Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 691816, 7 pages
http://dx.doi.org/10.1155/2014/691816

Research Article

Active Grasping Control of Virtual-Dexterous-Robot Hand with Open Inventor

Jinbao Chen, Dong Han, and Zhuang Peng

College of Astronautics, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China

Correspondence should be addressed to Jinbao Chen; chenjbao@nuaa.edu.cn

Received 13 November 2013; Accepted 24 December 2013; Published 30 January 2014

Academic Editor: Weichao Sun

The grasping technology of the virtual-dexterous-robot (VDR) hand plays a key role in the teleoperation of the space robot. For a grasp task in the virtual environment, the collision detection and virtual force calculation need to be implemented. Firstly, a tree-structure virtual scene including a VDR hand and a target object is built up with open inventor (OIV). Secondly, the collision manager provided by OIV is used for collision detection and the oriented bounding box (OBB) is adopted to improve the real-time performance of collision detection. Thirdly, an algorithm is proposed for calculating the virtual force by using the contact deformation. And the contact deformation is calculated according to the transformation matrix between the coordinate systems of the two contact objects. Furthermore, the contact friction is also calculated by this matrix. Considering the virtual force, the modified stable grasping conditions are proposed which are suitable for the virtual grasping. Then a method is proposed for implementing the grasp or release operation by translating different nodes in the tree-structure scene, which can avoid destroying the tree structure. Finally, the viability and effectiveness of the proposed algorithms are proved by simulation experiments.

1. Introduction

Virtual reality (VR) technology has been applied to many fields such as teleoperation [1–4] and health [5–7]. The operator can control a VDR hand in the virtual environment instead of the real robot hand to reduce stress and time delay effect [8].

The grasping technology of the VDR hand lets operators manipulate the virtual object with a feeling of manipulating the real object. There are two problems that need to be solved for grasping in the virtual environment: collision detection and virtual force calculation.

In order to give the operator sense of immersion, the collision detection is necessary in the virtual environment. So, Wan et al. [9] implemented the Voxmap-PointShell (VPS) method as the collision detection engine and reduced the computational time by collision proxy. And Cameron et al. [10] detected the collision through calculating the distance between two objects. This algorithm only applies to simple geometric shapes. In this paper, SoCollisionManager is used for collision detection, which is a class in OIV. OIV is an object-oriented, cross-platform 3D graphics toolkit for the development of interactive 3D graphics applications using C++ or Java. And OIV provides many useful functions for users such as collision detection. SoCollisionManager is an efficient collision manager and can simplify the collision-detection algorithm. For real-time collision detection, the bounding boxes are added to the geometrical models of the VDR hand and object.

The virtual environment is required to possess sufficient verisimilitude. Thus, the virtual force, particularly with regard to tactility, is of vital importance [11]. Wan et al. [12] proposed a 4-layer virtual hand model and used the Hooke's law to compute the feedback force according to the intersection results between the haptic layer and the target model. This method is complex and is not suitable for the VDR hand model. Zhou [13] calculated the virtual force according to the fingertip displacement of the VDR hand obtained from the data glove. In this paper, an algorithm is proposed for calculating the virtual force based on the contact deformation. And the contact deformation can be got according to the transformation matrix from object coordinate system to fingertip coordinate system. Since the transformation matrices are got through the functions in

OIV rather than the data glove, the algorithm is more real time and robust. Furthermore the contact friction can also be calculated using this algorithm in the virtual environment.

Considering the grasping stability in the virtual environment, the grasping conditions are usually based on the position and number of the contact points [14]. In order to enhance the reliability, the force constraints are added to the grasping conditions in this paper.

Section 2 of this paper introduces the VDR hand model built up with OIV and ProE. The implementation of collision detection, virtual force, stable grasping, and virtual operation with OIV is, respectively, introduced in Sections 3, 4, 5, and 6. The validation of the proposed algorithms is presented in Section 7. Finally a brief conclusion is drawn in Section 8.

2. VDR Hand Model

It is difficult to build up a complex model (especially for modeling complex 3D surface) by directly using the functions of OIV [3]. The ProE is 3D modeling software and is convenient for building up complex 3D models. So, the high-fidelity VDR hand is built up with ProE and OIV (Figure 1). Firstly, each part of the hand model in ProE is imported into OIV. Then, these parts are defined as the named nodes in OIV. Finally, these nodes are assembled into a tree-structure VDR hand model with OIV functions according to the certain parent-child relationship between the nodes. And the parent-child relationship is determined by the assembly sequence of the real robot hand. The tree-structure virtual scene model includes the VDR hand model and a target object. And the tree structure of the virtual scene or the relationship between the nodes is shown in Figure 2, where "Root" node represents the root node of the whole virtual scene and "Object" node represents the target object in the virtual environment. Furthermore, the parent-child relationship plays a key role in the VDR hand motion.

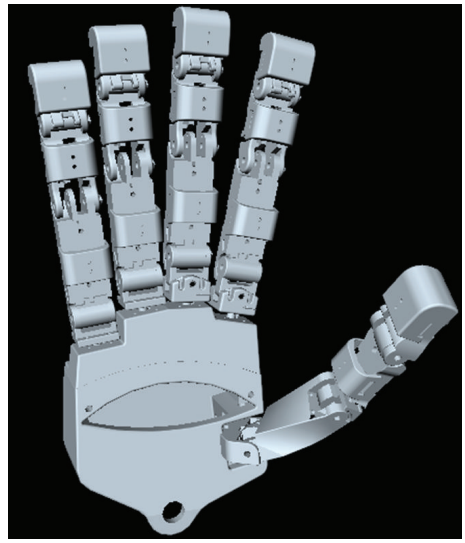

Figure 1: The VDR hand model in OIV.

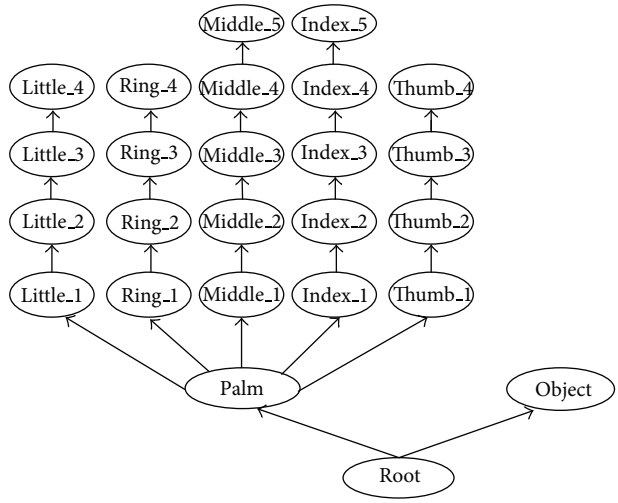

Figure 2: The parent-child relationship between the nodes.

3. Collision Detection

Collision detection is used for avoiding the interpenetration between the virtual objects and protecting the real robot. SoCollisionManager as the collision manager in OIV has three members that are related to the realization of collision detection: a *scene*, an *object*, and a *transformation*. The *scene* is the "Root" node in Figure 2. It means that the collision scene includes the whole VDR hand and the target object. And the *object* is the "Object" node which represents the target object. It means that the collision manager can detect collisions between the target object and other objects within the collision scene. The *transformation* is an attribute node in OIV and acts on the nodes of the VDR hand or the "Object" node. When the referenced *transformation* changes, the collision manager looks for a collision between the *scene* (the VDR hand) and the *object* (the target object).

For real-time collision detection, the OBB algorithm is adopted. The OBB algorithm is able to detect all contacts between complex geometries [15] and close the object closely [16]. SoBoundingbox as a class in OIV provides a three-dimensional graphics bounding box based on OBB, which makes the creation of OBB relatively easy. The dynamic OBBs are added to the fingertips of the VDR hand and the static OBB is added to the target object [17].

4. Virtual Force Calculation

For stable grasping, the operator needs to know the contact force when there is a collision between the fingers and the target object. The dynamical equation of the contact force in the jth contact point can be expressed as

$$f_{cj} = m\ddot{x} + b\dot{x} + kx, \tag{1}$$

where f_{cj} represents the contact force in the jth contact point; m, b, and k are, respectively, the inertial mass, damping coefficient, and stiffness coefficient; x, $\dot{x}$, and $\ddot{x}$ are, respectively, the deformation, velocity, and acceleration of the contact. Therefore calculating x, $\dot{x}$, and $\ddot{x}$ is needed to get the contact force f_{cj} in the virtual environment.

Assuming that the relative position of the fingertip and the target object is as shown in Figure 3 (O_f is the coordinate system of the fingertip center; O_o is the coordinate system of the target object; R_f represents the radius of the curved surface of the fingertip; D_o represents the distance between the point O_o and the contact surface; T_x and T_y represent the translation distances from the point O_o to the point O_f), x is given by

$$x = R_f + D_o - T_x \geq 0. \tag{2}$$

In Figure 3, R_f and D_o are, respectively, determined by the structure of the fingertip and the size of the target object. However, T_x and T_y are calculated by the transformation matrix from O_o-coordinate system to O_f-coordinate system. And $\mathbf{T}$ is defined as the transformation matrix and is given by

$$\mathbf{T} = \begin{bmatrix} n_x & o_x & a_x & T_x \\ n_y & o_y & a_y & T_y \\ n_z & o_z & a_z & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3}$$

SoGetMatrixAction, a class in OIV, provides an efficient method to obtain the transformation matrix. When SoGetMatrixAction acts on a path, it can obtain the accumulation of all the transformation matrices in the path. Since the fingertip of the VDR hand and the target object cannot be on the same path in OIV, the matrix $\mathbf{T}$ cannot be obtained directly. Figure 2 shows that the path from "Root" node to "Object" node can be established and the path from "Root" node to fingertip node can also be established. Thus, the matrix $\mathbf{T}$ can be expressed as

$$\mathbf{T} = {\mathbf{T}_{\mathrm{RO}}}^{-1}\mathbf{T}_{\mathrm{RF}}, \tag{4}$$

where $\mathbf{T}_{\mathrm{RO}}$ represents the transformation matrix from "Root" node to "Object" node and $\mathbf{T}_{\mathrm{RF}}$ represents the transformation matrix from "Root" node to fingertip node. The inverse of $\mathbf{T}_{\mathrm{RO}}$ can be got by "getInverse()" function of SoGetMatrixAction and the matrix $\mathbf{T}_{\mathrm{RF}}$ can be got by "getMatrix()" function of SoGetMatrixAction. Using the functions provided by OIV makes the algorithm more simple and is conducive to the real-time computing.

Assuming that the contact deformation is calculated once every t s after collision is detected and x_i $(i \geqq 1)$ is defined as the ith contact deformation, $\dot{x}_i$ and $\ddot{x}_i$ are given by

$$\dot{x}_i = \frac{x_i - x_{i-1}}{t} \quad (i \geqq 2) \tag{5}$$

$$\ddot{x}_i = \dot{x}_i - \dot{x}_{i-1} = \frac{\left(x_i - 2x_{i-1} + x_{i-2}\right)}{t^2} \quad (i \geqq 3), \tag{6}$$

where (6) shows that it needs to be calculated three times to get the virtual contact force f_{cj} at least.

5. Stable Grasping Conditions

For the real dexterous robot hand, the stable grasping conditions are given by the following [18, 19].

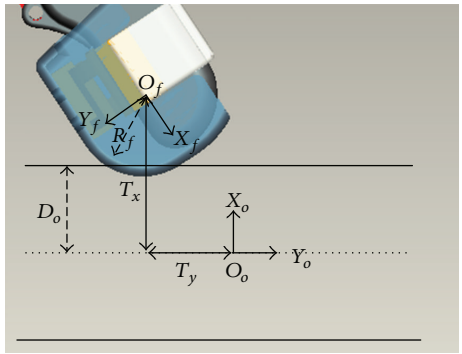

Figure 3: The relative position of the fingertip and target object.

(a) The contact points should be symmetrically distributed in a plane and the central axis of the target object should be in this plane and the external force (the external force does not include the grasping force in this paper) should be as far as possible in this plane.

(b) The grasping force and torque acting on the target object should keep balance with the external force and torque.

(c) The friction cone constraint should be met:

$$\sqrt{F_{cjx}^2 + F_{cjy}^2} \leq \mu_j F_{cjz}, \tag{7}$$

where μ_j represents the maximum static friction coefficient of the jth contact point; F_{cjx}, F_{cjy}, and F_{cjz} represent the component force in x, y, and z coordinate axis, respectively.

Condition (a) can be met by grasp position planning which is decided by the operator. Conditions (b) and (c) should be modified so that they can be implemented in the virtual environment.

Apparently, the virtual contact force calculated in Section 4 dose not include the contact friction. But the contact friction can be calculated by using the method proposed in Section 4. Considering the contact friction, the force analysis of the contact between the fingertip and target object is shown in Figure 4, where F_{cj} represents the contact force including the contact friction; f_j represents the contact friction in the jth contact point; f_{cj} is defined in (1). The coordinate system O_{f0} is established in the joint axis of the fingertip and can be transformed into the coordinate system O_f by translation. Therefore, θ_j or the angle between f_{cj} and F_{cj} can be calculated by the transformation matrix $\mathbf{T}$ which is defined in (3). Then, (7) can be modified as

$$\tan\theta_j \leq u_j. \tag{8}$$

In the virtual environment, the stable grasping conditions are given by the following:

(a′) the same as condition (a),

(b′) $\sum_{j=1}^{n} F_{cj} = F_0$, where n represents the number of the contact points, F_0 represents the external force acting on the target object, and $F_{cj} = f_{cj}/\cos\theta_j$,

(c′) $\tan\theta_j \leqq \mu_j$.

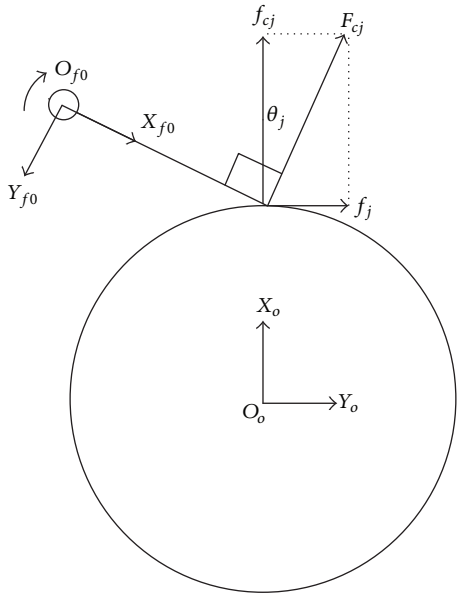

Figure 4: Force analysis of the contact point.

6. Grasping in the Virtual Environment

The grasping task in the virtual environment is divided into six segments: *move*, *approach*, *grasp*, *translate*, *place*, and *release* [11]. When the stable grasping conditions in the virtual environment are met, the virtual target object can be grasped stably by the VDR hand. When the conditions are not met any more, the virtual target object is released.

OIV is an object-oriented 3D graphics toolkit and the virtual scene is tree-structure as shown in Figure 2. Thus, the child nodes inherit the properties of the parent nodes, which means that the motion of the parent nodes can influence the motion of the child nodes. So, the target object can be easily moved or released by the VDR hand. When the stable grasping conditions in the virtual environment are met, the VDR hand and the target object are moved together by operating the "Root" node in Figure 2, which means that the VDR hand is moving the target object. And when the conditions are not met any more, the VDR hand and the target object are separated by operating the "Palm" node in Figure 2, which means that the virtual target object is released. This method of operation does not destroy the relationship between the nodes, so that the collision detection still works when the target object is operated.

7. Validation and Analysis

7.1. Validation and Analysis of the Virtual Force. Let $t = 0.01$ s, $m = 0.1$ kg, $b = 1000$ N/(m/s), $k = 10,000$ N/m, $\mu_j = 0.55$, and $R_f = 6$ mm in this paper. And the target object is a sphere in the virtual environment. The radius of the sphere is 10 mm and thus $D_o = 10$ mm.

Let the middle finger of the VDR hand approach the sphere. And the angular displacements of the finger joints and the calculated virtual force are recorded in the course of contact.

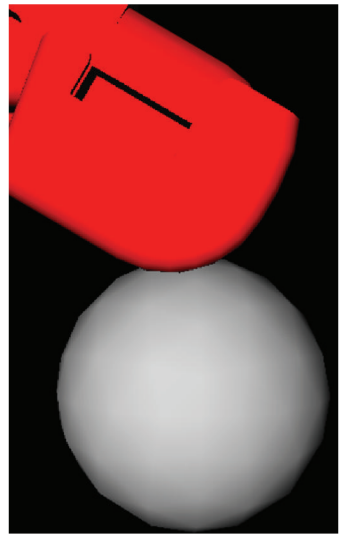

Figure 5: Collision in the virtual environment.

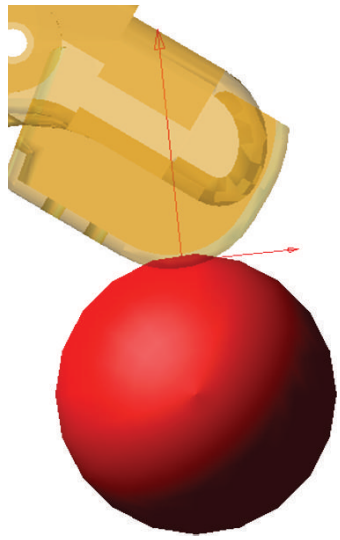

Figure 6: Collision in ADAMS.

In order to validate the calculated virtual force efficiently, the dexterous robot hand model, being the same with the VDR hand, is built up in ADAMS software (the force calculated by ADAMS is approximately equal to the real force) at first. Then the same sphere in ADAMS is placed in the same position with the virtual sphere. Thirdly, the middle finger in ADAMS moves according to the recorded angular displacements. Finally, the contact force calculated by ADAMS can be got in the same conditions with the virtual environment. Figures 5 and 6 are, respectively, the collision in the virtual environment and ADAMS.

Process the force data obtained from ADAMS and virtual environment with MATLAB. And the results are shown in Figures 7 and 8, where Figure 7 is the contrast figure of the calculated virtual force and the force calculated by ADAMS without including the contact friction; Figure 8 is the contrast figure including the contact friction.

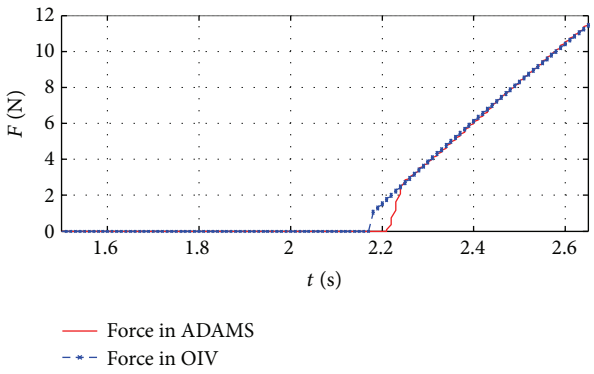

Figure 7: The calculated virtual force and force calculated by ADAMS without including friction.

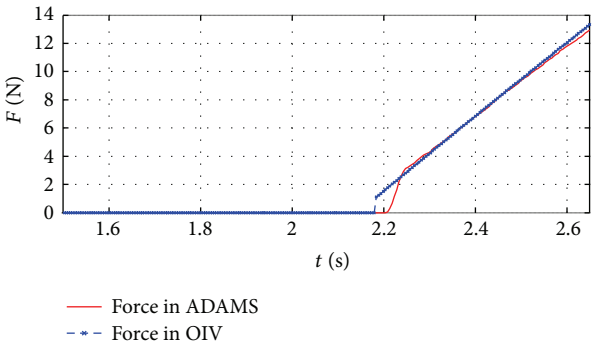

Figure 8: The calculated virtual force and force calculated by ADAMS including friction.

Figures 7 and 8 show that the average error between the calculated virtual force and the force calculated by ADAMS is less than 0.2 N. This demonstrates the viability of the proposed algorithm for calculating virtual force.

7.2. Validation and Analysis of the Stable Grasping. The user operation interface is shown in Figure 9. And the interface consists of a data window and a VR window. The operator can observe the angular displacements of the fingers, the calculated contact force, and the grasping status in the data window. The VR window can display the motion of the VDR hand in real time. The friendly user interface can reduce the stress of the operator.

In the experiment, the operator uses a data glove to control the VDR hand to grasp a cylinder in the virtual environment. When a part of the VDR hand collides with the cylinder, this part turns red. It can alert the operator in time to avoid excessive contact force. Figure 10 shows that the fingertips of the thumb, middle finger, and index finger have collided with the cylinder. And the operator can adjust the motion of the three fingers slightly according to the data displayed in the data window. When the stable grasping conditions are met, the operator can move the cylinder using the VDR hand in the virtual environment, as shown in Figure 11.

The cylinder is released when the operator adjusts the motion of the fingers so that the conditions are not met, as shown in Figures 12 and 13.

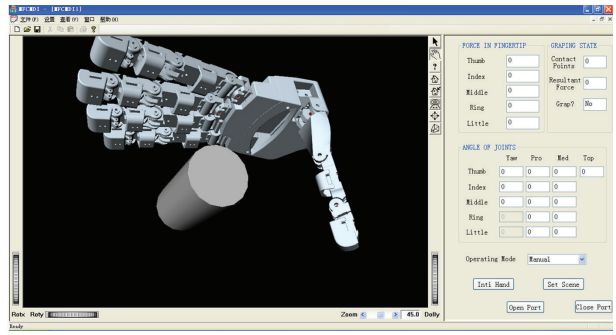

Figure 9: User operation interface.

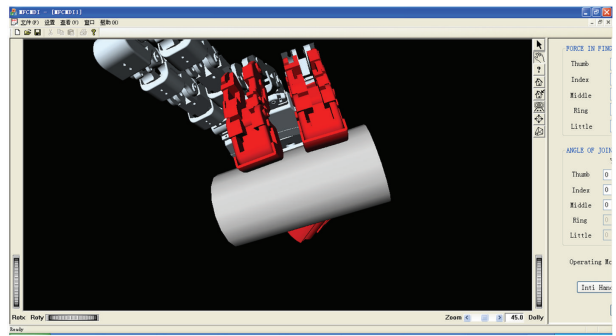

Figure 10: Three fingers collide with the cylinder.

In the course of the validation, the VDR hand completed the grasp, translation, and release operations. And the results demonstrate the viability of the proposed grasping algorithm. In addition, the proposed method of collision detection is proved to be real time and the modified stable grasping conditions can be implemented in the virtual environment.

8. Conclusion

In order to solve the problems of grasping in the virtual environment, the functions provided by OIV are adopted. SoCollisionManager and OBBs are used for the real-time collision detection. The virtual force is related to the contact deformation according to the dynamical equation of the contact force. For calculating the contact deformation, this paper uses the transformation matrix **T** which can be obtained by the functions of SoGetMatrixAction. Furthermore, the contact friction in the virtual environment can also be calculated by the transformation matrix. Considering the limitation of the virtual force calculation in the virtual environment, the traditional stable grasping conditions are modified so that the operator can grasp the virtual object stably according to the modified conditions. The tree-structure virtual scene makes the grasping more flexible.

However, there are still some problems that need to be solved. For example, this paper only calculates and verifies the virtual contact force in the curved surface of the fingertip. And the following research should focus on the virtual contact force calculation in the other parts of the fingers for the power grasping.

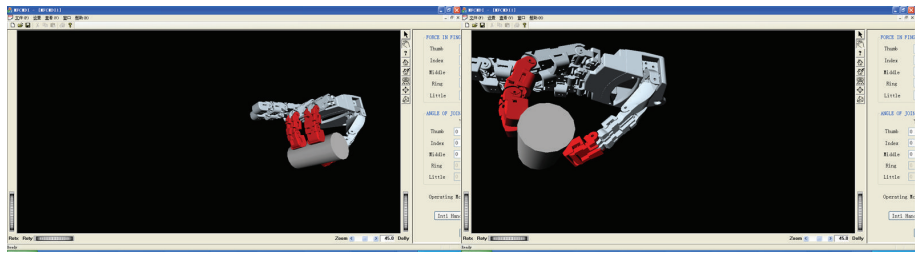

Figure 11: Operator moves the cylinder using the VDR hand.

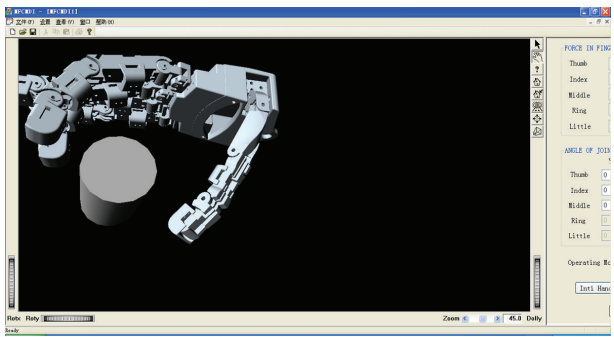

Figure 12: The stable grasping conditions are not met.

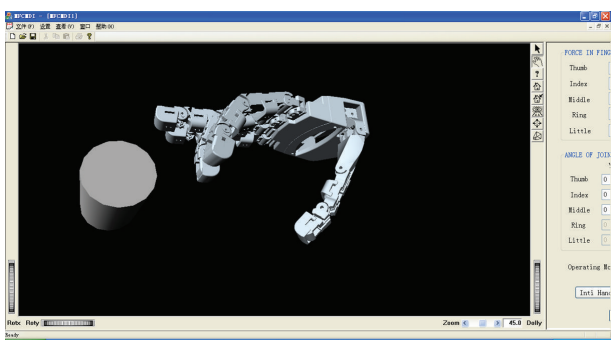

Figure 13: The cylinder is released.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant no. 51105196) and Natural Science Foundation of Jiangsu Province (Grant no. BK2011733).

References

[1] C.-P. Kuan and K.-Y. Young, "VR-based teleoperation for robot compliance control," *Journal of Intelligent and Robotic Systems*, vol. 30, no. 4, pp. 377–398, 2001.

[2] H. Liu, K. Sun, Z. W. Xie et al., "Research on the satellite on-orbit self-servicing testbed," *Advanced Robotics*, vol. 22, no. 2-3, pp. 299–317, 2008.

[3] J. Zainan, L. Hong, W. Jie, and H. Jianbin, "Virtual reality-based teleoperation with robustness against modeling errors," *Chinese Journal of Aeronautics*, vol. 22, no. 3, pp. 325–333, 2009.

[4] K. Belghith, F. Kabanza, and L. Hartman, "Using a randomized path planner to generate 3D task demonstrations of robot operations," in *Proceedings of the IEEE International Conference on Autonomous and Intelligent Systems (AIS '10)*, pp. 1–6, Póvoa de Varzim, Portugal, June 2010.

[5] J. Tang, C. Carignan, and S. Gattewar, "Virtual environment for robotic tele-rehabilitation," in *Proceedings of the IEEE 9th International Conference on Rehabilitation Robotics (ICORR '05)*, pp. 365–370, Chicago, Ill, USA, July 2005.

[6] D. Jack, R. Boian, A. S. Merians et al., "Virtual reality-enhanced stroke rehabilitation," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 9, no. 3, pp. 308–318, 2001.

[7] J. Arata, H. Takahashi, P. Pitakwatchara et al., "A remote surgery experiment between Japan and Thailand over Internet using a low latency CODEC system," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '07)*, pp. 953–959, Roma, Italy, April 2007.

[8] H. Kawasaki, K. Nakayama, and G. Parker, "Teaching for multi-fingered robots based on motion intention in virtual reality," in *Proceedings of the 26th Annual Conference of the IEEE Electronics Society (IECON '00)*, pp. 428–433, Nagoya, Japan, October 2000.

[9] H. Wan, Y. Luo, S. Gao, and Q. Peng, "Realistic virtual hand modeling with applications for virtual grasping," in *Proceedings of the ACM SIGGRAPH International Conference on Virtual Reality Continuum and Its Applications in Industry (VRCAI '04)*, pp. 81–87, New York, NY, USA, June 2004.

[10] C. R. Cameron, L. W. DiValentin, R. Manaktala et al., "Hand tracking and visualization in a virtual reality simulation," in *Proceedings of the IEEE Systems and Information Engineering Design Symposium (SIEDS '11)*, pp. 127–132, University of Virginia, April 2011.

[11] H. Kawasaki, K. Nakayama, T. Mouri, and S. Ito, "Virtual teaching based on hand manipulability for multi-fingered robots," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '01)*, pp. 1388–1393, Seoul, Republic of Korea, May 2001.

[12] H. Wan, F. Chen, and X. Han, "A 4-layer flexible virtual hand model for haptic interaction," in *Proceedings of the IEEE International Conference on Virtual Environments, Human-Computer Interfaces, and Measurements Systems (VECIMS '09)*, pp. 185–190, Hong Kong, May 2009.

[13] J. W. Zhou, *Research of Tele-Operation Technology Based on Force Modification*, Huazhong University of Science & Technology, Wuhan, China, 2012, (Chinese).

[14] T. Ullmann and J. Sauer, "Intuitive virtual grasping for non haptic environments," in *Proceeding of the 8th Pacific Conference on Computer Graphics and Applications*, pp. 373–457, Hong Kong, October 2000.

[15] S. Gottschalk, M. C. Lin, and D. Manocha, "OBBTree: a hierarchical structure for rapid interference detection," in *Proceeding of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, pp. 171–180, New York, NY, USA, August 1996.

[16] Q. Song, L. Z. Song, and F. J. Kang, "Research and application of bounding box collision detection technique," *Computer Engineering and Applications*, vol. 45, no. 24, pp. 238–240, 2009 (Chinese).

[17] Z. N. Jiang, *Research on Space Robot Teleoperation Based on Virtual Reality and Local Autonomy*, Harbin Institute of Technology, Harbin, China, 2009, (Chinese).

[18] G. Baud-Bovy and J. F. Soechting, "Two virtual fingers in the control of the tripod grasp," *Journal of Neurophysiology*, vol. 86, no. 2, pp. 604–615, 2001.

[19] J. Kerr and B. Roth, "Analysis of multi-fingered hands," *International Journal of Robotics Research*, vol. 4, no. 4, pp. 3–17, 1986.