INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Semi-interactive morphogenesis

Jean Combaz — Fabrice Neyret

N° 5302

Septembre 2004

Thème 3

Rapport de recherche

ISSN 0249-6399 ISRN INRIA/RR--5302--FR+ENG

Semi-interactive morphogenesis

Jean Combaz * , Fabrice Neyret *

Thème 3 — Interaction homme-machine, images, données, connaissances
Projet Evasion

Rapport de recherche n° 5302 — Septembre 2004 — 14 pages

Abstract: This paper presents a method to simulate growth phenomena, and its application to the modeling of complex organic shapes (e.g., plants organs) and folded surfaces.

Our main contribution is the interactive and stable resolution of the mechanical problem of growth-induced deformations, based on the minimization of the energy due to the various constraints in the shell. From this, we propose a new modeling approach based on a set of growing tools: The user can apply 'hot spots', 'hot curves', or paint growing parameters on the surface to grow. Growth can be simulated either simultaneously to the user interaction, or once all parameters have been settled on the surface (which allows the use of textures of parameters and procedural operations). The main parameters are the intensity and anisotropy of growth, as well as their variations over time. Geometric constraints and plasticity can also be considered.

As our results show shapes can fold, bend, and curl as in nature, which deforming tools such as displacement map could not achieve. We demonstrate our tool with an interactive session and a gallery of shapes easily produced.

Key-words: Morphogenesis, growth, folds, surface modeling, procedural modeling, physical model, user interaction.

* projet EVASION INRIA Rhône-Alpes et Laboratoire GRAVIR/IMAG, laboratoire commun CNRS, Institut National Polytechnique de Grenoble INRIA et Université Joseph Fourier

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe, 38334 Montbonnot Saint Ismier (France)
Téléphone : +33 4 76 61 52 00 — Télécopie +33 4 76 61 52 52

Morphogénèse semi-interactive

Résumé : Ce rapport présente une méthode permettant de simuler des phénomènes de croissance, ainsi que ses applications pour la modélisation de formes organiques complexes ou des surface plissées.

Notre principale contribution est la résolution stable et en temps interactif du problème mécanique des déformations induites par la croissance, grace à la minimisation de l'énergie des contraintes qui s'appliquent à une coque. Nous proposons une nouvelle approche de la modélisation basée sur un ensemble d'outils de croissance: l'utilisateur peut appliquer des 'points chauds', des 'contrours de croissance' ou peindre directement les paramètres de croissance sur la surface. La croissance peut être simulée soit simultanément aux interactions de l'utilisateur, soit une fois que tous les paramètres aient été fixés sur la surface (ce qui autorise l'utilisation d'une texture de ces paramètres ainsi que les opérations procédurales). Les principaux paramètres sont l'intensité et l'anisotropie de la croissance, ainsi que leurs variations au cours de temps. On peut également tenir compte de la plasticité ainsi que de contraintes géométriques.

Comme le montre nos résultats, les formes peuvent se plisser, plier ou se tordre (comme dans la nature), ce que des outils de déformations basés sur des cartes de déplacements ne peuvent réussir de manière simple.

Mots-clés : Morphogénèse, croissance, plis, modélisation surfacique, modélisation procédurale, modèle physique, interface utilisateur.

1 Introduction and Previous Work

Natural shapes are often complex, globally as well as in their details. Modeling them using classical geometric modelers can thus be especially tedious. A number of these natural shapes result from a growth phenomena [Tho17] (e.g. plants, organic shapes) or can be modeled as such (e.g. folded surfaces, geological shapes). Even though growth-induced deformations can be physically simulated similarly to cloth animation or deformable objects [BW98, TPBF87], real-world parameters are usually unknown. Moreover, artists want to keep enough control so a blind off-line simulation is not desirable.

Various tools have been introduced to model or enrich a shape by deforming it either at global (e.g. Free-Form Deformations [SP86, Coq90]) or local scale (e.g. displacement maps [Coo84, WMF$^+$00]). Similarly, several virtual sculpting tools have been proposed either for the overall shaping (e.g. [BL95, FCG00]) or to edit and enrich an existing shape (e.g. Maya Artisan [Ali]). Although the user interaction is more intuitive with these tools than using a classical geometric modeler, the user still has to explicitly specify where each bulge or pit should be. For shapes that are complex (either globally or locally) this can be very tedious.

We would like to let the user specify the global and local aspect of the shape without having to define each very detail. [CN02] introduces the *expansion texture* paradigm (summarized in the next section) to shape folded surfaces: the user specifies expansion (i.e. growth) parameters using either a brush or a texture map. The surface folds under the local constraints caused by the growth. Since the parameters are attached to the surface, even large amplitude of growth can be simulated and still yields natural looking results. Oppositely, surface displacement tools cannot be applied for huge amplitudes: the displacement is done along the normal and cannot curl the shapes. Tools like Maya Artisan [Ali] allows to iterate displacement relying on the new normals. But such a geometric growth quickly degenerates into unnatural and self-intersecting surfaces. This does not occur with expansion textures due to the use of a mechanical model to simulate the deformations.

The [CN02] model has similarities with physical models such as the ones used for deformable surfaces (e.g. [TPBF87]). However, the mechanical model is tuned to give controllable results (e.g. fold wavelength) and the solving is quasi-static (the purpose is to simulate the shape at equilibrium, not the dynamics).

Our model draws on [CN02] and extends it to permit huge deformations at interactive rate (i.e. we are not limited to adding details) and to increase the flexibility. Thus, our tool applies to the modeling of a large variety of shapes (and not simply folds). For that we revisit the mechanical model in order to make it more general, steady and efficient as detailed in section 3.

From this new engine we propose a new modeling approach based on a set of growing tools. These allow both local and global shaping, and both interactive and semi-procedural specification of growth. We describe these tools in section 4.

In section 5, we demonstrate our tool with interactive sessions, as well as samples of typical shapes that are easily produced (e.g., plant organs, and folded shapes like a tablecloth or an unmade bed).

2 The expansion texture formalism

Since we draw on and expand the approach of [CN02], we share the same formalism. We present here a summary (see the paper for more details).

The growth is modeled by local anisotropic expansions which acts on a free form surface discretized by a triangular mesh. The growth does not modify explicitly the surface but a *reference state* defined by the rest length $\bar{l}_e$ of each edge e, the rest mean curvature $\overline{\kappa}_v$ of each vertex v and the graph connectivity of the mesh (see figure 1). This *virtual* reference state (updated by the growth) defines locally the ideal shape the surface should attain. The 'real' shape is obtained by a mechanical solver which minimizes the constraints, i.e. the differences between the mesh and the reference state. Note that the surface has to be dynamically remeshed.

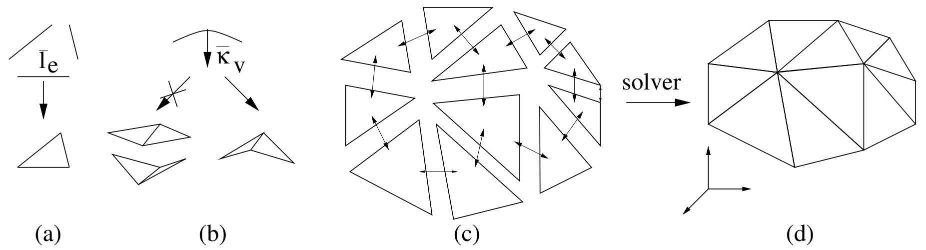

Figure 1: *The* virtual reference state *is defined by the rest lengths* $\bar{l}_e$ (a), *the rest mean curvatures* $\overline{\kappa}_v$ (b) *and the graph connectivity* (c) *of the mesh. The solver finds the shape* (d) *that minimizes the difference with the reference state (modified by the growth).*

The local expansion is described by an *expansion tensor field* over the surface. An expansion tensor can be represented by a 2×2 positive definite symmetric matrix $\mathcal{D}$. It can be defined by three parameters: two expansion rates D_1 and D_2 relative to the eigenvector directions, and an angle θ which defines the orientation of the eigenvectors basis relative to the local $2D$ frame. The expansion along a vector $\bar{l}$ is then given by $\mathcal{G}_{\mathcal{D}}(\bar{l}) = \sqrt{\vec{l}^t \mathcal{D} \vec{l}}$.

The growth parameters the user can tune are: the rate and the anisotropy of the expansion (i.e. D_1, D_2, θ) at every active locations; and desired wavelength and regularity λ_1, λ_2 of folds or curls. External constraints can be added by the user, such as static collision with solid objects and partial or total attachment (i.e. $1D$, $2D$ or $3D$ constraints) of some surface locations. The local growth parameters interactively painted by the user or predefined using a paint program are stored in a map – the *expansion texture*. A parameterization of the surface is thus required.

3 Our mechanical model and solver

We want to allow huge expansion in order to deal with a large range of morphogenesis situations, and we want to provide more precise tools to the user (see section 4). For this, our model must not depend on any global parameterization (which would be soon overstretched). A very stable model is also required: we rely on an energetic formalism close to thin shells mechanical model. We also want to provide the user with higher level controls (i.e. semi-procedural, or semi-interactive), by allowing him to tune how long a growth will last, according to which activation pattern (e.g. delayed starting, progressive decay...).

3.1 Expansion model

To avoid a global (u, v) parameterization we store the various local data at the vertices or at extra nodes within the faces. For extra nodes we define a radius of influence r and an influence kernel $w(x)$. We define a local $2D$ frame for each vertex using a local geodesic polar parameterization [WW94].

For growth data defined at vertices we model the expansion of the reference length $\bar{l}_e$ of edge e as the average of the expansion at the tips: $\mathcal{G}(e) = \frac{1}{2}(\mathcal{G}_{\mathcal{D}_{P_1}}(\vec{l_e}) + \mathcal{G}_{\mathcal{D}_{P_2}}(\vec{l_e}))$ where $\mathcal{D}_{P_1}$ and $\mathcal{D}_{P_2}$ are the tensors at the edge tips P_1 and P_2. For growth data defined at an arbitrary node N we define $\mathcal{D}_{P_i} = w(\|P_i - N\|)\mathcal{D}_N$. For growth data defined on an arbitrary curve $\mathcal{C}$ (based on control points $\{N_i\}$) we define $\mathcal{D}_{P_i} = w(d(P_i, \mathcal{C}))\mathcal{D}_N$ where $d(P, \mathcal{C})$ is the distance from a point P to the curve $\mathcal{C}$.

As stated above and detailed in section 4 we account for the duration of growth in time. The expansion of an edge e between iteration steps corresponding to time t_1 and t_2 is $\boldsymbol{\mathcal{G}}(e) = \mathcal{G}(e)^{t_2 - t_1}$.

We also take *plasticity* into account if desired, which avoids accumulating undissipated stress in the surface. This permit smoother shapes in case of huge expansion. When the difference between the reference length $\bar{l}_e$ and the corresponding real length exceeds a threshold α_{plast} we operate a relaxation over $\bar{l}_e$:

`if` $\bar{l}_e < (1 - \alpha_{plast}) l_e$
  $\bar{l}'_e = (1 - k_{plast})\bar{l} + k_{plast}(1 - \alpha_{plast}) l_e$
`else if` $\bar{l}_e > (1 + \alpha_{plast}) l_e$
  $\bar{l}'_e = (1 - k_{plast})\bar{l} + k_{plast}(1 + \alpha_{plast}) l_e$
`endif`

3.2 Our energy-based mechanical model

Contrary to [CN02] we do not compute explicitly the displacement of the vertices, but we define an energy E of deformation between the surface and the reference state. This is closer to mechanical formalism and this ease the accounting of new phenomena. We use three components: $E = E_{membrane} + E_{bending} + E_{pressure}$. This definition is inspired by the models of thin shells (see [GHDS03] for details) where the energy is the sum of two components similar to $E_{membrane}$ and $E_{bending}$.

The expression we propose below for these two energies are easily to calculate and derive. It produces realistic deformations. However, more accurate (and thus time-consuming) formulations could be used in the scope of off-line complex growth simulation. The *pressure* is not a classical notion here and was introduced by [CN02] to control the wavelength and the regularity of the shape. We adapt this term to our energetic formalism.

3.2.1 Membrane energy

$E_{membrane}$ measures the stretching (change of area) and the shearing of the surface (change in length but not area). We opted for a finite element expression of the stress. We integrate on the mesh the density of elastic deformation energy $E^t_{membrane}$ of the triangles t:

$$\begin{aligned} E_{membrane} &= \sum_t \overline{A}_t E^t_{membrane} \\ \text{where} \qquad E^t_{membrane} &= \frac{1}{2}\sum_{i=1}^{2}\sum_{j=1}^{2} \sigma^t_{ij}\epsilon^t_{ij} \end{aligned}$$

$\overline{A}_t$ is the reference area of the triangle t, ϵ^t_{ij} (the *strain tensor*) measures the deformation of t respective to its reference state, and σ^t_{ij} (the *stress tensor*, which describes the internal forces which acts on the triangle) can be deduced using the Hook's law (see for instance [OH99] for details).

3.2.2 Bending energy

$E_{bending}$ controls the flexure of the surface. In mechanics, models classically measure a squared difference between curvature tensors at current and rest states. We prefer to use the mean curvature – which is scalar – because it is faster to evaluate (see [DMSB99]), easier to derive and the quality of the approximation is enough for our simulations. We integrate the difference between the mean curvature κ^v and the reference curvature $\overline{\kappa}^v$ along the surface. So, we have:

$$\begin{aligned} E_{bending} &= \sum_v \overline{A}_v E^v_{bending} \\ \text{where} \qquad E^v_{bending} &= (\kappa_v - \overline{\kappa}_v)^2 \end{aligned}$$

A_v is the area of the Voronoï cell which contains the vertex v.

3.2.3 Pressure energy

[CN02] relies on normal stress $\vec{F}_N$ for this term, thus turning the triangles compression into a displacement along the normal. This force succeeds in modulating the fold wavelength and the growth preferred direction. We adapted this term to define a pressure energy $E_{pressure}$:

$$\begin{aligned} E_{pressure} &= \sum_v \overline{A}_v E^v_{pressure} \\ \text{where} \quad E^v_{pressure} &= -\frac{1}{\overline{A}_v}(v - \widetilde{v})\vec{F}^v_N \end{aligned}$$

$\widetilde{v}$ is the $3D$ position of the vertex at a previous sub-step in the optimization solver.

3.3 The solver

To generate a new global equilibrium of the shape after a growth step, we solve the energy minimization using a classical Conjugate Gradient Optimization (see [PTVF] for details). The optimality of the steps provides a more robust convergence than the simple displacement used in [CN02] (corresponding to a constant step Gradient Optimization). We stop the optimization when the difference between the energies in two consecutive steps or the maximum stress along the surface are below a small threshold.

4 Growing tools

In nature, growth exists in a variety of modalities: in plants, cellular proliferation mostly occurs at apexes (i.e. tip of branches). This causes the 1D nature of branches. At some stage of the leaves development the surface enlarges due to the cellular division along the leaf edge. Geological rifts are another situation showing an expansion occurring along a curve. Animal tissues grow more homogeneously, but often with an important anisotropy: body proportions change from embryo to adult [WFM01]. The distribution of active and passive (expansion-wise) areas on a surface strongly influences the pattern of deformation due to mechanical constraints: an expanding spot can only bulge out of the surface. An expanding linear area crossing the surface might simply enlarge it (e.g. rifts). When the whole surface is expanding, any heterogeneity or local constraint will result in folds or curling.

In order to provide the user with most flexibility, we account for the same variety of modalities: Our tools can be *hot spots* (0D), *hot curves* (1D) or *hot surfaces* (2D). Formally these three kinds of active regions could all be simply painted in an expansion texture. The difference mostly lies in their specification and control: Spots and curves control points are located precisely on the mesh by relying on their barycentric coordinates within faces (and updating this data during remeshings). Hot surface data is defined at the mesh vertices.

4.1 Hot spots (0D tools)

A hot spot is defined by the growth parameters $(D_1, D_2, \theta, ...)$, a radius of influence r and an activation curve $I(t)$ (growth intensity in time, or by default a constant and a duration). Since a hot spot generates a bulge, a long activation results in the growth of a branch.

We implemented various spot tools such as rotating the frame over time, and splitting a spot into several spots within the radius of influence. As shown in the results section, this results into branching. It would be easy to connect this mechanism to a procedural event generator such as L-systems [PLH88] or [DL97]. We tried simple ones as illustrated in the result section.

4.2 Hot curves (1D tools)

A hot curve can share common growth parameters, which orientation can follows the local frame along the curve. We denote D_T the expansion along the curve (i.e. tangential) and D_N the expansion in the orthogonal direction (i.e. normal). To model an heterogeneous growth, growth parameters can be painted and attached to the control points (and thus to the surface), or specified by a 1D texture $I(u)$ (relying on a curvilinear parameterization u). The last case can typically be used for controlling the activation of growth in time: A user-defined 1D texture defines the growth intensity along a normalized range of time, and the local time offset and local time pace are interpolated along the curve.

The curve is attached to the surface and thus is distorted together with it (by the effect of its activity plus the one of other growing tools). For instance, this allows to define a very short curve across an apex (i.e. a hot spot) and to let it enlarge before activating it. This can be used to generates a leaf from a stem. It is easy to create automatically such a small curve from geometric parameters (e.g. the apex normal – which corresponds to the stem axis – , or the vertical), so a procedural tool can generate and trigger hot curves as illustrated in results.

Growth anisotropy along the curve can result in different shapes: for a straight curve, a totally radially oriented growth (i.e. $D_T = 1$ and $D_R > 1$) generates a plane. Adding a slight tangential growth ($D_T > 1$) yields tangential folds, while a slight tangential contraction ($D_T < 1$) would yield radial folds. For a curve, the tuning corresponding to the balanced case (i.e. a plane) depends on the local curvature γ. If the user wants to control the regularity of the generated surface (see the mushrooms example on figure 2(b)) a normalization has to be applied: in such case we define $D_T = 1 + \lambda(D_R - 1)r_e\gamma$, with λ a control factor (1 for a plane) and r_e the efficient influence radius $= \frac{r}{\int_0^r w(x)}$, where $w(x)$ defines the shape of the influence kernel.

4.3 Hot surfaces (2D tools)

The user interface for the 2D tool draws on and extends the [CN02] one: growth parameters are defined either by interactive painting (parameters are sampled at mesh vertices, orientation is given by the mouse direction) or by a 2D texture (a temporary (u, v) parameterization – e.g. a projection – is then defined on the targeted area). From this, we added several features. In the interactive case we store a time offset t_0 and a time pace dt at each painted vertex: this allows to control how long the growth will apply (i.e. while $(t - t_0)dt < 1$). This also permits to spread huge growth along time, thus allowing stable huge deformations. We also implemented various painting tools in order to apply growth 'brushes' and interpolation. For the latter, we rely on adapted Laplace interpolation [GS03]: the user paint values (spots or curves) which are spread by solving a diffusion equation (we used an implicit solver for efficiency). Interpolating tensors is not as simple as for scalars. Similarly to what [Tur01] does for vectors, we first interpolate the tensor coefficients. Then we rebuild an orthonormal frame and we reconstruct the interpolated tensor by scaling it with the separately interpolated eigenvalues.

5 Results

Our results are presented on figure 2. Animated sequences are framed.
• *(a)*: Using a hot spot (the initial shape is a sphere). Isotropic growth; anisotropic growth (folds appear in the most expanded side); with twisted frame; branchings by splitting the spot.
• *(b)*: Using a hot curve (the initial shape is a triangle). Mushrooms (the initial hot curve is a circle at the stem tip): without normalization; with the normalization. Leaves (the initial hot curve is a half-circle crossing the apex at the stem tip; its control points are figured on the first two images).
• *(c)*: Using a hot surface (and isotropic growth). Pumpkin: rigid lines (in grey) were painted on a sphere before the homogeneous growth. Leaves: growth is painted only on the edge, using various tunings (frequency, gradient). Flowers: painting growth only on the corners grows and curls them into petals.
• *(d)*: Modeling an unmade bed. A physical cloth simulation would require to know the initial state, the material parameters, and the history of forces. Using our model, the artist simply draws areas where to add 'extra matter' (resulting in folds) much like a sculptor would. The frequency is tuned by the user and the folds are oriented by the mouse direction.
• *(e)*: Table cloth. Starting with a cylinder, the user paints a totally tangential anisotropic expansion on the bottom, and a zero-expansion circle on the top. Then he interpolates the growth parameters in between. Note that the user can exaggerate the folds (as for the drapes in ancient paintings) on desire.
• *(f)*: A cabbage leaf modeled using an expansion texture (much like in [CN02]).
• *(g)*: Procedural growth. Apexes are split randomly. Note that our model produces a continuous growth, and that mechanical constraints move branches apart naturally.

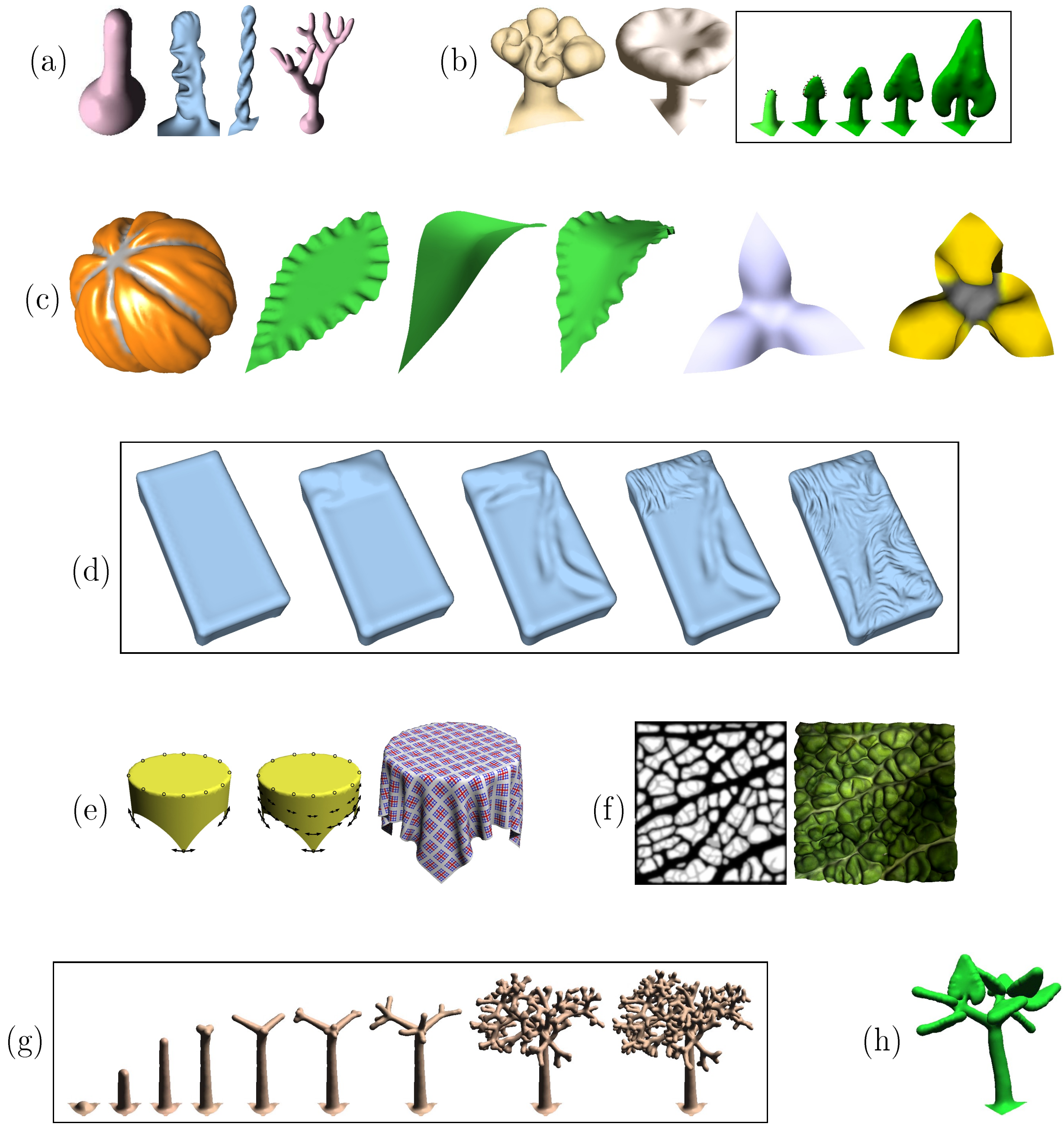

Figure 2: Results (sequences are framed). *(a)*: Using hot spots. *(b)*: Using hot curves. *(c-f)*: Using hot surfaces. *(g)*: Procedural growth.

6 Conclusion and Future Work

We have presented a new modeling paradigm based on the control of surface growth. The engine consists on a new efficient and stable mechanical model and solver able to simulate growth-induced deformations. Compared to physical simulations we are able to provide the user with fine control of the shape. Compared to traditional modeling tools and to sculpting tools we do not require the user to explicitly define each very detail. However, by tuning the radius of influence the user can indeed edit details whenever he wants. Most of the result we show could not have been modeled easily with any other modeling tool or physical simulation.

Our approach is able to model either global shapes or complex surface details, relying on interactive or semi-interactive operations. However, the user may sometime want an even higher level of control. Our model could be connected to procedural tools such as L-systems [PLH88] to generate complex structures. Similarly, expansion texture could be procedurally evaluated or simulated using noise [Per85] or reaction-diffusion [Tur91]. Finally, it would be interesting to add to our model a scheme for the semi-procedural synthesis of surface colors and textures.

References

[Ali] Alias. Maya Artisan. `www.alias.com/`.

[BL95] James R. Bill and Suresh K. Lodha. Sculpting polygonal models using virtual tools. In *Graphics Interface '95*, pages 272–279, May 1995.

[BW98] David Baraff and Andrew P. Witkin. Large steps in cloth simulation. In *Proceedings of ACM SIGGRAPH 98*, Computer Graphics Proceedings, pages 43–54, July 1998.

[CN02] Jean Combaz and Fabrice Neyret. Painting folds using expansion textures. In *Pacific Graphics*, october 2002. `www-imagis.imag.fr/Publications/2002/CN02`.

[Coo84] Robert L. Cook. Shade trees. In Hank Christiansen, editor, *Computer Graphics (SIGGRAPH '84 Proceedings)*, volume 18, pages 223–231, July 1984.

[Coq90] Sabine Coquillart. Extended free-form deformation: A sculpturing tool for 3D geometric modeling. In Forest Baskett, editor, *Computer Graphics (SIGGRAPH '90 Proceedings)*, volume 24, pages 187–196, August 1990.

[DL97] Oliver Deussen and Bernd Lintermann. A modelling method and user interface for creating plants. In *Graphics Interface '97*, pages 189–198, May 1997.

[DMSB99] Mathieu Desbrun, Mark Meyer, Peter Schröder, and Alan H. Barr. Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proceedings of ACM SIGGRAPH 99*, pages 317–324, August 1999.

[FCG00] Eric Ferley, Marie-Paule Cani, and Jean-Dominique Gascuel. Practical volumetric sculpting. *the Visual Computer*, 16(8):469–480, dec 2000. `www-imagis.imag.fr/Publications/2000/FCG00`.

[GHDS03] Eitan Grinspun, Anil N. Hirani, Mathieu Desbrun, and Peter Schröder. Discrete shells. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 62–67, 2003.

[GS03] Leo Grady and Eric Schwartz. Anisotropic interpolation on graphs: The combinatorial Dirichlet problem. Technical Report CAS/CNS-TR-03-014, Boston University, July 2003.

[OH99] J.F. O'Brien and J.K. Hodgins. Graphical modeling and animation of brittle fracture. In *SIGGRAPH'99 Conference Proceedings*, pages 137–146. ACM SIGGRAPH, 1999.

[Per85] Ken Perlin. An image synthesizer. In B. A. Barsky, editor, *Computer Graphics (SIGGRAPH '85 Proceedings)*, volume 19(3), pages 287–296, July 1985.

[PLH88] Przemyslaw Prusinkiewicz, Aristid Lindenmayer, and James Hanan. Developmental models of herbaceous plants for computer imagery purposes. In *Computer Graphics (Proceedings of SIGGRAPH 88)*, volume 22, pages 141–150, August 1988.

[PTVF] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Reciepes in C++, The Art of Scientific Computing (2nd edition)*. Cambridge University Press. ISBN 0-521-75033-4.

[SP86] Thomas W. Sederberg and Scott R. Parry. Free-form deformation of solid geometric models. In *SIGGRAPH '86 Proceedings*, volume 20, pages 151–160, August 1986.

[Tho17] D'Arcy Wentworth Thompson. *On Growth and Form*. Cambridge University Press, Cambridge, 1917.

[TPBF87] Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. Elastically deformable models. In *Computer Graphics (Proceedings of SIGGRAPH 87)*, volume 21, pages 205–214, July 1987.

[Tur91] Greg Turk. Generating textures for arbitrary surfaces using reaction-diffusion. In Thomas W. Sederberg, editor, *Computer Graphics (SIGGRAPH '91 Proceedings)*, volume 25, pages 289–298, July 1991.

[Tur01] Greg Turk. Texture synthesis on surfaces. In *Proceedings of ACM SIGGRAPH 2001*, pages 347–354, August 2001.

[WFM01] Marcelo Walter, Alain Fournier, and Daniel Menevaux. Integrating shape and pattern in mammalian models. In *Proceedings of ACM SIGGRAPH 2001*, pages 317–326, August 2001.

[WMF+00] Xiaochuan Corina Wang, Jérome Maillot, Eugene L. Fiume, Victor Ng-Thow-Hing, Andrew Woo, and Sanjay Bakshi. Feature-based displacement mapping. *Rendering Techniques 2000: 11th Eurographics Workshop on Rendering*, pages 257–268, June 2000.

[WW94] William Welch and Andrew Witkin. Free-form shape design using triangulated surfaces. In *Proceedings of ACM SIGGRAPH 94*, pages 247–256, July 1994.

Contents

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399