VECIMS 2009 - International Conference on Virtual Environments, Human-Computer Interfaces and Measurements Systems
Hong Kong, China
May 11-13, 2009

A 4-Layer Flexible Virtual Hand Model for Haptic Interaction

Huagen Wan, Feifei Chen
State Key Lab of CAD&CG
Zhejiang University
Hangzhou, China
hgwan@cad.zju.edu.cn

Xiaoxia Han
Department of Information Science and Electric Engineering
Zhejiang University
Hangzhou, China

Abstract—**Virtual hand interactions play key roles in virtual environments. The recent addition of force feedback to virtual reality simulations has enhanced their realism, especially when dexterous manipulation of virtual objects is concerned. In the past decades, much effort has been made on virtual hand modeling from the perspectives of computer animation and human computer interaction. However, much less attention is paid on haptic modeling of flexible virtual hand. In this paper, we propose a 4-layer flexible virtual hand model for virtual hand haptic interaction. The skin layer, kinematics layer, collision detection layer and haptic layer are integrated into a sophisticated virtual hand to simulate the human hand's natural anatomy in its appearance and motion, and to reflect the area contact feature of force feedback datagloves. The infrastructure and details of the flexible virtual hand model are discussed. Experimental results show that the proposed flexible virtual hand demonstrates good performance in virtual hand haptic applications.**

Keywords-virtual hand model; haptic interaction; collision detection; dataglove

I. INTRODUCTION

The human hand is a dexterous organ with complex shape and motion. It provides an important interface to the physical world. Virtual reality (VR) technology is revolutionizing the way of interacting with computers. In a virtual environment, a virtual hand, either rigid or flexible, is often used as the avatar of the human hand [1, 2]. It provides a natural interface to the computer synthesized virtual world.

To drive a flexible virtual hand, a dataglove is usually used to track the user's finger motions. Nowadays, datagloves have become a main kind of VR input devices [3], moreover, force feedback datagloves provide the user with the sensations involved by force feedback during the manipulation of virtual objects [4]. Especially, haptic feedback is mandatory when the graphics is corrupted (simulating poor visibility) or when the manipulated object is partly or totally occluded, or when the environment is dark [5].

Indeed, a large number of applications have been foreseen for haptic interaction, for instances, virtual sculpture, virtual surgery, education, entertainment, and industry applications including virtual prototyping, training, and maintenance [7]. Therefore, the modeling of the virtual hand, including shape modeling, kinematics modeling, and haptic modeling, is fundamental and important for 3D interaction, and is required by a wide range of virtual reality applications. In the past decades, much effort has been paid on virtual hand modeling in the community of computer animation and human computer interaction [7-12], however, much less attention is paid on haptic modeling of virtual hand [5,13,14].

In this paper, we investigate the modeling of a flexible virtual hand for haptic applications. A 4-layer model is proposed which consists of skin layer, kinematics layer, collision detection layer and haptic layer. Specifically, our virtual hand model simulates the human hand's natural anatomy in its appearance and motion, and reflects the area contact feature of force feedback datagloves. The infrastructure and details of the proposed flexible virtual hand model are discussed in the sections II, III, and IV. Experimental results are demonstrated in the section V, and finally a brief conclusion is drawn in the section VI.

II. INFRASTRUCTURE OF OUR FLEXIBLE VIRTUAL HAND

The human hand is a complex organ of a human being. It's not trivial to build the geometry model of a virtual hand, not to mention to set up its kinematics and haptic model. Basically, our proposed flexible virtual hand consists of 4 layers, namely, skin layer, kinematics layer, collision detection layer, and haptic layer.

As illustrated in Figure 1, the 4 layers are integrated into a sophisticated virtual hand model to facilitate virtual hand haptic interaction. The user's hand motion data are captured by 3D tracker and dataglove and used to drive the skeleton structure of the kinematics layer of the virtual hand. While the skeleton transformations are directly transferred to the collision detection layer and the haptic layer, the skin layer's deformation is driven by the skeleton transformations as described in the section III. Note that the skin layer's deformation is also constrained by whether there are contacts between the collision detection layer and other virtual objects in the scene. The feedback force computation is performed between the haptic layer and other virtual objects, which will be described in the section IV.

Supported by the National Natural Science Foundation of China (60673197), and the 863 High Technology Plan of China (2006AA01Z130).

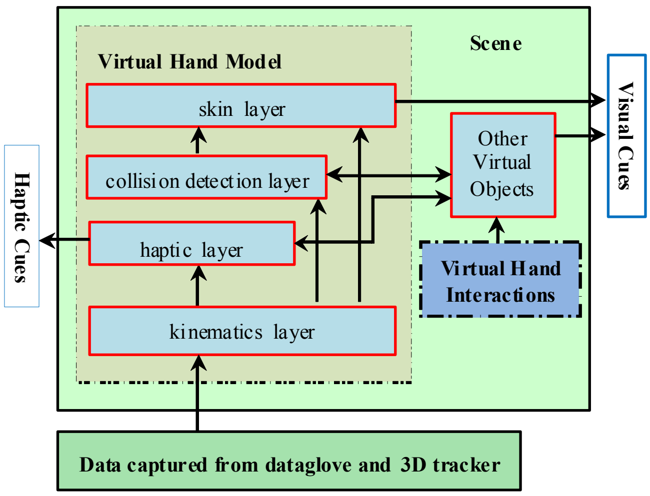

Figure 1. Infrastructure of our flexible virtual hand

A. Skin Layer

The skin layer is the surface geometry of the virtual hand. The complexity of the hand structure makes its shape modeling a complicated and tedious process. Many modeling techniques, from polygonal modeling, parametric surface modeling, to implicit modeling, have been proposed for modeling geometry of the human hand.

Polygon mesh is a fundamental representation of 3D objects. For virtual hand modeling, the built polygonal model should on the one hand be accurate enough to reflect the human hand shape, and on the other hand, it should not be too complicated to hinder the real time simulation of hand motion. Though it is often very tedious since large amount of user interaction is inevitable to construct the polygonal mesh of a human hand, the development of subdivision surfaces seems to help alleviate the heavy burden of lots of user input [15], and many mesh modeling tools exist. To facilitate rendering and motion control, we construct the geometry of the virtual hand with triangular mesh, based on the knowledge of the hand shape and its anatomic structure [16]. Besides, the balance between the amount of triangles and the visual appearance is also taken into account.

B. Collision Detection Layer

Real-time collision detection is used to automatically identify whether there are interferences between the virtual hand and virtual objects. In general, collision detection requires a significant computational overhead, especially when involving deformable models. However, collision detection should be computationally efficient since real-time feedback is fundamental for haptic interactions.

To some extent, visual realism is of more interest, rather than accuracy, for virtual hand operations. We think that collision detection between virtual hand and virtual objects is a more qualitative issue rather than a quantitative one as far as haptic interaction is concerned. Realizing this, we build simplified structures for the palm and each joint of the virtual hand, and use these simplified geometries as the collision detection layer. In order to prevent the penetration of virtual hand into virtual objects, a simplified structure is a bit larger than its corresponding geometry. The classic software toolkit, RAPID, is used as the collision detection engine between the collision detection layer and virtual objects in the scene [17].

III. KINEMATICS MODEL

The human hand is a complex structure with extra articulation that enables us to grasp, hold, and operate a wide variety of objects. The kinematics layer of the virtual hand is determined by the hand's skeleton structure. For a human hand, each finger has three phalanges (proximal, middle, and distal); the thumb has two (proximal and distal). Correspondingly, each finger has three joints (distal interphalangeal joint (DIJ), proximal interphalangeal joint (PIJ) and metacarpophalangeal joint (MPJ)) with DIJ and PIJ each having one degree of freedom (DOF) and MPJ having two DOFs; the thumb also has three joints (thumb IJ, thumb MPJ, and trapeziometacarpal joint (TMJ)) with TMJ and thumb MPJ each having two DOFs and thumb IJ having one DOF. The palm has two DOFs [16]. Therefore, the hand motion is highly constrained by the joints. More complicated, a joint is often constrained by other joints when in motion.

The extra articulation of the hand makes it difficult to realistically simulate the motion and muscle deformation of the hand with simple kinematics models. Usually, a three-layer model which consists of the skeleton layer, the muscle layer and the skin layer is adopted to handle the virtual hand deformation [9]. However, due to strict computational time limit of haptic interaction, we use a simpler two-layer model and employ skeletal subspace deformation (SSD) to handle virtual hand kinematics [18, 19].

As illustrated in Figure 2, the kinematics model of the virtual hand consists of the skin layer (i.e. surface geometry) and the kinematics layer. The skin layer is the triangular mesh used for displaying purpose. Its deformation is driven by the kinematics layer. The kinematics layer is actually a hierarchical skeleton structure. The skeleton structure is built based on the anatomic structure of the human hand [16]. Each finger is abstracted as a joint chain, where each joint has a local coordinate system, and includes such information as the joint position, orientation, rotation angle and a pointer to the next joint. The joint rotations are controlled by flex data (rotation angles) captured by a dataglove (e.g. the CyberGlove dataglove [3]).

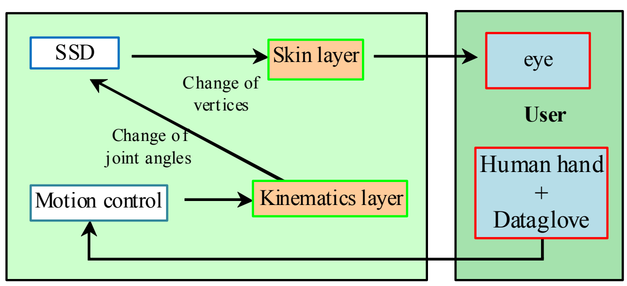

Figure 2. Kinematics model of our flexible virtual hand

The process of deforming the skin can be described as follows. Firstly, the user's finger rotations are captured by the dataglove, and the captured flex data directly control the rotations of the joints in the skeleton structure of the kinematics layer. Secondly, the rotations of the joints drive the deformation of the skin layer according to the SSD which is evaluated by the weighted blending of an affine transformation of each joint by (1).

$$v_j = \sum_{i=1}^{n} w_i M_i v_{j0} \tag{1}$$

Where n is the number of joints, v_j is the j-th vertex in an arbitrary pose, v_{j0} is the j-th vertex in its rest pose, M_i is the affine transformation matrix defined by flex angles of joints and hand motion, and w_i is a joint weight that defines the contribution of the i-th joint's transformation to the j-th vertex. The weight w_i is assigned by the user to control deformation.

We use graphics processing unit (GPU) to accelerate the SSD computation. The vertices, normals, joint weights and joint indices are all stored in textures and transferred to GPU. The affine transformations of each joint are also transferred to GPU. Figure 3 lists the pseudo code of the fragment program for SSD computation. Note that the joints which contribute to any vertex are limited to at most 4 to facilitate the GPU data storage.

```
// Pixel Shader – SSD Computation
uniform sampler2DRect vertex, normal, weight, jointIdx;
uniform mat4 world2joint[32], joint2world[32];

void main()
{
        vec2 coords = gl_TexCoord[0].xy;
        vec4 w = texture2DRect(weight, coords);
        ivec4 jIdx = ivec4(texture2DRect(jointIdx, coords));
        gl_FragData[0] = vec4(0.0);
        vec4 tmpnormal = vec4(0.0);
        for(int i = 0; i < 4; i++) {
                vec4 wlocal = world2joint[jIdx[i]]*texture2DRect(vertex, coords).rgba;
                gl_FragData[0] += joint2world[jIdx[i]]*wlocal*w[i];
                wlocal = world2joint[jIdx[i]]*texture2DRect(normal, coords).rgba;
                tmpnormal += joint2world[jIdx[i]]*wlocal*w[i];
        }
        gl_FragData[1] = vec4(normalize(tmpnormal.xyz), 0.0);
}
```

Figure 3. Pseudo code of the fragment program for SSD computation

IV. HAPTIC LAYER AND FEEDBACK FORCE COMPUTATION

Basically, the integration of haptic feedback within a virtual environment raises many problems at both hardware and software levels. During the past decade, much effort has been made to develop haptic rendering algorithms for various haptic devices. These methods can be classified into categories according to the avatars used: point-based methods, ray-based methods and object-based methods [6]. However, a current major limitation for the design of haptic interfaces is our poor knowledge concerning human haptic perception. Indeed, both psychological and physiological issues of haptic perception should be taken into account as far as haptic interaction is concerned.

A. Haptic Layer

We think that the haptic layer should fully respect the feature of area contacts between virtual fingertips and target objects in order to present realistic force. As a result, we propose a simple yet effective haptic layer of the virtual hand. As illustrated in Figure 4, the haptic layer is composed of a cluster of line segments whose end points form a reasonable sample of the estimated contact area of a virtual fingertip. The haptic layer is generated as follows. Firstly, a grid a bit narrower than the fingertip is created in front of the fingertip, and the center of the grid (COG) is projected along the grid normal toward the back face of the fingertip to get an apex point (AP). Secondly, the AP is connected with each of the grid points to form a pyramid, and the pyramid is trimmed by the front surface of fingertip which results in a grid of intersection points (green points in Figure 4). Finally, the AP is connected with each of the intersection points to form a line segment cluster which consists of the haptic layer of our virtual hand model.

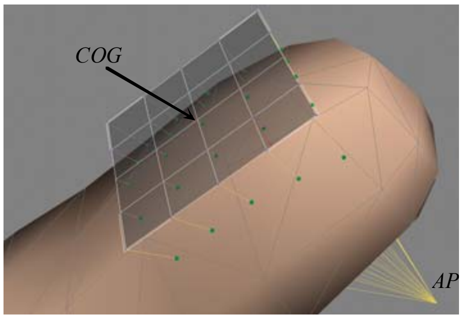

Figure 4. Construction of haptic layer

B. Feedback Force Computation

As the haptic layer is composed of many line segments, we perform force computation first by detecting whether the line segments intersect with the target model. We perform voxelization on target models in the pre-processing stage to gain efficiency for intersection tests as it has proven to be a very significant way for accelerating such computations [14,20,21].

Given a line segment and a voxelized target model, following two steps are taken to check whether the line segment intersects with the target model. Firstly, all the voxels intersecting with the line segment are identified to reduce unnecessary computation since only the facets of the target model contained in these voxels may intersect with the line segment. We extend the method proposed by [22] to perform such intersection tests. Secondly, the contact point between the line segment and the target model is determined. As only triangles contained in the intersecting voxels need to be checked, we extend the algorithm presented in [23] to calculate the nearest intersections between the line segment and each triangle contained in the intersecting voxels. The above process continues till all line segments of the haptic layer are dealt with, and the results (e.g. intersection points, indices of intersecting triangles) are recorded for force computation.

We use the Hooke's law to compute the feedback force to each fingertip based on the collision detection results [14]. Assume that a line segment L intersects a triangle M of the target model at the point P (Figure 5). Let P_s and P_e be the start point and end point of the line segment L (The equation of L is $L(t) = P_s + t(P_e - P_s)/\|P_e - P_s\|, (0 \le t \le 1)$, the intersection point P is represented by $L(t_0)$). Then the direction of the feedback force F_i generated by L is the same as the normal of M, and its magnitude is calculated by (2).

$$F_i = kx = kd\cos\alpha \tag{2}$$

where k is the stiffness of the target model, x is the penetration depth, and d is the penetration length along L:

$$d = \|P_e - P\| = (1 - t_0)\|P_e - P_s\| \tag{3}$$

It is worth noting that $\|P_e - P_s\|$ can be calculated in advance during the pre-processing stage.

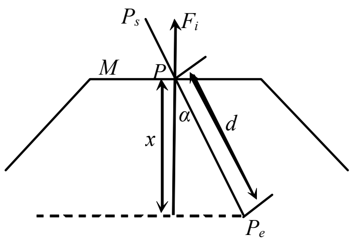

Figure 5. Feedback force computation

Assume there are m line segments of the haptic layer intersecting the target model, the feedback force can be calculated as an average of the feedback forces generated by each line segment:

$$F = \sum_{i=1}^{m} F_i / m \tag{4}$$

V. EXPERIMENTAL RESULTS

The proposed flexible virtual hand model has been implemented with C++, and tested on a PC with Intel Core 2 Quad Q6600 2.40GHz CPU with 2G RAM. The graphics hardware is NVIDIA GeForce 8800GTS. The force feedback device used was the CAS-dataglove with PEDfinger which is a force feedback system (Figure 6) developed by Institute of Automation, Chinese Academy of Sciences to help users feel virtual 3D objects [24,25].

The virtual hand model has more than 6,500 triangles. Figure 7~10 show its skin layer, collision detection layer, kinematics layer and haptic layer respectively. Figure 11 demonstrates the virtual scene for testing the performance of our flexible virtual hand model. Figure 12 shows a sphere grasped by the flexible virtual hand. Figure 13 shows a cube grasped by the virtual hand. In both Figure 12 and Figure 13, the red line segments shown at the back face of each fingertip indicate the feedback forces' magnitudes. In our tests, the overall update frame rate is over 60 frames/second. As a result, the user can feel continuous force feedback when he/she interacts with virtual objects The computational overhead in general includes the rendering of the virtual hand and the scene, skin layer deformation, collision detection between the collision detection layer and the virtual objects, and feedback force computation, etc.

Figure 6. CAS-dataglove with PEDfinger

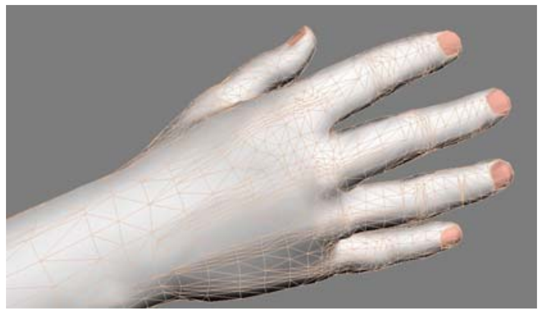

Figure 7. Skin layer

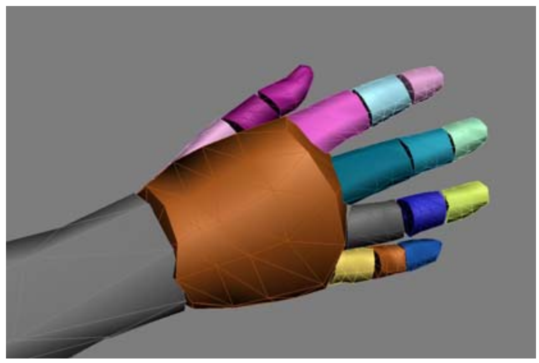

Figure 8. Collision detection layer

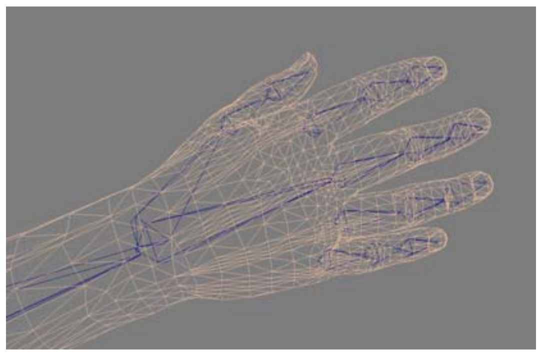

Figure 9. Kinematics Layer

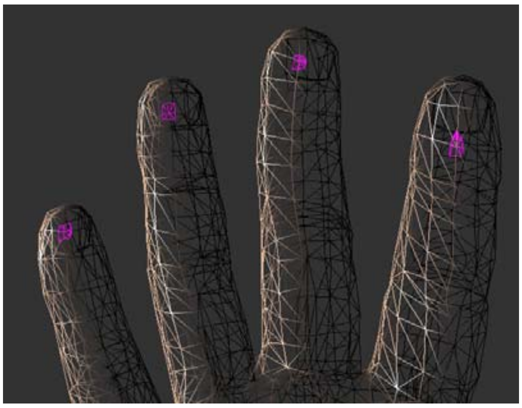

Figure 10. Haptic layer

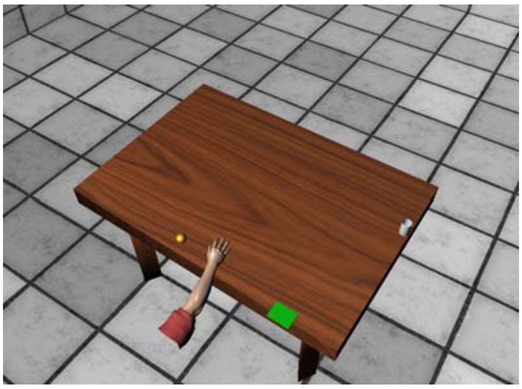

Figure 11. Virtual scene

Figure 12. Flexible virtual hand grasps a sphere

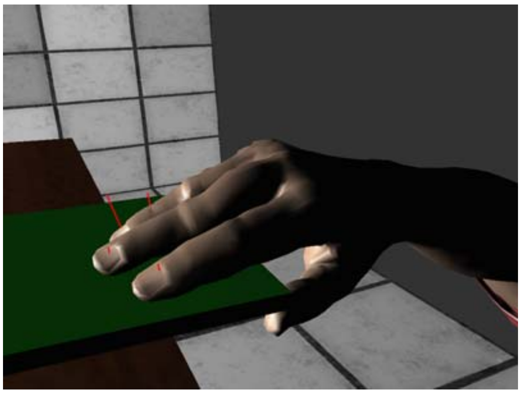

Figure 13. Flexible virtual hand grasps a cube

VI. CONCLUSIONS

Haptic devices are used to extend a human being's sense of touch into a virtual world, in which the user can feel the geometry and other properties of virtual objects. Virtual hand haptic interactions play key roles in virtual environments, especially when dexterous manipulation of virtual objects is concerned. In this paper, a 4-layer flexible virtual hand model which consists of skin layer, collision detection layer, kinematics layer and haptic layer has been proposed for virtual hand haptic applications.

In this sophisticated infrastructure which simulates the human hand's natural anatomy in its appearance and motion, the user's hand motion data captured by 3D tracker and dataglove are used to drive the kinematics layer, and directly transferred to the collision detection layer and the haptic layer, while the skin layer's deformation is driven by the skeleton transformations as well as constrained by the collision detection results between the collision detection layer and other virtual objects in the scene.

The feedback force computation is performed between the haptic layer and other virtual objects. As a multiple-point sampling scheme is used to sample the finger tips, the area contact feature of force feedback dataglove is well reflected. Experimental results show that the user can feel continuous force feedback when he/she interacts with virtual objects during our test on a PC with an in-house developed force feedback dataglove.

ACKNOWLEDGMENT

The authors would like to thank Professor Qingxiu Du and Professor Kui Yuan with Institute of Automation, Chinese Academy of Sciences for providing the dataglove (CAS-glove with PEDfinger) and the digital image shown in Figure 6.

REFERENCES

[1] U. Jayaram, H. Tirumali, S. Jayaram, "A tool/part/human interaction model for assembly in virtual environments," Proceedings of ASME DETC'00, Baltimore, Maryland, September 2000.

[2] H. Wan, Y. Luo, S. Gao, Q. Peng, "Realistic virtual hand modeling with applications for virtual grasping," Proceedings of ACM SIGGRAPH International Conference on VRCAI, Singapore, June 2004.

[3] Wireless Data Glove: The CyberGlove® II System. http://www.immersion.com/3d/products/cyber_glove.php.

[4] CyberGrasp™ Exoskeleton. http://www.immersion.com/3d/products/cyber_grasp.php.

[5] V. Popescu; G. Burdea, M. Bouzit. Virtual reality simulation modeling for a haptic glove. In Proceedings of the Computer Animation'99, Geneva, Switzerland, 1999.

[6] S. Laycock, A. Day, "A survey of haptic rendering techniques,". Computer Graphics Forum, Vol.26, No.1, pp.50-65, 2007.

[7] K. Hui, N. Wong, "Hands on a virtually elastic object," The Visual Computer, Vol.18, No.3, pp.150-163, 2002.

[8] VirtualHand® SDK. http://www.immersion.com/3d/products/virtualhand_sdk.php.

[9] L. Moccozet, Z. Huang, M. Thalmann, D. Thalmann, "Virtual hand interactions with 3D world," Proceedings of Multimedia Modeling97, Singapore, pp.307-322, 1997.

[10] J. Lee, S. Yoon, "Realistic human hand deformation," Computer Animation and Virtual Worlds, Vol.17, Issue 3-4, July 2006, pp.479-489.

[11] T. Rhee, U. Neumann, J. Lewis, "Human hand modeling from surface anatomy," Proceedings of the 2006 symposium on Interactive 3D graphics and games, March, 2006, Redwood City, California.

[12] M. Thalmann, R. Laperriere, D. Thalmann, "Joint–dependent local deformations for hand animation and object grasping," Proceedings of Graphics Interface'88, pp.26-33, June 1988.

[13] A. Maciel, S. Sarni, O. Buchwalder, R. Boulic, D. Thalmann, "Multi-finger haptic rendering of deformable objects," Proceedings of Eurographics Symposium on Virtual Environments, 2004.

[14] H. Wan, X. Han, Z. Zhou, "Realistic force rendering with applications for virtual assembly," Journal of Computational Information Systems, Vol.3, pp.303-308, February 2007.

[15] T. Sederberg, , J. Zheng, D. Sewell, M. Sabin, "Non-uniform recursive subdivision surfaces," Proceedings of SIGGRAPH98, pp.387-394, 1998.

[16] W. Kapit, L. Elson. The anatomy coloring book. 2nd edition, Harper Collins. 1993.

[17] S. Gottschalk, M. Lin, D. Manocha, "Obb-Tree: A hierarchical structure for rapid interference detection," Proceedings of the ACM Siggraph'96, New Orleans, LA, August 1996.

[18] M. Thalmann, R. Laperriere, D. Thalmann, "Joint–dependent local deformations for hand animation and object grasping," Proceedings of Graphics Interface'88, pp.26-33, June 1988.

[19] T. Rhee, J. Lewis, U. Neumann, "Real-time weighted pose-space deformation on the GPU," Computer Graphics Forum, Vol 25, No 3, pp.439-448, In Proceedings of Eurographics06, Vienna, Austria, September 2006.

[20] A. Kaufman. Introduction to volume graphics. SIGGRAPH'99 course notes, course 41, pp.24-47, 1999.

[21] J. Huang, R. Yagel, V. Filippov, Y. Kurzion, "An accurate method for voxelizing polygon meshes," Proceedings of ACM 1998 Symposium on Volume Visualization, Research Triangle Park, NC, October 1998.

[22] W. Andrew. Fast ray-box intersection. Graphics gems, Academic Press Professional, Inc., San Diego, CA, 1990.

[23] T. Moeller, B. Trumbore, "Fast, minimum storage ray-triangle intersection," Journal of Graphics Tools, Vol.2, No. 1, pp. 21-28, 1997.

[24] Y. Fu, K. Yuan, H. Zhu, Q. Du, "Motion Modeling and Software System Design for CAS-Glove Data Glove," Journal of System Simulation, Vol.16, No.4, pp.660-662,666, 2004. (in Chinese)

[25] K. Yuan, H. Zhu, "A new kind of force-reflecting device and its control," Proceedings of IEEE Conference on Intelligent Robots and System.Lausanne,Switzerland, pp. 2920-2924, 2002.