the petal (the central stroke may be omitted). The system returns a flat petal object (figure 5-1). Next, the user draws modifying strokes; these strokes are interpreted as cross-sections of the object (figure 5-1 (b), (c), (d)). Modifying strokes have two modes: global and local. In the global mode, a modifying stroke deforms the entire object, while in the local mode, only part of the object is deformed (figure 5-1 (d)). Users can switch between the two modes by selecting a button. To add realism, users can also add noise and texture.

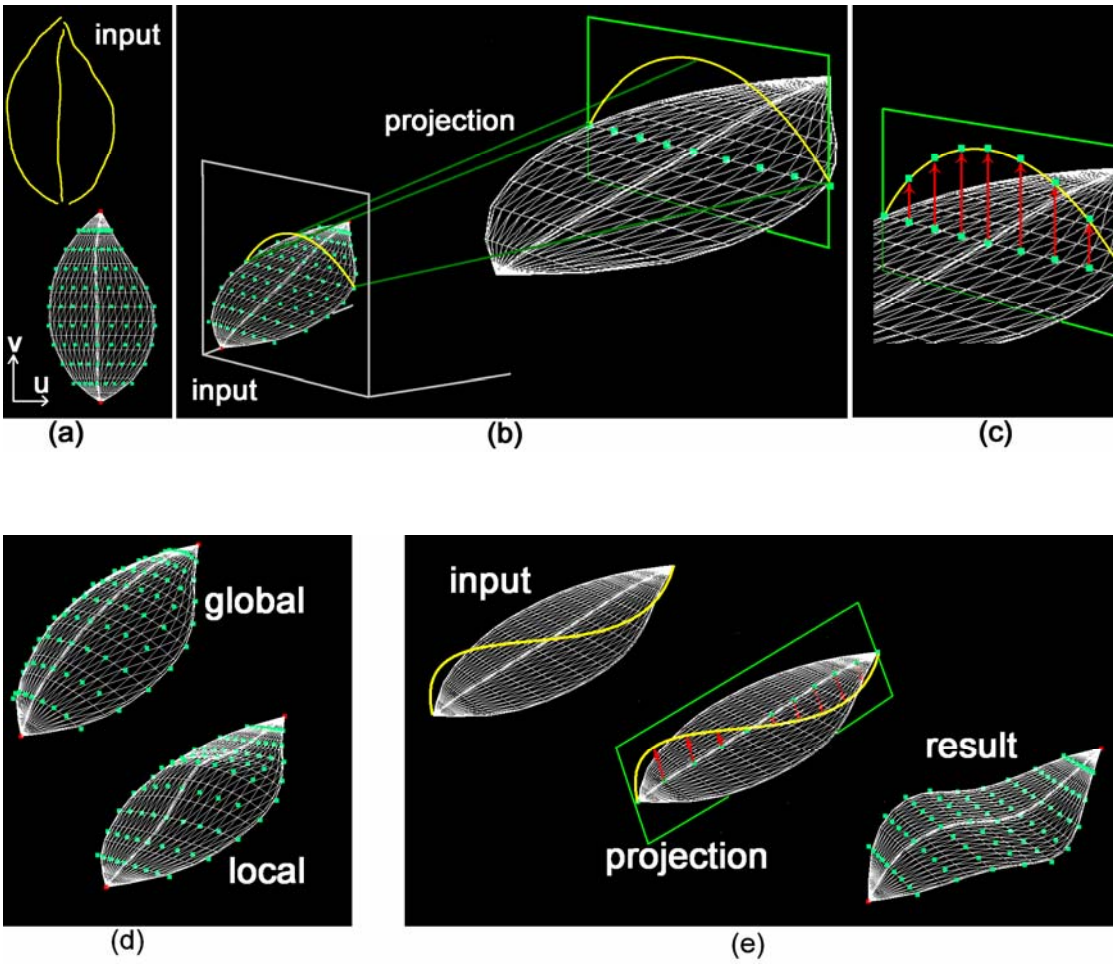


figure 5-2 : *Petal modeling 2*

(a) Initial creation. (b) (c) The system maps the 2D stroke. (d) Resulting geometry in global and local modes. (e) An example of a modifying stroke along the vertical direction.

A petal object is implemented as a B-spline surface. When the initial three outline strokes are drawn, the system generates control points of the B-spline surface, shown in figure 5-2 (a). We parameterize the surface using u and v coordinates, where the u-axis corresponds to horizontal direction and the v-axis corresponds to vertical direction. The system saves the plane on which the initial surface lies as a base plane. Modifying strokes move control points perpendicular to this base plane. If a user draws a modifying stroke in the u direction, the system first finds the control point nearest to the stroke's starting point on the screen. Control points that have the same v value as the base point are marked as target control points. The system projects the stroke on a plane that passes through target control points and is perpendicular to the base plane (Figure 5-2 (b)). Next, the system moves target control points to the projected stroke (Figure 5-2 (c)). In the global mode, the system moves all control points on the surface, and in the local mode it moves only neighboring points (figure 5-2 (d)). The displacement amount smoothly decays toward the petal's top and bottom. When a modifying stroke is drawn in the v direction, the system projects the stroke to a plane containing the central axis, perpendicular to the base plane (Figure 5-2 (e)). The system then moves control points so that all points with the same v-coordinates move the same amount. In this case, there is no difference between global and local modes.

5.2. Inflorescences

The interface for modeling the geometry of an inflorescence is very simple. After selecting an inflorescence pattern and adjusting its parameters in the structure editor (figure 4-3 : *Inflorescences editor*), the user draws the selected inflorescence's central axis as a 2D freeform stroke. The system then creates the 3D geometry of the inflorescence, displaying the curves that represent the axis and branches during the drawing operation. When the user completes drawing the stroke, the system creates a mesh for the stem and places the flower objects on branch terminals (figure 5-3).

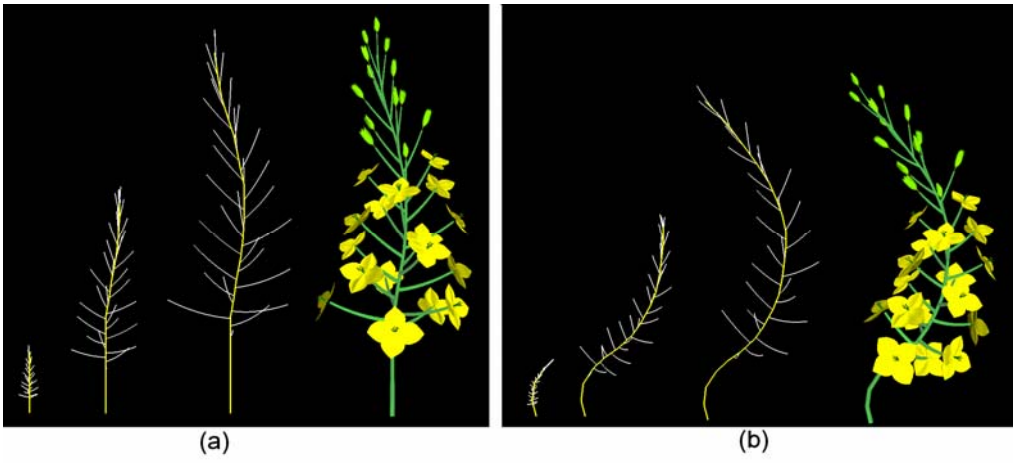


figure 5-3 : *A geometry editor for inflorescences*
The user draws the axis of the inflorescences freehand and the system provides the real time feedback during drawing.

Our system automatically adds appropriate depth to a user-drawn 2D stroke. Typical existing approaches first define a work plane that is almost perpendicular to the view direction and project the user-drawn stroke onto it [Cohen et al. 1999; Tobita and Rekimoto 2003]. A drawback of this approach is that it cannot create the typical shapes of stems such as spirals, and it requires that strokes be drawn twice. Our approach requires input of a single stroke and generates a 3D curve with a similar appearance regardless of viewing direction around the axis.

For example, when a user draws a sine curve, it creates a 3D spiral stroke. We achieve this

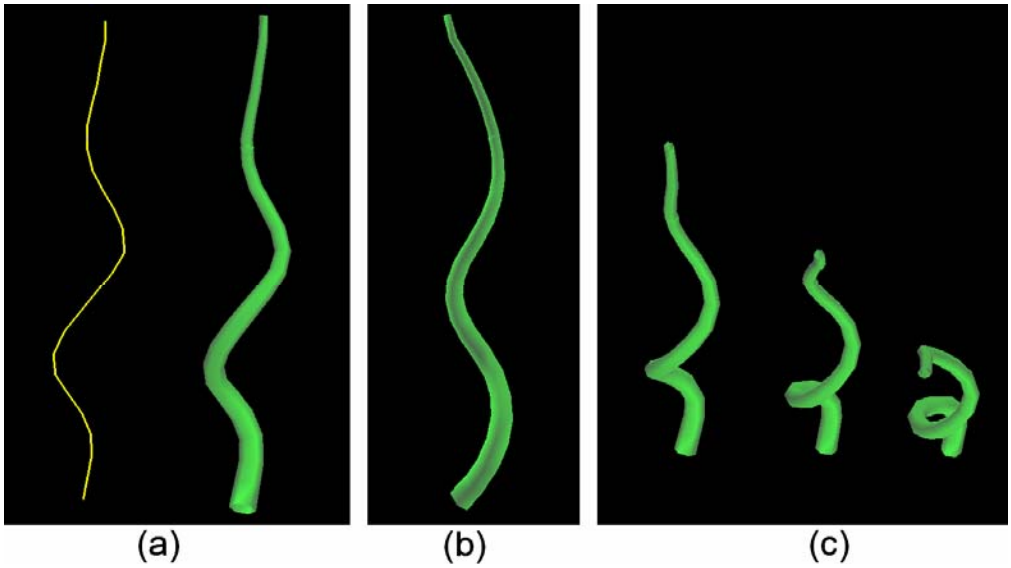


figure 5-4 : *automatic depth calculation*

(a) A stroke drawn by the user and the resulting 3D geometry models. (b) The model viewed from the right side. (c) The model viewed from higher perspectives.

effect by adding depth to the curve, so that the resulting curve has a constant curvature in 3D space (figure 5-4). Our algorithm is a specialized version of the energy-minimizing curve reconstruction proposed by Pentland and Kuo [1989]. The detailed algorithm is as follows.

We assume that a user draws a stroke on the *x*-*y* plane and that the viewing direction is in the positive *z* direction. The initial stroke is represented as follows:

$$stroke = \left\{ v_i \mid v_i = (x_i, y_i, z_i), z_i = 0 \right\}$$

where the *y*-axis corresponds to the vertical direction. We resample the input stroke so that vertices are equally spaced along the *y* direction. Our algorithm receives the stroke with *x* and *y* values as input and returns a new stroke with appropriate *z* values. To achieve this, our algorithm assumes that the resulting stroke has a constant curvature in 3D space along the *y*-axis, *i.e.*:

$$\left(\frac{d^2x}{dy^2}\right)^2 + \left(\frac{d^2z}{dy^2}\right)^2 = const$$

We compute z values by solving this equation. We first decide the constant value by taking the maximum squared value of the second derivatives of x along the axis. Given the constant value and the second derivatives of x, we can calculate absolute values of the second derivative of z by solving the above equation.

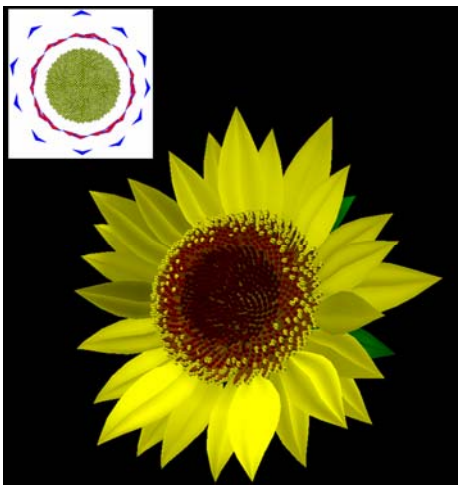
Direct solution of this formula yields only absolute values. The next task is to determine the signs of the second derivatives of z. We assume that the second derivative of z_0 is positive and determine the signs sequentially, so that successive signs change when the first derivatives of x cross zero.

Given the signed second derivatives of z, we calculate values for z by integrating them twice. We set z_0 to be 0 and adjust the first derivative of z_0 (the initial branch slope in the depth direction) so that the last z also becomes zero.

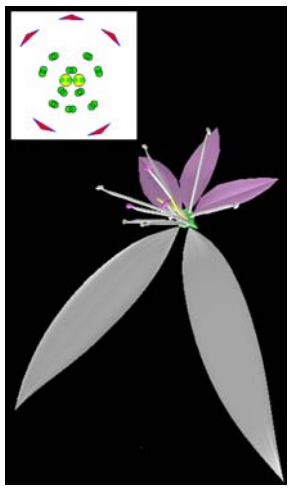
Chapter 6. Results

figure 6-1 shows flower models designed by the authors using our system with the corresponding floral diagrams and inflorescence patterns. Since our system provides a simple, intuitive user interface for defining complex structures and geometries, it took less than 40 minutes to design these complete flower models from scratch.

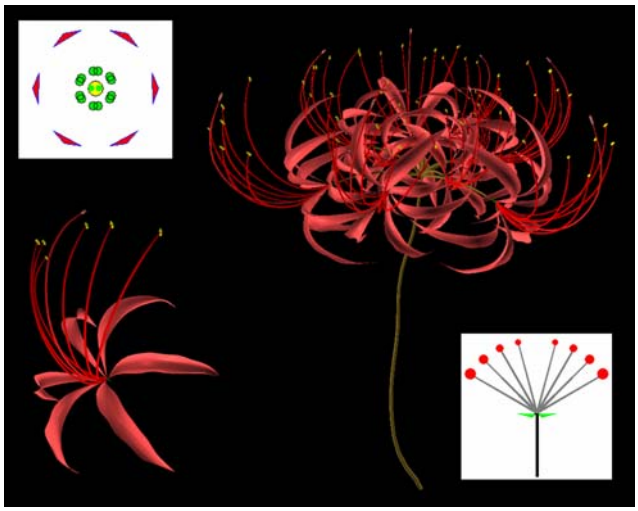
We also performed a preliminary user study to test the usability of our prototype system. Subjects are four university students who were novices of 3D modeling systems. We gave them short tutorial of our system in less than 20 minutes. After this, we showed flower pictures and asked them to create 3D flower models based on the pictures. Subjects were allowed to consult books to learn the floral diagrams and inflorescence patterns of the target plants. It took less than 40 minutes for them to design the complete flower models. figure 6-2 show the flower models designed by test user with corresponding pictures.



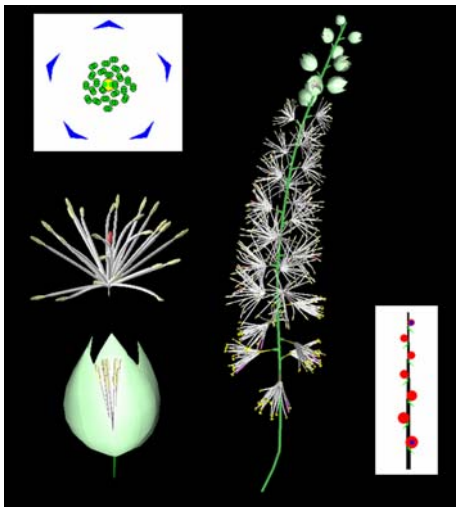
(a) Sun flower (40min)



(b) Saxifraga stolonifera (15min)



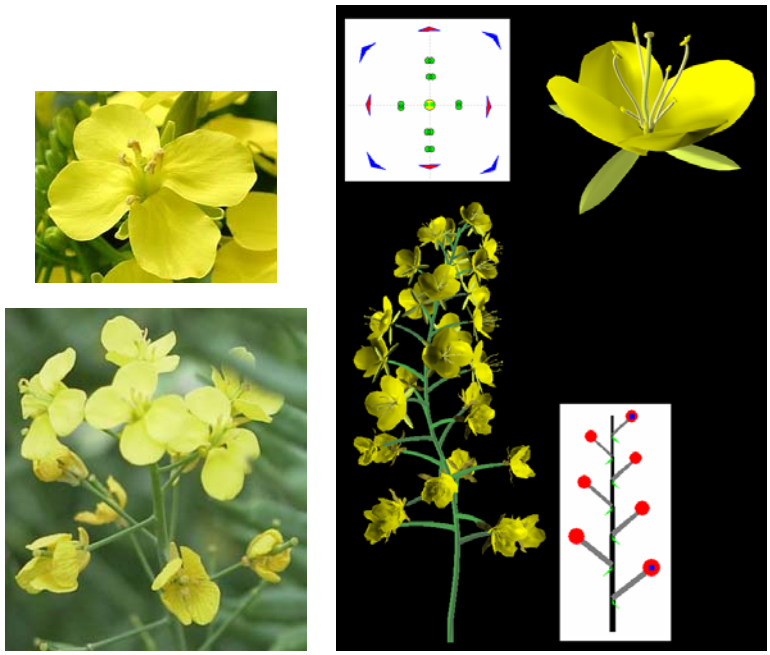
(c) Lycoris radiate (40 min)



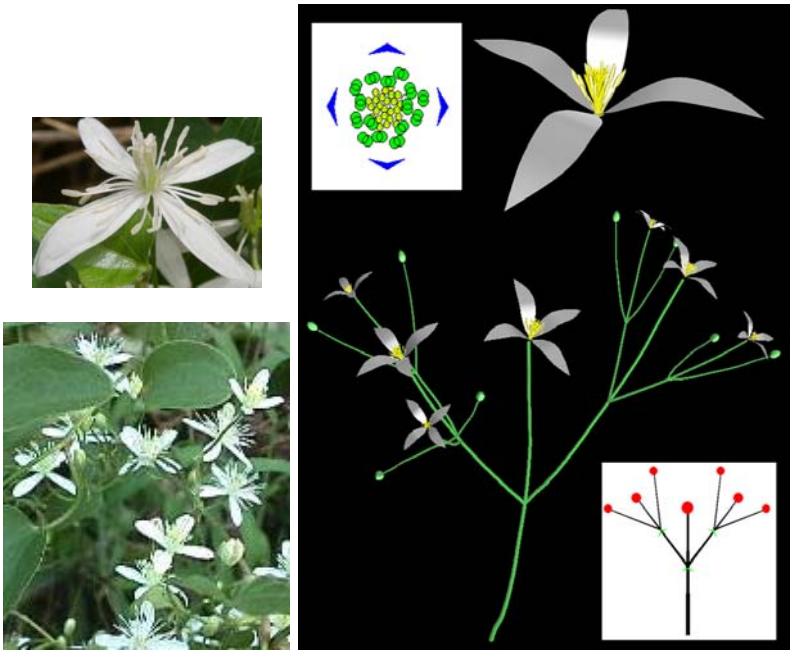
(d) Cimicifuga acerina (30 min)

figure 6-1 : *Example models*

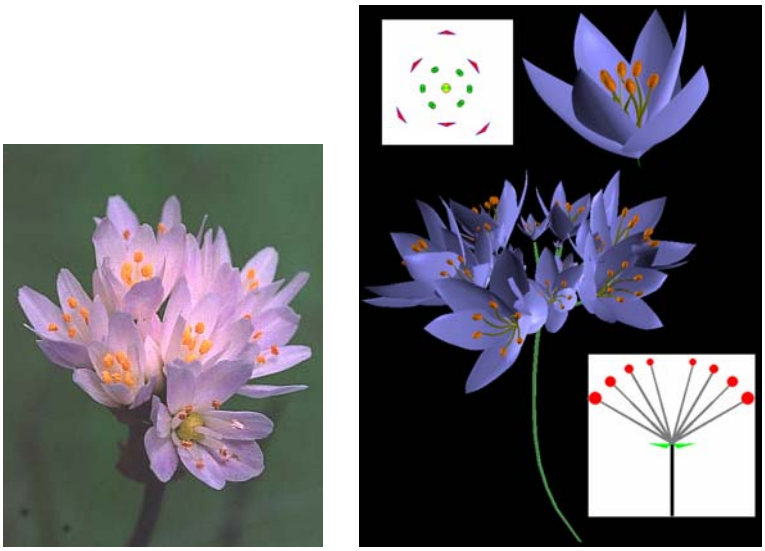
Example models designed by the author and the approximate time to complete each model.



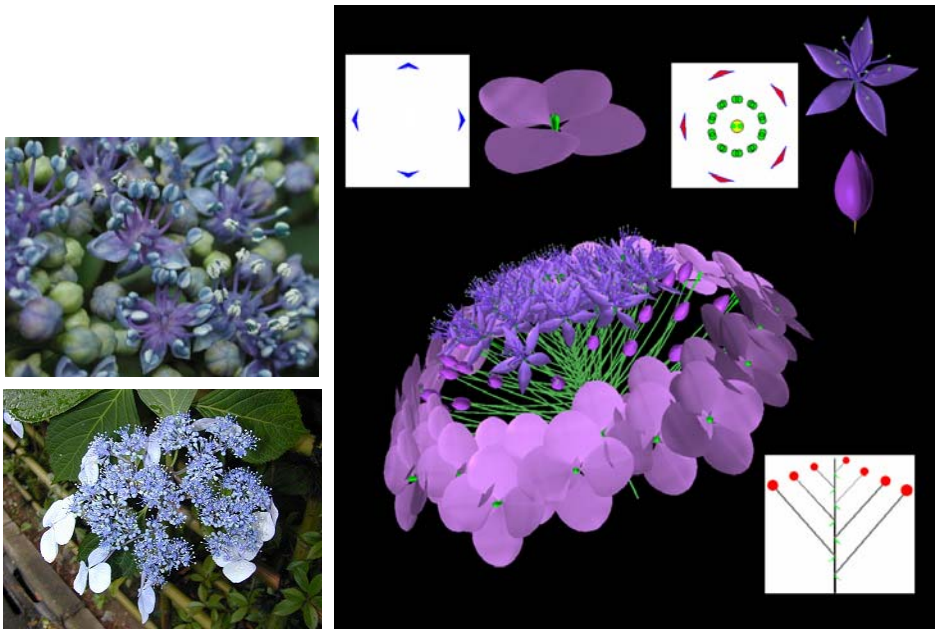
(a) Brassica rapa (30min)



(b) Clematis terniflora (30min)



(c) Allium roseum (30min)



(d) Hydrangea (45min)

figure 6-2 : *Results of user study*

Flower models designed by test user with the approximate time to complete each model.

Chapter 7. Discussion

In this paper, we propose a system for efficiently modeling flowers with correct botanical structures. We introduce floral diagrams and inflorescences, which were developed by botanists to describe structural information about flowers. We also propose a specialized sketch-based geometry editor for floral elements. Our current implementation supports eight inflorescence branching patterns, shown in figure 7-1. These are typical patterns selected from three inflorescence groups: indeterminate, determinate, and compound. Our results show that we can model plants successfully using these patterns, and it is probable that other branching patterns can be supported in a similar manner.

One limitation of the current system is that our inflorescence editor is not able to support the creation of a gradual progression of developmental flower stages. In addition, there are a few shapes that our geometry editor cannot create; for example, it is impossible to create petal-like shapes that do not have an elliptical outline.

The basis of our approach is the importance of separating structure editing from geometry editing. Our approach could be useful for modeling other targets with complicated structures and geometry, such as trees, insects, four-footed animals, etc.; in the future we would like to deal with these targets. Another interesting direction would be to extend our system to support entire plant structures. We are also interested in creating a flower arrangement application; this application would require a combination of biological and artistic knowledge, and would therefore be an interesting challenge.

We consider this work to be an example of an application-customized sketch-based interface; the success of the interface depends in part on balancing correct choice in expressive interface components against application needs: too-general components may allow users to make mistakes easily; too-limited ones may restrict user ability to reach goals, and may require a greater variety of components, which will be difficult to learn. The proper design rules for making such choices have yet to be elucidated; we hope that our system provides an instance from which such rules may someday be drawn.

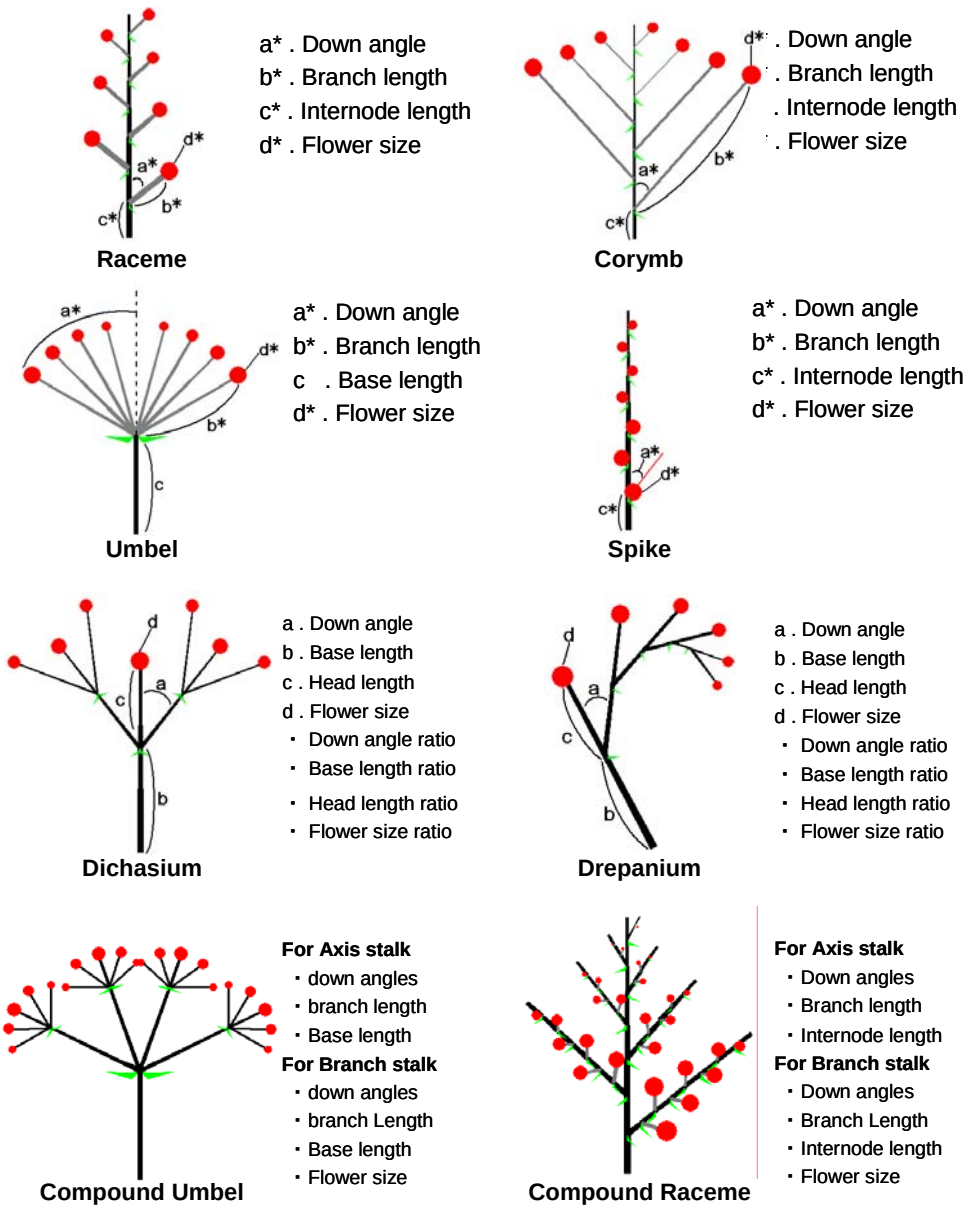


figure 7-1 : *Inflorescence patterns and their parameters in our current implementation*

The parameters with the superscript '*' are pair of numbers to be linearly interpolated along the stem. There are also some common parameters that are not shown in the figure: phototropism direction, stem hardness, stem width, rotate angle, and the number of branches. Dichasium and Drepanium patterns have additional “ratio” parameters for all parameters that determine the ratio of a child branch’s parameter values to those of a parent branch.

Chapter 8. References

Bell, A. D. 1991. Plant Form: An Illustrated Guide to Flowering Plant Morphology. Oxford University Press.

Boudon, F., Prusinkiewicz, P., Federl, P., Godin, C., and Karwowski, R. 2003. Interactive Design of Bonsai Tree Models. In Proceedings of Eurographics 2003: Computer Graphics Forum, 22, 3, 591-599.

Cohen, J., Markosian, L., Zeleznik, R., Hughes, J., and Barzel, R. 1999. An Interface for Sketching 3D Curves. In Proceedings of ACM I3D 99, 17-21.

Deussen, O. and Lintermann, B. 1997. A Modeling Method and User Interface for Creating Plants. In Proceedings of Graphics Interface 97, 189-197.

Deussen O. and Lintermann, B. 1999. Interactive Modeling of Plants. IEEE Computer Graphics and Applications, 19, 1, 56-65.

Eggli, L., Hsu, C., Elber, G., and Bruderlin, B. 1997. Inferring 3D Models from Freehand Sketches and Constraints. Computer-Aided Design, 29, 2, 101-112.

Hara, N., 1994. Syokubutu Keitaigaku (Plant Morphology). Sasakura Shoten 1994 (In Japanese).

Igarashi, T., Matsuoka, S., and Tanaka, H. 1999. Teddy: A Sketching Interface for 3D Freeform Design. In Proceedings of ACM SIGGRAPH 99, ACM, 409-416.

Ijiri, T., Igarashi, T., Takahashi, S., and Shibayama, E., 2004 Sketch interface for 3d modeling of flowers. In Technical Sketch SIGGRAPH ’04.

Lindenmayer, A. 1968. Mathematical Models for Cellular Interactions in Development, I & II. Journal of Theoretical Biology, 280-315.

Lintermann, B. and Deussen, O. 1996. Interactive Modelling and Animation of Branching Botanical Structures. In Proceedings of Eurographics Workshop on Computer Animation and Simulation 96, 139-151.

Lipson, H. and Shpitalni, M. 1996. Identification of Faces in a 2D Line Drawing Projection of a Wireframe Object. IEEE Transactions on Pattern Analysis and Machine Intelligence, 18, 10, 1000-1012.

Měch, R. and Prusinkiewicz, P. 1996. Visual Models of Plants Interacting with Their Environment. In Proceedings of ACM SIGGRAPH 96, ACM, 397-410.

Okabe, M., Owada, S., and Igarashi. T., 2005. Interactive design of botanical trees using freehand sketches and example-based editing. In Proceedings of Eurographics 2005: Computer Graphics Forum, 24, 3, 487-496.

Pentland, A. and Kuo, J. 1989. The Artist at the Interface. Vision and Modeling Technical Report 114, MIT Media Lab.

Prusinkiewicz, P., and Lindenmayer, A. 1990. The Algorithmic Beauty of Plants. Springer-Verlag, New York, 1990. With J. S. Hanan, F. D. Fracchia, D. R. Fowler, M. J. M. de Boer, and L. Mercer.

Prusinkiewicz, P., Hammel, M., Hanan, J., and Měch, R. 1996. L-systems: From the Theory to Visual Models of Plants. In Proceedings of the 2nd CSIRO Symposium on Computational Challenges in Life Sciences.

Prusinkiewicz, P., James, M., and Měch, R. 1994. Synthetic Topiary. In Proceedings of ACM SIGGRAPH 94, ACM, 351-358.

Prusinkiewicz, P., Mündermann L., Karwowski, R., and Lane, B. 2001. The Use of Positional Information in the Modeling of Plants. In Proceedings of ACM SIGGRAPH 2001, ACM, 289-300.

Pugh, D. 1992 Designing Solid Objects Using Interactive Sketch Interpretation, Computer Graphics, 25, 2, 117-126.

Runions, A., Fuhrer, M., Lane, B., and Federl, P. 2005. Model and visualization of leaf venation patterns. ACM Transactions on Graphics, 24, 3, 702-711.

Shimizu, T., 2001. Syokubutu Yougo Jiten (Dictionary of Botanical Terms). Yasaka Shobou (in Japanese).

Streit, L., Federl, P., and Sousa, M. C., Modelling Plant Variation Through Growth. In Proceedings of Eurographics 2005: Computer Graphics Forum, 24, 3, 497-506.

Tanaka, T., Naito, S., and Takahashi, T. 1989. Generalized Symmetry and its Application to 3D Shape Generation. Visual Computer, 5, 83-94.

Tobita, H., and Rekimoto, J. 2003. Flat3D: A Shared Virtual 3D World System for Creative Activities and Communication, IPSJ JOURNAL, 44, 2, IPSJ, 245-255 (in Japanese).

Zeleznik, R. C., Herndon, K. P., and Hughes, J. F. 1996. SKETCH: An Interface for Sketching 3D Scenes. In Proceedings of ACM SIGGRAPH 96, ACM, 163-170.