

# Physical Model for Interactive Deformation of 3D Plant



Shenglian Lu, Xinyu Guo, Chunjiang Zhao and Changfeng Li

National Engineering Research Center for Information Technology in Agriculture, Beijing 100097, China

**Abstract**—Modeling the deformation of 3D plant is a challenge in computer graphics. This paper presents a simulation method for physically simulating interactive deformation of 3D plant models. This method creates a tetrahedral mesh from the initial triangular plant model, the tetrahedral mesh is then used for dynamic response calculation of collision or interaction, the original triangular mesh is deformed along with the tetrahedral mesh. A capsule-based method and a spatial hashing based method are used for efficient and accurate collision detection. Smooth deformation effects and real-time simulation on 3D plant models demonstrate the effectiveness of our method. The main contribution of this paper is the proposed method can handle the geometric complexity of various plants by a simple model.

**Index Terms**—physical-based deformation simulation, deformable objects, tetrahedral mesh

## I. INTRODUCTION

There is a wide range of virtual reality (VR) applications that can benefit from 3D plant models, such as plant growth simulation, landscape design, entertainment, training and teaching. Up to day lots of methods and software packages have been proposed for creating 3D plant models (e.g., L-system, AMAP and Xfrog). Highly realistic static 3D plant models can be obtained by using these tools. The geometric accuracy or visual reality is their focus in these tools. However, less works has been focused on the deformable physical character of plant. In fact, the body of a real plant is deformable. It would be bended or deformed under exterior force. One of challenges for simulating the deformable characteristics of 3D plant models exists in the geometric complexity of plant. Another challenge is the real time simulation requirement. Although the raising computational power of the last decades allowed for adapting selected methods known from engineering sciences for interactive simulation, and a number of approaches for simulating deformable objects have been presented over the last decade, the problem of simulating the deformable characteristics of 3D plant models has not been solved satisfyingly, yet.

Basing on the fact that less work has focused on real-time deformation simulation of plant models in virtual environment (VE), this paper presents a tetrahedral mesh-based method for interactive simulating deformation of 3D plant models, and to

improve user's experience in VE. The proposed method uses capsule-based and spatial hashing based detection for efficient collision detection and response calculation. The capsule-based method subdivides a triangular plant model into capsules which are linked as mass-spring system, and is used for coarse collision detection. The spatial hashing based method is used for more accurate collision computation. We show the effectiveness and flexibility of the approach by several examples.

## II. PREVIOUS WORK

There has been a great deal of previous work on simulating the motions of plant, including plant growth, motion in the wind and so on, such as the work demonstrated in [1]. Some physically-based models had been used to create natural shape of plant [2, 3]. Recently some people had simulated physically the natural deformed shape of plant leaves [4, 5]. However, none of them address the problem of simulating deformation of plant model in interactive VE. This problem of modeling deformable objects has been studied for many years, and currently it is still one of the most hot research issues in computer graphic. In the following we summarize the most closely related work. For an extended discussion on deformation models, we will refer the readers to recent summaries on the simulation of deformation [6] and collision detection [7].

Mesh-based methods are the early way for performing shape deformations by direct mesh optimization, such as [8, 9]. The key of these methods is to explicitly preserve the local shape properties by applying user-specified deformations. Similarly, skeleton-based methods restrict the space of natural deformations by inferring the deformations on the skeleton structure of the shape [10] or by exploring the set of example poses [11], in which inverse-kinematics are generally used. Although mesh-based methods provide a high degree of freedom in manipulating the shape, they commonly require a tedious process of weight or parameter selection for constraining and estimating per-vertex deformations. Many methods have been designed to improve efficiency, such as multiresolution methods [12]. Some researchers have proposed techniques that use deformable models with a significantly reduced dimensionality as compared to the full geometric complexity (e.g., [13, 14]).

Meshless shape deformation is another method that was recently introduced in computer graphics for physically based

animations (e.g., [15, 16]). The classical meshless deformation is meshless finite elements. Chang et al. [17] presents a meshless method for animating elastic solids, using linear superposition of approximate analytical. Adams et al. [18] propose a novel meshless algorithm that allows proper deformation modeling with the flexibility of traditional meshless algorithms such as easy sampling and a smooth and consistent deformation field representation.

While volumetric shape representations such as tetrahedral meshes or similar structures have been widely used in physical simulation [13, 14]. The benefit of a volumetric deformation framework is that it better prevents unintuitive shape transformations, such as local self-intersections of opposing surfaces. Furthermore, it enables distance preservation not only on an object's surface, but also throughout its interior, which makes the deformations resistant to changes in volume and cross-sectional areas.

Our method proposed in this paper use a representation of tetrahedral mesh. Apart from the different application, this paper presents an efficient spatial hashing-based mesh deformation algorithm. Based on the spatial hashing approach, we implemented the collision detection of triangular plant models. This algorithm does not require complex data structures, and can be run in real time in interactive applications with realistic effects.

### III. PROPOSED METHOD

This section will detail the techniques of our proposed method. The input of our algorithm is a (usually high resolution) triangular mesh to be deformed, which consists of a set of vertices and triangles. From this we generate a tetrahedral mesh.

The first step of our pipeline after generating the tetrahedral mesh is classifying tetrahedrons in the tetrahedral mesh and vertices in the initial input triangular mesh according to their AABB (Axis-Aligned Bounding Box). Then these AABB are indexed by using a spatial hashing approach. The spatial hashing structure is used for collision detection in interactive deformation. Our approach of deformation simulation augments the tetrahedral mesh plausibly approximate to nonlinear deformation behavior by solving only linear equation systems.

#### 3.1 Generating Tetrahedral Mesh

Our method relies on a tetrahedral mesh representation of the input shape that can be easily generated for triangle-mesh objects. Many methods have been proposed for generating and optimizing tetrahedral mesh, the one we use in this paper is shown in [19]. The method proposed in [19] can handle unorientable, non-manifold or damaged surfaces, and is therefore particularly suitable for the generation of tetrahedral mesh from triangular plant models. It is difficult to generate closed volume from this kind of surface, so this method introduces a concept "pseudo volume". The pseudo volume of a surface means namely the space that is intuitively occupied by

the surface. This method of generating tetrahedral mesh needs some users' interactions for adjusting the outside and smooth degree of the generated tetrahedral mesh. This is useful for users to obtain better simulated deformation results. Fig. 1 shows an initial plant model and the generated tetrahedral mesh.



Fig.1. Example of a 3D plant model and its tetrahedral mesh

#### 3.2 Physical Equations for Deformation

To compute the collision responses, we look the preprocessed volumetric mesh as an elastic object consists of tetrahedral elements, while these tetrahedral elements are connected by mass and spring. We can use physical equations for simulating mass-spring system to handle the collision responses.

At the early stage of each step during the simulation, we calculate the internal force and exterior force on each vertex. The internal force is the elastic force comes from the changing of position of a vertex, while the exterior force generally consists of gravity, action force caused by collision, wind force and user's interactive force. Using the linearized elastic forces, the linear algebraic formula of motion for an entire mesh becomes:

$$M\ddot{u} + D\dot{u} + Ku = f \quad (1)$$

Where  $M$  is the mass matrix,  $D$  is the global damping matrix,  $K$  is the stiffness matrix; vector  $u$  is the displacement of the nodes in the mesh, as such  $u = x - x_0$ , where vectors  $x$  and  $x_0$  are the actual and the rest positions of the nodes. While  $f$  is the external force acting on the elastic object.

To get stable computation at large time steps by using implicit method to formula (1), we can get the following formulas [20]:

$$\Delta u = h(\dot{u} + \Delta v) \quad (2)$$

$$(M - h\mu I + h^2 K) = h(\mu \dot{u} - Ku - f - hK\dot{u}) \quad (3)$$

Where  $h$  is the time step,  $\Delta u$  is the variation of  $u$  in a time step,  $\Delta v$  is the variation of  $\dot{u}$  in a time step,  $\mu$  is the damping coefficient,  $I$  is an unit matrix.

Formula (3) is a symmetrical positive definite large scale sparse linear equations, which can be solved by using preconditioned conjugate gradient method with acceptable

calculation speed. Then  $\Delta v$  can be calculated from formula (2) based on the results of formula (3).

### 3.3 Collision Detection

Collision detection and response is usually the most time consuming process for the overall simulation. We use two phase policies for efficient and accurate collision detection. The first phase is handled by using a capsule-based method for coarse collision detection, and a spatial hashing is used for detecting collision occurring in between two tetrahedral objects or between a tetrahedral object and a rigid object.

#### 1) Capsule-based collision detection

Capsule-based method has been extensively used for collision detection in CG. A capsule can be defined with two parameters: the radius  $r$  and the height  $h$ , as Fig. 2 shows.

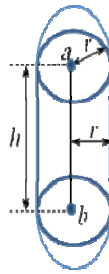


Fig. 2. Parameters description of a capsule.

In our method, capsule is used to represent the plant model. Firstly, we use the user-defined parameters  $r$  and  $h$  to subdivide a plant model into a series of capsules, these capsules are then joined by using mass-spring. While others objects in the virtual environment are represented as triangular mesh. So the collision between two plant models or a plant model and another object can be look as collision in between two capsules or a capsule and a triangle.

The detection of collision between a capsule and a triangle can be done easily. As Fig. 3 shows, this can be turn to the problem calculating the minimum distance of the triangle to line segment  $ab$ , if the minimum distance is small than  $r$ , collision exists. This method can be easily applied to dynamic situation. When a capsule moving with velocity  $v$ , the question for detecting collision of the capsule with a triangle is to check when the distance of the triangle to the line segment  $ab$  equals to  $r$ .

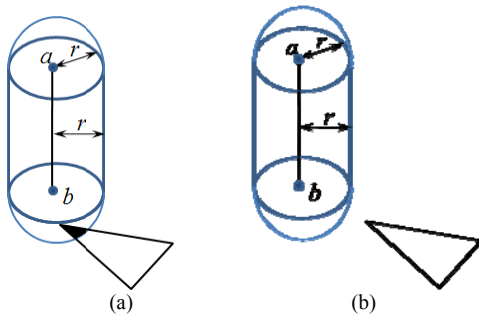


Fig. 3. Collision detection between capsule and triangular primitive: (a) intersection; (b) no intersection.

The intersection detection between two capsules is another situation. As Fig. 4 shows, the intersection between two capsules can be classified as three categories: intersection between two hemispheres, intersection between the hemisphere and the cylinder, and intersection occur in the two cylinders respectively. The first kind intersection can be checked by using the discriminant:

$$|\overline{a_1 a_2}| \leq 2r \quad (4)$$

Where  $a_1$  and  $a_2$  are the center of the two hemispheres respectively. The second kind intersection can be checked by using the following group discriminants:

$$\begin{cases} |\overline{a_1 a_2} - \text{dot}(\overline{a_1 a_2}, \vec{n})| \leq 2r \\ \text{dot}(\overline{a_1 a_2}, \vec{n}) \geq 0 \\ \text{dot}(\overline{b_1 a_2}, \vec{n}) \geq 0 \end{cases} \quad (5)$$

Where  $a_1$  and  $b_1$  are the two end points of the axis in the cylinder,  $\vec{n}$  is the axis vector, and  $a_2$  is the center of the hemisphere. The intersection between two cylinders can be checked by calculating the minimum distance of their axis line segments, intersection exists if the distance is small than  $2r$ .

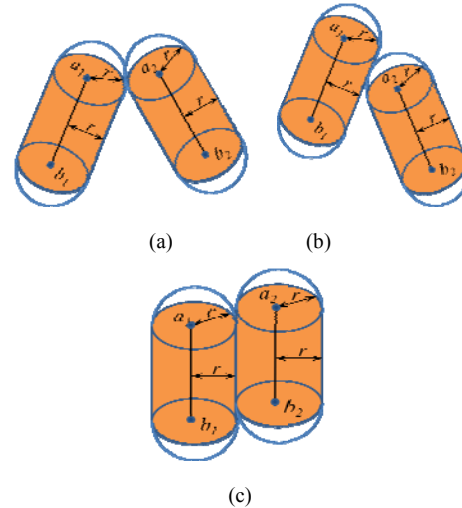


Fig. 4. Intersection detection between two capsules: (a) intersection between two hemispheres; (b) intersection between a hemisphere and a cylinder; (c) intersection occur in two cylinders.

#### 2) Spatial hashing based collision detection

Capsule-based method is used for coarse intersection detection among plant models. When intersection is detected, we use a spatial hashing method basing on [21] for more accurate detection, with calculation of penetration depth, which can be used for further collision response. The consistent process enables this method can not only be used for collision detection, but also for self-collision detection. This is very important to simulating deformation of plant model, which is a complex and non-convex outline.

This method works by creating spatial hash table for the initial vertices and tetrahedral mesh of the plant model. It subdivides all plant models into a lot of small AABB. Firstly, all vertices in each plant model in the virtual environment are classified basing on their locating in the small 3D grid unit. Then all of tetrahedral elements are also classified basing on the same grid units. When carry out the intersection detection, if a tetrahedral element is detected being intersecting with some AABB, all vertices locating in the AABB need to do intersection detection with the AABB. The barycentric coordinate of the tetrahedral element is calculated for estimating if the vertex has penetrated the tetrahedral element.

Teschner et al. [20] have used two spatial hashing methods for the initial vertices and the tetrahedral mesh of a 3D plant model respectively. For spatial hashing of the initial vertices, firstly a user-defined grid unit size is used to discretize the position of each vertex. The discretization bases on the following rule:

$$v(x', y', z') = ([x/l], [y/l], [z/l])$$

Where  $l$  represents the user-defined size of grid unit size;  $(x, y, z)$  is the coordinate of vertex  $v$  in the mesh, so  $(x', y', z')$  is the discretized coordinate of  $v$ . Then a hash function  $hash$  is used to map these discretized vertices to a one dimensional index  $h$ , as such the information of the vertex is saved to position of  $h$  in the hash table, where  $h = hash(x', y', z')$ . For the selection of hash function, Teschner et al. had given some rules, and they used the one as follows:

$$hash(x,y,z) = (x p_1 \text{ xor } y p_2 \text{ xor } z p_3) \text{ mod } n$$

Where  $(x,y,z)$  is the position of a vertex,  $p_1, p_2$  and  $p_3$  are three very large primes, their values are 73856093, 19349663 and 83492791 respectively. And  $n$  is the size of the hash table.

The spatial hashing for the tetrahedral mesh is used for actual intersection test in between a vertex  $p$  and a tetrahedron. This spatial hash table is created by traversing all tetrahedral elements. Firstly, the maximum and minimum AABB in the tetrahedral mesh are calculated, and these values are executed a modular operation with a user-defined grid unit size. The key code element of each grid unit in the AABB of the tetrahedral element is then calculated. Finally, a traversing is used to the AABB of each tetrahedral element from the minimum key code element to the maximum key code element. All vertices which can be found from the hash table index need to test intersection.

### 3.4 Related Parameters

#### 1) Parameters for the capsule

Two parameters are used in capsule-based collision detection method: the radius  $r$  and the height  $h$ . They influence the simulation in two aspects: one is the efficiency for collision detection, the other is the effect of natural deformation of the plant model, because the capsules of a plant model also serve as its physical deformation model by connecting these capsules as mass-spring objects. The smaller of  $r$  and  $h$ , theoretically, the more natural of the deformation effects would be. But smaller  $r$  and  $h$  also mean more time consuming in collision detection.

#### 2) Parameters for the spatial hash

Both the hash table size and grid unit size would influence

the performance of the collision detection significantly. In fact, the larger of the hash table size is, the less important of the quality of the hash function. Experiment has shown that larger hash table can reduce the probability of mapping different 3D coordinates to a same hashing index. On the other hand, larger hash table may bring some decrease to the performance due to memory management. Method has been used to handle this problem [20]. The grid unit size for spatial hash dividing would influence the amount of geometric elements of the simulated objects, because all these geometric elements need to be mapped to the same hash table. To get an optimum performance, the grid unit size needs to be specified approximately to the average length of the tetrahedral elements.

## IV. RESULTS

We implemented the proposed algorithm in C++ with OpenGL on a PC with a 2.8 GHz Pentium processor and a NVIDIA GeForce 7900 GS graphics card.

Firstly we simulate the effects of collision detection and response in between two plant models. The plant model using in this experiment is corn, each corn model consists of 925 vertices and 1184 triangles, with 1156 tetrahedral elements and 452 vertices in the corresponding tetrahedron. Two plant models are used in this example, and they are placed close to make some organs penetration mutually. These penetrations can be eliminated by using the collision detection and response method proposed in this paper, as Fig. 5 shows.

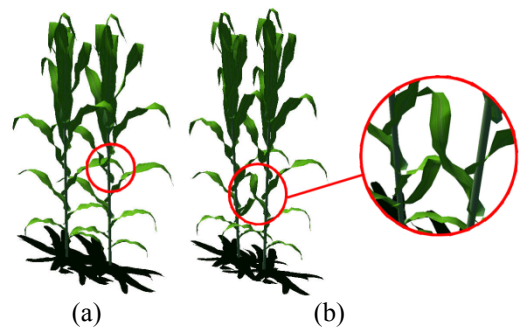


Fig. 5. Collision detection and response effects: (a) penetration phenomenon without collision response; (b) effect after using collision response.



Fig. 6. Natural stand effects of a cucumber model under different capsule height.

The second application example is applied to test the natural stand of a cucumber model. The 3D plant model has a height of 210cm, which consists of 22004 vertices and 30244 triangles. The number of the tetrahedral elements is 1431 with 521 vertices in the generated tetrahedron. By fixing the bottom capsule to the ground, we test the influence of parameters  $h$  and  $r$  to the natural stand effect of the plant model, by varying the two parameters in our experiments. We find that the influence of  $r$  is weak compared to  $h$ . As Fig. 6 shows, we use four different  $h$  values to simulate the natural stand shape of the cucumber model under gravity. It can be concluded that both larger and smaller  $h$  are not suitable for natural deformation of plant model. The simulation is run in real time with 75fps.

We also find that the parameter  $r$  is important to the interactive deformation simulation. Under suitable parameter  $h$ , appropriate  $r$  needs to be selected for natural interactive deformation. Using the same plant model and capsule height  $h$ , we vary the parameter  $r$  to test the effects. As Fig. 7 shows, the parameter  $r$  are 25cm, 10cm and 5cm in (a), (b) and (c) respectively, with the same  $h$  of 20cm. In which the red line represents the user's manual drag force.

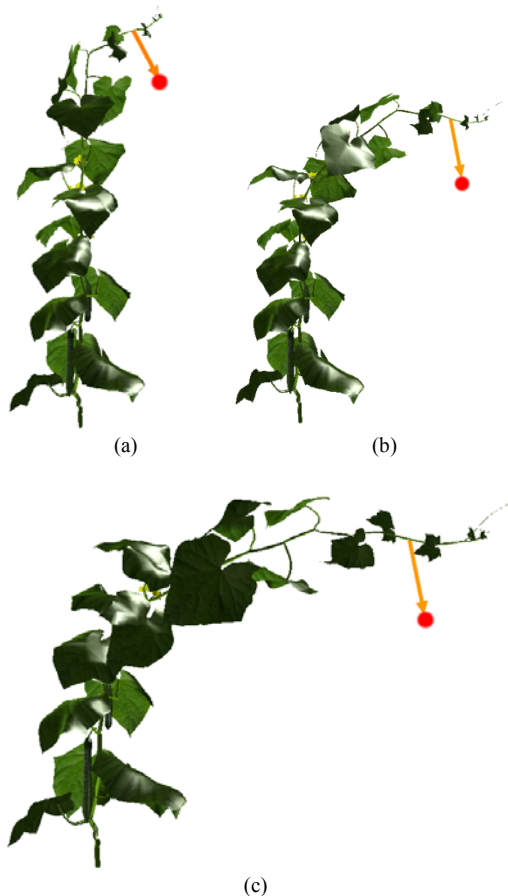


Fig. 7. Interactive deformation effects under different parameter  $r$ : (a)  $r=25\text{cm}$ ; (b)  $r=10\text{cm}$ ; (c)  $r=5\text{cm}$ .

The above experimental examples demonstrate that the proposed approach for simulating interactive deformation of 3D plant models is effective and flexible. Realistic effects of plant standing in nature, penetration processing between two plant models, and dynamic collision and response in plants

scenes can be achieved. Although our approach is based on tetrahedrons, it is not restricted to tetrahedrons and can be used to other object primitives. Currently, the method used in this paper detects whether a vertex penetrates a tetrahedron, but it does not detect whether an edge intersects with a tetrahedron. This can be improved in the further work.

## V. CONCLUSION

We have introduced a method for interactive deformation simulation of 3D plant models by combing collision detection and dynamical self-collisions detection for deformable objects. Difference from traditional methods, we use capsule-based and spatial hashing based detection for two phase processes, and achieve real-time simulation results. The experimental examples show that environment of up to 20k tetrahedrons can be performed in real-time, and independent from the number of objects. The proposed method can not only be used to a wide range of plant models, but also to other complex scenes.

But the limitations of our method are also evident. This method did not consider the difference of plasticity of different plant organs. As such it is difficult to be used as a quantitative method for simulating the deformation of a plant under different exterior force. We will combine our approach with the real material attributes of plant organ in the near future.

## ACKNOWLEDGEMENT

This work was supported by the National Natural Science Foundation for Young Scientists of China under Grant No. 61003118 and Beijing Training Program Foundation for Excellent Talent under Grant No.2010D002020000005.

## REFERENCES

- [1] J. Beaudoin and J. Keyser. Simulation levels of detail for plant motion, In Proc. ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 297-304, 2004.
- [2] C. Jirasek and P. Prusinkiewicz. A biomechanical model of branch shape in plants. In Proc. of the Western Computer Graphics Symposium, Whistler, Canada, pp. 23-26, 1998.
- [3] J. Hart, B. Baker and J. Michaelraj. Structural simulation of tree growth and response. The Visual Computer, 19: 151-163, 2003.
- [4] X. Y. Chi, B. Sheng, Y. Chen and E. H. Wu. Physically Based Simulation of Weathering Plant Leaves. Chinese Journal of Computers, 32(2): 221-230, 2009. (in Chinese with English abstract)
- [5] S. L. Lu, C. J. Zhao and X. Y. Guo. Venation Skeleton-Based Modeling Plant Leaf Wilting. International Journal of Computer Games Technology, Vol. 2009, Article ID 890917, 2009.
- [6] A. Nealen, M. Müller, R. Keiser, E. Boxerman and M. Carlson. Physically based deformable models in computer graphics. Computer Graphics Forum, 25 (4): 809-836, 2006.
- [7] Teschner M, Kimmerle S, Heidelberger B, et al. Collision detection for deformable objects. Computer Graphics Forum, 24(1): 61-81, 2005.
- [8] L. Shi, Y. Yu, N. Bell and W. W. Feng. A fast multi-grid algorithm for mesh deformation. ACM Transactions on Graphics, 25(3):1108-1117, 2006.
- [9] O. Sorkine, Y. Lipman, D. Cohenor, M. Alexa, C. Rossl, and H-P. Seidel. Laplacian surface editing. Proc. of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing, Eurographics Association, pp. 179-188, 2004.

- [10] X. Shi, K. Zhou, Y. Tong, M. Desbrun, H. Bao and B. Guo. Mesh puppetry: cascading optimization of mesh deformation with inverse kinematics. *ACM Transactions on Graphics*, 26(3): 81, 2007.
- [11] R. W. Sumner, M. Zwicker, C. Gotsman and J. Popović. Mesh-based inverse kinematics. *ACM Transaction on Graphics*, 24 (3): 488-495, 2005.
- [12] M. Botsch and L. Kobbelt. Multiresolution Surface representation based on displacement volumes. *Computer Graphics Forum*, 22(3): 483-491, 2003.
- [13] N. Galoppo, M. A. Otaduy, P. Mecklenburg, M. Gross and M. C. Lin. Fast simulation of deformable models in contact using dynamic deformation textures. *Proc. Eurographics/SIGGRAPH Symposium on Computer Animation*, pp. 73-82, 2006.
- [14] M. Botsch, M. Pauly, M. Wicke, and M. Gross. Adaptive space deformations based on rigid cells. *Computer Graphics Forum*, 26(3): 339-347, 2007.
- [15] X. Guo and H. Qin. Real-time meshless deformation: collision detection and deformable objects. *Computer Animation and Virtual Worlds*, 16(3-4): 189-200, 2005.
- [16] M. Pauly, R. Keiser, B. Adams, P. Dutré, M. Gross, L. J. Guibas. Meshless animation of fracturing solids. *ACM Transactions on Graphics*, 24(3): 957-964, 2005.
- [17] J. Chang and J. J. Zhang. Mesh-Free Deformations. *Computer Animation and Virtual Worlds*, 15(3-4): 211-218, 2004.
- [18] B. Adams, M. Ovsjanikov, M. Wand, H-P. Seidel and L. J. Guibas. Meshless modeling of deformable shapes and their motion. *Proc. Eurographics/ACM SIGGRAPH Symposium on Computer Animation*, Dublin, Ireland, pp.77-86, 2008.
- [19] J. Spillmann, M. Wagner and M. Teschner. Robust tetrahedral meshing of triangle soups. *Proc. Vision, Modeling, Visualization VMV'06*, Aachen, Germany, Nov. 22-24, 2006.
- [20] Y.M. Kang and H.G. Cho. Bilayered approximate integration for rapid and plausible animation of virtual cloth with realistic wrinkles. *Proceedings of Computer Animation*, IEEE Computer Society, pp 203-214, 2002.
- [21] M. Teschner, B. Heidelberger, M. Müller, D. Pomeranets and M. Gross. Optimized spatial hashing for collision detection of deformable objects. *Proceedings of Vision, Modeling, Visualization VMV'03*, pp. 47-54, 2003.

Technology Commission of Ministry of Agriculture, China. He also is an adjunct professor of several universities.

Dr. Zhao received the Excellent Young Scientist Award from China Society of Agricultural Sciences at 1998, and the Excellent Scientist Award from Ministry of Science and Technology of China at 2001. His current research focuses on digital agriculture, virtual reality and GIS.

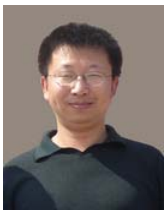


**Changfeng Li** is a graphics programmer at the National Engineering Research Center for Information Technology in Agriculture, China. His research interests include modeling, animation, and realistic rendering of organic objects.



and image processing.

**Shenglian Lu** received his M.S. degree in computer science, in 2005. He joined the National Engineering Research Center for Information Technology in Agriculture (NERCITA) in 2008 after receiving his Ph.D. degree in Mechanical Engineering from Shanghai Jiao Tong University. Currently he serves as an assistant research fellow in Computer Graphics. His research interests include simulating, modeling and visualizing of complex biological objects, physically-based modeling,



**Xinyu Guo** received his Ph.D. degree in agricultural science from China Agricultural University, in 2001. Currently he is a Research Fellow at the National Engineering Research Center for Information Technology in Agriculture. His current research focuses on digital agriculture, plant structure-function modelling and virtual reality.



Project of Intelligent Agricultural Information Technology Application, National High-Tech R&D Program of China; Member of Science and

**Chunjiang Zhao** received his M.S. degree from Beijing Academy of Agri. & Forestry Sciences in 1988 and a Ph.D. degree from China Agricultural University, in 1991. Currently he is the director of National Engineering Research Center for Information Technology in Agriculture. He also server on many social and academic assignment, including the Leading Scientist for Modern Agriculture Projects in National High-Tech Research and Development 863 Program of China; Chief Expert in the