COMPUTER ANIMATION AND VIRTUAL WORLDS
Comp. Anim. Virtual Worlds 2015; **26**:433–443
Published online 22 April 2015 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/cav.1647

SPECIAL ISSUE PAPER

Boundary-dominant flower blooming simulation

Jianfang Li[1], Min Liu[1], Weiwei Xu[2], Haiyi Liang[3]* and Ligang Liu[1]

[1] School of Mathematical Sciences, University of Science and Technology of China, China
[2] Department of Computer Science, Hangzhou Normal University, China
[3] CAS Key Laboratory of Mechanical Behavior and Design of Materials, University of Science and Technology of China, China

ABSTRACT

This paper presents a new physics-based simulation method for flower blossom, which is based on biological observations that flower opening is usually driven by a boundary-dominant morphological transition in a curved petal. We use an elastic triangular mesh representing a flower petal and adopt in-plane expansion to induce global bending. Out-of-plane curl plays an auxiliary role in reducing the curvatures of cross-sections. We also propose to adapt semi-implicit Euler time integrator for fast simulation results, which has intrinsic damping and at least one order precision. Our system allows users to control the blossoming process by simply specifying a growth curve, which is easy to design because of the boundary-dominant property. Experimental results show that our physics-based system runs faster and generates more realistic and convincing blossom results than the existing simulation methods. Copyright © 2015 John Wiley & Sons, Ltd.

KEYWORDS
flower blossom; boundary-dominant; physics-based simulation

Supporting information may be found in the online version of this article.

***Correspondence**
Haiyi Liang, CAS Key Laboratory of Mechanical Behavior and Design of Materials, University of Science and Technology of China,Hefei, Anhui 230027,China.
E-mail: hyliang@ustc.edu.cn

1. INTRODUCTION

Flower blooming earns much wonderfulness in plant developing: a flower bud expands, opens, warps backward, wrinkles, and finally grows into an adult flower [1]. Traditionally, studies on flower blooming rely on manual recordings, as shown in the time-lapse videos [2,3], or recovering dynamic geometry with 3D acquisition devices [4]. However, such workflows are tedious and time consuming.

Because of its attractiveness, simulation of flower blossom is important in computer animations. Generally, it is difficult to manually create convincing animations as it requires much expert knowledge of flowers for the users. Ijiri *et al.* [5] introduced a surface-based method for animating flower blossom, which was based on a previous observation that the flower blossom is caused by differences in the cell expansion rate between the inner and outer sides of petals [6]. However, users have to specify many growth parameters manually to obtain reasonable results. Moreover, the method cannot produce some realistic effects such as wrinkles, which makes the results less convincing.

A recent study on biological mechanism of flower blooming shows that the dynamic deployment of petals during blooming is a boundary-driven morphological transition in a curved lamina [7]. Specifically, the boundaries of petals grow faster than the inner parts of petals [7]. Inspired by this observation, we develop a new flower blooming simulation method. Our method follows the boundary-dominant blooming mechanism and assumes that the bending procedure of a petal is passively induced by an in-plane expansion with boundary-dominant scheme. It represents a petal as an elastic triangular mesh, and a growth map is prescribed to specify the different growing rates of the petal where its boundaries have larger growth rates than its inner parts. Thanks to the symmetric (or nearly symmetric) shape of a petal around its central axis, we simplify the growth map into a growth curve along the cross-section of the petal, which makes it easier to specify the growth parameter in the simulation.

We trigger the deformations of a petal using the mass-spring model in our simulation with stretching and bending energy. The joint angle between petal and receptacle increases automatically, producing natural and convincing flower opening animation. Out-of-plane curl growth assists to reduce the curvatures of cross-sections, and only a single parameter is required. Wrinkles on petal boundaries can be produced, making the flower blossom more realistic. We also engage connected springs, which constrain the vertices of flower bud, to simulate the locking force at the initial blooming stage.

For fast simulation, we turn to semi-implicit Euler integrators [8], and we introduce a parameter into the semi-implicit Euler time integration method so that the integration method has intrinsic damping and preserves at least one order precision at the same time. The introduced damping effect can efficiently stabilize the flower blooming simulation where the rest lengths and the rest angles change frequently.

2. RELATED WORK

Modeling. The L-system and its variants [9–11] are well-known tools for plants modeling, but they focused on branching structure, not dealing with growth of petal tissues of plants. Ijiri *et al.* [12] presented a system according to floral diagrams and inflorescences to create models of 3D flowers, while Qin *et al.* [13] proposed a flower modeling using L-system with Bezier surface. Reconstruction methods were also suggested. Yan *et al.* [14] provided a flower reconstruction method from a single photograph, while Zhang *et al.* [15] developed a method to obtain flower model from 3D point cloud of a single view of flower. Besides, a volume data reconstruction method was presented by Ijiri *et al.* [16] to create flower model from flower CT. Generally speaking, with these flower modeling or reconstruction methods, 3D flower models could be created, but these methods created only static flower models and were not suitable to create flower blooming models varying in time sequence. Although Li *et al.* [4] suggested a framework employing a forward–backward analysis method to analyze the 4D point cloud data from the recordation of plant growth, several 3D acquisition devices were needed to get a sequence of flower opening models, which were usually expensive.

Simulation. Some researchers provided methods for leaf developing simulation. Wang *et al.* [17] employed modified Navier–Stokes equations to simulate plant leaf growth. Xiao *et al.* [18] used a minimalist model and the differential contraction strain field on the leaf to simulate the curled dry leaf, while Jeong *et al.* [19] adopted a double-layer triangular surface to simulate leaf drying phenomenon. Beautiful leaf developing animation could be created with the simulation methods. But as flower opening is different from leaf developing, these methods are not appropriate to produce flower opening animation. Coen and Rolland-Lagan and their colleagues [6,20] expanded the local regions to simulate the development of petal surface, but their experiments were limited to flattened 2D shapes. Ijiri *et al.* [5] introduced a surface-based method for flower opening simulation. They specified the growth parameters of flower by five growth maps: direction map, elongation and width growth maps for inner and outer surfaces. They let inner surface grow faster than outer surface to bend the petal. However, these growth maps were not easily designed, and growth maps varying in time were necessary to obtain convincing results. In application, their method requires too many parameters for users to specify, and it was difficult to provide these growth maps for graphics application. Lu *et al.* [21] provided a deformation method for flower opening simulation. They represented a petal as a parameter equation and changed the corresponding growth parameters to deform the petal. Both methods were needed to increase the joint angle between petal and receptacle at every growth iteration to deploy the flower. Our method employs fewer growth parameters, and the growth parameters are easy to provide based on boundary-dominant growth. The growth scheme makes joint angle increase automatically, able to create natural flowering animations with wrinkles on petal boundaries. Lu *et al.* [22] suggested a visualization model for growth simulation of flower based on cylinder deformation. They began from a part of cylinder and employed different perturbation functions to deform the surface into a petal. This method could not create convincing flower opening animation.

Two semi-iterative tools of 3D modeling system were recommended to model folded surfaces [23,24]. Both of them deformed elastic surfaces through changing local regions and minimizing corresponding energy. However, these tools were designed for only a single surface and did not provide any mechanism to deploy a flower bud or bud-like structure.

3. FLOWER BLOOMING SIMULATION MODEL

Petal notations. A textured flower petal is shown in Figure 1; and we define some notations. *Tip* is the top part of the petal. *Petal base* is the bottom part of the petal, which is attached to flower receptacle. *Petal axis* refers to the center curve of the petal. *Center plane* passes through petal axis and the petal is symmetric to it. *Center part* refers to the part around petal axis. There has two *boundary* curves in a petal. *Cross-section* refers to a curve on the petal surface and is symmetric to the petal axis.

3.1. Physical Model

As a flower petal is much like a thin shell, we employ the well-known mass-spring model [20,25–27] as our physical simulation tool. Figure 2 shows a mass-spring system derived from a petal mesh, and the system consists of the following components:

- **Mass.** Every vertex of the petal mesh is thought as a mass point.
- **Linear spring.** Vertices are connected with their neighbors by linear springs.
- **Angular spring.** A virtual angular spring exists between normals of two adjacent triangles.

DOI: 10.1002/cav

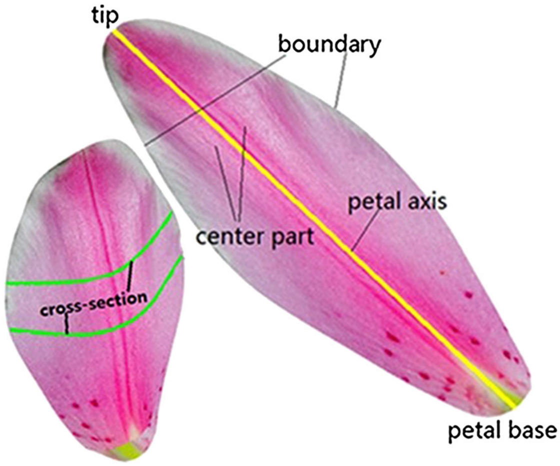


Figure 1. Notations of a petal. Center plane, to which the petal is symmetric, is not shown here.

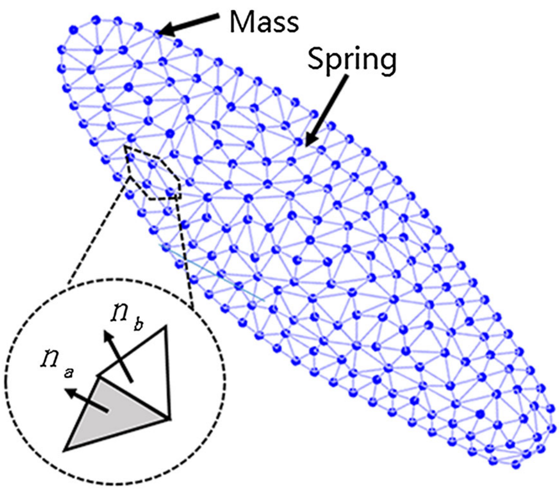


Figure 2. A mass-spring system derived from a petal mesh. n_a and n_b are the two unit normals of two adjacent triangles.

Denote $\mathbf{X} = \{\mathbf{x}_i, i = 1, 2, \cdots, w\}$ as the set of vertices of the mesh. Denote $\mathbf{E}$ as its edges and $\partial\mathbf{E}$ as the boundary edges. $\mathring{\mathbf{E}} = \mathbf{E} - \partial\mathbf{E}$ is denoted as the inner edges. Following [28], we define the whole energy $U(\mathbf{X})$ of the system as

$$U(\mathbf{X}) = U_s(\mathbf{X}) + U_b(\mathbf{X}) \quad (1)$$

$$U_s = \sum_{[\mathbf{x}_i, \mathbf{x}_j] \in \mathbf{E}} \frac{1}{2} K_s \left(||\mathbf{x}_i - \mathbf{x}_j|| - L_{ij}^0 \right)^2 \quad (2)$$

$$U_b = \sum_{[\mathbf{x}_i, \mathbf{x}_j] \in \mathring{\mathbf{E}}} K_b \left(1 - cos \left(\theta_{ij} - \theta_{ij}^0 \right) \right) \quad (3)$$

where L_{ij}^0 is the rest edge length of $[\mathbf{x}_i, \mathbf{x}_j]$, θ_{ij} is the dihedral angle between two adjacent triangles sharing edge $[\mathbf{x}_i, \mathbf{x}_j]$, θ_{ij}^0 is the rest dihedral angle, and K_s and K_b are the stretching stiffness and bending stiffness, respectively. The internal force vector $\mathbf{f}(\mathbf{X}) \in R^{3w}$ takes the following form:

$$\mathbf{f}(\mathbf{X}) = -\nabla U(\mathbf{X}) \quad (4)$$

By Newton's law we have

$$\ddot{\mathbf{X}} = M^{-1}\mathbf{f}(\mathbf{X}(t)) \quad (5)$$

where $M \in R^{3w \times 3w}$ is the mass matrix, and the dot refers to derivative with respect to time t.

For the mass-spring system used in our framework, the growth of the rest length of a linear spring is called the *in-plane expansion*, and the growth of the rest angle of an angular spring is called the *out-of-plane curl*.

3.2. Global Bending Induced by In-plane Expansion

Boundary-dominant scheme. Recently, Liang *et al.* [7] highlighted the role of boundaries in flower opening and showed that the dynamic deployment of petals during blooming was a boundary-driven morphological transition in a curved lamina. This inspires us to make a flower petal bend passively using in-plane expansion growth with a boundary-dominant scheme, specifically, let the boundaries of a petal grow at a high rate and center part at a low rate. The internal force accumulates asynchronously and induces the global bending. This is not an intuitive scheme; here, we turn to Figure 3 to explain this mechanism.

In-plane expansion. We formulate the in-expansion as follows:

$$L_{ij}^0(t) = [1 + A(t)\alpha(\mathbf{x}_i, \mathbf{x}_j)]L_{ij}^0(0), [\mathbf{x}_i, \mathbf{x}_j] \in \mathbf{E} \quad (6)$$

where $L_{ij}^0(t)$ is the rest length of spring $[\mathbf{x}_i, \mathbf{x}_j]$ at time t, $A(t) \in [0, 1]$ is an increasing function, which means gradual growth, and $\alpha(\mathbf{x}_i, \mathbf{x}_j)$ is the growth ratio of spring $[\mathbf{x}_i, \mathbf{x}_j]$. The increasing function $A(t)$ depicts the whole growth rate of a petal. A linear function $A(t)$ means the petal uniformly grows, while usually an S-like curve reflects the real growth of a plant [29].

3.3. Parameter Setting

Our system describes the maximal growth of a spring by growth ratio. Because a petal mesh has a 2D surface, the growth ratios can be given by a growth map [5], which is usually a color image. In the growth map, the RGB values specify the maximal growth ratios. In the simulation, when a growth map is provided, a parametrization that maps the petal mesh to growth map must be calculated to load growth parameters for each spring. However, the petal is not developable and has an ellipse-like structure that has two boundary curves, while the growth map has four boundary edges, these result to difficulty in calculating an

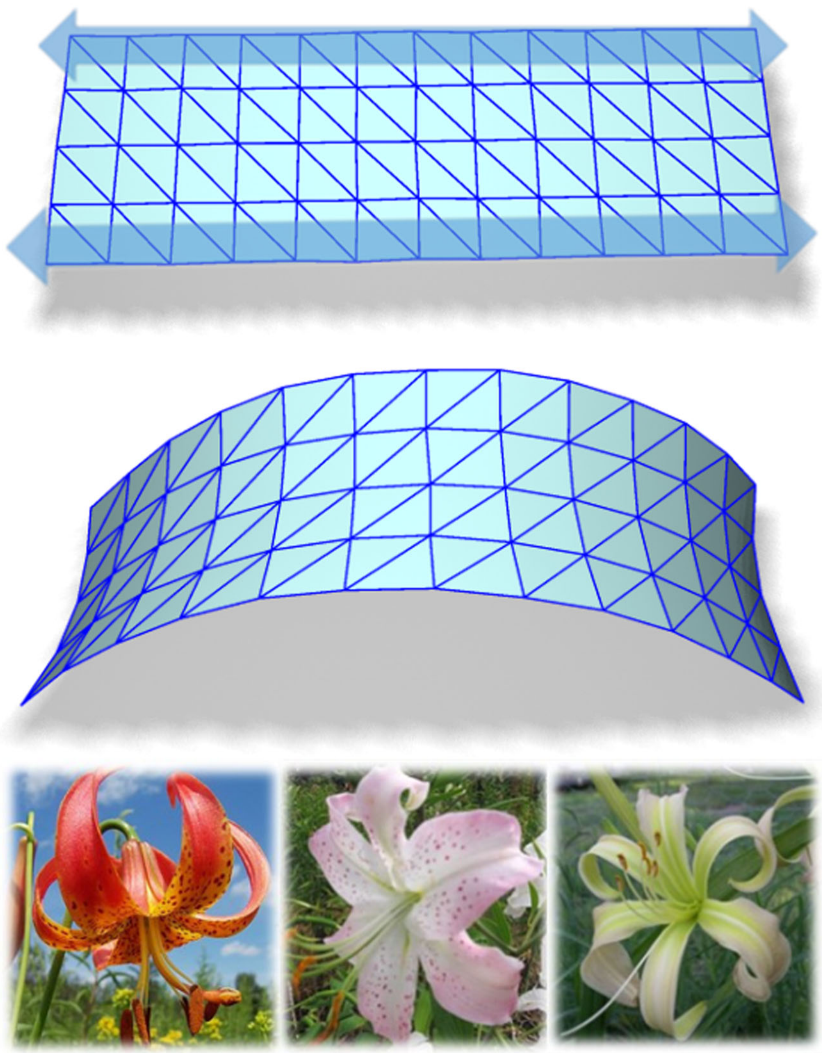


Figure 3. Sole in-plane expansion with boundary-dominant scheme induces the rectangle surface (top) to bend passively, and the rectangle surface deforms into a saddle-like surface (middle), which is the structure of adult petal (bottom).

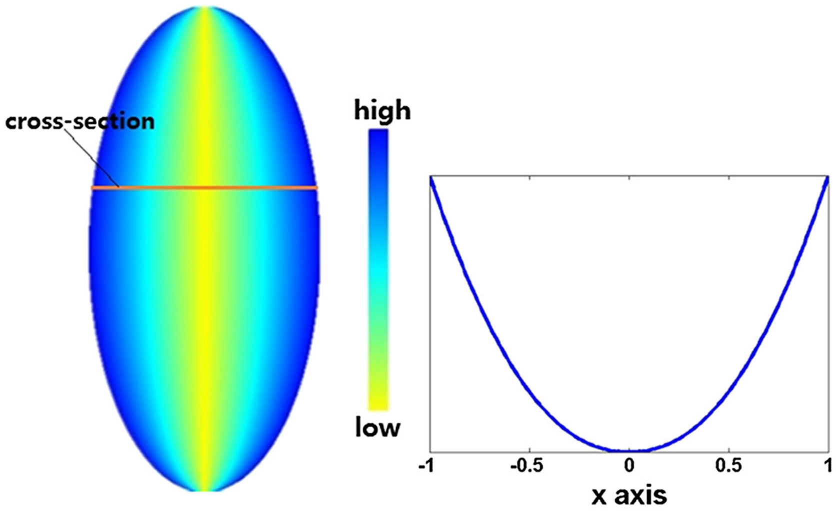
appropriate parametrization. Thanks to the symmetry (or near symmetry) of flower petal and the boundary-dominant scheme, a growth map can be reduced to a growth curve under some reasonable assumptions. We assume that the boundaries of a petal have equivalent growth ratios and that the petal axis has equivalent growth ratios. We also assume growth ratios of boundaries gradually decrease to the growth ratios of petal axis along cross-sections. With these reasonable assumptions, the growth ratios along each cross-section share the same values if they are normalized; thus, the growth ratios of a petal can be represented by a growth curve (Figure 4). Using the growth curve, we can automatically assign the parameters from a cross-section to another one without the necessity to calculate the mesh parametrization.

Buckling. Although boundary-dominant growth scheme can bend a flower petal naturally, appropriately spatially varying parameters are preferred. If the boundaries of a petal grow too fast, the petal will stop bending at an undesired state. This is owning to physical instability: buckling [28,30]. The petal buckles its boundaries severely and loses the boundary-driven force to bend. This phenomenon is shown in Figure 5.

4. TIME INTEGRATION

Equation (5) is usually solved by numerical integrators in dynamics simulation. Implicit time integrators are often the choice to achieve large time step [31]. In the implicit integrator, a nonlinear equation needs to be solved every iteration, involving the calculation of Hessian matrix of the energy function. Due to the high nonlinearity of angular spring, the calculation of Hessian is cumbersome and error-prone [32]. In cloth simulation, Baraff *et al.* [31] assumed that the normal and edge vectors were of constant

Comp. Anim. Virtual Worlds 2015; **26**:433–443 © 2015 John Wiley & Sons, Ltd.
DOI: 10.1002/cav



(a) (b)

Figure 4. Under some reasonable assumptions, the growth ratios of a petal, which is represented by an ellipse (a) can be reduced to a growth curve (b) where each cross-section is normalized into [-1,1].

(a) $0.2x^2$ (b) $0.2x^4$ (c) $0.2x^6$

Figure 5. The petal bends naturally with growth curve $0.2x^2$ (a). A little buckling is found in (b), while growth curve $0.2x^6$ results in severe buckling and undesired opening (c).

magnitudes, making the calculation easy. However, their assumptions are violated in our case because the vectors are not constant anymore due to the growth control (see Equation (6)).

Semi-implicit time integration. Alternatively, we turn to semi-implicit time integrator [33] for fast simulation, and we propose the following method:

$$\mathbf{X}_{n+1} = \mathbf{X}_n + h\mathbf{V}_n + \frac{h^2}{2}M^{-1}\mathbf{f}(\mathbf{X}_n) \tag{7}$$

$$\mathbf{V}_{n+1} = \mathbf{V}_n + hM^{-1}[\eta\mathbf{f}(\mathbf{X}_n) + (1-\eta)\mathbf{f}(\mathbf{X}_{n+1})] \tag{8}$$

where $\mathbf{X}_n$ is the position vector at nth iteration, $\mathbf{V}_n$ is the velocity vector, and M is the mass matrix, h is the time step, and $\eta \in \left[0, \frac{1}{2}\right]$ is a parameter which we call damping factor. For $\eta = \frac{1}{2}$, the semi-implicit scheme is equivalent to the well-known Verlet method [34]. However, We found that the Verlet method caused the vibration of some regions of the petal mesh in the simulation. This might be due to the momenta preservation property of the Verlet method, while the momenta in flower simulation is hard to control because its deformation energy changes frequently due to growth control. We thus tried to add explicit damping into the velocity update, similar to [35], to stabilize the simulation:

$$\mathbf{V}_{n+1} = d\mathbf{V}_n + \frac{h}{2}M^{-1}[\mathbf{f}(\mathbf{X}_n) + \mathbf{f}(\mathbf{X}_{n+1})] \tag{9}$$

where d represents the damping typically close to 1. In our experiments, we found it difficult to control the damping via this method. A large d makes things little better, while a small one dampens much and makes the whole petal buckles. We believe this is due to the explicit damping, which dampens a specific term, such as the velocity term in Equation (9). After many iterations, the velocities of vertices do not coordinate with each other and induce the buckling. Here, we propose a semi-implicit method with intrinsic damping when $\eta < \frac{1}{2}$ in Equation (8). From Equation (7) and Equation (8) we know the semi-implicit integrator has at least one order precision. Using the method in our simulation, we usually adjust the damping factor $\eta \in \left[0, \frac{1}{2}\right]$ to control the damping.

5. BLOOMING EFFECTS CONTROL

Center plane constraint. Usually a flower petal is symmetric to a plane, called center plane, which passes the petal axis. But due to the modeling ability of designers or remeshing, the petal mesh has nonsymmetric topology about the petal axis and this causes the nonsymmetric growth that may result in the adult petal with a little skew (Figure 6). In the simulation, to deal with this, we constrain the distance from vertices of springs, which intersect the petal axis to center plane unchanged. This can be imposed via manipulation of velocity update, keeping the corresponding velocities parallel to center plane.

Petal base. The petal base of natural flower is attached to receptacle. In the present method, the vertices belonging to petal base $\mathbf{X}_{\text{base}}$ are fixed in the simulation. Mathematically, this constraint reads

$$-\nabla_{\mathbf{x}} U(\mathbf{X}) = \mathbf{0}, \mathbf{x} \in \mathbf{X}_{\text{base}} \tag{10}$$

In the simulation, we set the corresponding forces zero every iteration. As another benefit of the boundary-dominant scheme the joint angles between petals and receptacle increase automatically and naturally.

Connection spring. Occasionally, there are tissue connections that prevent the flower bud from opening, such as lily's lock mechanism that the boundaries of outer petals are locked in midribs of inner petals. We mimic the tissue connections by springs, which connect the vertices of two petals. As the flower starts to open, the connection springs are extended longer. While connected spring is stretched to a critical point, the spring will be broken and the flower starts to open.

Cross-section. Ubiquitously, as the flower blooms, the curvatures of cross-sections get small and adult petals are flatter than bud petals. In our framework, out-of-plane curl growth is auxiliary to reduce the curvatures of cross-sections:

$$\theta_{ij}^0(t) = [1 - B(t)\beta]\theta_{ij}^0(0), [\mathbf{x}_i, \mathbf{x}_j] \in \mathring{\mathbf{E}} \tag{11}$$

where $\theta_{ij}^0(t)$ is the rest dihedral angle corresponding to inner spring $[\mathbf{x}_i, \mathbf{x}_j]$ at time t, $\beta \in [0, 1]$ is the growth parameter for out-of-plane curl, and $B(t) \in [0, 1]$ is an increasing function, which also means gradual growth. In our simulation, we usually take $B(t)$ and $A(t)$ (see Equation (6)) as the same functions. Figure 7 presents different effects of parameter β. Usually, as the parameter gets larger, the curvatures of cross-sections get lower.

(a) (b) (c)

Figure 6. A resulting petal is viewed from different sides. The tip is a little skew.

(a) $\beta = 0.1$ (b) $\beta = 0.5$ (c) $\beta = 0.9$

Figure 7. Usually a large parameter β results in small curvatures of cross-sections.

Comp. Anim. Virtual Worlds 2015; **26**:433–443 © 2015 John Wiley & Sons, Ltd.
DOI: 10.1002/cav

Wrinkles. Some species of flowers have wrinkles on boundaries such as asiatic lily whose boundaries wrinkle as flower opens. The wrinkles result from unequal growth between petal boundaries and center part to some specific degree [36,37]. The petal buckles its boundaries and generates wrinkles [7]. In the simulation, we employ an additional growth ratio for petal boundaries to generate wrinkles. Then the growth of petal boundaries has form

$$L_{ij}^{0}(t) = [1 + A(t)(\alpha(\mathbf{x}_i, \mathbf{x}_j) + \alpha_{\text{add}})]L_{ij}^{0}(0) \tag{12}$$

where α_{add} refers to the additional growth ratio and $[\mathbf{x}_i, \mathbf{x}_j] \in \partial\mathbf{E}$. Because thin plate or shell with low stiffness is easier to buckle [38], we scale the stiffness of boundary springs with a positive factor less than 1 to generate wrinkles (Figure 8).

Collision detection. A flower usually has several petals, and during blooming, collisions between petals happen frequently. To efficiently capture the collisions, we set a bounding box in which the flower blooms and discretize the bounding box into small cuboids. Only the vertices and triangles in a same cuboid or adjacent cuboids can collide. In the simulation, we take the method used in [5] to handle collision, which is based on the highly specialized arrangements of flower petals. If a vertex of petal penetrates an adjacent petal we push it back with a small offset (the current offset is 0.01). To avoid the vertex and triangle to collide, next iteration, we cull off the part parallel to the normal of triangle from the vertex's velocity.

6. EXPERIMENTAL RESULTS

We have implemented our simulation system in a dual-CPU 2.4 GHz computer with 4 GB memory. In our system, the mesh of a flower is uniformly scaled with an average edge length of 1. We have tested several kinds of flowers with different features and different number of petals to illustrate the performance of our system, as shown in the paper (also in the accompanying video).

(a) (b) (c)

Figure 8. The petal bends outward and wrinkles its boundaries naturally (a). A different viewpoint of the adult petal is shown (b). Real lily petals with wrinkles (c).

Figure 9. Four frames of a blooming magnolia (top) and its corresponding meshes (bottom).

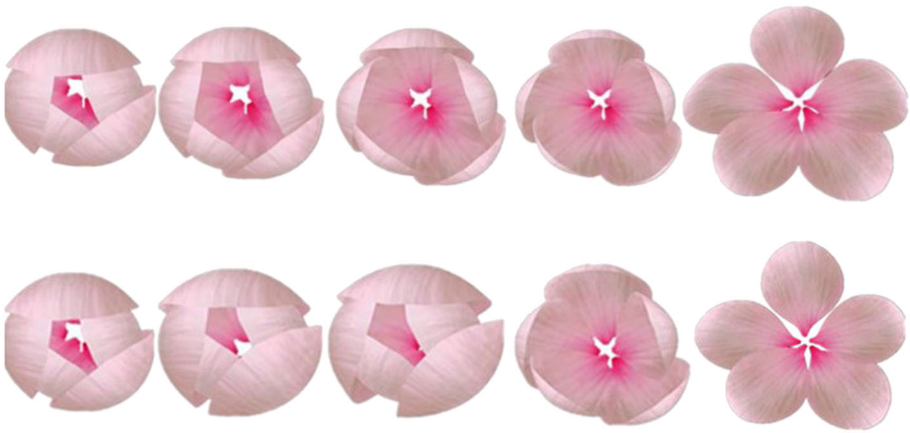


Figure 10. Two animations of plum blossom. The first one does not own connected springs in the simulation (top), but the second one does. In the second one, the flower opens slowly at the beginning and then undergoes a quick opening time (bottom).

Figure 11. While the flower opens, petals curve outward and curvatures of cross-sections decrease.

Figure 12. When applied with the same asiatic lily model, our method produces more convincing animations (b). Results produced by **[5]** depend on too many growth parameters, and resulting petals are featureless (a). Our results have beautifully curved petals, and wrinkles on boundaries can be generated (b). Besides, it takes fewer time with our method.

Table I. Example, stretching stiffness, bending stiffness, growth curve, vertex count, face count, edge count, and total simulation time. In all our experiments, $h = 1.0$ is used and we consider every vertex has mass of 1.

Example	K_s	K_b	Growth curve	#V	#F	#E	Total time (seconds)
Figure 9	0.1	0.0005	$0.1x^2$	1805	3077	4876	35
Figure 10	0.1	0.0009	$0.2x^2$	2433	4427	6855	57
Figure 11	0.1	0.0005	$0.15x^2$	1800	3080	4874	35
Figure 12 (b)	0.1	0.0005	$0.2x^2$	2133	3786	5913	44

DOI: 10.1002/cav

6.1. Blooming of Various Flowers

Magnolia. There are many kinds of magnolias. The magnolia shown in Figure 9 has six petals and does not have wrinkles on petal boundaries. We can see that it opens naturally with only in-plane expansion.
Plum blossom. Plum blossom has five similar petals and no wrinkles are observed on petal boundaries. Here, we create two animations about it. Besides the same in-pane expansion and out-of-plane curl growth parameters, we also apply several connected springs to the second one. Both of them show striking attractiveness (see Figure 10), while the second one is more vivid. The flower bud opens slowly at the beginning. When the opening force is strong enough, the connected springs are broken and the flower undergoes an abrupt opening.
Lily. Figure 11 shows a blooming lily. At the initial stage, the cross-sections of petals have large curvatures. As the flower opens, the petals bend outward and the curvatures of cross-sections decrease. Finally, the flower stops growing at a pose with naturally curved petals.

6.2. Evaluations

Comparison. Another lily model is employed in comparison with [5]. This model contains 2133 vertices, 3786 faces, and 5913 edges, and results are shown in Figure 12. It takes about 11.5 minutes to complete the simulation with the method of Ijiri *et al.* [5], while it only needs 44 seconds with our boundary-dominant method. Besides efficiency of time, our method requires fewer parameters, which are easy to provide based on the boundary-dominant scheme. Our resulting flowers have naturally curved petals and wrinkles can be produced which beautify much the flower.
Timing. In our experiments, we first run our system with only one petal. After obtaining convincing results, we simulate the whole flower blooming. It usually takes within 10 seconds to complete the simulation with only one petal, and this is fast enough for experiments with various parameters. The results in Figures 5, 6, 7, and 8 involve the same petal with 323 vertices, 567 faces, and 889 edges, and each simulation of them takes 4 seconds. Table I lists the information and the simulation time of the results in Figures 9, 10, 11, and 12.

7. CONCLUSION AND FUTURE WORK

We have presented a biology-based method for flower blooming simulation, through which natural flower opening results can be obtained with only a few growth parameters. Our method takes a boundary-dominant growth scheme to make flower bend passively and uses out-of-plane curl to get various results. Some recipes are proposed to produce more convincing effects. We also suggest to use the semi-implicit time integration to adjust the damping, which can efficiently stabilize the simulation and create pleasing results. In our experiences, the simulation is fast and growth parameters are easy to provide according to the specific growth scheme. The results demonstrate the power of our method, and we believe this system is helpful for flower modeling and blooming animation.
Limitation and future work. Because of the requirements of the boundary-dominant scheme, one limitation of the method is that it is not suitable for flowers having fused petals, such as trumpet flower. But this is one of our ongoing researches, and in the future we want to tackle it.

ACKNOWLEDGEMENTS

We would like to thank the anonymous reviewers for their constructive and valuable comments. Thanks also goes to Ijiri *et al.* [5] for providing some models for this paper. The work is supported by the NSF of China (Nos. 61222206, 61272392, 61322204, 11272303, 11132009) and One Hundred Talent Project of the Chinese Academy of Sciences.

REFERENCES

1. Harampolis A, Rizzo J. *The Flower Recipe Book*. Artisan Division of Workman Publishing: New York, USA, 2013.
2. Pichelhofer S. Elfleurcam elphel 333 flower lily blossom timelapse 2010. https://www.youtube.com/watch?v=KDUZq1tK-Z4).
3. Footage Ultra HD 4K 8K Time Lapse Stock Video. Flowering trees - uhd ultra hd 2k 4k 5k time lapse stock footage royalty-free blue screen 2012. http://www.youtube.com/watch?v=_YTSLCvta4Y.
4. Li Y, Fan X, Mitra NJ, Chamovitz D, Cohen-Or D, Chen B. Analyzing growing plants from 4d point cloud data. *ACM Transactions on Graphics* 2013; **32**(6): 157:1–157:10.
5. Ijiri T, Yokoo M, Kawabata S, Igarashi T. Surface-based growth simulation for opening flowers. In *Proceedings of Graphics Interface 2008*. Canadian Information Processing Society: Toronto, Canada, 2008; 227–234.
6. Coen E, Rolland-Lagan A-G, Matthews M, Bangham JA, Prusinkiewicz P. The genetics of geometry. *Proceedings of the National Academy of Sciences of the United States of America* 2004; **101**(14): 4728–4735.
7. Liang H, Mahadevan L. Growth, geometry, and mechanics of a blooming lily. *Proceedings of the National Academy of Sciences* 2011; **108**(14): 5516–5521.
8. Kincaid DR, Cheney EW. *Numerical Analysis: Mathematics of Scientific Computing*, Vol. 2. American Mathematical Society: Providence, USA, 2002.

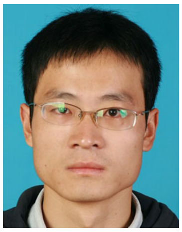
9. Lindenmayer A. Mathematical models for cellular interactions in development I. filaments with one-sided inputs. *Journal of theoretical biology* 1968; **18**(3): 280–299.
10. Prusinkiewicz P, Hammel M, Hanan J, Mech R. L-systems: from the theory to visual models of plants. In *Proceedings of the 2nd CSIRO Symposium on Computational Challenges in Life Sciences*, 1996.
11. Prusinkiewicz P, Hammel MS, Mjolsness E. Animation of plant development. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*. ACM: New York, USA, 1993; 351–360.
12. Ijiri T, Owada S, Okabe M, Igarashi T. Floral diagrams and inflorescences: interactive flower modeling using botanical structural constraints. In *ACM Transactions on Graphics (TOG)*, Vol. 24. ACM: New York, USA, 2005; 720–726.
13. Qin P, Chen C. Integration of l-systems with bezier surfaces. *IJCSNS* 2006; **6**(2): 65–68.
14. Yan F, Gong M, Cohen-Or D, Deussen O, Chen B. Flower reconstruction from a single photo. In *Computer Graphics Forum*, Vol. 33. Wiley Online Library: Hoboken, USA, 2014; 439–447.
15. Zhang C, Ye M, Fu B, Yang R. Data-driven flower petal modeling with botany priors. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE: Piscataway, New Jersey, USA, 2014; 636–643.
16. Ijiri T, Yoshizawa S, Yokota H, Igarashi T. Flower modeling via x-ray computed tomography. *ACM Transactions on Graphics* 2014; **33**(4): 48:1–48:10.
17. Wang IR, Wan JWL, Baranoski GVG. Physically-based simulation of plant leaf growth. *Computer Animation and Virtual Worlds* 2004; **15**(3-4): 237–244.
18. Xiao H, Chen X. Modeling and simulation of curled dry leaves. *Soft Matter* 2011; **7**(22): 10794–10802.
19. Jeong S, Park S-H, Kim C-H. Simulation of morphology changes in drying leaves. In *Computer Graphics Forum*, Vol. 32. Wiley Online Library: Hoboken, USA, 2013; 204–215.
20. Rolland-Lagan A-G, Bangham JA, Coen E. Growth dynamics underlying petal shape and asymmetry. *Nature* 2003; **422**(6928): 161–163.
21. Lu L, Wang L, Yang XD. A flower growth simulation based on deformation. In *Information Science and Engineering, 2008. ISISE'08. International Symposium on*, Vol. 2. IEEE: Piscataway, New Jersey, USA, 2008; 216–218.
22. Lu L, Song W, Wang L. A visualization model for simulating the growth of flower and fruit. In *World Automation Congress (WAC), 2010*. IEEE: Piscataway, New Jersey, USA, 2010; 223–227.
23. Combaz J, Neyret F. Painting folds using expansion textures. In *Computer Graphics and Applications, 2002. Proceedings. 10th Pacific Conference on*. IEEE: Piscataway, New Jersey, USA, 2002; 176–183.
24. Combaz J, Neyret F. Semi-interactive morphogenesis. In *Shape Modeling and Applications, 2006. SMI 2006. IEEE International Conference on*. IEEE: Piscataway, New Jersey, United States, 2006; 35–35.
25. Prusinkiewicz P, Lindenmayer A. *The Algorithmic Beauty of Plants*. Springer-Verlag: New York, 1990. With Hanan JS, Fracchia FD, Fowler DR, de Boer MJM and Mercer L.
26. Zhang D, Yuen MMF. Cloth simulation using multilevel meshes. *Computers & Graphics* 2001; **25**(3): 383–389.
27. Nealen A, Müller M, Keiser R, Boxerman E, Carlson M. Physically based deformable models in computer graphics. In *Computer graphics forum*, Vol. 25. Wiley Online Library: Hoboken, USA, 2006; 809–836.
28. Seung HS, Nelson DR. Defects in flexible membranes with crystalline order. *Physical Review A* 1988; **38**(2): 1005–1018.
29. Karadavut U, Kayiş SA, Palta Ç, Okur O. A growth curve application to compare plant heights and dry weights of some wheat varieties. *American-Eurasian Journal of Agricultural & Environmental Sciences* 2008; **3**: 888–892.
30. Choi K-J, Ko H-S. Stable but responsive cloth. In *ACM SIGGRAPH 2005 Courses,* SIGGRAPH '05: New York, NY, USA, 2005. ACM.
31. Baraff D, Witkin A. Large steps in cloth simulation. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '98. ACM: New York, NY, USA, 1998; 43–54.
32. Grinspun E, Hirani AN, Desbrun M, Schröder P. Discrete shells. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Eurographics Association: Aire-la-Ville, Switzerland, 2003; 62–67.

33. Kincaid DR, Cheney EW. *Numerical Analysis: Mathematics of Scientific Computing*, Vol. 2. American Mathematical Society: Providence, USA, 2002.
34. Huang W, Leimkuhler B. The adaptive verlet method. *SIAM Journal on Scientific Computing* 1997; **18**(1): 239–256.
35. Liu T, Bargteil AW, O'Brien JF, Kavan L. Fast simulation of mass-spring systems. *ACM Transactions on Graphics (TOG)* 2013; **32**(6): 214:1–214:7.
36. Sharon E, Marder M, Swinney HL. Leaves, flowers and garbage bags: making waves. *American Scientist* 2004; **92**(3): 254–261.
37. Prusinkiewicz P, de Reuille PB. Constraints of space in plant development. *Journal of experimental botany* 2010; **61**(8): 2117–2129.
38. Carraro C, Nelson DR. Grain-boundary buckling and spin-glass models of disorder in membranes. *Physical Review E* 1993; **48**(4): 3082–3090.

SUPPORTING INFORMATION

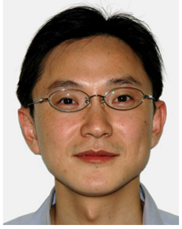
Supporting information may be found in the online version of this article.

AUTHORS' BIOGRAPHIES



Jianfang Li PhD candidate. honour-respect@163.com. His main research interest is computer graphics.

Min Liu PhD candidate. leasent @mail.ustc.edu.cn. His main research interest is computer graphics.



Weiwei Xu PhD, Professor. weiwei.xu.g@gmail.com. His main research interests include computer graphics and physically-based simulation, and so on.

Haiyi Liang PhD, Professor. hyliang @mail.ustc.edu.cn.

Ligang Liu PhD, Professor. lgliu @ustc.edu.cn. His main research interests include computer graphics, geometry processing, image processing, and so on.