Human-Robot Gesture Interaction: From Simulation Verification to Rapid Implementation

Xingqiang Jian[1,3,a], Xing Ma[1,2,b], Chunyang Mu[1,3,c,*] and Guohe Tian[3]
[1]The Key Laboratory of Intelligent Information and Big Data Processing of Ningxia Province, North Minzu University, Yinchuan, China
[2]School of Electrical and Information Engineering, North Minzu University, Yinchuan, China
[3]College of Mechatronic Engineering, North Minzu University, Yinchuan, China
a. 1176979724@qq.com, b. maxingsky@126.com, c. muchunyang@126.com

Keywords: Gesture Interaction, Leap Motion, Robot Control, Simulated Robot, ROS Toolbox.

Abstract: This paper proposes to use a Leap Motion controller to capture human gestures and translate them into control commands to control the robot. In order to verify the feasibility of the gesture interactive control algorithm in advance, simulation verification was first performed in the MATLAB simulation environment. Then we proposed a specific implementation scheme, which can deploy the control algorithm to an actual robot through ROS Toolbox. The results showed that the interactive control of the simulated robot can be quickly implemented in the simulation environment, some errors of the gesture interactive control algorithm can be avoided in advance, and the direct natural operation of the robot by gesture interaction can be easily developed and effectively implemented.

1. Introduction

Today, robots have become the most beneficial helpers for humans to complete social practices. But since the invention of robots, robots have rarely been able to accomplish human tasks in a very short time. In the process of using the robot, the operator generally needs to formulate the corresponding task actions for the robot through complex program editing, which often requires a lot of time and effort. The programming of robot is often limited to professionals, which greatly limits the use of robots. Therefore, the research on the interaction between robot and humans is expected to enable the robotic system to form an effective system in contact with humans. Robot will need to be able to obtain sufficient information from their environment, detect people, and establish connections with people.

The interactive control method of the robot has gradually changed from the traditional human-computer interaction method such as keyboard, mouse, and touch screen to a more natural human-computer interaction method. For example, gesture recognition and interaction based on visual[1], interaction based on speech processing[2], interaction based on recognition of human intent based on brain waves and EMG signals[3], and interaction based on torque sensor， which have greatly reduced the threshold for using robots[4].These interactions allowing non-professionals to control the robot in a simpler, more convenient and direct way.

In robotics, gesture recognition is used to control robots in a simple way with hand signals. Basically a device with a machine vision system captures hand positions and assigns an action to those movements that have a certain pattern[5].When using image processing technology, it is difficult to implement because different hand recognition schemes are needed, such as pattern recognition, tracking, and color recognition[6].While Leap Motion Controller is a very popular motion sensor in the field of gesture interaction. It can obtain motion data and gestures of human hands very well. And eliminate the shortcomings of image processing technology and increase the speed of real-time work.As shown in the Figure 1, Leap Motion can interact with objects in the virtual environment in real time.

Figure 1: Leap Motion interactive with the Virtual environment.

Human-computer interaction is a technology that realizes human-computer dialogue through computer input and output devices in an effective way. We believe that in the development of gesture interaction algorithms, it is necessary to introduce simulation technology to verify gesture interaction in advance to maintain its effectiveness. Therefore, this paper presents a more simple and direct way to control the robot, we use the advantages of MATLAB programming efficiency to achieve the development and simulation testing of interactive control systems, so that we can find and correct errors in the simulation in time.

2. Structure and Principle

In this section, we will introduce the process of constructing a gesture interactive control system in the MATLAB development environment.

Firstly, the gesture and motion data of the operator are collected from the Leap Motion controller, and the gestures are defined and classified by means of pattern recognition. The recognized gestures and data are used to directly interact with the robot and control the robot's motion and gripper state. The implementation of gesture interaction is first verified in the simulation and then actually deployed through the ROS toolbox[7], which will be described in detail in Part III. The whole process shows as below Figure 2.

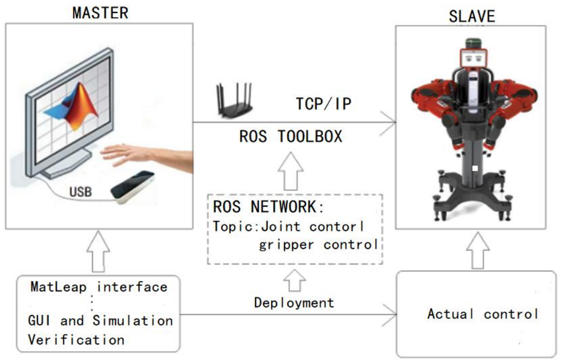


Figure 2: Software architecture for implement interaction method.

In Figure 2, we can also replace the data of gesture information based on leap motion with other methods, such as monocular vision or binocular depth camera, for example, Kinect. Slave can be other robots, as long as it supports the ROS system. This means that as long as the gesture interaction algorithms verified in simulation can be deployed to different robot systems, this reflects the universality of interactive algorithm development. If we change the robot system, we can implement the same interaction algorithm to control other robots.

In this paper, we use Leap Motion to obtain gesture information, first verify the reliability of interactive control in the simulation, and then carry out specific deployment through the ROS toolbox.

3. System Development

The entire development process is divided into three parts. First, the data obtained through the Leap Motion is obtained in MATLAB. Then build a robot model and perform interactive control tests and result display on the GUI interface through gesture definition and motion matching. Finally, the gesture-translated commands are issued directly to the robot through the ROS toolbox

3.1. Capture Leap Motion Signals

Leap Motion uses two infrared cameras and three infrared LEDs to form a body-sensing sensor with depth information. Using a dual-camera controller to simulate human eyes, and using the principle of stereo vision, it can coordinate positioning of space objects. The field of view displayed by Leap Motion is roughly concentrated in an inverted quadrangular pyramid above the device, and the effective range of its detection is about 25 mm to 600 mm. Figure 3 is the hand shape and data display interface captured in the field of view of the Leap Motion device, reflecting the detectable range of the sensor.

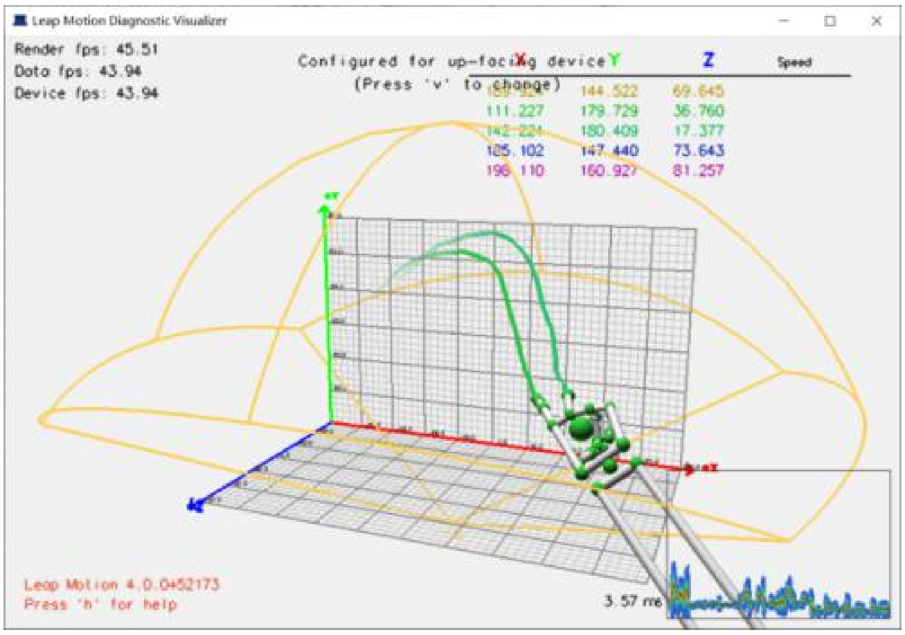


Figure 3: The Visualizer of Leap Motion device.

Most of Leap Motion is used to implement interaction and control robots using the official software development kit (LeapSDK). The development language also only has the development interface for C ++, C #, Java, Python, Unity languages, which requires a high programming ability of developers. As we all know, MATLAB is a simple and efficient programming language, users don't need too much programming ability.

In this paper, Matleap was used. Matleap is a MATLAB mex interface for the Leap Motion Controller[8].We can use this interface to generate matleap.mexw64. The matleap.mexw64 is a binary file (MEX type file) which is a type of dynamic linker. The MATLAB interpreter can automatically load and execute this file. In this way, data acquisition of Leap Motion can be realized in the MATLAB environment. Figure 4 shows the movement track of the hand in space by Leap Motion capturing the position of the hand and leaving it in the MATLAB figure.

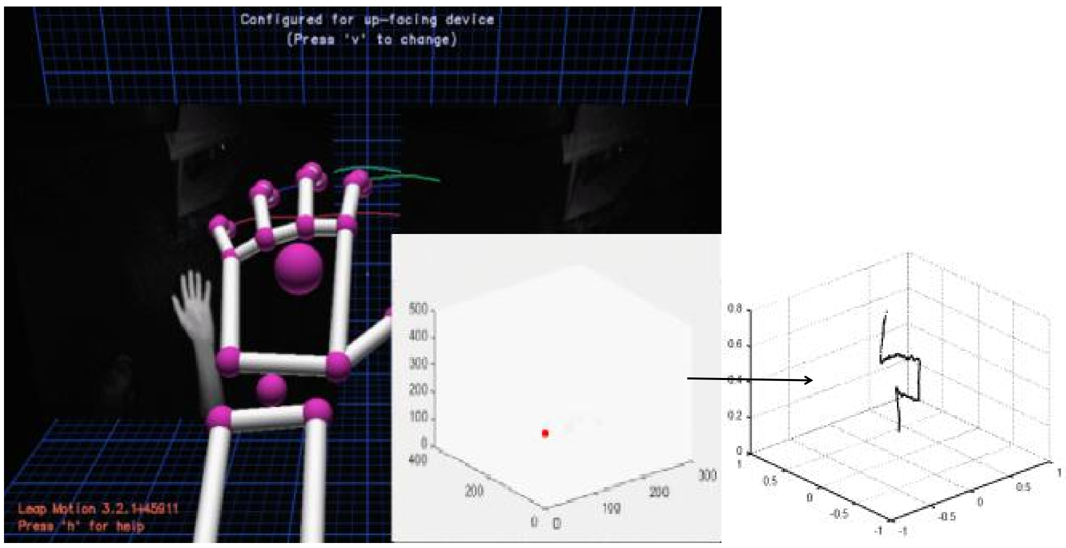


Figure 4: Leap Motion capturing the position of the human hand.

3.2. Interactive Simulation

In this section, we first establish a simple GUI interface and kinematics model of the robot, and then test in simulation based on the defined gesture and interactive control mode. Interactive simulation control is mainly used to control the Cartesian space of the simulated robot, to control the single-axis joint space of the simulated robot, and to control the opening and closing of the gripper.

Analysis the Baxter robot, the simplified model of the left arm link is as follows Figure 5, which illustration and reflects the transformation from the arm pillar to the elbow joint e1[9].

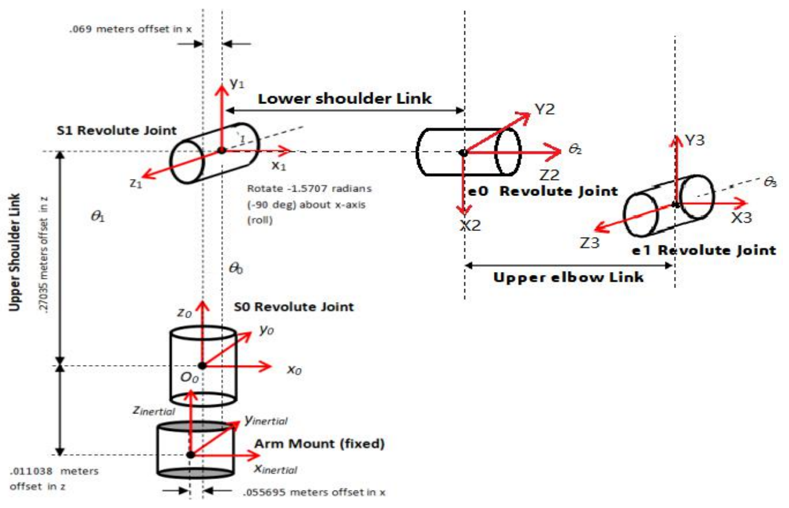


Figure 5: Baxter robot link model.

In this way, DH parameters of Baxter's left arm can be obtained according to the characteristics of its left arm and the URDF file of the Baxter robot, as shown in Table 1.

After knowing the DH parameters, we can quickly use the robotic toolbox, a very powerful tool by Peter Corke[10].to build the Baxter robot simulation model and use the forward and inverse kinematic functions in the toolbox for simulation control.

Use the following matrix transformation to transform the XYZ coordinates of the leap motion with the coordinates of the end of the left arm of the Baxter robot.

$$ {}_{L}^{B}T = \begin{vmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{vmatrix} \tag{1}$$

Table 1: DH Parameter of Baxter left arm.

i	θ/(deg)	d/m	α/m	α/(deg)	Range /(rad)
1	θ1	0.2703	0.069	-90	-1.7016-1.7016
2	θ2+90	0	0	90	-2.147-1.047
3	θ3	0.3644	0.069	-90	-3.0541-3.0541
4	θ4	0	0	90	-0.005-2.618
5	θ5	0.3743	0.01	-90	-3.059-3.059
6	θ6	0	0	90	-1.5707-2.094
7	θ7	0.2295	0	0	-3.059-3.059

The hand motion captured by leap motion can be mapped to the end of the robot. The mapping algorithm relationship is as follows.

$$P_L = P_C + ({}_{L}^{B}T * offsetLeap \quad XYZ\ ) * Scaling \tag{2}$$

And the Pc represents the current pose of the left arm of Baxter robot.

$$P_C = {}_{7}^{0}T = {}_{1}^{0}T(\theta_1)\,{}_{2}^{1}T(\theta_2)\ldots{}_{7}^{6}T(\theta_7) \tag{3}$$

Where

$$ {}^{i-1}T_i = \begin{vmatrix} c\theta_i & s\theta_i c\alpha_i & s\theta_i s\alpha_i & a_i c\theta_i \\ c\theta_i & c\theta_i c\alpha_i & -c\theta_i s\alpha_i & a_i s\theta_i \\ 0 & s\alpha_i & c\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{vmatrix} \tag{4} $$

Which represents the conversion relationship between adjacent coordinates. Where sθ is sin(θ), cθ is con(θ),cα is cos(α),sα is sin(α), Rj is rotation around the J axis, Tj is translation along the J axis, and J is the Z axis or X Axis, i = 1, 2, ... 7.θ is the corresponding joint angle. The parameter offset*LeapXYZ* is the relative offset of the Palm center. Scaling is sensitivity for movement of robot.

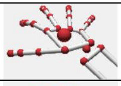
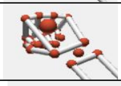
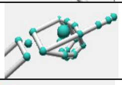
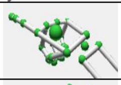
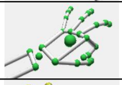
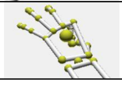
$$ \text{offset}\,L\text{eap}XYZ = [\Delta X, \Delta Y, \Delta Z] \tag{5} $$

The Cartesian space mapping algorithm can be used to map the up, down, left and right movement of the hand to the up, down, left and right movement of the left arm of the Baxter robot.

Because the data tracking list of each frame object provided by Leap Motion includes Hands, Fingers and Pointables. We uses some properties provided by the hand object to design gestures. A part of gesture recognition results and corresponding interaction control methods are shown in the Table 2.

The GUI interface is shown in figure 6.The coordinate area is the simulation model of the Baxter left arm. The lower left corner indicates that the control mode is Cartesian space motion control and single-axis joint space motion control. Different control modes are directly reflected by the movement of the robot model. In the lower right corner, there are detection gesture recognition results and displayed gripper controlled states.

Table 2: Gesture information and explanation.

Gesture	Identify	Interactive action
	Palm Open	Gripper Grasp
	Palm Close	Gripper Close
	A Finger (Positive)	Joint1 Moving (Forward)
	A Finger (Negative)	Joint1 Moving (Inverse)
	Three Fingers (Positive)	Joint4 Moving (Forward)
	Three Fingers (Positive)	Joint4 Moving (Inverse)

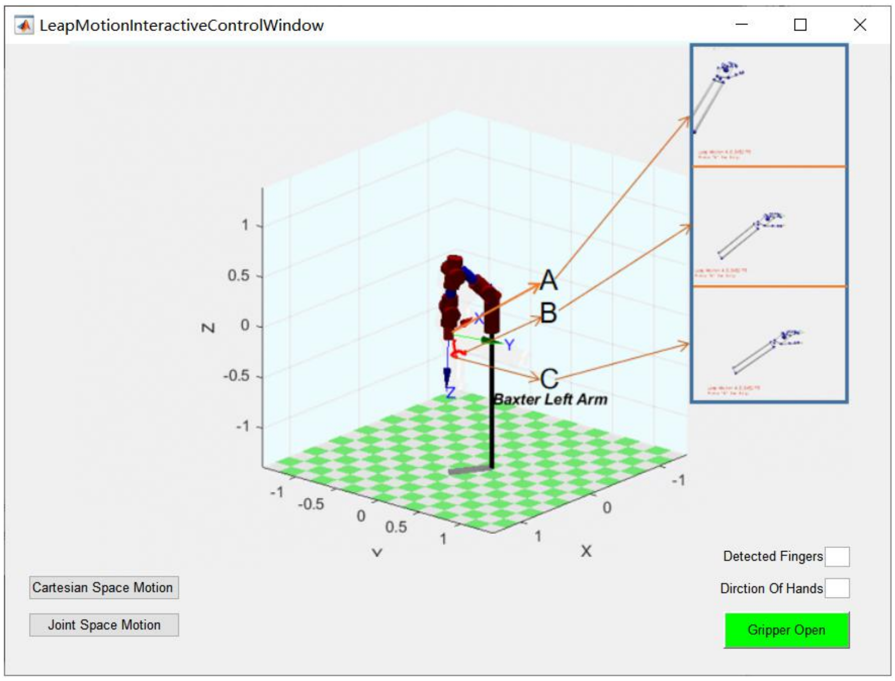
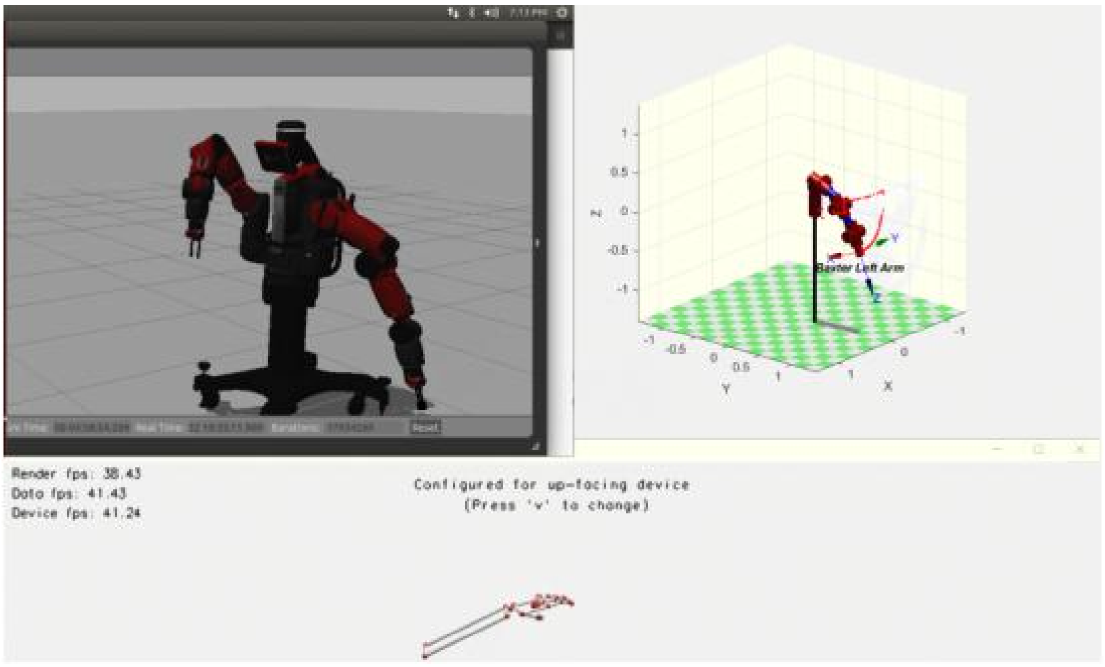


Figure 6: GUI, Gesture interaction for Cartesian space motion.

Figure 6 just shows the gesture motion captured based on leap motion, and it is reflected on the robot through the mapping algorithm. The red trajectory left at the end of the left arm reflects the motion control of the robot. We extracted three key points and attached the corresponding gestures and spatial positions to the upper right corner of the GUI.

As shown in the Figure 7, we use gesture recognition to control independent joints of the robot's single axis. In order to reflect the consistency of interactive control to the robot simulation model, we also introduced the Gazebo simulator at the same time. The robot in the two simulation environments show the consistency of motion. The Baxter robot in the two simulation environments show almost the same motion state under the gesture interactive control with Leap Motion controller. This also shows the feasibility of interactive control algorithms in the real-time nature of gesture interaction based on Leap Motion controller.


Render fps: 38.43
Data fps: 41.43
Device fps: 41.24

Configured for up-facing device
(Press 'v' to change)

Figure 7: Gesture interaction for Joint space motion.

4. Deployment

In the above, we have verified the feasibility of Leap Motion gesture interactive control algorithm through simulation. Next, the interactive control algorithm can be directly deployed on the actual robot According to the principle and structure of the interactive control system shown in Figure 2, a robot control system based on Leap Motion gesture interaction is built. The ROS toolbox is used to implement the control instructions transmitted from the ROS Global Node on MATLAB to the ROS Master node manager on the Baxter robot. The following Figure 8 is a schematic diagram of the implementation of a Leap Motion gesture interactive control system.

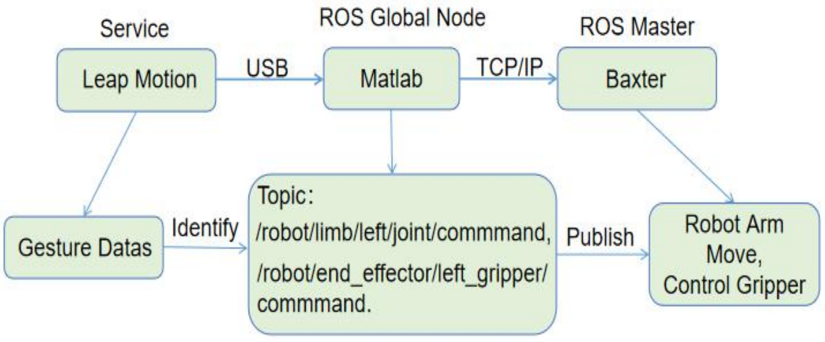


Figure 8: Gesture interaction based on ROS toolbox.

The recognized gestures have been defined in Table 2. In this section we interpret gestures as corresponding robot control commands. Via TCP / IP, a corresponding control message is published to a node whose topic is / robot / limb / left / joint / command or / robot / end_effector / left_gripper / command. In this way, joint motion control and grip control of the Baxter robot can be realized.

We also designed a simple task, in which anyone can control the robot through gesture interaction to achieve grabbing and placing. The simple pick and place task is shown in Figure 9, this experiments also show that the proposed gesture interaction is effective.

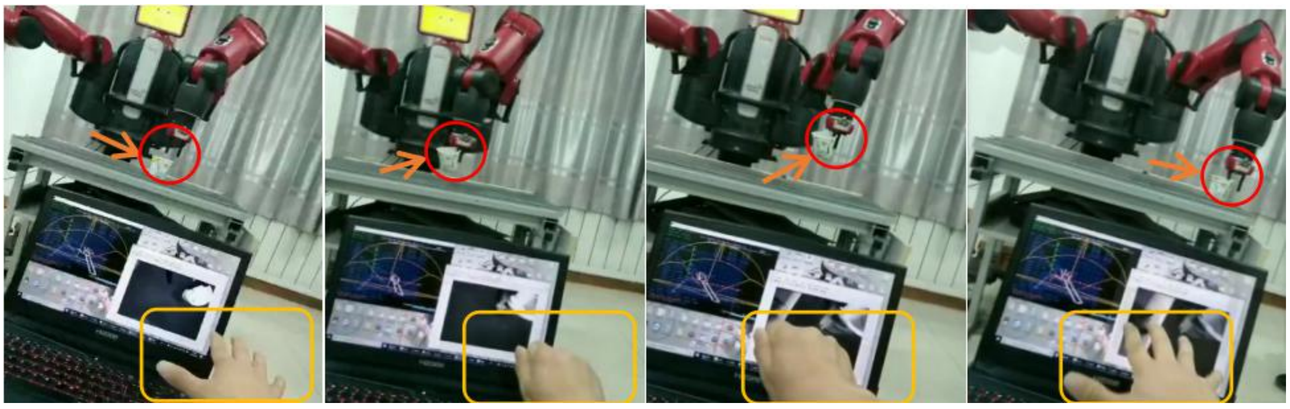


Figure 9: Pick and place based on Leap Motion Controller.

5. Conclusions

As described in this article, the main goal of our research is to use gesture interaction to achieve a more natural operation of the robot to replace the traditional robot operation mode and reduce the threshold of robot use. In this article, we propose that the introduction of simulation during the development of gesture interaction can verify the feasibility of gesture interaction in advance. We specifically describe the introduction of external real signals from Leap Motion sensors for interactive control research in a simulated environment, and derive a control method for reliable interaction with Baxter robots. And deploy the actual interactive control scheme through ROS toolbox.

The experimental results show that gesture-based interaction and remote control of the robot through the network protocol (TCP/IP) can accomplish some specific tasks. In our proposed interactive software structure, the proven interactive control method is basically applicable to robots supporting ROS systems. This represents the universality and portability of gesture interactive control, and it also means that when the same interactive control algorithm is transplanted, the development period can be greatly reduced to ensure the reliability of the system.

Acknowledgments

This work was financially supported by Natural Science Foundation of Ningxia Province (No. 2019AAC03121); the Key Laboratory of Intelligent Information and Big Data Processing of Ningxia Province, North Minzu University (No. 2019KLBD002); University Science and Technology Innovation Platform of Ningxia Province (Industry-University Cooperative Research Base for Key parts of advanced equipment and system innovation); Research and Practice Project of Graduate Education and Teaching Reform in 2018 (No. YJG201858).

References

[1] Cancedda L, Cannavò A, Garofalo G, et al. Mixed Reality-Based User Interaction Feedback for a Hand-Controlled Interface Targeted to Robot Teleoperation[C]//International Conference on Augmented Reality, Virtual Reality and Computer Graphics. Springer, Cham, 2017: 447-463.

[2] Du G, Chen M, Liu C, et al. Online Robot Teaching With Natural Human–Robot Interaction[J]. IEEE Transactions

on Industrial Electronics, 2018, 65(12): 9571-9581.

[3] Mao X, Li W, Lei C, et al. A Brain–Robot Interaction System by Fusing Human and Machine Intelligence[J]. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 2019, 27(3): 533-542.

[4] Li P, Wu D, Xu S, et al. A novel methodology of calculating the human-machine interactive force for a head-neck exoskeleton[J]. Journal of Mechanical Science and Technology, 2018, 32(9): 4383-4397.

[5] Mr. Rajesh D. Savatekar, Mr. A.A.Dum. Design Of Control System For Articulated Robot Using Leap Motion Sensor[J]. International Research Journal of Engineering and Technology (IRJET), 2016, 3.

[6] Sato Y. Study of Intelligent Control of an Arm Robot Equipped with a C-MOS Camera Using a New Type of Image Processing[C]//Proceedings of the 54th Annual Meeting of the ISSS-2010, Waterloo, Canada. 2010, 54(1).

[7] https://ww2.mathworks.cn/products/ros.html

[8] https://github.com/jeffsp/matleap

[9] Ju Z, Yang C, Ma H. Kinematics modeling and experimental verification of baxter robot[C]//Proceedings of the 33rd Chinese Control Conference. IEEE, 2014: 8518-8523.

[10] https://www.mathworks.com/matlabcentral/fileexchange/68542-robotics-toolbox-for-matlab.