Hand Gesture Based Robot Control System Using Leap Motion

Sunjie Chen[1], Hongbin Ma[1,2(✉)], Chenguang Yang[3], and Mengyin Fu[1,2]

[1] School of Automation, Beijing Institute of Technology,
Beijing 100081, People's Republic of China
mathmhb@gmail.com
[2] State Key Lab of Intelligent Control and Decision of Complex Systems,
Beijing Institute of Technology, Beijing 100081, People's Republic of China
[3] School of Computing and Mathematics, Plymouth University,
Plymouth PL4 8AA, UK
cgyang82@gmail.com

Abstract. Gesture based human-robot interface is a highly efficient robot control strategy for its simple operation and high availability. This paper develops a hand gesture based robot control system using Leap Motion. The process that the robot responds to human's hand gesture contains noise suppression, coordinate transformation and inverse kinematics. A Client/Server structured robot control system is developed, which provides the function of controlling virtual universal robot UR10 with hand gesture. Finally, experimental results demonstrate that the system is effective and practical.

Keywords: Leap motion · Gesture · Robot · Human-robot Interface · Simulation

1 Introduction

Nowadays, robots are so popular that programming seems not to be efficient enough to control them. That is, we need to create a more practical easy-to-use robot control system. Voice recognition analyzes voice signal with models such as hidden Markovian models (HMMs) such that the voice can be interpreted as text information by the robots. However, there has a large amount of dialect so that a big database needs to be built. Analyzing meaning of voice must refer to specific context, so it will be difficult as well. EEG samples physiological electric signals when brain is active. However, the approach of using EEG to reflect the user's requirement and intention is not robust or reliable enough. Gesture recognition samples the characteristic of hand action to reflect the user's requirement or

This work was partially supported by the National Natural Science Foundation in China (NSFC) under Grants 61473038 and 61473120. Also supported in part by State Key Laboratory of Robotics and System (HIT) SKLRS-2014-MS-05.

H. Liu et al. (Eds.): ICIRA 2015, Part I, LNAI 9244, pp. 581–591, 2015.
DOI: 10.1007/978-3-319-22879-2_53

intention. Hence, gesture based control system is low-cost, highly flexible and efficient.

Gesture tracking is an important function of gesture based human-robot interaction. There are two methods for gesture tracking research, one relies on vision, and the other one bases on data glove. The former one can be affected by the light, complexion, and it is difficult to capture detail. The latter one may depend more on the device [1].

Low-cost sensors, the Kinect and the Xtion Pro, make it convenient to achieve gesture recognition. However, they concentrate more on body action. The Leap Motion is small in size, and has high precision and low power dissipation. It can detect details of hand action [2].

The Leap Motion has two cameras inside, which can take photos from different directions to obtain hand action information in 3D space. The detection range of Leap Motion is between 25mm and 600mm upon the sensor. The shape of the detection space is similar to an inverted cone [3]. Figure 1 depicts the coordinate system for the Leap Motion.

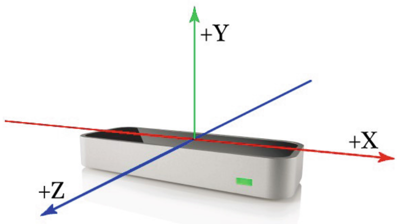

Fig. 1. Coordinate system of Leap Motion

Recently, Xu *et al.* [4] put forward a method of remote interaction through detecting the position of palm based on the Leap Motion, and mapping the physical space and information space. As one typical application of the Leap Motion, a MIDI controller using the Leap Motion was reported in the literature [5] and it is also well known that many games can be played with motion sensors such as Kinect. With the fast development of motion sensing technology, the Leap Motion can serve as an excellent replacement of Kinect for desktop applications. After extensive testing, one may easily draw a conclusion that the motion sensing technology is more practical and more attractive than traditional way. As to the dynamic hand gesture recognition, Wang *et al.* [6] proposed an effective method which can recognize dynamic hand gesture by analyzing the information of motion trajectory captured by the Leap Motion.

This paper develops a hand gesture based robot control system using the Leap Motion. The system first obtains gesture information using the Leap Motion. Then, with the aid of algorithms, it analyzes data and understands robot control signals from the gesture information. Finally, the system achieves the goal of controlling the robot with gesture in real time.

2 Hardware and Software

The software development with the Leap Motion should base on the official SDK and the driver. When it is powered on, the Leap Motion sends hand action information periodically. Every package information is called a frame [7]. The sensor will assign these information with an ID. With the ID, the user may call functions such as Frame::hand(), Frame::finger() to check any object's information.

With C++ compiler of Microsoft Visual Studio 2010 on Windows or Gnu C/C++ Compiler on Linux, we can implement functions of reading and analyzing data from Leap Motion.

V-REP is a robot simulator with integrated development environment. Each object/model can be individually controlled via an embedded script, a plugin, a ROS node, a remote API client, or a custom solution [8].

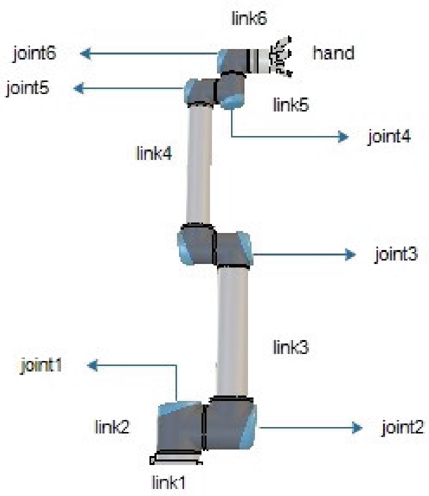

Fig. 2. UR10&BarrettHand

UR10 robot is produced by Universal Robots. Unlike traditional industrial robot, UR10 is light and cheap. It weights 28kg, and can load 10kg [9]. Its working range is 1300mm.(see Figure 2)

The BarrettHand is a multi-fingered programmable grasper with the dexterity to secure target objects of different sizes, shapes and orientations. Even with its low weight (980 g) and super compact (25mm) base, it is totally self-contained [10].

3 Control System

To achieve the goal of controlling robot with gesture, this paper develops a Client/Server structured remote human-robot interface control software [11].

First, the client sends a signal to the server to inform that the client is prepared to accept next action command. Then, the server gets the signal, and receives current frame information from the Leap Motion. The server finishes the work about noise suppression, coordinate transformation, inverse kinematics and packs the control signals sending to V-REP as well. Finally, V-REP runs the simulation. Figure 3 shows the process of the loop.

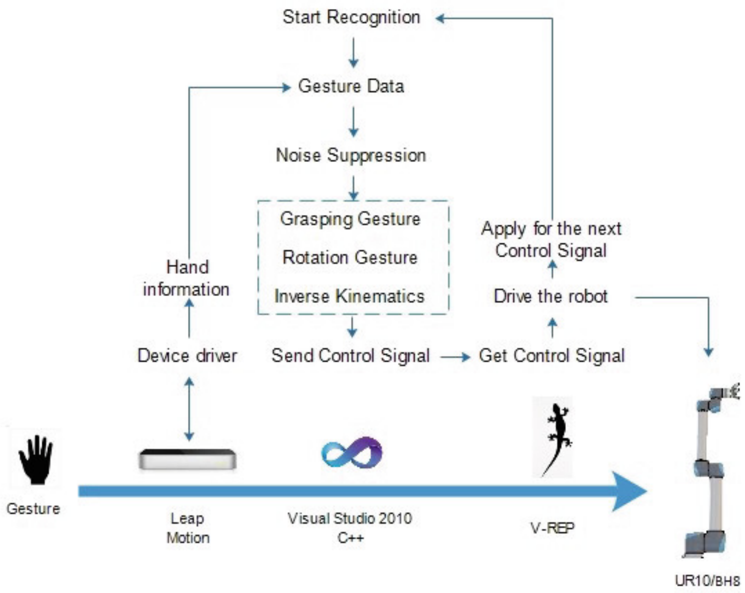

Fig. 3. Control system structure

3.1 Noise Suppression

The data, including the position of palm, direction vector of finger and normal vector of palm, will be handled by the server. The precision of the Leap Motion can be 0.01mm in theory. Actually, there are some destabilizing factors which will affect the precision, such as shaking of hand, magnetocaloric effect, calculating, etc. This paper adapts a speed-based low-pass filter [12] to eliminate noise. The point of this method is changing the cut-off frequency of low-pass filter with the velocity of palm. The filter can be mathematically described by

$$\hat{X}_i = \alpha X_i + (1 - \alpha)\hat{X}_{i-1} \tag{1}$$

where X_i is a vector containing the coordinates and direction information given by Leap Motion, $\hat{X}_i$ is the vector after filter, and α is a factor which can be calculated by

$$\alpha_i = \frac{1}{1 + \tau_i/T_i} \tag{2}$$

$$\tau_i = \frac{1}{2\pi f_{ci}} \tag{3}$$

where T_i is the period of updating the data, τ_i is a time constant, f_{ci} is cut-off frequency, which can be determined by

$$f_{ci} = f_{cmin} + \beta \mid \hat{V}_i \mid \tag{4}$$

where $\hat{V}_i$ is a derivative of $\hat{X}_i$, representing the velocity of palm. Based upon experience, make

$$f_{cmin} = 1HZ, \beta = 0.5 \tag{5}$$

3.2 Grasping Gesture

We add a hand on the terminal of robot, which can execute grasping task. We preset every joint of finger which can rotate between $0 - 120°$. When the hand is open, joint rotation angle is defined as 0. On the contrary, when the hand is closed, the angle is 120°. We use a coefficient μ to describe the level of grasping. With the API of the Leap Motion SDK, the parameter μ can be given by `hand::grabStrength()`, and the finger joint angle is 120μ.

3.3 Rotation Gesture

To describe the rotation gesture mathematically, we build the coordinate system (Figure 4) for the hand, just the same as Leap Motion. Therefore, the problem of hand rotation is equivalent to the problem of coordinate rotation. For example, the coordinate system for Leap Motion is named as frame A, and for hand it is named as frame B. Starting with the frame B coincident with frame A. Rotate frame B first about $\hat{Y}_B$ by an angle $\alpha(\alpha \in [0, 360°])$, then about $\hat{X}_B$ with an angle $\beta(\beta \in [-90°, 90°])$, finally, about $\hat{Z}_B$ by an angle $\gamma(\gamma \in [0, 360°])$. We know the orientation of frame B relative to frame A. So if we can obtain the Euler angles of these rotation process, we know how to control joints 4, 5, 6 to reappear the rotation gesture.

Because all rotations occur about axes about frame B, the rotation matrix is

$${}^{A}_{B}R = R_Y(\alpha)R_X(\beta)R_Z(\gamma) = \begin{pmatrix} c\alpha & -s\alpha & 0 \\ s\alpha & c\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} c\beta & 0 & s\beta \\ 0 & 1 & 0 \\ -s\beta & 0 & c\beta \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & c\gamma & -s\gamma \\ 0 & s\gamma & c\gamma \end{pmatrix} \tag{6}$$

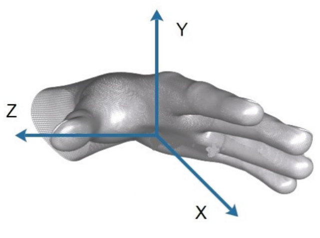

Fig. 4. Coordinate system for hand

$$ {}^A_BR = \begin{pmatrix} s\alpha s\beta s\gamma + c\alpha c\gamma & s\alpha s\beta c\gamma - c\beta s\gamma & s\alpha c\beta \\ c\beta s\gamma & c\beta c\gamma & -s\beta \\ c\alpha s\beta s\gamma - s\alpha c\gamma & c\alpha s\beta s\gamma + s\alpha s\gamma & c\alpha c\beta \end{pmatrix} \tag{7} $$

where ${}^A_BR$ is a rotation matrix that specifies the relationship between coordinate system A and B. And $c\alpha$ means cosine of angle α and $s\alpha$ means sine of angle α.

According to the definition of rotation matrix, we have

$$ {}^A_BR = \left({}^A\hat{X}_B \ {}^A\hat{Y}_B \ {}^A\hat{Z}_B \right) \tag{8} $$

where the unit vectors giving the principal directions of coordinate system B, when written in term of coordinate system A, are called ${}^A\hat{X}_B, {}^A\hat{Y}_B, {}^A\hat{Z}_B$. We can obtain the normal vector of hand with the function Hand::palmNormal(), and ${}^A\hat{Y}_B$ is in the opposite direction with the normal vector. We also can get the vector from palm to finger with the function Hand::direction(), and ${}^A\hat{Z}_B$ is in the opposite direction with it as well. What's more, we can obtain that

$$ {}^A\hat{X}_B = {}^A\hat{Y}_B \times {}^A\hat{Z}_B \tag{9} $$

when the coordinate system B employs a right-handed Cartesian coordinate system.

Assuming

$$ {}^A_BR = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \tag{10} $$

Now, all the things have been prepared. First, we can get angle β which satisfies

$$ -s\beta = r_{23} \tag{11} $$

Then, we can get angle α which satisfies

$$ s\alpha c\beta = r_{13} \tag{12} $$

$$ c\alpha c\beta = r_{33} \tag{13} $$

Finally, we can get angle γ which satisfies

$$ c\beta s\gamma = r_{21} \tag{14} $$

$$ c\beta c\gamma = r_{22} \tag{15} $$

We control joint 4 to rotate angle β, make joint 5 rotate angle α, and make joint 6 rotate angle γ. Reappearing the rotation gesture is achieved.

3.4 Inverse Kinematics

Every frame of the position of the palm can be read when the hand is tracked. We use the palm position information to control joints 1, 2, 3 of robot with inverse kinematics. At the beginning, we build coordinate system for joints 1,

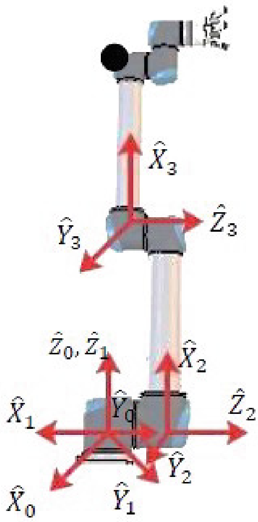

Fig. 5. Coordinate system for UR10

2, 3 (Figure 5). The position of black point, denoted by (x, y, z), is given by Hand::palmPosition().

The coordinate system for robot is not coincident with that for the Leap Motion, hence

$$x = palmPosition()[2]/150 \tag{16}$$

$$y = palmPosition()[0]/150 \tag{17}$$

$$z = palmPosition()[1]/150 - 0.4 \tag{18}$$

Table 1 shows the link parameters (Denavit-Hartenberg parameters), whose definitions are given in Table 2 [13], for UR10. In this paper, we can ignore the length between $\hat{X}_1$ and $\hat{X}_2$, as well as the length between $\hat{X}_2$ and $\hat{X}_3$.

Table 1. Link parameters for UR10

i	α_{i-1}	a_{i-1}	d_i	θ_i
1	0	0	0	θ_1
2	$-90°$	0	0	θ_2
3	0	L_1	0	θ_3

We compute each of the link transformations:

$$ {}^0_1T = \begin{pmatrix} c_1 & -s_1 & 0 & 0 \\ s_1 & c_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{19}$$

Table 2. Definitions of symbols

Symbol	Definition
a_i	the distance from $\hat{Z}_i$ to $\hat{Z}_{i+1}$ measured along $\hat{X}_i$
α_i	the angle from $\hat{Z}_i$ to $\hat{Z}_{i+1}$ measured along $\hat{X}_i$
d_i	the distance from $\hat{X}_{i-1}$ to $\hat{X}_i$ measured along $\hat{Z}_i$
θ_i	the angle from $\hat{X}_{i-1}$ to $\hat{X}_i$ measured along $\hat{Z}_i$
L_1	the length of link3
L_2	the length of link4

$$ {}^{1}_{2}T = \begin{pmatrix} c_2 & -s_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ s_2 & c_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{20} $$

$$ {}^{2}_{3}T = \begin{pmatrix} c_3 & -s_3 & 0 & L_1 \\ s_3 & c_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{21} $$

where c_1 (or s_1) means cosine (or sine) of angle θ_1, and c_{12} means $cos_1 cos_2$.

Then,

$$ {}^{0}_{3}T = {}^{0}_{1}T \quad {}^{1}_{2}T \quad {}^{2}_{3}T \begin{pmatrix} c_{123} - c_{13}s_2 & -c_{12}s_3 - c_{13}s_2 & -s_1 & L_1 c_{12} \\ s_1 c_{23} - s_{123} & -s_{13}c_2 - s_{123} & c_1 & L_1 s_1 c_2 \\ s_2 c_3 + c_2 s_3 & -s_{23} + c_{23} & 0 & L_1 s_2 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{22} $$

The position of black point relative to frame 3 is

$$ {}^{3}P = \begin{pmatrix} L_2 \\ 0 \\ 0 \end{pmatrix} \tag{23} $$

The position of black point relative to frame 0 is

$$ {}^{0}P = \begin{pmatrix} X \\ y \\ z \end{pmatrix} \tag{24} $$

Then,

$$ \begin{pmatrix} {}^{0}P \\ 1 \end{pmatrix} = {}^{0}_{1}T \, {}^{1}_{2}T \, {}^{2}_{3}T \begin{pmatrix} {}^{3}P \\ 1 \end{pmatrix} \tag{25} $$

From equation (25), we can get ${}^{0}_{3}T$, assuming

$$ {}^{2}_{3}T = \begin{pmatrix} r_{11} & r_{12} & r_{13} & l_1 \\ r_{21} & r_{22} & r_{23} & l_2 \\ r_{31} & r_{32} & r_{33} & l_3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{26} $$

Now it is easy to obtain the value of θ_i $(i = 1, 2, 3)$.

4 Experiment

An experiment was designed to test the system performance.The user first puts his hand upon the Leap Motion, and then can do the gestures such as translation, grasping and rotation. The results of simulation shown in Figure 6 demonstrate that the system can respond correctly and quickly to the gesture, which means that the system is efficient and practical.

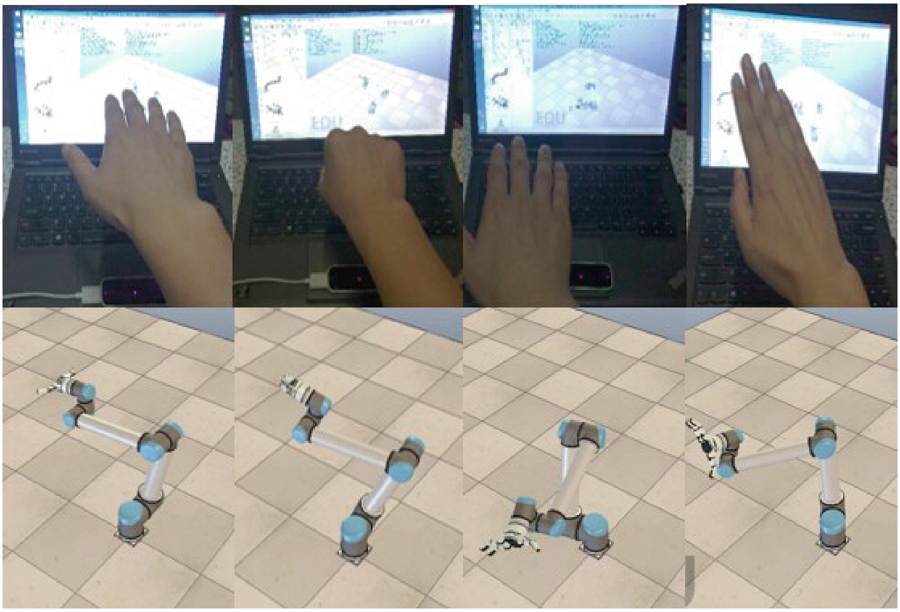

Fig. 6. Experiment result

Then, we test the accuracy of the system. The workspace of robot is a sphere with radius $1.4470m$. When the user conducts an action, the response time of system is limited in $0.1s$. The user can put his hand at any position upon Leap Motion. We get the position $(29.7574\text{mm}, 175.155\text{mm}, 40.302\text{mm})$ instantly. In theory, the terminal position of robot should be $(0.2687\text{m}, 0.1983\text{m}, 0.7677\text{m})$, and the real position is $(0.2498\text{m}, 0.3629\text{m}, 0.7573\text{m})$. The open loop error is 4.65%. Do more experiments, the average open loop error is limited in 5%. That is the system has high precision.

Finally, we add a table into the scene, and put a cup on the table. We select five people to experience the system by grasping the cup with the Leap Motion (see Figure 7). The testers' user experiences and feedbacks were recorded. Results demonstrate that people are satisfied the system. Users also suggest that grasping gesture should be adapted to different shape things.

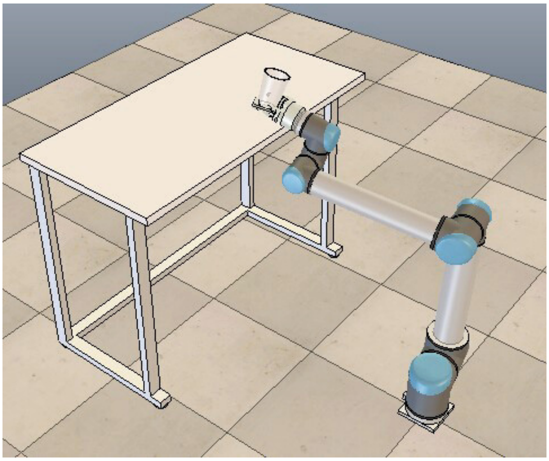

Fig. 7. Grasping the cup

5 Result

This paper has developed a hand gesture based robot control system using the Leap Motion. The system contains noise suppression, coordinate transformation and inverse kinematics, achieving the goal of controlling a robot with hand gesture. The system has advantages in terms of simple operation, high flexibility and efficiency. This robot control system does not have any complex menu or buttons. This system considers more about users experience in their daily life, so the control gesture desired will be natural and reasonable. It can be used in tele medicine, family nursing care, etc. In the future work, we can continue to improve the stability of the system. Meanwhile, we hope to improve the performance of the robot, increase the number of robots, and recognize more than one hands to control different robots, respectively.

References

1. Pan, J.J., Xu, K.: Leap motion based 3D. gesture. China Science Papaer **10**(2), 207–212 (2015)
2. Jiang, Y.C.: Menacing motion-sensing technology, different leap motion. PC. Fan **11**, 32–33 (2013)
3. The principle of leap motion. http://www.3dfocus.com.cn/news/show-440.html
4. Xu, C.B., Zhou, M.Q., Shen, J.C., Luo, Y.L., Wu, Z.K.: A interaction technique based on leap motion. Journal of Electronics & Information Technology **37**(2), 353–359 (2015)
5. Pan, S.Y.: Design and feature discussion of MIDI. controller based on leap motion. Science & Technology for Chinas Mass Media **10**, 128–129 (2014)

6. Wang, Q.Q., Xu, Y.R., Bai, X., Xu, D., Chen, Y.L., Wu, X.Y.: Dynamic gesture recognition using 3D trajectory. In: Proceedings of 2014 4th IEEE International Conference on Information Science and Technology, Shenzhen, Guanzhou, China, pp. 598–601 (April 2014)
7. Elons, AS., Ahmed, M., Shedid, H., Tolba, M.F.: Arabic sign language recognition using leap motion sensor. In: 2014 9th International Conference on Computer Engineering & Systems (ICCES), Cairos, pp. 368–373 (December 2014)
8. V-rep introduction. http://www.v-rep.eu/
9. UR10 introduction. http://news.cmol.com/2013/0530/33267.html
10. Barretthand introduction. http://wiki.ros.org/Robots/BarrettHand
11. Qian, K., Jie, N., Hong, Y.: Developing a gesture based remote human-robot interaction system using Kinect. International Journal of Smart Home **7**(4), 203–208 (2013)
12. Casiez, G., Roussel, N., Vogel, D.: 1 filter: a simple speed-based low-pass filter for noisy input in interactive systems. In: Proceedings of the 2012 ACM Annual Conference on Human Factors in Computing Systems, Austin, TX, USA, pp. 2527–2530 (May 2012)
13. Craig, J.J.: Introduction to Rbotics: Mechanics and Control, 3rd edn. China Machine Press, Beijing (2006)