Neurocomputing 275 (2018) 2093–2103

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Robot teaching by teleoperation based on visual interaction and extreme learning machine

Yang Xu[a], Chenguang Yang[a,*], Junpei Zhong[b], Ning Wang[c], Lijun Zhao[d]

[a] Key Lab of Autonomous Systems and Networked Control, School of Automation Science and Engineering, South China University of Technology, Guangzhou 510641, China
[b] National Institute of Advanced Industrial Science and Technology (AIST), Aomi 2-3-26, Tokyo, Japan
[c] Chinese University of Hong Kong Shenzhen Research Institute Building, Shenzhen, China
[d] State Key Laboratory of Robotics and System, Harbin Institute of Technology, Harbin 150001, China

ARTICLE INFO

Article history:
Received 26 June 2017
Revised 7 September 2017
Accepted 11 October 2017
Available online 2 November 2017

Communicated by Dr. H. Yu

Keywords:
Robot teaching
Teleoperation
Extreme learning machine (ELM)
Human-robot interaction (HRI)

ABSTRACT

Compared with traditional robot teaching methods, robots can learn various human-like skills in a more efficient and natural manner through teleoperation. In this paper, we propose a teleoperation method based on human-robot interaction (HRI), which mainly uses visual information. With only one teleoperation, the robot can reproduce a trajectory. There is a certain error between this trajectory and the optimal trajectory due to the cause of the human demonstrator or the robot. So we use an extreme learning machine (ELM) based algorithm to transfer the demonstrator's motions to the robot. To verify the method, we use a Microsoft KinectV2 to capture the human body motion and the hand state, according to which a Baxter robot in Virtual Robot Experimentation Platform (V-REP) will be controlled by the command. Through learning and training by the ELM, the robot in V-REP can complete a certain task autonomously and the robot in reality can reproduce this trajectory well. The experimental results show that the developed method has achieved satisfactory performance.

1. Introduction

With the recent rapid advances in robotics, the application of robots in industries has been extended to various fields. Through a teaching by demonstration (TbD) method, a robot can perform a task which is different from the previous one in a new working environment [1,2]. Traditionally, only after the professionals spend a lot of time for programming by keyboards or joysticks [3], the industrial robots can learn fixed skills on the assembly line. Apparently this approach is usually time consuming and not flexible to adapt to modern manufacturing. While a robot can be directly programmed by learning human-like manipulation skills from a skilful demonstrator through teleoperation. Therefore, this method can enable the robot to adapt to different tasks or environment efficiently.

Teleoperation based on HRI has been recently attracted much attention due to the advantages described above [4]. In [5], a TbD method is presented, enhanced by transferring the stiffness profile during HRI. In their work, muscle surface electromyography (sEMG) is collected and processed to extract the demonstrator's variable stiffness and hand grasping patterns. In [6], various hand gestures are recognized through the proposed HRI method based on hand guided demonstration. In [7], a teleoperation-based robot programming method is proposed. To verify the method, they develop a master-slave teleoperation system and an exoskeleton device is used as the HRI device.

There are many techniques or devices that are applied to HRI for enhanced performance. Generally, visual interaction has been one of the most widely utilized techniques [8]. Because visual interaction based on body motions tracking is comparative easy to implement, most of them are applied to capture human motion [9].

While in the TbD method for robot, neural networks have been widely applied. In [10], a TbD method for building an adaptive control system is presented. And the robot will improve its work performance by repeated a task with the help of the neural network. In [11], a neural learning scheme which can be used in estimating stable dynamical systems is presented. The result shows that the method is able to evaluate systems accurately. Slightly less than

* Corresponding author.
E-mail address: cyang@ieee.org (C. Yang).

https://doi.org/10.1016/j.neucom.2017.10.034
0925-2312/

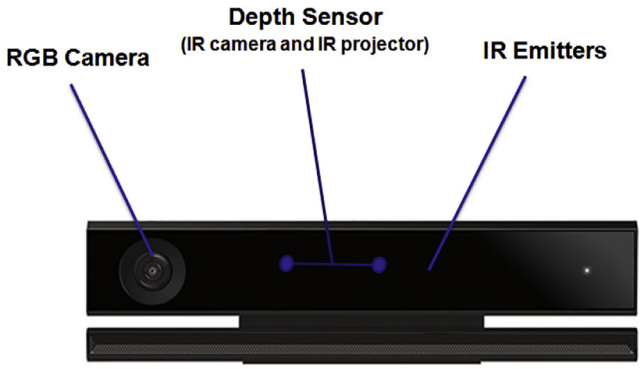

Fig. 1. The KinectV2 sensor [16].

the robot teaching based on neural learning is too complex to cost much time.

In this paper, we put forward a robot teaching method which uses a virtual teleoperation system based on visual interaction, and uses a neural learning method based on ELM. More specifically, in our work, Microsoft second generation of motion capture device, which is called Kinect V2, is used to track human body motion and the hand state. A simulation experiment has been conducted based on the V-REP platform where the Baxter robot is guided to learn the demonstrator's motion skills. And a learning algorithm based on ELM [12], which can teach the robot to learn some skills from human, is developed. Compared with robot teaching based on other neural network, this ELM requires less training samples, and has a high generalization capacity [13].

The rest of the paper is structured as follows: In Section 2, we introduce Kinect sensor, V-REP and its remote API. In Section 3, we present system included a virtual teleoperation system and a learning and training system. In Section 4, the design methodology about the system is introduced. In the Section 5, a space vector approach, a data processing method and a TbD method based ELM is presented. Finally, the experimental results are revealed in Section 6 followed by the conclusion given in Section 7.

2. Preliminaries

2.1. Kinect sensor

The second-generation Kinect for Windows is used in our work. Kinect has an RGB color camera, an IR emitter, and a depth sensor which is composed of an IR camera and an IR projector. With these devices above, Kinect Sensor provides full-body 3D motion capture, facial recognition and other capabilities [14]. Compared with the Kinect V1, Kinect V2 allows us to track up down 25 body joints [15], included the fists and thumbs. Because of such an advantage, the Kinect V2 can recognize the hand state. Such as Open, Closed and Lasso. The Lasso state means you close your hand and with the middle and extend your middle finger and index finger. (Fig. 1).

2.2. V-REP

V-REP is an open source robot simulator with an integrated development environment [17]. The procedure of construction and simulation of a robot is illustrated as follows:

1. Set up the robot model and work environment in V-REP scene: On the construction of a robot model, one way is to create the model in other software, then save it to a certain file format, such as OBJ, STL, 3DS, DXF, COLLADA or URDF, and finally import it to V-REP. Another way of doing it is to use the model directly which is provided by the internal model browser.

2. Using the programming approaches to control each object/model: V-REP is a highly customizable simulator, because it has 6 programming approaches to control each model [18]. That is to say, a model in V-REP could be controlled by an embedded script, an add-on, a plugin, a remote API client, a ROS node or a custom client/server, respectively.

3. Select the physics engine and set the simulation parameters: To simulate real-world physics and object interactions, V-REP provides 4 physics engines (Bullet Physics, ODE, Vortex Dynamics and Newton Dynamics). So the dynamics calculations can be faster and easier, and be customizable. [19].

In this work, a model of Baxter robot is used for simulation, and two API clients are developed to control the Baxter in V-REP.

2.3. The Remote API in V-REP

As mentioned above, The V-REP remote API allows to control a simulation or the V-REP itself from an external application or a remote hardware. There are two separate entities in the remote API functionality, namely the client side and the server side [20], respectively. The server side is usually a V-REP scene. And the client can be a C++ application or an application written in MATLAB . In order to ensure real-time communication between a V-REP scene and an API client, a socket communication is used, which could reduce the lags and the network load to a large extent. Therefore, it could allow an external application to communicate with V-REP in a synchronous manner.

The remote API in V-REP provides four types of operation modes which can decide one of the mechanisms to execute function calls. They are blocking function calls, non-blocking function calls, data streaming and synchronous operation, respectively. The process of transferring data in blocking function calls and non-blocking function calls are shown in Fig. 2. Using a blocking function call, we should wait for a reply from the API client. Using a non-blocking function call, do not need to wait for a reply, we are able to send command and data to V-REP.

2.4. Baxter robot

Baxter is a semi-humanoid industrial robot developed by Rethink Robotics. Baxter has a 2-DOF head and two 7-DOF arms (shoulder joint: S0, S1; elbow joint: E0, E1; wrist joint: W1, W2, W3) [21]. In addition to being applied in industry, it can also be used in scientific research, especially in HRI and human-robot collaboration (HRC) [22]. As is shown in Fig. 3, the Baxter robot in V-REP is similar to a real Baxter robot, except the head and the animated face. They share the same kinematics and dynamics.

3. The architecture of the system

As is shown in Fig. 4, the system we design contains a virtual teleoperation system and a training and learning system. In the first stage, a human demonstrator controls the Baxter in V-REP by Kinect. In the second stage, a neural network will be used to train and learn the data, which is recorded in the first stage. And then the output data is sent to the Baxter, to make it complete the previous task.

The virtual teleoperation system, which is the simulation of the real one, can verify the proposed algorithm in the virtual physical environment. The core of the virtual system is the verisimilitude of the model. Therefore, it is able to allow the human user to obtain quite real interactive experience.

The virtual teleoperation system we proposed, which is based on HRI, consists of a human demonstrator, a Kinect, a computer with V-REP. Separately, the human demonstrator occupies the dominant position in the teleoperation system; the Kinect is used

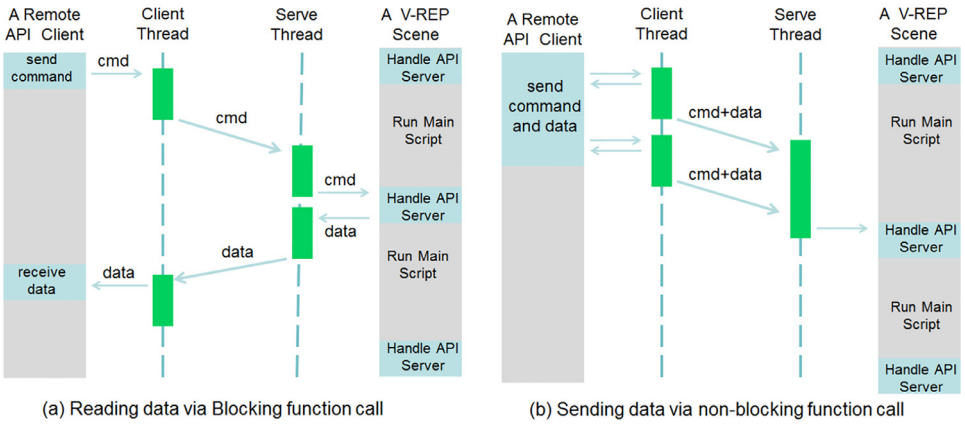

Fig. 2. The process of transferring data in blocking function calls and non-blocking function calls modified from [20].

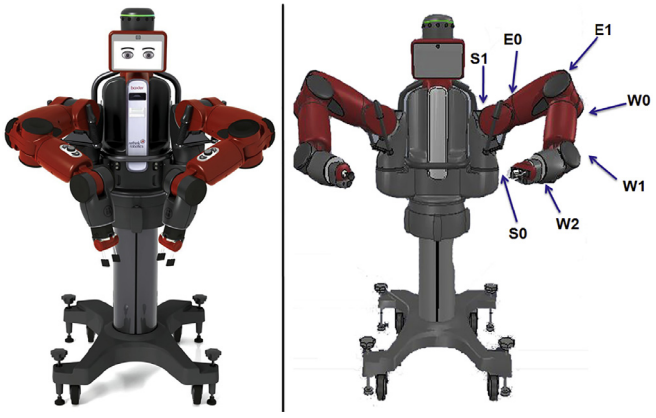

Fig. 3. The robot on the left is the Baxter in reality [23] and the robot on the left is the Baxter in V-REP.

to capture the human body motion and the hand state; and V-REP offers the virtual robot model and work environment. The control target is the Baxter robot's arms in V-REP, which are expected to follow the movement of the human demonstrator.

The training and learning system we proposed consists of a database, a learning neural network and also a computer with V-REP. Separately, the database can record and store robot joint angles; the a learning neural network based on ELM can get the data from data storage, and then train it by its three-layer neural network structure; and V-REP offers another simulation scene to test the effectiveness of the learning neural.

4. Design methodology

4.1. Acquiring information from Kinect

Kinect skeletal tracking is not affected by ambient lighting because of the infrared information. 3D depth images can be captured by the Kinect due to the mechanism of binocular vision [26].

There are three steps for the Kinect to capture the demonstrator's body information: At first, Kinect adopts the method of image segmentation to distinguish the human body from the complex background. Then Kinect finds the object in the image that is more likely to be human and evaluates depth of field image to identify different parts of the human body. Finally, it uses the results of the previous phase, and generates a skeletal system based on the 25 traced joint points.

4.2. Communication between Kinect and V-REP

To connect the Kinect V2, the computer should use a USB 3.0 port. And we install the Kinect for Windows SDK V2.0 and V-REP PRO EDU in the computer.

In the virtual teleoperation system, the client side is a custom C++ application and the server side is the V-REP scene.

To enable the Remote API on the client side, the C-language files provided by V-REP should be included in the C++ project,

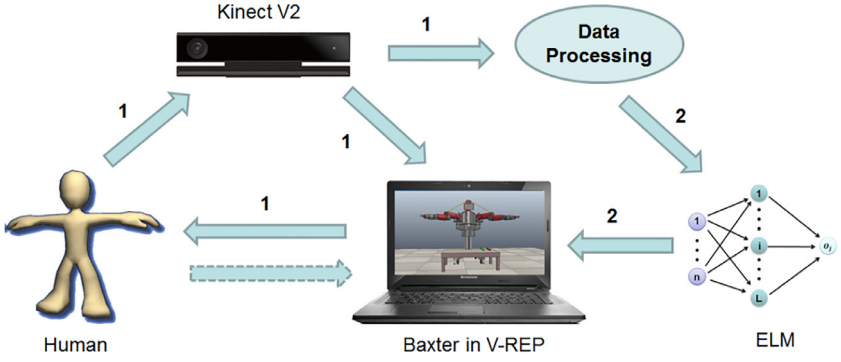

Fig. 4. The architecture of the system. The teleoperation system consists of a human [24], Kinect V2 [16], a V-REP sence . And the a training and learning system consists of a database, ELM [25] and another V-REP sence.

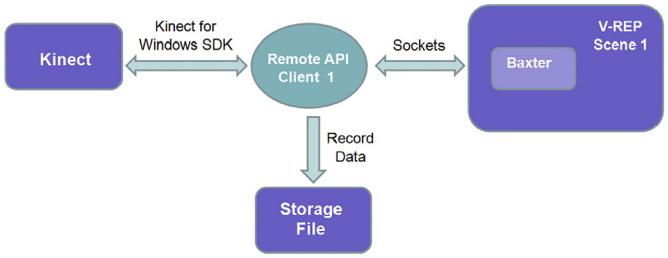

Fig. 5. The communication structure of the virtual teleoperation system.

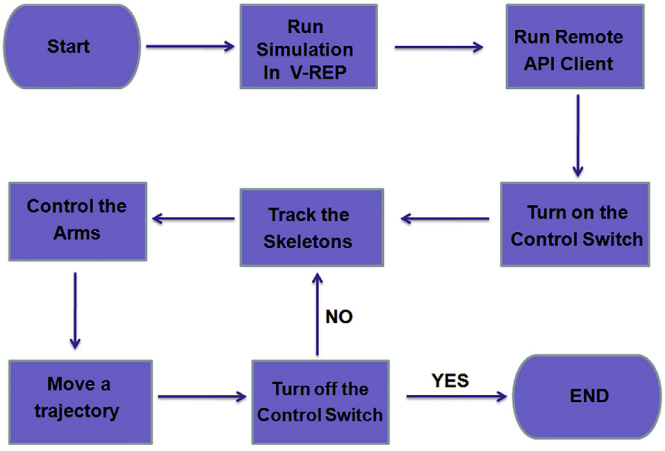

Fig. 6. The flow chart of the control scheme. After turning off the control switch, you can turn on it again by using the hand state.

such as extApi.h, extApi.c, extApiPlatform.h and extApiPlatform.c. To enable the remote API on the server side, the remote API plugin should be loaded when the V-REP is started up.

To control the Baxter in V-REP, the blocking function call is used to get the data of every robot joint in V-REP. So the client side can get information about the robot joints in V-REP. While the non-blocking function call is used to send the joint angle to the robot in V-REP. And then the C++ application will be used to record the data of the joint angle and save it in a file.

4.3. Design of the control scheme

In the teleoperation system whose communication structure is shown in Fig. 5, the human body motion is applied to control the movement of robot's arms, while the hand state is applied to control switching modes. The flow chart of the control scheme is shown in Fig. 6.

To turn on the control switch, the teleoperation system uses the sequence of the left hand as follows: Lasso⇒ Open hand ⇒ closed hand. As a result, the Baxter follows the movement of the human demonstrator. On the contrary, to turn off the control switch, it uses the sequence of the left hand as follow: Lasso⇒ Closed hand ⇒ Open hand. The robot grippers are controlled in such a way: to open/close the human demonstrator's right/left hand could open/close the robot right/left gripper.

4.4. The process of training and trajectory reproduction

Through human demonstration based on HRI, the robot has completed certain tasks, and the data of the joint angle is recorded and saved in a storage file. As the time of robot teaching is different, there is a difference in the time dimension of the collected experimental data. To ensure the data is valid, and has the same dimension, we need to process them. The joint angle of the robot changes over time, so each of them is a time series, which can be aligned by Dynamic Time Warping (DTW) algorithm. And then the data will be trained by ELM to learn the action of human demonstrator. (Fig. 7).

MATLAB is used to send the joint angle which are processed by DTW and ELM to the Baxter in V-REP. Similar to the C++ application, a blocking function call and a non-blocking function call are used as the same function in the communication between MATLAB and V-REP. The blocking function call is used to get the data of every robot joint in V-REP. And the non-blocking function call is used to send the joint angle, which is output by the ELM, to the robot in V-REP. The difference is that some M files provided by V-REP need to be included in the folder. Such as remApi.m, remoteApiProto.m and simpleSynchronousTest.m.

5. Key techniques

5.1. Space vector approach

The key of controlling the Baxter by Kinect is how to calculate the human joint angle. Kinect is able to get the 3D Cartesian coordinates of the joints of a human body. In a 3D space, the distance between two points $A(x_1, y_1, z_1)$ and $B(x_2, y_2, z_2)$ can be calculated by the following equation:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \tag{1}$$

And the Vector $\vec{AB}$ can be expressed as $\vec{AB} = (x_2 - x_1, y_2 - y_1, z_2 - z_1), d = |\vec{AB}|$ And in 3D space, the law of Cosines can calculate the angle between two joints. A joint in Kinect coordinate can be expressed as a vector. The joint 1 is $\vec{AB}$, and the joint 2 is $\vec{BC}$, so the angle between two joints can be calculated as,

$$cos(\vec{AB}, \vec{BC}) = \frac{\vec{AB} \cdot \vec{BC}}{|\vec{AB}| \cdot |\vec{BC}|} \tag{2}$$

According to the above equation, the coordinates returned by Kinect can be converted to corresponding vectors. And the respective angles of the joints can be calculated by the law of Cosines.

Getting all the location coordinates from Kinect, we can build the geometry model of human left arm as shown in Fig. 8. In Cartesian space of Kinect, The directed straight line OX, OY and OZ form a coordinate system. To calculate the shoulder pitch angle ∠OEF, we get the vectors $\vec{OE}$ and $\vec{EF}$ from three points and then calculate the angle. And using the same method, we can get the

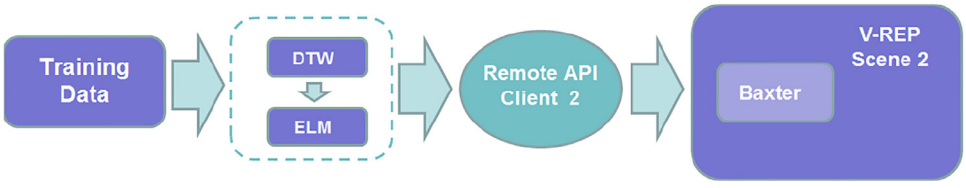

Fig. 7. The communication structure of the training and learning system.

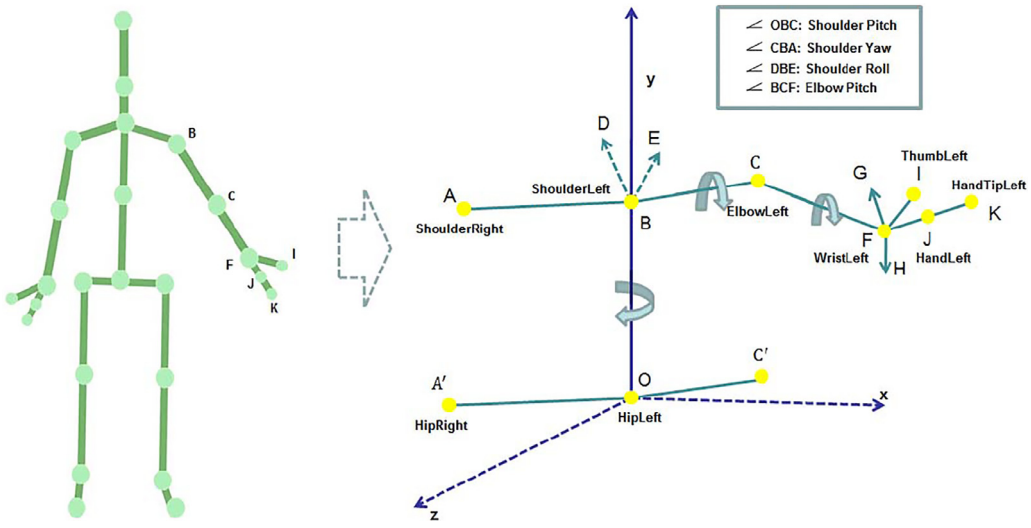

Fig. 8. The human skeleton obtained by Kinect V2 and the geometry model of human left arm.

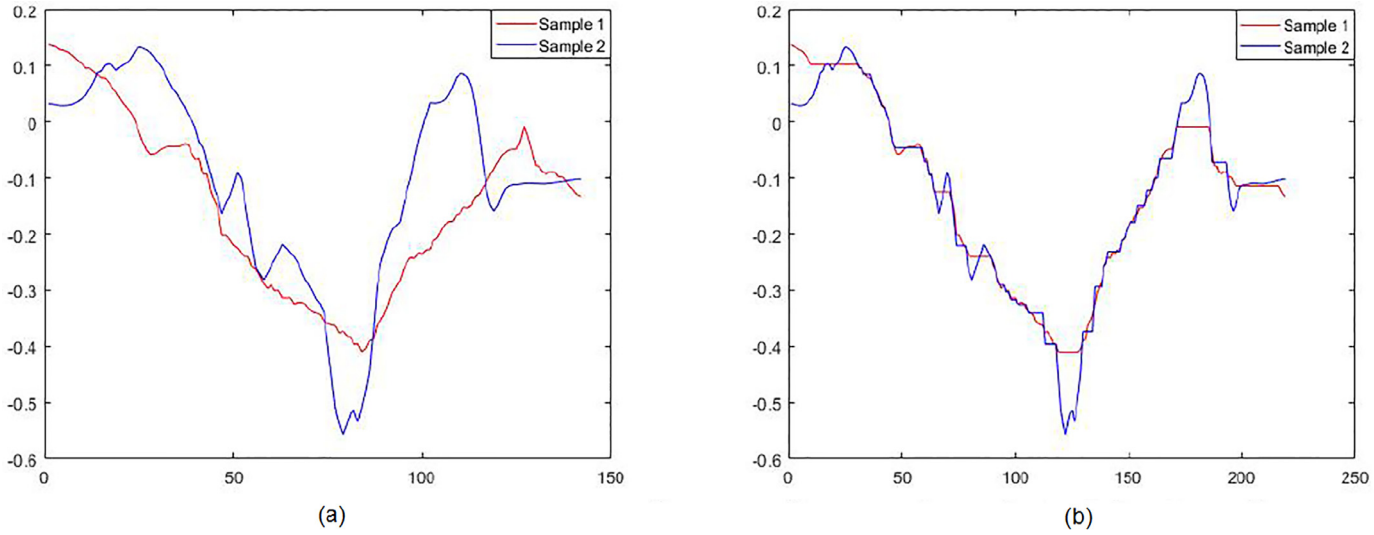

Fig. 9. (a)The two sample values of the joint angle S0.(b)Two sample values are aligned by DTW.

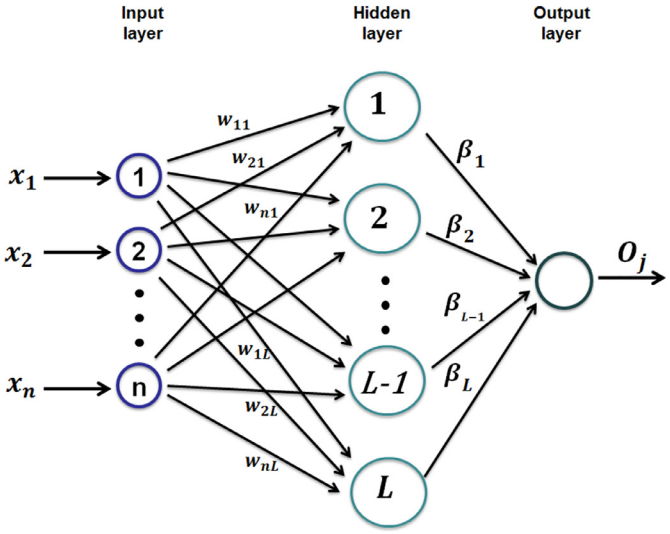

Fig. 10. Extreme learning machine.

elbow pitch ∠EFG. Project Points D, O and F to the plane XOZ, we can calculate the shoulder yaw angle ∠KOJ.

Through vector product, we can get the $\overrightarrow{LE}$ and $\overrightarrow{ME}$, which can be calculated as

$$\begin{cases} \overrightarrow{LE} = \overrightarrow{EF} \times \overrightarrow{FG} \\ \overrightarrow{ME} = \overrightarrow{EF} \times \overrightarrow{DE} \end{cases} \tag{3}$$

So the calculation of shoulder roll ∠LEM can be solved. In the same way, we can calculate the elbow roll, which is the angle between $\overrightarrow{LE}$ and $\overrightarrow{GN}$, and the hand yaw, which is the angle between $\overrightarrow{GN}$ and $\overrightarrow{GQ}$.

And the Baxter coordinate system is different with the Kinect coordinate system. So we should map the coordinate axis between them, which can be calculated as

$$\begin{cases} X_{Baxter} = -Z_{Kinect} \\ Y_{Baxter} = X_{Kinect} \\ Z_{Baxter} = Y_{Kinect} \end{cases} \tag{4}$$

So using the space vector approach, we can calculate the human joint angle and then use them to control the robot in V-REP by Kinect.

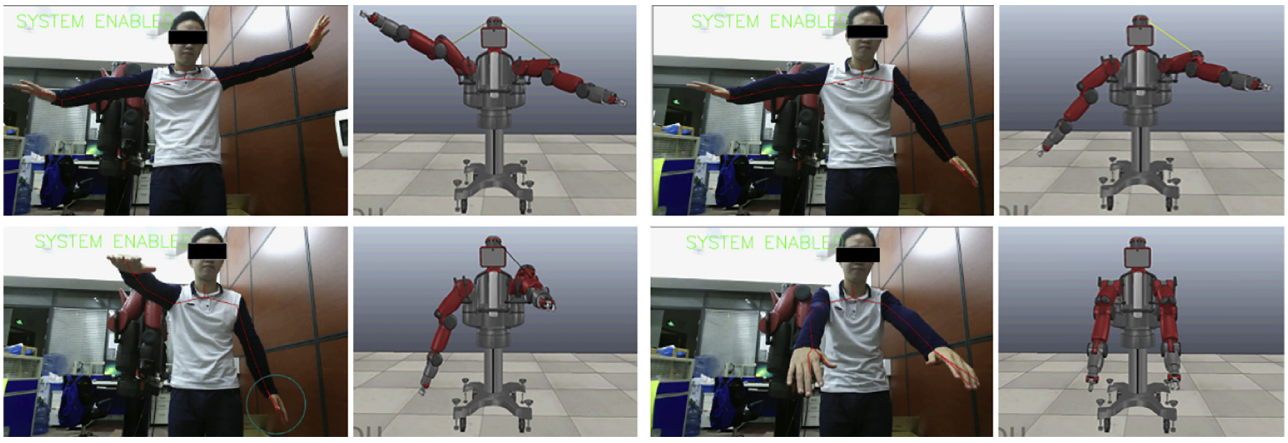

Fig. 11. The experiment results 1:human body motion is used to control the Baxter in V-REP. The first experimental results show that Baxter can follow the motions of human in a real time manner.

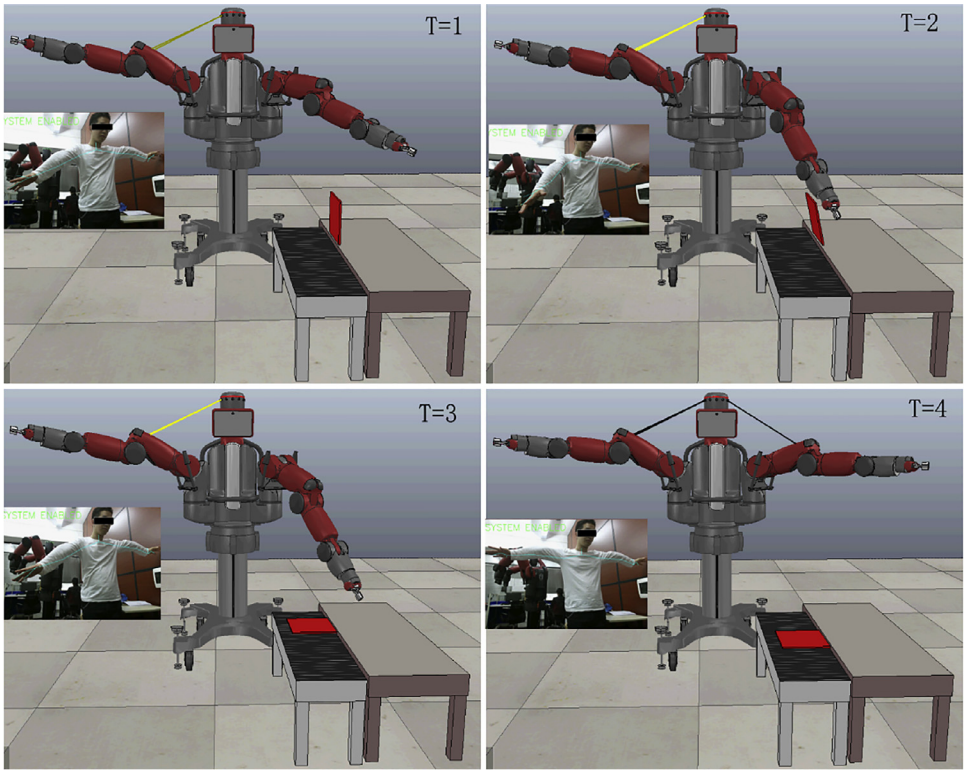

Fig. 12. The experiment results 2: Controlled by the human with a Kinect, the Baxter can push down the workpiece on the desk.

5.2. Dynamic time warping

DTW is a well-known algorithm to process the time series signal, which not only can calculate the similarity between two temporal sequences, but also can align the timeline of them with a dynamic programming.

Given two lengths of different time series,$x[x_1, x_2, \ldots, x_m]$ and $y[y_1, y_2, \ldots, y_m]$. Suppose the matrix WP is the match path between two given time series bulit by DTW, where

$$WP = \begin{bmatrix} wp_{11} & wp_{21} & \cdots & wp_{L1} \\ wp_{12} & wp_{22} & \cdots & wp_{L2} \end{bmatrix} \tag{5}$$

L is the length of the match path. And $wp_{k1} \in \{1, 2, \ldots, m\}, wp_{k2} \in \{1, 2, \ldots, n\}, k \in \{1, 2, \ldots, L\}$.

Under the constraint of match path WP, the distance between x and y can be expressed as

$$\rho(WP, x, y) = \sum_{k=1}^{L} (x_{wp_{k1}} - y_{wp_{k2}})^2 \tag{6}$$

where $[wp_{k1}, wp_{k2}]^T$ is the kth column of the matrix WP. And the match path WP must satisfy the following constraint:

1. Boundary condition:
$[wp_{11}, wp_{12}] = [1, 1]$ and $[wp_{L1}, wp_{L2}] = [m, n]$.
2. Monotonicity condition:
For $k = 1, 2, \ldots, L$, $wp_{11} \leqslant wp_{21} \leqslant \ldots \leqslant wp_{k1}$ and $wp_{12} \leqslant wp_{22} \leqslant \ldots \leqslant wp_{k2}$.
3. Step size condition:
For $k = 1, 2, \ldots, L-1$, $wp_{k+1,1} - wp_{k1} \leqslant 1$ and $wp_{k+1,2} - wp_{k2} \leqslant 1$.

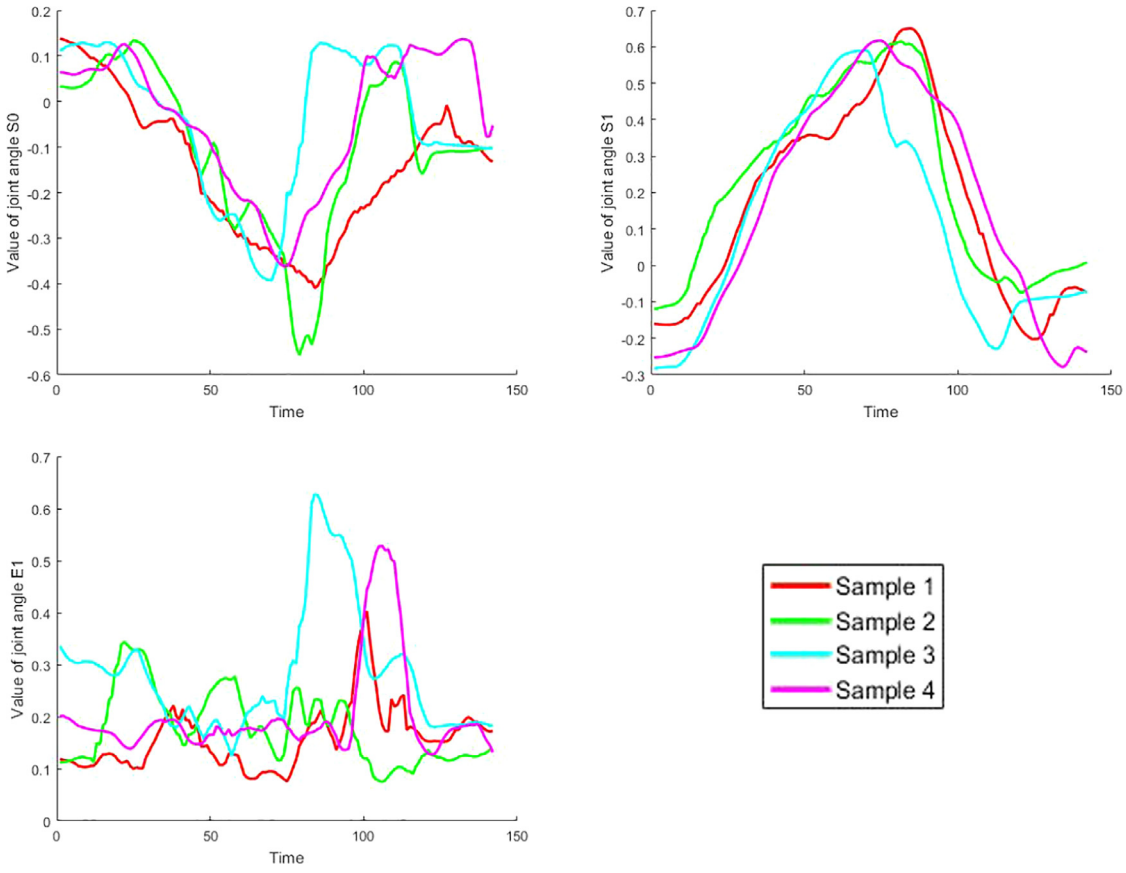

Fig. 13. Four randomly selected training samples. It can be seen that the difference between each set of data is very large.

Table 1
the value of the hidden layer biases β.

Joint	1	2	3	4	5	6
S0	0.6029	0.7756	0.3044	.8459	0.0224	0.8578
S1	0.5672	0.1846	0.0722	0.3726	0.9727	0.8949
E1	0.6948	0.3170	0.9502	0.0344	0.4387	0.3815
Joint	7	8	9	10	11	12
S0	0.8620	0.4827	0.7025	0.7762	0.0673	0.1802
S1	0.1044	0.9107	0.5869	0.8424	0.2982	0.7795
E1	0.7655	0.7952	0.1869	0.4898	0.4456	0.6463

There are multiple paths that satisfy the above constraints. The goal of DTW is to find the optimal match path, which can be expressed as

$$DTW(x, y) = min(\rho(WP, x, y)) \tag{7}$$

Using the dynamic programming to solve $DTW(x, y)$, it builds a accumulated cost matrix R which has a dimension of $m \times n$. $R(i, j)$ can be expressed as

$$R(i,j) = (x_i - y_j)^2 + \begin{cases} 0 & \text{if } i = 1 \text{ and } j = 1 \\ R(i, j-1) & \text{else if } i = 1 \text{ and } j > 1 \\ R(i-1, j) & \text{else if } i > 1 \text{ and } j = 1 \\ min(R(i-1, j), & \\ \quad R(i, j-1), R(i-1, j-1)) & \text{otherwise} \end{cases} \tag{8}$$

Using the DTW algorithm to process the sample, We can align them on the timeline, as shown in Fig. 9.

5.3. TbD based on extreme learning machine

Using the virtual teleoperation system, we can control the Baxter to move according to a trajectory. However, in the process of accomplishing a task, human action is not necessarily optimal. Therefore, Controlled by teleoperation, the robot's trajectory will have some deviation compared with the target trajectory. Through the neural network learning based on ELM, the robot trajectory can approach the target trajectory. The dynamical system can be expressed by a first-order autonomous ordinary differential equation

$$\dot{s} = f(s) + \varepsilon \tag{9}$$

where s denotes the robot's joint angles, and $\dot{s}$ is the first derivative of s. The dataset is $\{s, \dot{s}\}_{t=0}^{T_1,\ldots,T_L}$, ε is a zero mean gaussian noise [27]. The goal is to obtain an estimation of $\hat{f}$ from f.

To achieve this goal, we use a method based on ELM, which is more efficient than the traditional learning algorithm under the same conditions. To use the ELM in the teleoperation system,the goal is to learn the mapping $f : s \to \dot{s}$ based on the dataset $\{s, \dot{s}\}_{t=0}^{T_1,\ldots,T_L}$. As shown in Fig. 10, to a neural network with a hidden layer, the input layer has n nodes, which is the dimension of s. In the hidden layer, the target function is,

$$\dot{s} = o = \sum_{i=1}^{L} \beta_i f_i(s) = \sum_{i=1}^{L} \beta_i g(\omega_i^T s + b_i) \tag{10}$$

where g is activation function, $W=(\omega_1, \omega_2, \ldots,\omega_L)^T$ is the input weights, which has dimension $L \times d$, and $\beta=(\beta_1, \beta_2, \ldots,\beta_L)^T$ is the output weights, which also has dimension $L \times d$, $b=(b_1, b_2, \ldots,b_L)$ is the hidden layer biases, $\omega_i^T s$ is inner product of W and s.

The learning goal of a single hidden layer neural network is making the output error minimized. So ELM solves the problems

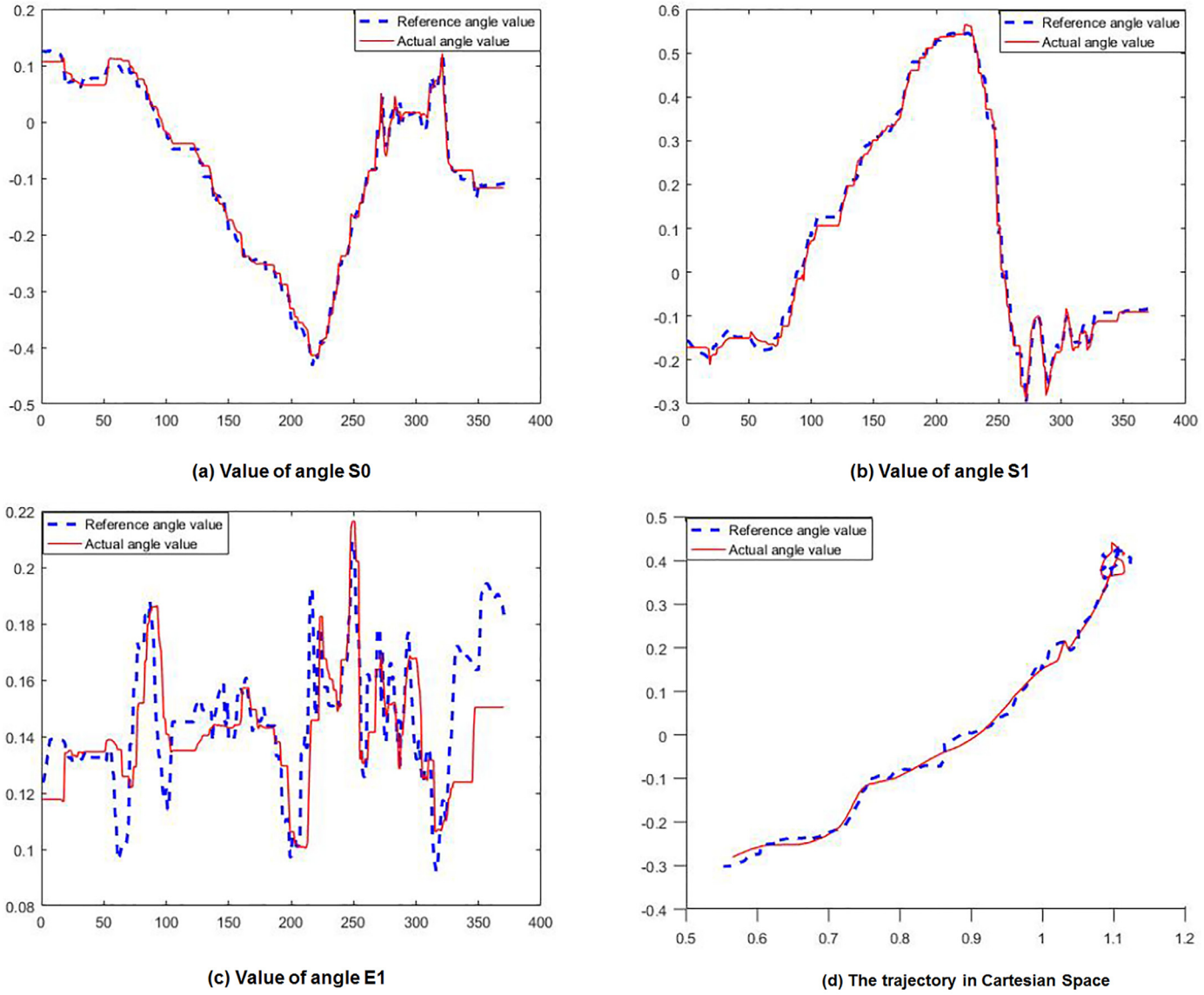

Fig. 14. The output data of the ELM. (a),(b),(c) are the joint angles of S0, S1 and E1. And (d) is the trajectory of the Baxter arm moving downward in Cartesian space. The reference angle value are output by the ELM network. And the actual angle value are recorded from the real Baxter robot.

as follows:

$$\min_{\beta} \|H\beta - O\| \tag{11}$$

where

$$H(\omega_1, \omega_2, \ldots, \omega_L, b_1, b_2, \ldots, b_L, s_1, s_2, \ldots, s_L) \tag{12}$$

$$= \begin{bmatrix} g(\omega_1^T s_1 + b_1) & \cdots & g(\omega_L^T s_1 + b_L) \\ \vdots & \ddots & \vdots \\ g(\omega_1^T s_L + b_1) & \cdots & g(\omega_L^T s_L + b_L) \end{bmatrix}$$

is the output of the hidden layer node, O=(o_1, o_2, ...,o_L)^T is expected output. In the system, O is the target value which is generated by demonstration.

Once the input weights and hidden layer biases are fixed, the output matrix of the hidden layer H is uniquely determined. Then the problem about training single hidden layer neural network can translate into a problem about solving a linear system. The solution is

$$\beta = H^{+}O \tag{13}$$

where H^{+} is the Moore–Penrose generalized inverse of the matrix H.

For any $t \in R$, the activation functions $g(t)$ should be continuous and continuously differentiable. We use a kind of activation functions which satisfy that

$$\begin{cases} g(t) = 0, & t = 0 \\ g(t) > 0, & \forall t \in R \end{cases} \tag{14}$$

In order to satisfy the above properties, a bipolar sigmoid function

$$g(t) = \frac{2}{1 + e^{-t}} - 1 \tag{15}$$

is used. The sigmoid functions are continuous and continuously differentiable, so they have not an impact on the performance of ELM.

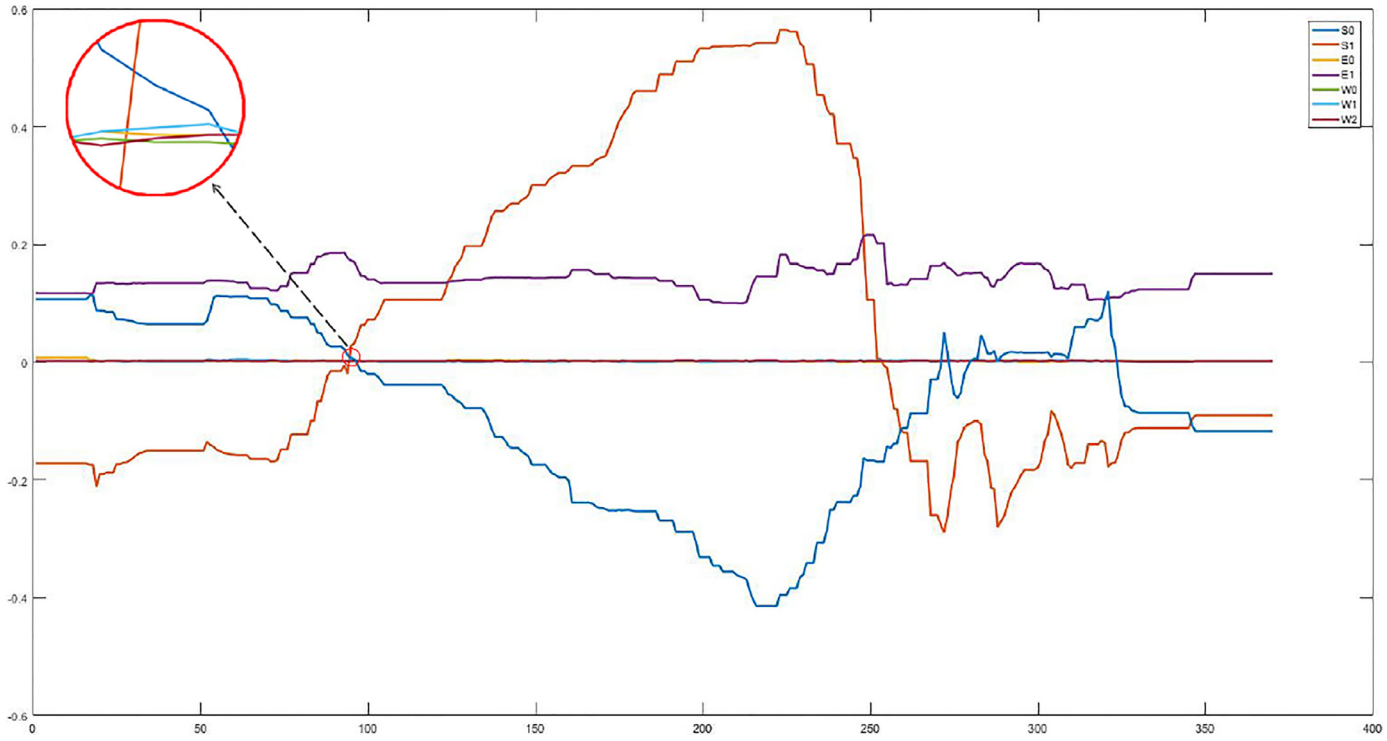

Fig. 15. The joint angle recorded from the real Baxter robot. It can be seen that the other values are approximately 0.

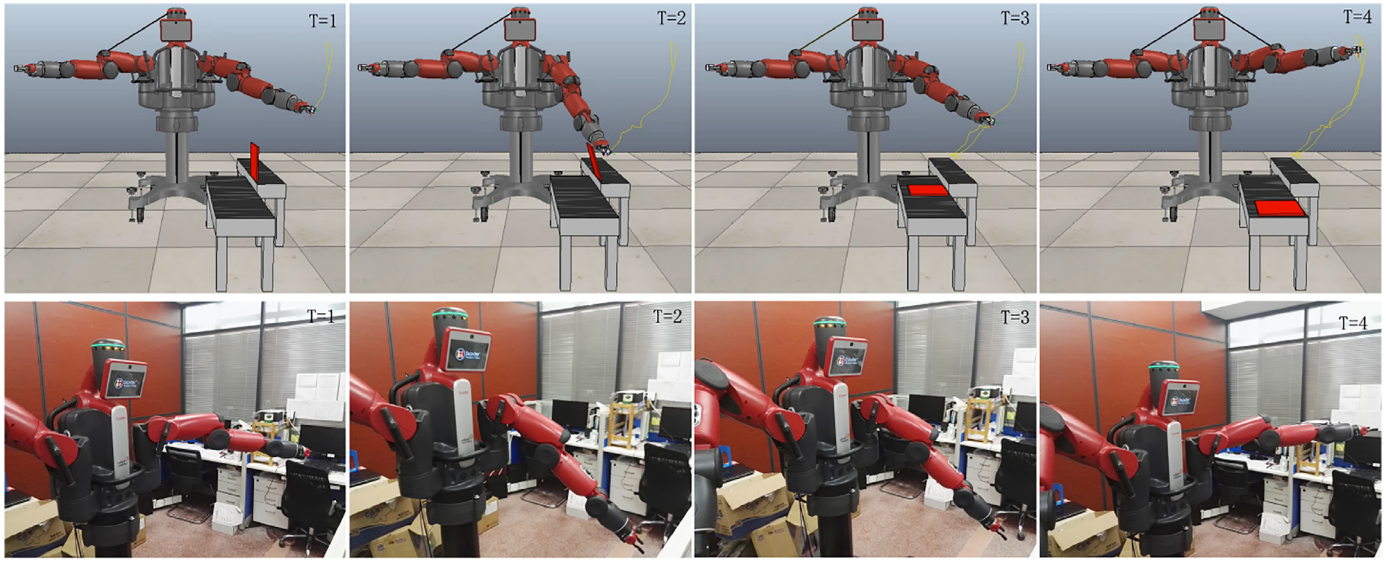

Fig. 16. The experiment results 3: Through learning and training by ELM, the Baxter in V-REP and in reality can reproduce the action of the human demonstrator.

6. Experiment and results

6.1. The Effectiveness of the virtual teleoperation system

In order to verify the validity of the proposed method, we build a virtual teleoperation platform which mainly consists of Kinect and V-REP. At first we verify the effectiveness of controlling the robot by human body motions. Four motions are designed to verify that the Baxter robot arms can be moved flexibly in the virtual space.

As shown in Fig. 11, the first two motions show that the Baxter robot arms can swing up and down controlled by the human demonstrator. And the rest motions show that the Baxter robot arms can swing back and forth. In addition, it can be seen that the motion of the human arms and the motion of the robot are symmetrical. That is to say, the human left arm controls the robot's right arm, while the human right arm controls the robot's left arm. Therefore, the robot is able to follow the human motion accurately. The experimental results show that this virtual teleoperation system can make the human control the Baxter in V-REP in a real-time and accurate manner.

6.2. The collection and processing of experimental data

To verify the effectiveness of the TbD based on ELM, We designed a simulation scene. Controlled by the Kinect, the Baxter

in V-REP moves its arm and pulls down a square workpiece on the desk. Then the workpiece is transferred to a conveyor belt. In the above process, we have recorded the robot joint angle, which changes with time. Finally, we have repeated this process more than 20 times. (Fig. 12) Because the robot movement time is different in each experiment, the length of the experimental data is different. To ensure that the length of each set of data same, some data are interpolated. Then we can get a sample with dimension 142×3 from each experiment. After the processing of DTW, we can get a set of data with 317×3.

6.3. Data training and learning based on ELM

We implement the algorithm based on ELM in MATLAB. To find the optimal number of hidden layer nodes, we take a test in which the number of nodes changes from 1 to 100. The result shows that the situation of the 12 nodes has highest accuracy. And then we use ELM to train the input data, and finally get three group of output data. As shown is Fig. 14, they are the values of the robot's corresponding joint angles, respectively.

In the following experiment, the data is send to V-REP through MATLAB. Controlled by the MATLAB, the Baxter in V-REP can reproduce the trajectory which is provided by the ELM. In another scene we design, the Baxter can transfer a workpiece from a conveyor belt to another one. And it perform well in repetitive tasks.

And we also used the real Baxter robot to verify the effectiveness of ELM. We send the data from the neural network directly to the robot. At the same time, each time we enter a set of joint angle values, we record the actual value of the joint angle. The joint angle value we sent and get shown in Fig. 14. It can be seen that the actual robot can reproduce the trajectory which is trained by the ELM. And the actual value of the joint angle we record is shown in Fig. 15.

In addition, from Figs. 13 and 14, we can know that these robot trajectories are mainly related to S0 and S1. During the downward movement of the robot arm, the minimum value of the S0 is approximately equal to -0.4, and the maximum value of the S1 is approximately equal to 0.55. Thus, Therefore, as shown in Fig. 16, the robot can learn to move a trajectory by teleoperation based on HRI. And the neural network based on ELM can obtain the main features of this trajectory.

7. Conclusion

In this paper, we have developed a virtual teleoperation system based on visual interaction. The human body motion is used to control the robot's arms, and gesture are used to control the beginning and end of the simulation. In addition, through a TbD method based on ELM, the system can transfer the human motions to the robot. We use Kinect to acquire the body skeleton data and hand states. Then we use V-REP, to build a Baxter robot and its work environment. To verify the effectiveness of the TbD based on ELM, we control the Baxter in V-REP to move its end-effector according to a certain trajectory. Through learning and training, the robot in V-REP can autonomously repeat such tasks and the robot in reality can reproduce this trajectory well. The results show that robot can not only learn the motion from human demonstrator in a natural manner, but also perform well in repeating a certain trajectory.

Since there exits a certain distance between human eyes and the computer screen, a good performance can not be always achieved. In the future, the virtual reality technique will be further introduced into the developed teleoperation system, in order to enable human demonstrator to obtain enhanced teaching experience by capturing 3D information.

Acknowledgement

This work was partially supported by National Nature Science Foundation (NSFC) under Grant 61473120, Guangdong Provincial Natural Science Foundation 2014A030313266 and International Science and Technology Collaboration Grant 2015A050502017, Science and Technology Planning Project of Guangzhou 201607010006, State Key Laboratory of Robotics and System (HIT) Grant SKLRS-2017-KF-13, and the Fundamental Research Funds for the Central Universities 2017ZD057.

References

[1] S. Calinon, D. Bruno, M.S. Malekzadeh, T. Nanayakkara, D.G. Caldwell, Human-robot skills transfer interfaces for a flexible surgical robot., Comput. Methods Programs Biomed. 116 (2) (2014) 81–96.
[2] L. Peternel, T. Petri, E. Oztop, J. Babi, Teaching robots to cooperate with humans in dynamic manipulation tasks based on multi-modal human-in-the-loop approach, Auton. Robots 36 (1) (2014) 123–136.
[3] J.J. LaViola, O.C. Jenkins, Natural user interfaces for adjustable autonomy in robot control, IEEE Comput. Graph. Appl. 35 (3) (2015) 20–21.
[4] T.B. Sheridan, Human-robot interaction: status and challenges., Hum. Factors: J. Hum. Factors Ergon. Soc. 58 (4) (2016).
[5] C. Yang, P. Liang, Z. Li, A. Ajoudani, C.-Y. Su, A. Bicchi, Teaching by demonstration on dual-arm robot using variable stiffness transferring, in: Proceedings of IEEE International Conference on Robotics and Biomimetics (ROBIO), IEEE, 2015, pp. 1202–1208.
[6] L. Zhao, X. Li, P. Liang, C. Yang, R. Li, Intuitive robot teaching by hand guided demonstration, in: Proceedings of 2016 IEEE International Conference on Mechatronics and Automation (ICMA), IEEE, 2016, pp. 1578–1583.
[7] H. Lee, J. Kim, T. Kim, A robot teaching framework for a redundant dual arm manipulator with teleoperation from exoskeleton motion data, in: Proceedings of IEEE-RAS International Conference on Humanoid Robots, 2014, pp. 1057–1062.
[8] C Yang, H Ma, M Fu, Human-Robot Interaction Interface[M], Adv. Technol. Mod. Rob. Appl. (2016) 257–301.
[9] C.D. Mutto, P. Zanuttigh, G.M. Cortelazzo, Time-of-Flight Cameras and Microsoft Kinect (TM), Springer Publishing Company, Incorporated, 2012.
[10] S. Liu, H. Asada, Teaching and learning of deburring robots using neural networks, in: Proceedings of IEEE International Conference on Robotics and Automation, 1993, IEEE, 1993, pp. 339–345.
[11] K. Neumann, A. Lemme, J.J. Steil, Neural learning of stable dynamical systems based on data-driven Lyapunov candidates, in: Proceedings of 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2013, pp. 1216–1222.
[12] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, Extreme learning machine: theory and applications, Neurocomputing 70 (1) (2006) 489–501.
[13] G.B. Huang, An insight into extreme learning machines: random neurons, random features and kernels, Cogn. Comput. 6 (3) (2014) 376–390.
[14] S. Zennaro, Evaluation of Microsoft kinect 360 and Microsoft kinect one for robotics and computer vision applications [J]. (2014).
[15] X. Xu, R.W. McGorry, The validity of the first and second generation Microsoft kinect for identifying joint center locations during static postures, Appl. Ergon. 49 (2015) 47–54.
[16] (https://blogs.msdn.microsoft.com/kinectforwindows/2014/06/05/pre-order-your-kinect-for-windows-v2-sensor-starting-today/).
[17] E. Rohmer, S.P. Singh, M. Freese, V-REP: a versatile and scalable robot simulation framework, in: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2013, pp. 1321–1326.
[18] M. Freese, S. Singh, F. Ozaki, N. Matsuhira, Virtual robot experimentation platform V-REP: a versatile 3D robot simulator, in: Proceedings of International Conference on Simulation, Modeling, and Programming for Autonomous Robots, Springer, 2010, pp. 51–62.
[19] S. Ivaldi, V. Padois, F. Nori, Tools for dynamics simulation of robots: a survey based on user feedback, Comput. Sci. 1402.7050 (2014).
[20] (http://www.coppeliarobotics.com/helpFiles/).
[21] A.N. Vazquez, W. Jabi, A collaborative approach to digital fabrication:a case study for the design and production of concrete pop-up structures, Int. J. Arch. Comput. 13 (2) (2015) 195–216.
[22] H. Reddivari, C. Yang, Z. Ju, P. Liang, Z. Li, B. Xu, Teleoperation control of Baxter robot using body motion tracking, in: Proceedings of International Conference on Multisensor Fusion and Information Integration for Intelligent Systems (MFI), 2014, 2014, pp. 1–6.
[23] (http://www.hizook.com/files/users/3/Baxter_Robot_from_RethinkRobotics_8.jpg).
[24] (http://i282.photobucket.com/albums/kk248/Dawisch/CartoonGuy-1.jpg).
[25] G. Huang, G.B. Huang, S. Song, K. You, Trends in extreme learning machines: a review, Neural Netw. Off. J. Int. Neural Netw. Soc. 61 (2015) 32–48.
[26] L. Yang, L. Zhang, H. Dong, A. Alelaiwi, A. El Saddik, Evaluating and improving the depth accuracy of kinect for windows v2, IEEE Sens. J. 15 (8) (2015) 4275–4285.

[27] S.M. Khansari-Zadeh, A. Billard, Learning stable nonlinear dynamical systems with Gaussian mixture models, IEEE Trans. Robot. 27 (5) (2011) 943–957.

Yang Xu received the B.Eng. degree in automation from the South China University of Technology, Guangzhou, China, in 2016, and is currently pursuing the M.S. degree in the South China University of Technology, Guangzhou, China. His research interests include human-robot interaction, robot imitation learning, and machine learning.

Chenguang Yang received the B.Eng. degree in measurement and control from Northwestern Polytechnical University, Xi'an, China, in 2005, and the Ph.D. degree in control engineering from the National University of Singapore, Singapore, in 2010. He received postdoctoral training at Imperial College London, UK. He is the recipient of the Best Paper Award from the IEEE Transactions on Robotics and a number of international conferences. His research interests lie in robotics, automation and computational intelligence.

Junpei Zhong is currently a research scientist at Artificial Intelligence Research Center of National Institute of Advanced Industrial Science and Technology (AIST), Tokyo, Japan. He received the B.Eng degree from South China University of Technology in 2006, M.Phil from the Hong Kong Polytechnic University in 2010 and doctoral degree (with "magna cum laude") from University of Hamburg in 2015. He has been awarded the Marie-Curie fellowship for his doctoral study from 2010 to 2013. From 2014 to 2016, he has participated in different EU and Japanese funded projects at University of Hertfordshire, Plymouth University and Waseda University before joining AIST. His research interests are machine learning, computational intelligence and cognitive robotics.

Ning Wang received the B.Eng. degree in measurement and control technologies and devices from the College of Automation, Northwestern Polytechnical University, Xi'an, China, in 2005, the M.Phil. and Ph.D degree in electronic engineering from the Department of Electronic Engineering, The Chinese University of Hong Kong, Hong Kong, China, in 2007 and 2011, respectively. She was working as Post-doc fellow at the Department of Computer Science & Engineering, The Chinese University of Hong Kong from 2011 to 2013, and was research fellow at the School of Computing, Electronics and Mathematics, Plymouth University, United Kingdom from 2014 to 2015. Her research interests lie in signal processing and machine learning, with applications in robust speaker recognition, biomedical pattern recognition, intelligent data analysis, and human-robot interaction.

Lijun Zhao received the bachelor degree from Beijing Institute of Technology in 1996, Beijing, China, master and Ph.D. degrees from Harbin Institute of Technology(HIT), Harbin, Heilongjiang, China, in 2002 and 2009, respectively, all in Mechatronics Engineering. He is current the supervisor of masters with State Key Laboratory of Robotics and Systems, Robotics Institute, Harbin Institute of Technology, China. His research interests include mobile robot 3D environment mapping, perception, navigation and planning in robotics.