Neural-Learning-Based Telerobot Control With Guaranteed Performance

Chenguang Yang, *Senior Member, IEEE*, Xinyu Wang, Long Cheng, *Senior Member, IEEE*, and Hongbin Ma, *Member, IEEE*

***Abstract*—In this paper, a neural networks (NNs) enhanced telerobot control system is designed and tested on a Baxter robot. Guaranteed performance of the telerobot control system is achieved at both kinematic and dynamic levels. At kinematic level, automatic collision avoidance is achieved by the control design at the kinematic level exploiting the joint space redundancy, thus the human operator would be able to only concentrate on motion of robot's end-effector without concern on possible collision. A posture restoration scheme is also integrated based on a simulated parallel system to enable the manipulator restore back to the natural posture in the absence of obstacles. At dynamic level, adaptive control using radial basis function NNs is developed to compensate for the effect caused by the internal and external uncertainties, e.g., unknown payload. Both the steady state and the transient performance are guaranteed to satisfy a prescribed performance requirement. Comparative experiments have been performed to test the effectiveness and to demonstrate the guaranteed performance of the proposed methods.**

***Index Terms*—Collision avoidance, guaranteed performance, neural networks (NNs), telerobot control.**

I. Introduction

In THE last few decades, the teleoperated robots, also known as telerobots, have been widely applied for human unfriendly tasks such as handing radioactive material and searching in dangerous environment. In comparison to the fully automatic robot manipulators used to perform routine task under static and structured environment, telerobots could work in dynamic and unstructured environments to perform more diverse tasks. A stereoscopic images processing-based teleoperated system is proposed in [1], where the stereoscopic images are displayed to the operator to provide assistance in manipulative tasks. In [2], an environment to synthesize the Internet-based teleoperation systems is studied, and the identified delay parameters of an Internet segment are exploited to design a stable controller. In most conventional teleoperation systems, the environmental information is supposed to be feedback to the operators directly, and they thus have to take care of every single interaction with the environment around the manipulator. The operator could manipulate a telerobot in this manner easily when the environment is simple, but in a dynamic and uncertain environment, this approach could result in extreme huge burden to the operator.

Typically, a telerobot system is subject to two types of uncertainties that affect the control performance. One is concerned with the external environment around the robot, e.g., potential obstacles. The other is concerned with the internal uncertainties such as unknown dynamics and varying payload. For the external uncertainties, the partial feedback of telerobot's environment information to the operator may limit the application range of a teleoperation system, while comprehensive feedback may distract the operator from focusing on the task. A visual sensing-based teleoperation method is designed in [5], in which the operator is able to control the motion of each single joint of the robot manipulator, by transferring his/her hand-arm motion captured by the vision system in real time. In such kind of joint-to-joint teleoperation system, human operators take full control of the manipulator including each degree of freedom (DOF). The workload of the operator is thus imaginably high. Therefore, the shared control strategy has attracted much attention [3], [4]. In the shared control framework, a telerobot is partially automatically controlled to assist the neuromotor control of the human operator.

To decrease the workload of the human operator, we consider to employ the shared control framework, and embed an automatic collision avoidance mechanism into the teleoperation system, to enable the telerobot safely interact with a dynamic environment. In this manner, the operator is able to focus on manipulation of the end-effector of the telerobot, which could avoid collision automatically with little influence to the end effector, by using redundancy mechanism. In the previous studies on manipulator collision avoidance, the redundancy of the manipulator is commonly used. The redundant manipulators are studied in [6] and [7], in which the secondary goal is described by a homogeneous solution in the joint space. The solution is decomposed into a particular and a homogeneous component, such that the multiple goals can be

Manuscript received April 19, 2016; accepted May 22, 2016. Date of publication June 21, 2016; date of current version September 14, 2017. This work was supported in part by the Engineering and Physical Sciences Research Council under Grant EP/L026856/2 and Grant EP/J004561/1, in part by the National Natural Science Foundation of China under Grant 61422310 and Grant 61473038, and in part by the Beijing Natural Science Foundation under Grant 4162066. This paper was recommended by Associate Editor P. X. Liu. *(Corresponding author: Chenguang Yang.)*

C. Yang is with the Zienkiewicz Centre for Computational Engineering, Swansea University, Swansea SA1 8EN, U.K. (e-mail: cyang@theiet.org).

X. Wang and H. Ma are with the School of Automation and the State Key Laboratory of Intelligent Control and Decision of Complex System, Beijing Institute of Technology, Beijing 100081, China.

L. Cheng is with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2016.2573837

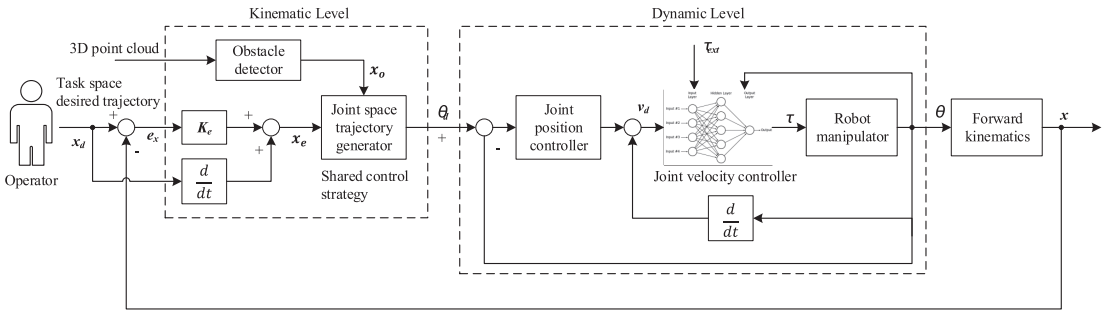


Fig. 1. System diagram.

effectively prioritized. However, the methods mentioned above may face troubles if an obstacle is too close to the base of the robot manipulator. The number of the remaining DOF near the base may not be enough for the manipulator to move along the direction opposite to the obstacle's movement. Thus, the manipulator may encounter a problem of nonsolvable kinematics. A dimension reduction method is thus developed in this paper to solve the this problem and to better exploit the redundancy mechanism. We also consider restoration of the manipulator back to the natural posture, when the obstacle moves far away from the manipulator. For this purpose, a simulated parallel system is introduced based on the kinematics the telerobot manipulator. Therefore, the proposed method could provide a performance guaranteed teleoperation at the kinematic level in an uncertain environment with dynamic obstacles.

The internal uncertainties mainly come from the unmodeled dynamics [8]–[11]. The dynamic control methods of a robot manipulator can be categorized as either model-free control or model-based control. The model-based control methods usually yield a better control performance [13], while most model-free approach may not produce a good transient response. In fact, the control performance heavily depends on the model accuracy, but a perfect dynamics model of the robots could never be available in advance. In addition, the unknown or varying payload makes it impossible to obtain an accurate dynamics model in advance. To solve such problems, the approximation-based control methods have been developed and have been successfully applied on a wide range of practical systems, e.g., formation control [14], multi-agent's consensus control [15], and the robotic manipulator control [16]. The rationale of these approximation enabled control methods is that when the system dynamics satisfy certain conditions, the uncertain nonlinearity can be approximated by tools such as neural network (NN), polynomial approach, wavelet network, and fuzzy logic system [12].

In [20], a multilayer feedforward NN control is proposed for the robot manipulators to compensate for the unknown dynamics. Due to neural learning process, the transient performance of adaptive NN control is usually not discussed, while in this paper, we combine adaptive neural control with an error transformation technique to achieve guaranteed tracking performance at the dynamic level. At the kinematic level, the developed technique ensures obstacle avoidance, and at dynamic level, the NN-based control design is seamlessly integrated with control design at the kinematic level. To our best knowledge, there is little research work in the past to investigate simultaneously guaranteed control performance at both kinematic and dynamic levels.

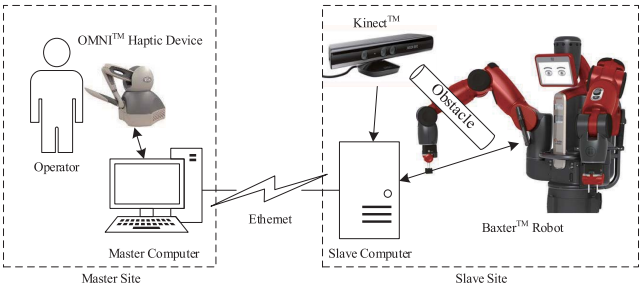


Fig. 2. Components used in the experiment setup [24].

The control strategies designed at both kinematics and dynamics levels are shown in Fig. 1. The design at the kinematic level is to generate a reference trajectory in the joint space for the end-effector of the manipulator to follow operator's command and to simultaneously achieve collision avoidance. The goal of design at the dynamic level is to ensure that the reference trajectory can be tracked satisfying a specified performance requirement in the presence uncertainties.

II. Preprocessing

A. System Components

As illustrated in Fig. 2, a human operator teleoperates the telerobot manipulator by sending command trajectory to its end-effector using the Omni joystick connected to the master computer. One of the Baxter robot's arms is used as telerobot manipulator. All the seven joints (as shown in Fig. 5) of the manipulator will be employed in the experiment. The robot manipulator connected to a slave computer works together with a Kinect sensor to detect obstacles in the surrounding environment.

The Kinect sensor is a red, green, blue plus depth (RGB-D) image sensor developed by Microsoft, as shown in Fig. 3(b). It contains a RGB camera and a depth sensor based on inferred projector. From both the RGB and depth images, we are able to generate a colored 3-D point cloud, such that we could use the Kinect sensor to detect the surrounding environment of the telerobot.

The 6-DOF SensAble Omni joystick (SensAble haptic technologies), as shown in Fig. 3(a), is used in this paper. The

Fig. 3. Devices used in the teleoperation system. (a) Omni joystick. [Captured from: http://www.geomagic.com/]. (b) Kinect sensor. [Captured from: http://www.microsoft.com/].

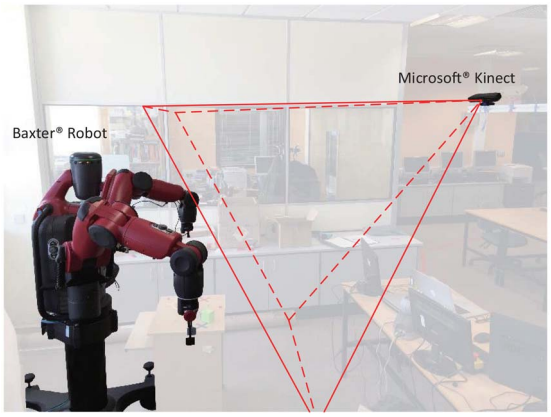


Fig. 4. Setup of the Kinect sensor.

first three joints decide the Cartesian space position of the tip/end-effector. The last three joints decide its orientation in the Cartesian space.

B. Workspace Matching

To make best use of the manipulator work space, the workspace matching between the robot and the Omni joystick is carried out, to make the scaled workspace of the Omni joystick to overlap with the workspace of the telerobot manipulator as much as possible. The point cloud of the end-effector points of both input device and manipulator is created based on Monte Carlo method. The matching process follows our previous work in [22]:

$$\boldsymbol{x_d} = \begin{bmatrix} \cos\beta & -\sin\beta & 0 \\ \sin\beta & \cos\beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \times (\boldsymbol{S_m x_m} + \boldsymbol{T_m}) \tag{1}$$

where $\boldsymbol{x_d} = [x_d\ y_d\ z_d]^T$ with unit m is the Cartesian position of the manipulator's end-effector, and $\boldsymbol{x_m} = [x_m\ y_m\ z_m]^T$ with unit mm is the Cartesian position of the Omni joystick's tip. The revolution angle $\beta = (\pi/4)$ is about the Z-axis of robot manipulator's base frame. The scaling factors and translations are chosen as $\boldsymbol{S_m} = \text{diag}\{0.0041, 0.0040, 0.0041\}$ and $\boldsymbol{T_m} = [0.701, 0.210, 0.129]^T$.

C. Coordinate Transformation

As shown in Fig. 4, a Kinect sensor is set up to detect the obstacles around the robot manipulator. To enable obstacle detection and collision avoidance, it is important to establish a transformation matrix $\boldsymbol{T}$ between the coordinate frames of the robot manipulator and of the Kinect. The $\boldsymbol{T}$ matrix can be obtained by a calibration method proposed in our previous work [23].

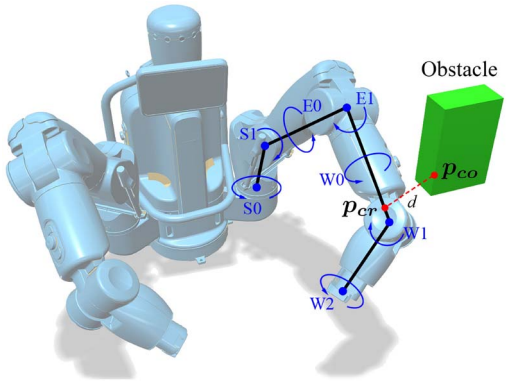


Fig. 5. Illustrations of the collision points $\boldsymbol{p_{cr}}$ and $\boldsymbol{p_{co}}$ and of the Baxter arm's joints.

First, we consider four noncollinear points, and measure their coordinates under both the robot coordinate frame and the Kinect coordinates frame. Let us denote XYZ as the coordinate frame of the robot, and $X'Y'Z'$ as the coordinate frame of the Kinect. Denote the coordinates of the four points as (x_1, y_1, z_1), (x_2, y_2, z_2), (x_3, y_3, z_3), (x_4, y_4, z_4) under coordinate frame XYZ, and as (x'_1, y'_1, z'_1), (x'_2, y'_2, z'_2), (x'_3, y'_3, z'_3), (x'_4, y'_4, z'_4) under coordinate frame $X'Y'Z'$, respectively. Based on these coordinates, we calculate the transformation matrix $\boldsymbol{T}$

$$\boldsymbol{T} = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \\ z_1 & z_2 & z_3 & z_4 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x'_1 & x'_2 & x'_3 & x'_4 \\ y'_1 & y'_2 & y'_3 & y'_4 \\ z'_1 & z'_2 & z'_3 & z'_4 \\ 1 & 1 & 1 & 1 \end{bmatrix}^{-1}. \tag{2}$$

The coordinate transformation between these two coordinates is obtained by $[x\ y\ z\ 1]^T = \boldsymbol{T}[x'\ y'\ z'\ 1]^T$.

D. Identification of Collision Points

First, the continuous k-means clustering method is used to over-segment the point cloud obtained from Kinect into superpixels. The superpixels on the robot will be identified using the robot skeleton model built according to its kinematics. Each robot manipulator's link is regarded as a segment in the 3-D space, as shown in Fig. 5. Based on the forward kinematics, the coordinates of each joint, namely, the Cartesian position of the endpoints of each segment can be calculated in the following manner:

$${}^iX_o = {}^0A_1\,{}^1A_2 \cdots {}^{n-1}A_i X_i \tag{3}$$

where ${}^iX_o = [{}^ix_o, {}^ny_o, {}^nz_o, 1]^T$ and $X_i = [x_i, y_i, z_i, 1]^T$ are augmented position vectors in the Cartesian space. The matrices ${}^{j-1}A_j$ are the homogeneous transform matrices between consecutive links [25]. Based on the segmented point cloud, we could easily generate a 3-D model of the robot in real-time [26], i.e., the red 3-D model consisting of spheres in Fig. 6.

The surrounding points of the 3-D model in the point cloud can be seen as obstacles. The collision points, $\boldsymbol{p_{cr}}$ and $\boldsymbol{p_{co}}$, are shown in Fig. 5. These two points, the former one on the

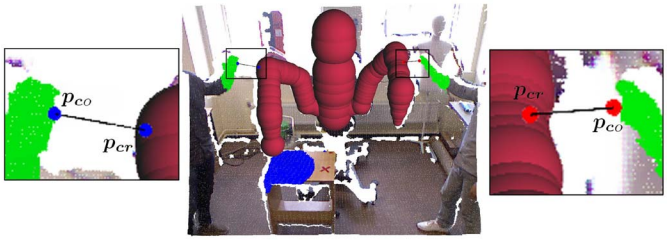


Fig. 6. Illustration of obstacle detection. The red 3-D model consisting of spheres is a simple 3-D model of the robot. The green points in the point cloud denote the obstacle. The blue and red points represent the collision points for the left and the right arms, respectively. The black line indicates the distance measured in between the robot manipulator and the obstacle.

TABLE I
NOMENCLATURE FOR DESIGN AT KINEMATIC LEVEL ($i = 1, 2, 3$)

$\dot{\boldsymbol{\theta}}$	Joint velocities of the manipulator.
$\dot{\boldsymbol{x}}_{\boldsymbol{e}}$	Cartesian velocity of end-effector.
$\boldsymbol{J_e}$	End-effector Jacobian matrix .
$\boldsymbol{x_d}$	Desired position of end-effector commanded by the operator.
$\boldsymbol{e_x}$	End-effector position error.
$\boldsymbol{J}^{\dagger}$	Pseudo-inverse of $\boldsymbol{J}$ defined as $\boldsymbol{J}^{\dagger} = \boldsymbol{J}^T(\boldsymbol{J}\boldsymbol{J}^T)^{-1}$.
$\boldsymbol{p_{co}}$	Collision point on the obstacle.
$\boldsymbol{p_{cr}}$	Collision point on the robot.
d	Distance between the collision points $\boldsymbol{p_{co}}$ and $\boldsymbol{p_{cr}}$.
$\boldsymbol{J_o}$	Collision point $\boldsymbol{p_{cr}}$'s Jacobian matrix.
v_{max}	Maximum collision avoiding velocity.
d_o	Threshold distance to avoid the obstacle.
d_c	Minimum allowed distance from manipulator to obstacle.
$\dot{\boldsymbol{\theta}}_{\boldsymbol{r}}$	Joint velocities of the parallel system.
$\boldsymbol{J_{ri}}$	Jacobian matrices of the joints S_1, E_1 and W_1.
$\dot{\boldsymbol{x}}_{\boldsymbol{ri}}$	Desired velocities of S_1, E_1 and W_1 for restoration.
$\boldsymbol{e_{ri}}$	Position errors of S_1, E_1 between parallel and real systems.

robot and the latter one on the obstacle, if of shortest distance d in between the robot manipulator and the obstacle.

As this paper does not focus on the obstacle detection method, the detection result is directly provided in Fig. 6, where the red 3-D model is the 3-D model of the robot, the green points in the point cloud denote the obstacle, the blue and the red points represent the collision points on the left and the right arms, respectively. The black line segments show the distance in between the robot manipulator and the obstacle.

III. CONTROL STRATEGY AT KINEMATIC LEVEL

The control to be designed at the kinematic level aims to generate a reference trajectory $\dot{\boldsymbol{\theta}}_d$ in the joint space, such that the manipulator could avoid potential collision, as shown in the left dashed box in Fig. 1. More specifically, the control goal is to make the telerobot manipulator's end-effector accurately follow reference trajectory in the Cartesian space commanded by the operator, while simultaneously avoid the obstacle automatically. For the convenience of the readers, the notations used in this section are presented in Table I.

A. Collision Avoiding

Without loss of generality, in this paper we only focus on the cases of collision avoidance that could be achieved using kinematic redundancy mechanism, i.e., the joint motion in the null space of Jacobian $\boldsymbol{J_e}$. Let us now consider one arm of the robot, the kinematics of which is given by

$$\dot{\boldsymbol{x}}_e = \boldsymbol{J_e}\dot{\boldsymbol{\theta}} \tag{4}$$

where the definitions of the joint velocity $\dot{\boldsymbol{\theta}}$, the end-effector velocity $\dot{\boldsymbol{x}}_{\boldsymbol{e}}$, and the Jacobian matrix $\boldsymbol{J_e}$ are provided in Table I. The $\boldsymbol{J_e}$ matrix can be described as

$$\boldsymbol{J_e} = [J_{e1}, J_{e2}, \ldots, J_{en}]$$

where J_{ei}, $1 = 1, 2, \ldots, n$, is the ith column of $\boldsymbol{J_e}$.

The first goal of control design at the kinematics level is to find joint velocities for the manipulator's end-effector to follow the desired position $\boldsymbol{x_d}$ commanded by the operator in the Cartesian space. In order to achieve that, a closed-loop kinematic control law is designed in Cartesian space as below:

$$\dot{\boldsymbol{x}}_e = \dot{\boldsymbol{x}}_d + \boldsymbol{K_e}\boldsymbol{e_x} \tag{5}$$

where $\boldsymbol{e_x} = \boldsymbol{x_d} - \boldsymbol{x_e}$ is the position error for the end-effector in the Cartesian space, defined as the difference between the desired position and the actual position. The positive definite $\boldsymbol{K_e}$ is a gain matrix to be specified by the designer.

The second goal is to avoid any potential collision with minimal effect on the first goal. This can be achieved by exploiting the kinematic redundancy mechanism of the manipulator in the joint space. If the DOF number of a manipulator is larger than the dimension number of the velocity of the end-effector, then the manipulator is regarded as of kinematic redundancy. The inverse kinematics of a kinematically redundant manipulator is not well defined because there are an infinite number of solutions. Using the pseudo-inverse of the Jacobian matrix $\boldsymbol{J}^{\dagger}$ defined in Table I, we give a general inverse kinematics solution as

$$\dot{\boldsymbol{\theta}} = \boldsymbol{J}^{\dagger}\dot{\boldsymbol{x}} + \left(I - \boldsymbol{J}^{\dagger}\boldsymbol{J}\right)z \tag{6}$$

where z is a vector to be used for collision avoidance design [6].

When the collision points are close to the manipulator arm, the manipulator could simply move toward the opposite direction of the obstacle's velocity. The desired avoiding velocity $\dot{\boldsymbol{x}}_o$ must satisfy the kinematic constraint described by

$$\dot{\boldsymbol{x}}_o = \boldsymbol{J_o}\dot{\boldsymbol{\theta}} \tag{7}$$

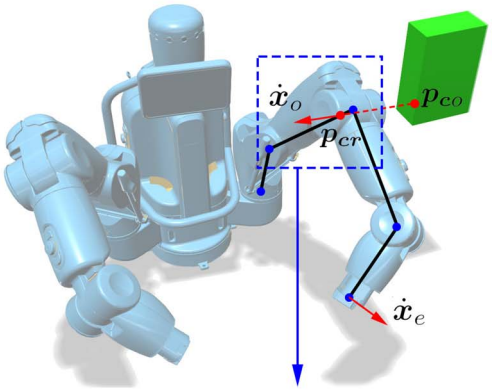
where the Jacobian matrix of the collision point $\boldsymbol{J_o}$ is defined in Table I. To reduce computational complexity, $\boldsymbol{J_o}$ can be simply chosen in the manner

$$\boldsymbol{J_o} = [J_{e1}, \ldots, J_{el}, \boldsymbol{0}, \ldots, \boldsymbol{0}]$$

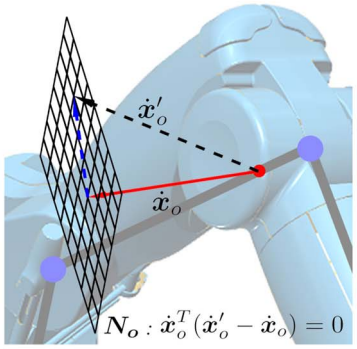
in which the first l columns are taken from $\boldsymbol{J_e}$, and the rest $n-l$ columns are simply zero vectors, where $l \leq n$ is the number of joints that are above the potential collision point $\boldsymbol{p_{cr}}$.

As mentioned above, the collision avoiding velocity can be designed to make the collision point $\boldsymbol{p_{cr}}$ move toward the direction opposite to the obstacle velocity's direction, as graphically illustrated in Fig. 7. It is reasonably to assume that the collision point $\boldsymbol{p_{cr}}$ should move faster when the obstacle is closer. Thus, we design the collision avoiding velocity $\dot{\boldsymbol{x}}_o$ as

$$\dot{\boldsymbol{x}}_o = \begin{cases} \boldsymbol{0}, & d \geq d_o \\ \gamma(d)\boldsymbol{v_{max}}, & d_c < d < d_o \\ \boldsymbol{v_{max}}, & d \leq d_c \end{cases} \tag{8}$$



(a)



(b)

Fig. 7. Illustration of the collision avoiding strategy. (a) Decision of a tentative avoiding velocity $\dot{x}_o$ (8). (b) Alternative avoiding velocity $\dot{x}'_o$ satisfying $\dot{x}'_o = J_o\dot{\theta}$ and $\dot{x}'_o - \dot{x}_o$ fall onto the normal plane N_o of $\dot{x}_o$.

where $\gamma(d) = ((d_o - d)/(d_o - d_c))$, $d = \|p_{cr} - p_{co}\|$ is defined as the distance in between the obstacle and the robot manipulator; $v_{max} = v_{\max}(p_{cr} - p_{co})/d$ is the maximum avoiding velocity vector with opposite direction to the obstacle velocity's direction.

Remark 1: Instead of using the potential field [21] which needs to generate a force vector to be embedded into joint torque input, we employ a simple yet efficient algorithm (8) to decide the avoiding velocity at the kinematic level rather than at the dynamic level. In comparison to the method proposed in [21], our proposed method is of less computational load and easier to be implemented. Moreover, a restoring control will also be developed later such that the manipulator restores the natural posture when the obstacle is gone.

B. Dimension Reduction Method

Consider that when the potential collision point is too close to the manipulator's base, i.e., shoulder mount point, there may be not enough number of DOF left for the robot to achieve the avoiding velocity $\dot{x}_o$, which is of 3-D in the Cartesian space. The rank number of Jacobian matrix J_o could be smaller than the number of dimension of $\dot{x}_o$, e.g., if a potential collision point falls in between the Baxter arm's elbow joint E_1 and its shoulder joint S_0, then the rank number of J_o will be only 2, i.e., only two columns are nonzero vectors. In this case, there may be no solution for the inverse problem of (7) given a 3-D $\dot{x}_o$.

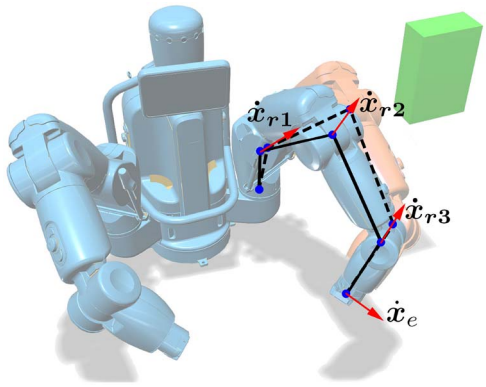


Fig. 8. Simulated parallel system. The skeleton of the actual manipulator is represented by the solid black line, and the dashed black line represent an artificial manipulator simulated in the parallel system.

However, there always exists an alternative avoiding velocity $\dot{x}'_o$ such that: 1) the solution $\dot{\theta}$ satisfying $\dot{x}'_o = J_o\dot{\theta}$ exists and 2) the vector $(\dot{x}'_o - \dot{x}_o)$ falls onto the normal plane N_o of $\dot{x}_o$. As shown in Fig. 7(b), the tentative avoiding velocity $\dot{x}_o$ can then be regarded as a projection of $\dot{x}'_o$ about the normal direction of the plane N_o, and thus the alternative avoiding velocity $\dot{x}'_o$ will a play similar role as $\dot{x}_o$, to drive the manipulator away from a coming obstacle. Consider there is a right angle between vector $\dot{x}_o$ and vector $(\dot{x}'_o - \dot{x}_o)$. The following equality holds:

$$\dot{x}_o^T(\dot{x}'_o - \dot{x}_o) = 0. \tag{9}$$

Substituting $\dot{x}'_o = J_o\dot{\theta}$ into (9), we have

$$\dot{x}_o^T\dot{x}_o = \dot{x}_o^T J_o\dot{\theta}. \tag{10}$$

Consider that (10) is a scalar equation, e.g., $\dot{x}_o^T\dot{x}_o$ is just a a scalar. Therefore, even if the rank number of J_o reduces to 1, the inverse problem of (10) is still solvable. In the following design, we will replace (7) by (10). This dimension reduction method ensures the inverse kinematics solution of $\dot{\theta}$ always exist and thus allow us to more efficiently use the redundancy mechanism of the manipulator.

C. Restoring Control

To eliminate the effect caused by the obstacle when it is gone, ideally the manipulator should restore its natural posture once the obstacle has been removed. We design a parallel system of the manipulator, based on its kinematics, and then simulate its motion in real time ignoring the effect of the obstacle, as shown in Fig. 8. The motion of the simulated artificial manipulator in the parallel system is described by

$$\dot{\theta}_r = J_e^{\dagger}(\theta_r)\dot{x}_e \tag{11}$$

where given the actual manipulator's end effector's position, the joint velocities $\dot{\theta}_r$ of the parallel system are simply calculated from the inverse kinematics.

Consider the Baxter arm's joints shown in Fig. 5. It is observed that the posture of the arm is decided only by the

position of three joints, namely S_1, E_1, and W_1, the Cartesian positions of which are denoted as $\boldsymbol{x_{r1}}$, $\boldsymbol{x_{r2}}$, and $\boldsymbol{x_{r3}}$, respectively. In the restoring process, each of these three joints on the real manipulator is supposed to move to coincide with the same joints of the parallel system. The aforementioned dimension reduction method is applied here as well to avoid the kinematics nonsolvable problem and achieve the position control of all joints simultaneously. The manipulator needs to satisfy (4) and (12) when there is no obstacle around

$$\begin{bmatrix} \dot{\boldsymbol{x}}_{\boldsymbol{r1}} \\ \dot{\boldsymbol{x}}_{\boldsymbol{r2}} \\ \dot{\boldsymbol{x}}_{\boldsymbol{r3}} \end{bmatrix}^T \begin{bmatrix} \dot{\boldsymbol{x}}_{\boldsymbol{r1}} \\ \dot{\boldsymbol{x}}_{\boldsymbol{r2}} \\ \dot{\boldsymbol{x}}_{\boldsymbol{r3}} \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{x}}_{\boldsymbol{r1}} \\ \dot{\boldsymbol{x}}_{\boldsymbol{r2}} \\ \dot{\boldsymbol{x}}_{\boldsymbol{r3}} \end{bmatrix}^T \begin{bmatrix} \boldsymbol{J_{r1}} \\ \boldsymbol{J_{r2}} \\ \boldsymbol{J_{r3}} \end{bmatrix} \dot{\boldsymbol{\theta}} \tag{12}$$

where the Jacobian matrices $[\boldsymbol{J_{r1}^T}, \boldsymbol{J_{r2}^T}, \boldsymbol{J_{r3}^T}]^T$, and joint velocities $[\dot{\boldsymbol{x}}_{\boldsymbol{r1}}^{\boldsymbol{T}}, \dot{\boldsymbol{x}}_{\boldsymbol{r2}}^{\boldsymbol{T}}, \dot{\boldsymbol{x}}_{\boldsymbol{r3}}^{\boldsymbol{T}}]^T$ are defined in Table I. Define $\dot{\boldsymbol{x}}_{\boldsymbol{r}} = [\dot{\boldsymbol{x}}_{\boldsymbol{r1}}^{\boldsymbol{T}}, \dot{\boldsymbol{x}}_{\boldsymbol{r2}}^{\boldsymbol{T}}, \dot{\boldsymbol{x}}_{\boldsymbol{r3}}^{\boldsymbol{T}}]^T$ and $\boldsymbol{J_r} = [\boldsymbol{J_{r1}^T}, \boldsymbol{J_{r2}^T}, \boldsymbol{J_{r3}^T}]^T$. Then (12) can be rewritten as

$$\dot{\boldsymbol{x}}_{\boldsymbol{r}}^{\boldsymbol{T}} \dot{\boldsymbol{x}}_{\boldsymbol{r}} = \dot{\boldsymbol{x}}_{\boldsymbol{r}}^{\boldsymbol{T}} \boldsymbol{J_r} \dot{\boldsymbol{\theta}}. \tag{13}$$

The restoring velocity $\dot{\boldsymbol{x}}_{\boldsymbol{r}}$ is designed based on the feedback position errors, as below:

$$\dot{\boldsymbol{x}}_{\boldsymbol{r}} = \boldsymbol{K_r} \boldsymbol{e_r} \tag{14}$$

where $\boldsymbol{e_r} = [\boldsymbol{e_{r1}^T}, \boldsymbol{e_{r2}^T}, \boldsymbol{e_{r3}^T}]^T$ is vector of position error between simulated artificial manipulator and actual manipulator, and the positive definite gain matrix $\boldsymbol{K_r}$ is to be specified by the designer.

D. Control Design at Kinematic Level

Let us combine the inverse kinematics general solution (6), the dimension reduction equations (10) and (12), then we have the following equations:

$$\dot{\boldsymbol{x}}_{\boldsymbol{o}}^{\boldsymbol{T}} \boldsymbol{J_o} \boldsymbol{J_e^\dagger} \dot{\boldsymbol{x}}_{\boldsymbol{e}} + \dot{\boldsymbol{x}}_{\boldsymbol{o}}^{\boldsymbol{T}} \boldsymbol{J_o} \Big(\boldsymbol{I} - \boldsymbol{J_e^\dagger} \boldsymbol{J_e} \Big) \boldsymbol{z_o} = \dot{\boldsymbol{x}}_{\boldsymbol{o}}^{\boldsymbol{T}} \dot{\boldsymbol{x}}_{\boldsymbol{o}} \tag{15}$$

$$\dot{\boldsymbol{x}}_{\boldsymbol{r}}^{\boldsymbol{T}} \boldsymbol{J_r} \boldsymbol{J_e^\dagger} \dot{\boldsymbol{x}}_{\boldsymbol{e}} + \dot{\boldsymbol{x}}_{\boldsymbol{r}}^{\boldsymbol{T}} \boldsymbol{J_r} \Big(I - \boldsymbol{J_e^\dagger} \boldsymbol{J_e} \Big) \boldsymbol{z_r} = \dot{\boldsymbol{x}}_{\boldsymbol{r}}^{\boldsymbol{T}} \dot{\boldsymbol{x}}_{\boldsymbol{r}}. \tag{16}$$

The solutions of $\boldsymbol{z_o}$ and $\boldsymbol{z_r}$ can be derived from (15) and (16), as given in the following equations:

$$\boldsymbol{z_o} = \Big[\dot{\boldsymbol{x}}_{\boldsymbol{o}}^{\boldsymbol{T}} \boldsymbol{J_o} \Big(\boldsymbol{I} - \boldsymbol{J_e^\dagger} \boldsymbol{J_e} \Big) \Big]^\dagger \Big(\dot{\boldsymbol{x}}_{\boldsymbol{o}}^{\boldsymbol{T}} \dot{\boldsymbol{x}}_{\boldsymbol{o}} - \dot{\boldsymbol{x}}_{\boldsymbol{o}}^{\boldsymbol{T}} \boldsymbol{J_o} \boldsymbol{J_e^\dagger} \dot{\boldsymbol{x}}_{\boldsymbol{e}} \Big) \tag{17}$$

$$\boldsymbol{z_r} = \Big[\dot{\boldsymbol{x}}_{\boldsymbol{r}}^{\boldsymbol{T}} \boldsymbol{J_r} \Big(\boldsymbol{I} - \boldsymbol{J_e^\dagger} \boldsymbol{J_e} \Big) \Big]^\dagger \Big(\dot{\boldsymbol{x}}_{\boldsymbol{r}}^{\boldsymbol{T}} \dot{\boldsymbol{x}}_{\boldsymbol{r}} - \dot{\boldsymbol{x}}_{\boldsymbol{r}}^{\boldsymbol{T}} \boldsymbol{J_r} \boldsymbol{J_e^\dagger} \dot{\boldsymbol{x}}_{\boldsymbol{e}} \Big). \tag{18}$$

In order to smoothly switch in between the obstacle avoidance and the restoration, we employ a weighted sum of (17) and (18), and integrate it into the design of the desired joint velocities as

$$\dot{\boldsymbol{\theta}}_{\boldsymbol{d}} = \boldsymbol{J_e^\dagger} \dot{\boldsymbol{x}}_{\boldsymbol{e}} + \Big(\boldsymbol{I} - \boldsymbol{J_e^\dagger} \boldsymbol{J_e} \Big) [\alpha \boldsymbol{z_o} + (1 - \alpha) \boldsymbol{z_r}] \tag{19}$$

where the weight factor α is chosen in [0, 1], depending on the distance in between the obstacle and the manipulator, as

$$\alpha = \begin{cases} 0, & d \geq d_o \\ \dfrac{d_o - d}{d_o - d_r}, & d_r < d < d_o \\ 1, & d \leq d_r \end{cases} \tag{20}$$

where d_r is the distance threshold that the manipulator start to restore its original pose.

The control strategy at the kinematic level can be obtained by substituting (5) into (19), as

$$\dot{\boldsymbol{\theta}}_{\boldsymbol{d}} = \boldsymbol{J_e^\dagger} (\dot{\boldsymbol{x}}_d + \boldsymbol{K_e} \boldsymbol{e_x}) + \Big(\boldsymbol{I} - \boldsymbol{J_e^\dagger} \boldsymbol{J_e} \Big) [\alpha \boldsymbol{z_o} + (1 - \alpha) \boldsymbol{z_r}]. \tag{21}$$

Lemma 1: Consider the desired velocity $\dot{\theta}_d$ defined in (21). If joint velocity $\dot{\theta}$ completely follows $\dot{\theta}_d$, then the end-effector's position error e_x will asymptotically converge to zero.

Proof: See the Appendix. ■

IV. Control Strategy at Dynamics Level

The control strategy at dynamics level aims to make sure the manipulator follow the joint space trajectory generated from the kinematics level, as shown in Fig. 1. The radial basis function NN (RBFNN) is used to compensate for the unknown dynamics, especially that caused by the unknown payload, to guarantee the steady state performance of the controller. An error transformation method is employed in the design to guarantee the transient performance.

A. Radial Basis Function NN

The effectiveness of the linear-in-parameter RBFNN has been extensively tested by a large number of researchers, and it is theoretically proved that RBFNN is able to approximate any continuous function $\phi(\boldsymbol{\theta}) : \boldsymbol{R^m} \rightarrow \boldsymbol{R}$ arbitrarily close on a compact set $\boldsymbol{\Omega_z} \subset \boldsymbol{R^m}$ as [27], [28]

$$\phi(\boldsymbol{\theta}) = \boldsymbol{W^T Z}(\boldsymbol{\theta}) + \varepsilon_\phi, \ \forall \boldsymbol{\theta} \in \boldsymbol{\Omega_\theta} \tag{22}$$

where $\boldsymbol{W} = [\omega_1, \omega_2, \ldots, \omega_l]^T \in R^l$ is the ideal NN weight vector of constant elements, $\boldsymbol{\theta} \in \boldsymbol{\Omega_\theta} \subset R^m$ is the input vector, ε_ϕ is the bounded approximation error, and $\boldsymbol{Z}(\boldsymbol{\theta}) = [z_1(\boldsymbol{\theta}), z_2(\boldsymbol{\theta}), \ldots, z_l(\boldsymbol{\theta})]^T \in R^l$ is basis function with $z_i(\boldsymbol{\theta})$ chosen as Gaussian functions as below [29]

$$z_i(\boldsymbol{\theta}) = \exp \left[\frac{-(\boldsymbol{\theta} - \boldsymbol{\mu_i})^T (\boldsymbol{\theta} - \boldsymbol{\mu_i})}{\eta_i^2} \right], i = 1, 2, \ldots, l \tag{23}$$

where vector $\boldsymbol{\mu_i} = [\mu_{i1}, \mu_{i2}, \ldots, \mu_{im}]^T \in \boldsymbol{R^m}$ represents the note centers and η_i the variance. The value of the ideal weight vector $\boldsymbol{W}$ minimizes the approximation error ε_z for all $\boldsymbol{\theta} \in \boldsymbol{\Omega_\theta}$ in the following manner:

$$\boldsymbol{W} \stackrel{\text{def}}{=} \arg \min_{\boldsymbol{W'} \in \boldsymbol{R^l}} \Big\{ \sup \Big| \phi(\boldsymbol{\theta}) - \boldsymbol{W'^T S}(\boldsymbol{\theta}) \Big| \Big\}, \boldsymbol{\theta} \in \boldsymbol{\Omega_\theta}. \tag{24}$$

If the number of NN node l is sufficiently large and node centers μ_i are appropriately chosen, the approximation error $|\varepsilon_z|$ could be reduced arbitrarily small.

B. Control Design at Dynamic Level

1) Error Transformation and Joint Position Control Loop: Let us define the joint angle tracking error $\boldsymbol{e_\theta} = \boldsymbol{\theta} - \boldsymbol{\theta_d}$, and then employ the following error transformation functions [12], [30]:

$$e_{\theta i}(t) = \rho(t) R_i \left(P_i \left(\frac{e_{\theta i}(t)}{\rho(t)} \right) \right), \ i = 1, 2, \ldots, n \tag{25}$$

where

$$R_i(x) = \begin{cases} \dfrac{e^x - \delta}{1 + e^x}, & \text{if } e_{\theta i}(0) \geq 0 \\ \dfrac{\delta e^x - 1}{1 + e^x}, & \text{if } e_{\theta i}(0) < 0 \end{cases} \tag{26}$$

and $P_i(\cdot)$ defined below is the inverse function of $R_i(\cdot)$

$$P_i(x) = \begin{cases} \ln \dfrac{x + \delta}{1 - x}, & e_{\theta i}(0) \geq 0 \\ \ln \dfrac{x + 1}{\delta - x}, & e_{\theta i}(0) < 0 \end{cases} \tag{27}$$

with $\rho(t)$ denoting the tracking performance requirement, which is defined as

$$\rho(t) = (\rho_0 - \rho_\infty)e^{-pt} + \rho_\infty \tag{28}$$

where parameters δ, ρ_0, $\rho_\infty < \rho_0$ and p used above are positive constants to be specified by the designer, depending on the requirement of the tracking performance.

The joint position control loop aims to generate the desired joint angular velocities and to guarantee the transient performance of the manipulator. Define $\eta_i(t)$ as

$$\eta_i(t) = P_i\left(\frac{e_{\theta i}(t)}{\rho(t)}\right). \tag{29}$$

Then the joint position control loop is designed as

$$v_{di}(t) = -k_1 \rho(t)\eta_i(t) + \dot{\theta}_{di}(t) + \frac{\dot{\rho}(t)}{\rho(t)} e_{\theta i}(t). \tag{30}$$

Lemma 2 [12], [30]: If $\eta_i(t)$ is bounded, the tracking performance can be tailored by the performance requirement function $\rho(t)$.

Proof: From the definition of $R_i(\cdot)$, we have

$$\begin{aligned} -\delta < R_i(\cdot) < 1, \ \text{if } e_{\theta i}(0) \geq 0 \\ -1 < R_i(\cdot) < \delta, \ \text{if } e_{\theta i}(0) < 0. \end{aligned} \tag{31}$$

If $\eta_i(t)$ is bounded, then from the definition of $\eta_i(t)$, we have

$$\begin{aligned} -\delta\rho(t) < e_{\theta i}(t) < \rho(t), \ \text{if } e_{\theta i}(0) > 0 \\ -\rho(t) < e_{\theta i}(t) < \delta\rho(t), \ \text{if } e_{\theta i}(0) < 0. \end{aligned} \tag{32}$$

Thus, $\rho(t)$ can be seen as the bounding envelope of the error $e_{\theta i}(t)$. For the transient performance, the maximal amplitude of overshoot is bounded by $\delta\rho_0$ and the amplitude of maximal tracking error in the stable phase is bounded by $\max\{\rho_\infty, \delta\rho_\infty\}$.

The 5% settling time, i.e., the required time for the tracking error to reach and stay within a $100\% \pm 5\%$ range is bounded by $(\max\{1, \delta\}/p)\ln(\rho_0 - \rho_\infty/1.05\rho_\infty)$. Therefore, $\rho(t)$ actually regulates both the transient and steady performance. ■

Consider a Lyapunov function $V_1 = (1/2)\boldsymbol{\eta}^T(t)\boldsymbol{\eta}(t)$, which will be used later for stability analysis. Its derivative is given by

$$\dot{V}_1 = \frac{\boldsymbol{\eta}^T(t)\dot{P}(\boldsymbol{\eta}(t))\boldsymbol{e}_v(t)}{\rho(t)} - k_1 \boldsymbol{\eta}^T(t)\dot{P}(\boldsymbol{\eta}(t))\boldsymbol{\eta}(t) \tag{33}$$

where

$$\begin{aligned} \dot{P}(\boldsymbol{\eta}(t)) &= \operatorname{diag}\left(\dot{P}_1(R_1(\eta_1(t))), \ldots, \dot{P}_n(R_n(\eta_n(t)))\right) \\ \boldsymbol{v_d} &= [v_{d1}, v_{d2}, \ldots, v_{dn}]^T \\ \boldsymbol{e_v} &= \dot{\boldsymbol{\theta}} - \boldsymbol{v_d}. \end{aligned} \tag{34}$$

2) Neural-Learning and Joint Velocity Control Loop: The dynamics of the robot manipulator can be described as

$$\boldsymbol{M}(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + \boldsymbol{C}\left(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}\right)\dot{\boldsymbol{\theta}} + \boldsymbol{G'}(\boldsymbol{\theta}) + \boldsymbol{\tau_{ext}} = \boldsymbol{\tau} \tag{35}$$

where $\boldsymbol{M}(\boldsymbol{\theta})$ is the manipulator inertia matrix, $\boldsymbol{C}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})$ is the Coriolis matrix for the manipulator, $\boldsymbol{G'}(\boldsymbol{\theta})$ is the gravity terms and $\boldsymbol{\tau_{ext}}$ denotes the external torque caused by payload. For convenience, let us define $\boldsymbol{G}(\boldsymbol{\theta}) = \boldsymbol{G'}(\boldsymbol{\theta}) + \boldsymbol{\tau_{ext}}$.

The velocity control loop is used to achieve the desired joint angular velocities $\boldsymbol{v_d}$ by applying the control torque $\boldsymbol{\tau}$.

Design the control torque input as follows:

$$\boldsymbol{\tau} = -k_2\boldsymbol{e_v} + \hat{\boldsymbol{M}}\dot{\boldsymbol{v}}_{\boldsymbol{d}} + \hat{\boldsymbol{C}}\boldsymbol{v_d} + \hat{\boldsymbol{G}} + \hat{\boldsymbol{f}} - \frac{\dot{P}(\boldsymbol{\eta}(t))\boldsymbol{\eta}(t)}{\rho(t)} \tag{36}$$

where $\hat{\boldsymbol{G}}(\boldsymbol{\theta})$, $\hat{\boldsymbol{M}}(\boldsymbol{\theta})$, $\hat{\boldsymbol{C}}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})$, and $\hat{\boldsymbol{f}}$ are the estimates of $\boldsymbol{G}(\boldsymbol{\theta})$, $\boldsymbol{M}(\boldsymbol{\theta})$, $\boldsymbol{C}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})$, and $\boldsymbol{f}$, respectively, and $\boldsymbol{f}$ defined later in (41) is a function of $\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}, \boldsymbol{v_d}$, and $\dot{\boldsymbol{v}}_{\boldsymbol{d}}$.

The closed-loop dynamics can then be formulated as follows:

$$\begin{aligned} &\boldsymbol{M}\dot{\boldsymbol{e}}_{\boldsymbol{v}} + \boldsymbol{C}\boldsymbol{e_v} + \boldsymbol{k_2}\boldsymbol{e_v} + \frac{\dot{P}(\boldsymbol{\eta}(t))\boldsymbol{\eta}(t)}{\rho(t)} - \hat{\boldsymbol{f}} \\ &\quad = -\left(\boldsymbol{M} - \hat{\boldsymbol{M}}\right)\dot{\boldsymbol{v}}_{\boldsymbol{d}} - \left(\boldsymbol{C} - \hat{\boldsymbol{C}}\right)\boldsymbol{v_d} - \left(\boldsymbol{G} - \hat{\boldsymbol{G}}\right). \end{aligned} \tag{37}$$

Applying NN approximation technique, we have

$$\begin{aligned} \boldsymbol{M}(\boldsymbol{\theta}) &= \boldsymbol{W}_{\boldsymbol{M}}^T\boldsymbol{Z}_{\boldsymbol{M}}(\boldsymbol{\theta}) + \boldsymbol{\varepsilon}_M(\boldsymbol{\theta}) \\ \boldsymbol{C}\left(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}\right) &= \boldsymbol{W}_{\boldsymbol{C}}^T\boldsymbol{Z}_{\boldsymbol{C}}\left(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}\right) + \boldsymbol{\varepsilon}_C\left(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}\right) \\ \boldsymbol{G}(\boldsymbol{\theta}) &= \boldsymbol{W}_{\boldsymbol{G}}^T\boldsymbol{Z}_{\boldsymbol{G}}(\boldsymbol{\theta}) + \boldsymbol{\varepsilon}_G(\boldsymbol{\theta}) \\ \boldsymbol{f} &= \boldsymbol{W}_{\boldsymbol{f}}^T\boldsymbol{Z}_{\boldsymbol{f}}\left(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}, \boldsymbol{v_d}, \dot{\boldsymbol{v}}_{\boldsymbol{d}}\right) + \boldsymbol{\varepsilon}_f\left(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}, \boldsymbol{v_d}, \dot{\boldsymbol{v}}_{\boldsymbol{d}}\right) \end{aligned} \tag{38}$$

where $\boldsymbol{W_M} \in R^{nl \times n}$, $\boldsymbol{W_C} \in R^{2nl \times n}$, $\boldsymbol{W_G} \in R^{nl \times n}$, and $\boldsymbol{W_f}$ are the ideal NN weight matrices defined as

$$\begin{aligned} \boldsymbol{W_M} &= \left[\boldsymbol{W}_{\boldsymbol{M}_{i,j}}\right], \boldsymbol{W_C} = \left[\boldsymbol{W}_{\boldsymbol{C}_{i,j}}\right] \\ \boldsymbol{W_G} &= \operatorname{diag}\left(\boldsymbol{W}_{\boldsymbol{G}_i}\right), \boldsymbol{W_f} = \operatorname{diag}\left(\boldsymbol{W}_{\boldsymbol{f}i}\right) \end{aligned} \tag{39}$$

where $\boldsymbol{W}_{\boldsymbol{M}_{i,j}} \in R^l$, $\boldsymbol{W}_{\boldsymbol{C}_{i,j}} \in R^{2l}$, $\boldsymbol{W}_{\boldsymbol{G}_i} \in R^l$, and $\boldsymbol{W_{fi}} \in R^l$, $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, n$, as weight vectors defined in (24), are used to approximate element $\boldsymbol{M}_{i,j}(\boldsymbol{\theta}) \in R$, $\boldsymbol{C}_{i,j}(\boldsymbol{\theta}) \in R$, and $\boldsymbol{G}_i(\boldsymbol{\theta}) \in R$, respectively.

The matrices of radial basis functions, namely, $\boldsymbol{Z_M}(\boldsymbol{\theta})$, $\boldsymbol{Z_C}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})$, $\boldsymbol{Z_G}(\boldsymbol{\theta})$, and $\boldsymbol{Z_f}(\boldsymbol{\theta})$ are designed as follows:

$$\begin{aligned} \boldsymbol{Z_M}(\boldsymbol{\theta}) &= \operatorname{diag}(\boldsymbol{Z}_{\boldsymbol{\theta}}, \ldots, \boldsymbol{Z}_{\boldsymbol{\theta}}) \in R^{nl \times n} \\ \boldsymbol{Z_C}\left(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}\right) &= \operatorname{diag}\left(\begin{bmatrix} \boldsymbol{Z}_{\boldsymbol{\theta}} \\ \boldsymbol{Z}_{\dot{\boldsymbol{\theta}}} \end{bmatrix}, \ldots, \begin{bmatrix} \boldsymbol{Z}_{\boldsymbol{\theta}} \\ \boldsymbol{Z}_{\dot{\boldsymbol{\theta}}} \end{bmatrix}\right) \in R^{2nl \times n} \\ \boldsymbol{Z_G}(\boldsymbol{\theta}) &= \left[\boldsymbol{Z}_{\boldsymbol{\theta}}{}^T, \ldots, \boldsymbol{Z}_{\boldsymbol{\theta}}{}^T\right]^T \in R^{nl \times n} \\ \boldsymbol{Z_f}\left(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}, \boldsymbol{v_d}, \dot{\boldsymbol{v}}_{\boldsymbol{d}}\right) &= \left[\bar{\boldsymbol{Z}}^{\boldsymbol{T}}, \ldots, \bar{\boldsymbol{Z}}^{\boldsymbol{T}}\right]^T \in R^{4nl \times n} \end{aligned} \tag{40}$$

where $\boldsymbol{Z_\theta} = [z_1(\boldsymbol{\theta}), z_2(\boldsymbol{\theta}), \ldots z_l(\boldsymbol{\theta})]^T \in R^l$, $\boldsymbol{Z_{\dot\theta}} = [z_1(\dot{\boldsymbol{\theta}}), z_2(\dot{\boldsymbol{\theta}}), \ldots z_l(\dot{\boldsymbol{\theta}})]^T \in R^l$, $\boldsymbol{Z_{v_d}} = [z_1(\boldsymbol{v_d}), z_2(\boldsymbol{v_d}), \ldots z_l(\boldsymbol{v_d})]^T \in R^l$, $\boldsymbol{Z_{\dot v_d}} = [z_1(\dot{\boldsymbol{v}}_{\boldsymbol{d}}), z_2(\dot{\boldsymbol{v}}_{\boldsymbol{d}}), \ldots z_l(\dot{\boldsymbol{v}}_{\boldsymbol{d}})]^T \in R^l$, and $\bar{Z} = [\boldsymbol{Z_\theta}^T, \boldsymbol{Z_{\dot\theta}}^T, \boldsymbol{Z_{v_d}}^T, \boldsymbol{Z_{\dot v_d}}^T]^T \in R^{4l}$, with z_i, $i = 1, 2, \ldots, l$ defined in (23). The NN-based estimates of $\boldsymbol{G(\theta)}$, $\boldsymbol{M(\theta)}$, $\boldsymbol{C(\theta, \dot\theta)}$, and

$$\boldsymbol{f} = \boldsymbol{\varepsilon_M}\dot{\boldsymbol{v}}_{\boldsymbol{d}} + \boldsymbol{\varepsilon_C}\boldsymbol{v_d} + \boldsymbol{\varepsilon_G} \tag{41}$$

can be written as follows:

$$\begin{aligned}\hat{\boldsymbol{M}}(\boldsymbol{\theta}) &= \hat{\boldsymbol{W}}_{\boldsymbol{M}}^{\boldsymbol{T}}\boldsymbol{Z_M}(\boldsymbol{\theta})\\ \hat{\boldsymbol{C}}\left(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}\right) &= \hat{\boldsymbol{W}}_{\boldsymbol{C}}^{\boldsymbol{T}}\boldsymbol{Z_C}\left(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}\right)\\ \hat{\boldsymbol{G}}(\boldsymbol{\theta}) &= \hat{\boldsymbol{W}}_{\boldsymbol{G}}^{\boldsymbol{T}}\boldsymbol{Z_G}(\boldsymbol{\theta})\\ \hat{\boldsymbol{f}} &= \hat{\boldsymbol{W}}_{\boldsymbol{f}}^{\boldsymbol{T}}\boldsymbol{Z_f}\left(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}, \boldsymbol{v_d}, \dot{\boldsymbol{v}}_{\boldsymbol{d}}\right).\end{aligned} \tag{42}$$

By substituting (42) into (37), we have

$$\begin{aligned}&\boldsymbol{M}\dot{\boldsymbol{e}}_{\boldsymbol{v}} + \boldsymbol{C}\boldsymbol{e_v} + k_2\boldsymbol{e_v} + \frac{\dot{P}(\boldsymbol{\eta}(t))\boldsymbol{\eta}(t)}{\rho(t)}\\ &\quad = -\tilde{\boldsymbol{W}}_{\boldsymbol{M}}^{\boldsymbol{T}}\boldsymbol{Z_M}\dot{\boldsymbol{v}}_{\boldsymbol{d}} - \tilde{\boldsymbol{W}}_{\boldsymbol{C}}^{\boldsymbol{T}}\boldsymbol{Z_C}\boldsymbol{v_d} - \tilde{\boldsymbol{W}}_{\boldsymbol{G}}^{\boldsymbol{T}}\boldsymbol{Z_G} - \tilde{\boldsymbol{W}}_{\boldsymbol{f}}^{\boldsymbol{T}}\boldsymbol{Z_f} - \boldsymbol{\varepsilon_f}\end{aligned} \tag{43}$$

where $\tilde{\boldsymbol{W}}_{(\cdot)} = \boldsymbol{W}_{(\cdot)} - \hat{\boldsymbol{W}}_{(\cdot)}$.

Let us consider a second Lyapunov function as

$$\begin{aligned}V_2 = &\frac{1}{2}\boldsymbol{e_v^T M e_v} + \frac{1}{2}\mathrm{tr}\left(\tilde{\boldsymbol{W}}_{\boldsymbol{M}}^{\boldsymbol{T}}\boldsymbol{Q_M}\tilde{\boldsymbol{W}}_{\boldsymbol{M}}\right)\\ &+ \frac{1}{2}\mathrm{tr}\left(\tilde{\boldsymbol{W}}_{\boldsymbol{C}}^{\boldsymbol{T}}\boldsymbol{Q_C}\tilde{\boldsymbol{W}}_{\boldsymbol{C}} + \tilde{\boldsymbol{W}}_{\boldsymbol{G}}^{\boldsymbol{T}}\boldsymbol{Q_G}\tilde{\boldsymbol{W}}_{\boldsymbol{G}} + \tilde{\boldsymbol{W}}_{\boldsymbol{f}}^{\boldsymbol{T}}\boldsymbol{Q_f}\tilde{\boldsymbol{W}}_{\boldsymbol{f}}\right)\end{aligned} \tag{44}$$

where $\boldsymbol{Q_M}$, $\boldsymbol{Q_C}$, $\boldsymbol{Q_G}$, and $\boldsymbol{Q_f}$ are positive definite weight matrices to be specified. Note that $\dot{\tilde{\boldsymbol{W}}}_{(\cdot)} = -\dot{\hat{\boldsymbol{W}}}_{(\cdot)}$, we see the derivative of V can be written as

$$\begin{aligned}\dot{V}_2 = &-\boldsymbol{e_v^T}k_2\boldsymbol{e_v} - \boldsymbol{e_v^T}\boldsymbol{\varepsilon_f} - \frac{\boldsymbol{e_v^T}\dot{P}(\boldsymbol{\eta}(t))\boldsymbol{\eta}(t)}{\rho(t)}\\ &- \mathrm{tr}\left[\tilde{\boldsymbol{W}}_{\boldsymbol{M}}^{\boldsymbol{T}}\left(\boldsymbol{Z_M}\dot{\boldsymbol{v}}_{\boldsymbol{d}}\boldsymbol{e_v^T} + \boldsymbol{Q_M}\dot{\hat{\boldsymbol{W}}}_{\boldsymbol{M}}\right)\right]\\ &- \mathrm{tr}\left[\tilde{\boldsymbol{W}}_{\boldsymbol{C}}^{\boldsymbol{T}}\left(\boldsymbol{Z_C}\boldsymbol{v_d}\boldsymbol{e_v^T} + \boldsymbol{Q_C}\dot{\hat{\boldsymbol{W}}}_{\boldsymbol{C}}\right)\right]\\ &- \mathrm{tr}\left[\tilde{\boldsymbol{W}}_{\boldsymbol{G}}^{\boldsymbol{T}}\left(\boldsymbol{Z_G}\boldsymbol{e_v^T} + \boldsymbol{Q_G}\dot{\hat{\boldsymbol{W}}}_{\boldsymbol{G}}\right)\right]\\ &- \mathrm{tr}\left[\tilde{\boldsymbol{W}}_{\boldsymbol{f}}^{\boldsymbol{T}}\left(\boldsymbol{Z_f}\boldsymbol{e_v^T} + \boldsymbol{Q_f}\dot{\hat{\boldsymbol{W}}}_{\boldsymbol{f}}\right)\right].\end{aligned} \tag{45}$$

The neural learning law is thus designed as follows:

$$\begin{aligned}\dot{\hat{\boldsymbol{W}}}_{\boldsymbol{M}} &= -\boldsymbol{Q_M^{-1}}\left(\boldsymbol{Z_M}\dot{\boldsymbol{v}}_{\boldsymbol{d}}\boldsymbol{e_v^T} + \sigma_M\hat{\boldsymbol{W}}_{\boldsymbol{M}}\right)\\ \dot{\hat{\boldsymbol{W}}}_{\boldsymbol{C}} &= -\boldsymbol{Q_C^{-1}}\left(\boldsymbol{Z_C}\boldsymbol{v_d}\boldsymbol{e_v^T} + \sigma_C\hat{\boldsymbol{W}}_{\boldsymbol{C}}\right)\\ \dot{\hat{\boldsymbol{W}}}_{\boldsymbol{G}} &= -\boldsymbol{Q_G^{-1}}\left(\boldsymbol{Z_G}\boldsymbol{e_v^T} + \sigma_G\hat{\boldsymbol{W}}_{\boldsymbol{G}}\right)\\ \dot{\hat{\boldsymbol{W}}}_{\boldsymbol{f}} &= -\boldsymbol{Q_f^{-1}}\left(\boldsymbol{Z_f}\boldsymbol{e_v^T} + \sigma_f\hat{\boldsymbol{W}}_{\boldsymbol{f}}\right)\end{aligned} \tag{46}$$

where σ_M, σ_C, σ_G, and σ_f are positive parameters to be specified by the designer.

In the following analysis, the boundness of $\boldsymbol{\eta}(t)$ is established, such that both transient and steady state performance of the robot manipulator can be guaranteed. Now, let us consider an overall Lyapunov function as

$$V = V_1 + V_2. \tag{47}$$

The derivative of V is

$$\begin{aligned}\dot{V} = &-k_1\boldsymbol{\eta}^T(t)\dot{\boldsymbol{P}}(\boldsymbol{\eta}(t))\boldsymbol{\eta}(t) - \boldsymbol{e_v^T}k_2\boldsymbol{e_v} - \boldsymbol{e_v^T}\boldsymbol{\varepsilon_f}\\ &+ tr\left[\sigma_M\tilde{\boldsymbol{W}}_{\boldsymbol{M}}^{\boldsymbol{T}}\hat{\boldsymbol{W}}_{\boldsymbol{M}}\right] + tr\left[\sigma_C\tilde{\boldsymbol{W}}_{\boldsymbol{C}}^{\boldsymbol{T}}\hat{\boldsymbol{W}}_{\boldsymbol{C}}\right]\\ &+ tr\left[\sigma_G\tilde{\boldsymbol{W}}_{\boldsymbol{G}}^{\boldsymbol{T}}\hat{\boldsymbol{W}}_{\boldsymbol{G}}\right] + tr\left[\sigma_f\tilde{\boldsymbol{W}}_{\boldsymbol{f}}^{\boldsymbol{T}}\hat{\boldsymbol{W}}_{\boldsymbol{f}}\right].\end{aligned} \tag{48}$$

According to the definition of $\dot{P}(\boldsymbol{\eta}(t))$, we have $\boldsymbol{\eta}^T(t)\dot{P}(\boldsymbol{\eta}(t))\boldsymbol{\eta}(t) \geq (2/(1+\delta))\|\boldsymbol{\eta}(t)\|^2$. Consider the following inequality obtained by Young's inequality:

$$tr\left[\tilde{\boldsymbol{W}}_{(\cdot)}^{\boldsymbol{T}}\hat{\boldsymbol{W}}_{(\cdot)}\right] \leq -\frac{1}{2}\left\|\tilde{\boldsymbol{W}}_{(\cdot)}\right\|_F^2 + \frac{1}{2}\left\|\boldsymbol{W}_{(\cdot)}\right\|_F^2. \tag{49}$$

Then (48) can be further derived as

$$\begin{aligned}\dot{V} \leq &-\frac{2k_1}{1+\delta}\|\boldsymbol{\eta}(t)\|^2 - \left(k_2 - \frac{1}{2}\right)\|\boldsymbol{e_v}\|^2 + \varrho\\ &- \frac{\sigma_M}{2}\left\|\tilde{\boldsymbol{W}}_{\boldsymbol{M}}\right\|_F^2 - \frac{\sigma_C}{2}\left\|\tilde{\boldsymbol{W}}_{\boldsymbol{C}}\right\|_F^2 - \frac{\sigma_G}{2}\left\|\tilde{\boldsymbol{W}}_{\boldsymbol{G}}\right\|_F^2 - \frac{\sigma_f}{2}\left\|\tilde{\boldsymbol{W}}_{\boldsymbol{f}}\right\|_F^2\end{aligned} \tag{50}$$

where $\boldsymbol{\varrho} = (\sigma_M/2)||\boldsymbol{W_M}||_F^2 + (\sigma_C/2)||\boldsymbol{W_C}||_F^2 + (\sigma_G/2)||\boldsymbol{W_G}||_F^2 + (\sigma_f/2)||\boldsymbol{W_f}||_F^2 + (1/2)\boldsymbol{\epsilon_f^2}$ with $\boldsymbol{\epsilon_f}$ the upper limit of $\|\boldsymbol{\varepsilon_f}\|$ over Ω.

Obviously, if $\tilde{\boldsymbol{W}}_{\boldsymbol{M}}$, $\tilde{\boldsymbol{W}}_{\boldsymbol{C}}$, $\tilde{\boldsymbol{W}}_{\boldsymbol{G}}$, $\tilde{\boldsymbol{W}}_{\boldsymbol{f}}$, $\boldsymbol{\eta}(t)$, and $\boldsymbol{e_v}$ satisfy the following inequality:

$$\begin{aligned}&\frac{\sigma_M\left\|\tilde{\boldsymbol{W}}_{\boldsymbol{M}}\right\|_F^2 + \sigma_C\left\|\tilde{\boldsymbol{W}}_{\boldsymbol{C}}\right\|_F^2 + \sigma_G\left\|\tilde{\boldsymbol{W}}_{\boldsymbol{G}}\right\|_F^2 + \sigma_f\left\|\tilde{\boldsymbol{W}}_{\boldsymbol{f}}\right\|_F^2}{2}\\ &\quad + \frac{2k_1}{1+\delta}\|\boldsymbol{\eta}(t)\|^2 + \left(k_2 - \frac{1}{2}\right)\|\boldsymbol{e_v}\|^2 \geq \boldsymbol{\varrho}\end{aligned} \tag{51}$$

then we have $\dot{V} \leq 0$.

Using the LaSalle's theorem and Lemma 1, it is easy to establish stability and convergence results in the following Theorem 1.

Theorem 1: Consider the closed-loop dynamics system consisting of telerobot open loop dynamics (35), the torque control input defined in (36) with neural learning law specified in (46). Semi-global uniformly boundedness of all the closed-loop signals is guaranteed given bounded $\boldsymbol{\theta_d}$ and $\dot{\boldsymbol{\theta}}_{\boldsymbol{d}}$. Particularly, the error signals $\boldsymbol{\eta(t)}$ and $\boldsymbol{e_v}$ will converge to an invariant set $\Omega_s \subseteq \Omega$ defined

$$\begin{aligned}\Omega_s = \Bigg\{&\left(\|\boldsymbol{\eta(t)}\|, \|\boldsymbol{e_v}\|, \|\boldsymbol{W_M}\|, \|\boldsymbol{W_C}\|, \|\boldsymbol{W_G}\|, \left\|\boldsymbol{W_f}\right\|\right)|\\ &\times \frac{\sigma_M\left\|\tilde{\boldsymbol{W}}_{\boldsymbol{M}}\right\|_F^2}{2\boldsymbol{\varrho}} + \frac{\sigma_C\left\|\tilde{\boldsymbol{W}}_{\boldsymbol{C}}\right\|_F^2}{2\boldsymbol{\varrho}} + \frac{\sigma_G\left\|\tilde{\boldsymbol{W}}_{\boldsymbol{G}}\right\|_F^2}{2\boldsymbol{\varrho}}\\ &+ \frac{\sigma_f\left\|\tilde{\boldsymbol{W}}_{\boldsymbol{f}}\right\|_F^2}{2\boldsymbol{\varrho}} + \frac{2k_1}{(1+\delta)\|\boldsymbol{\varrho}}\|\boldsymbol{\eta}(t)\|^2\\ &+ \frac{(2k_2-1)}{2\boldsymbol{\varrho}}\|\boldsymbol{e_v}\|^2 \leq 1\Bigg\}.\end{aligned} \tag{52}$$

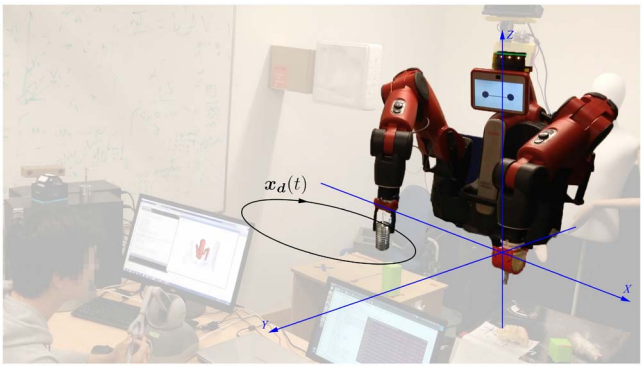


Fig. 9. Experiment setup for the unknown system dynamics estimation.

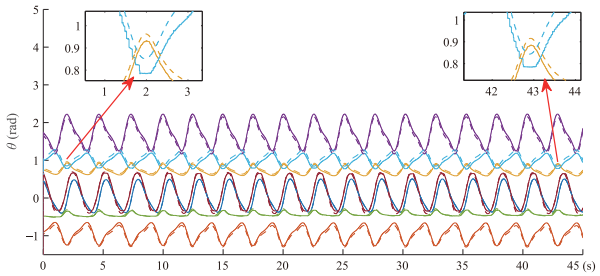


Fig. 10. Reference joint angles $\boldsymbol{\theta}_d$ and the actual joint angles $\boldsymbol{\theta}$ without the neural-learning. The dashed and solid lines indicates reference and actual joint angles, respectively. The lines of different colors indicate different joints.

V. Experiment Studies

A. Test of Neural-Learning Performance

The first group of experiments mainly test the compensation of the effect caused by the unknown dynamics and the uncertain payload. As shown in Fig. 9, the Baxter right arm's end-effector is controlled to move along a fixed trajectory specified by

$$\boldsymbol{x_d}(t) = \begin{bmatrix} 0.6 + 0.1\sin(2\pi t/2.5) \\ -0.4 + 0.3\cos(2\pi t/2.5) \\ 0.2 \end{bmatrix}. \tag{53}$$

A payload is held by the gripper, which has a weight of 1.3 kg. In order to achieve a high precision of the approximation of the 7-DOF robot dynamics, we choose three nodes for each input dimension, and employ totally $l = 3^7$ NN nodes for neural networks $\hat{\boldsymbol{M}}(\boldsymbol{\theta}) = \hat{\boldsymbol{W}}_{\boldsymbol{M}}^{\boldsymbol{T}}\boldsymbol{Z_M}(\boldsymbol{\theta})$ and $\hat{\boldsymbol{G}}(\boldsymbol{\theta}) = \hat{\boldsymbol{W}}_{\boldsymbol{G}}^{\boldsymbol{T}}\boldsymbol{Z_G}(\boldsymbol{\theta})$, $2l$ NN nodes for neural network $\hat{\boldsymbol{C}}(\boldsymbol{\theta}) = \hat{\boldsymbol{W}}_{\boldsymbol{C}}^{\boldsymbol{T}}\boldsymbol{Z_C}(\boldsymbol{\theta})^{\boldsymbol{T}}$ and $4l$ NN nodes for neural network $\hat{\boldsymbol{f}} = \hat{\boldsymbol{W}}_{\boldsymbol{f}}^{\boldsymbol{T}}\boldsymbol{Z_f}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}, \boldsymbol{v_d}, \dot{\boldsymbol{v}}_{\boldsymbol{d}})^{\boldsymbol{T}}$. While the NNs weight matrix are initialized as $\hat{\boldsymbol{W}}_{\boldsymbol{M}}(\boldsymbol{0}) = \boldsymbol{0} \in \mathbb{R}^{nl\times n}$, $\hat{\boldsymbol{W}}_{\boldsymbol{C}}(\boldsymbol{0}) = \boldsymbol{0} \in \mathbb{R}^{2nl\times n}$, $\hat{\boldsymbol{W}}_{\boldsymbol{G}}(\boldsymbol{0}) = \boldsymbol{0} \in \mathbb{R}^{nl\times n}$, and $\hat{\boldsymbol{W}}_{\boldsymbol{f}}(\boldsymbol{0}) = \boldsymbol{0} \in \mathbb{R}^{4nl\times n}$, where $l = 2187$ and $n = 7$.

Two comparative experiments are performed to verify the performance of the proposed neural-learning based controller. In the first experiment, the manipulator with payload is controlled by the controller without neural learning, while in the second experiments, the proposed neural learning is enabled.

For the first experiment, the reference joint angles $\boldsymbol{\theta}_d$ and the actual joint angles $\boldsymbol{\theta}$ are shown in Fig. 10; the joint angle errors $\boldsymbol{e}_\theta$ are shown in Fig. 11. We see that the joint angle errors are relatively high due to the heavy payload.

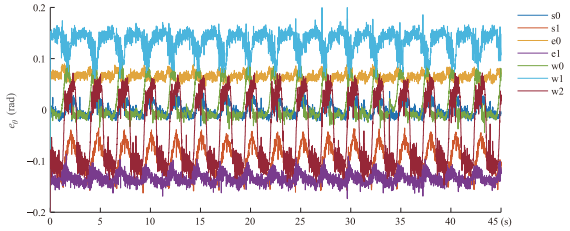


Fig. 11. Joint angle errors $\boldsymbol{e}_\theta$ without the neural-learning. The lines of different colors indicate different joints.

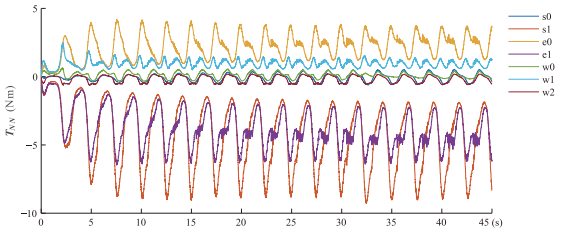


Fig. 12. Compensation torque $\boldsymbol{T_{NN}}$ generated by the neural learning. The lines of different colors indicate different joints.

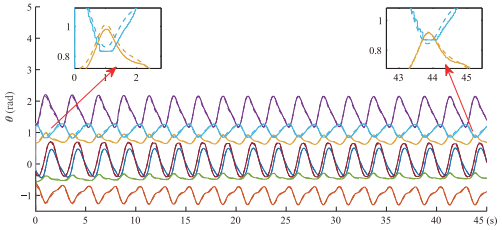


Fig. 13. Reference joint angles $\boldsymbol{\theta}_d$ and the actual joint angles $\boldsymbol{\theta}$ with the neural-learning. The dashed and solid lines indicates reference and actual joint angles, respectively. The lines of different colors indicate different joints.

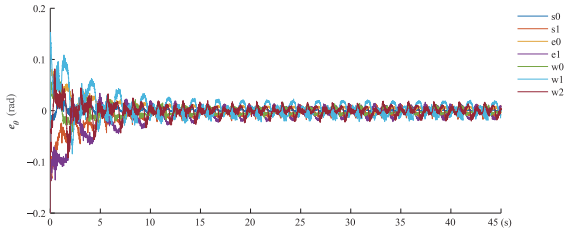


Fig. 14. Joint angle errors $\boldsymbol{e}_\theta$ with the neural-learning. The lines of different colors indicate different joints.

The experiment results with the proposed neural learning are shown in Figs. 12–14, where Fig. 12 shows the compensation torque $\boldsymbol{T_{NN}} = \hat{\boldsymbol{M}}\dot{\boldsymbol{v}}_{\boldsymbol{d}} + \hat{\boldsymbol{C}}\boldsymbol{v_d} + \hat{\boldsymbol{G}} + \hat{\boldsymbol{f}}$. When the payload is attached to the end effector, its gravity effect was more obvious than the dynamic uncertainties of the robot manipulator. Therefore, we particularly show the NN weight of $\hat{W}_G$ in Fig. 15. Fig. 13 shows the reference and actual joint angles and Fig. 14 shows the joint angle errors. We see that at the beginning, the compensation torque is zero and the joint angle error is as large as the results shown in Fig. 11. While later the weight matrices show the trend of convergence. Variance of same periodicity as the reference trajectory can be found in the compensation torque $\boldsymbol{T_{NN}}$. Along with the increment of the

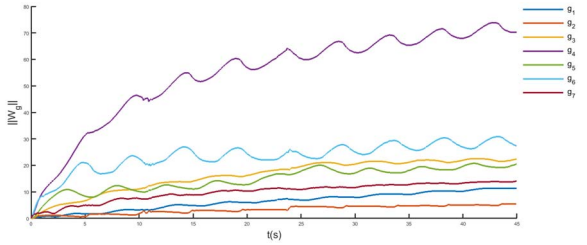


Fig. 15. Norm of each column vector of the NN weight $\hat{W}_G$, corresponding to each row of $\hat{\boldsymbol{G}}$.

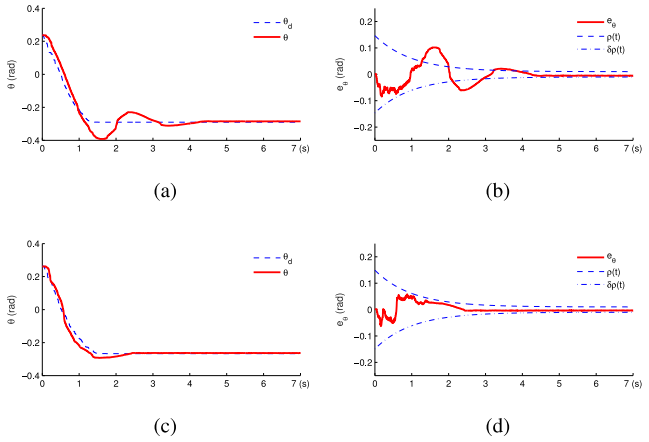


Fig. 16. Experiment results of the tracking performance test. Joint angle (a) without error transformation method, (b) error without error transformation method, (c) with the error transformation method, and (d) error with the error transformation method.

compensation torque, the joint angle errors reduce quickly and become satisfactory after a few cycles, as shown in Fig. 14.

B. Test of Tracking Performance

The second group of experiments mainly focus on the test of tracking performance of the dynamics controller. The end-effector of the manipulator is controlled to move along a straight line between two points, $\boldsymbol{P}_1 : (0.6, -0.2, 0.2)$ and $\boldsymbol{P}_2 : (0.6, -0.6, 0.2)$. Two comparative experiments are performed, with and without the error transformation method. The results of without the error transformation method are shown in Fig. 16(a) and (b). The results with the proposed error transformation method are shown in Fig. 16(c) and (d). Results of only one joint are shown as the results of other joints are very similar. In order to compare the performance between these two experiments, same performance functions $\rho(t)$ are drawn in Fig. 16(b) and (d). We see that the overshoot in the experiment without the error transformation is much larger and the settling time is much longer than the experiment with the error transformation method. It is worth to mention that the joint angle error of the experiment with the proposed method never exceeds the performance requirement during the transient process, as shown in Fig. 16(d). It means that the transient performance satisfies the prescribed requirement (28) with parameters $\delta = 1$, $\rho_0 = 0.15$, $\rho_\infty = 0.02$, and $p = 1.5$.

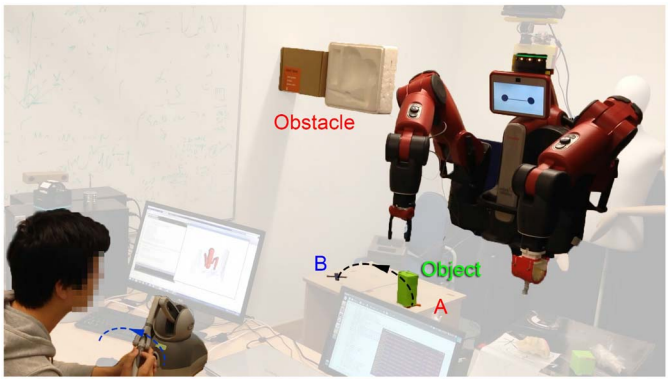
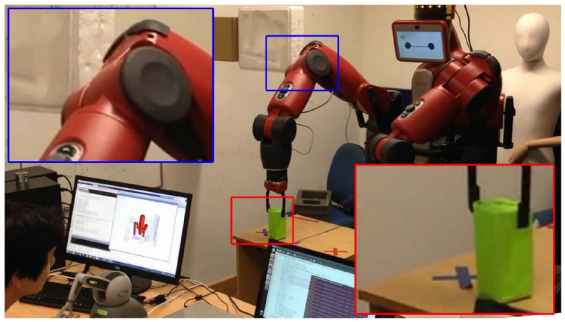
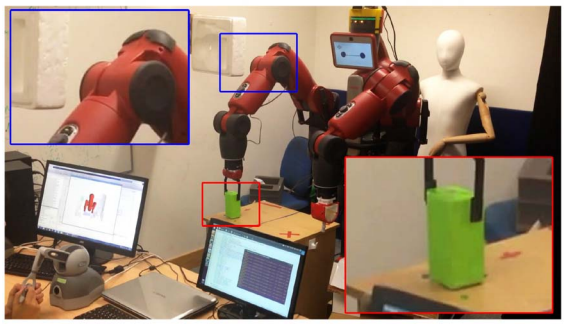


Fig. 17. Set-up of the collision avoidance experiment.



(a)



(b)

Fig. 18. Video frames of the collision avoidance experiments. (a) Without collision avoidance. (b) With collision avoidance.

C. Test of Collision Avoidance

In this group of experiments, the collision avoidance performance is tested. The manipulator is teleoperated by a human operator to move an object between two fixed positions *A* and *B*, as indicated by the red cross and the blue cross on the desk in Fig. 17, in which the green box is the object to be manipulated. Two experiments are performed, with and without the proposed collision avoidance method.

The test results are shown in Figs. 18 and 19. We see that when the robot elbow comes close to the obstacle, the manipulator controlled without the collision avoidance method collide with the obstacle and consequently cannot fulfil the task. In the contrast, the manipulator equipped with the proposed collision avoidance method can adjust its posture to avoid the obstacle and able to complete the task.

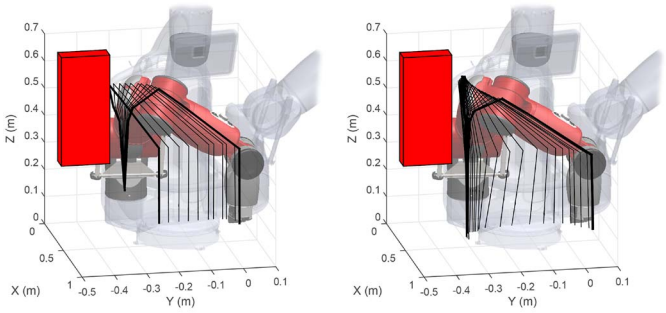


Fig. 19. Time series of the posture of the manipulator. The black lines indicates the manipulator arm, and the red box indicates the obstacle. (a) Without collision avoidance. (b) With collision avoidance.

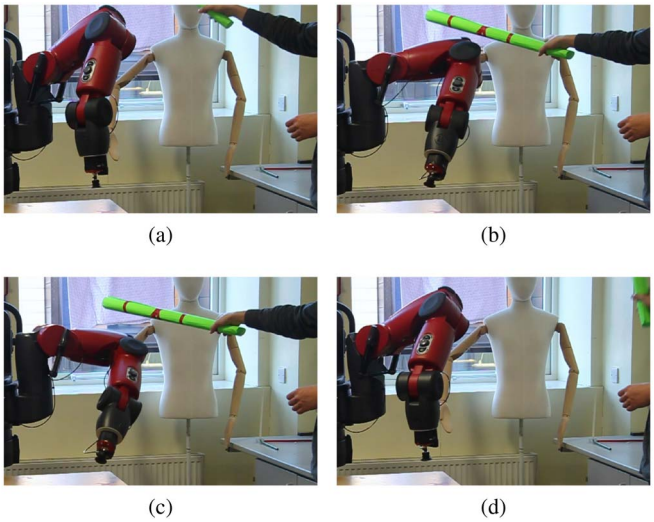


Fig. 20. Video frames of the restoring control experiment. (a) $t = 36$ s. (b) $t = 38$ s. (c) $t = 39$ s. (d) $t = 43$ s.

D. Test of Restoration Function

In the last experiment, the restoration function is tested. For ease of test, the operator holds the joystick statically such that the manipulator's end-effector is maintained at the fixed position, while the obstacle is moving in a dynamic manner. The test results are shown in Fig. 20. We see in the figure that when the obstacle is moving close to the manipulator, its elbow moves down in order avoid possible collision. It is noted that there is nearly no change of the end-effector's position during the collision avoidance. When the obstacle moves away, the manipulator automatically restores back to its previous posture.

VI. Conclusion

In this paper, a telerobot control method with guaranteed performance at both kinematic and dynamic levels is developed. On the kinematic level, a dimension reduction method is developed to overcome the kinematics nonsolvable problem while the manipulator is avoiding obstacle, and to achieve a more efficient use of the redundancy. A simulated parallel system of the manipulator is designed to achieve restoration back to the natural pose in the absence of obstacle. The human operator may not need to consider the surrounding obstacles any more when teleoperating the end-effector of the manipulator. On the dynamic level, a neural-learning based controller is designed to compensate for the uncertainties of manipulator dynamics and the payload. Both transient and steady state control performance are guaranteed by implementing the error transformation method. Extensive experiments have been performed on an arm of the Baxter robot to demonstrate the performance of our developed methods, which could be widely used for various types of applications of robot manipulators, for wide range of tasks such as exploration, rescue, and medical surgery.

Appendix

Proof of Lemma 1

Proof: Following our previous work in [24], we choose a positive definite Lyapunov function $V(\boldsymbol{e_x}) = (1/2)\boldsymbol{e_x}^T\boldsymbol{K_e}\boldsymbol{e_x}$, the derivative of which is $\dot{V} = \boldsymbol{e_x}^T\boldsymbol{K_e}\dot{\boldsymbol{x}}_{\boldsymbol{d}} - \boldsymbol{e_x}^T\boldsymbol{K_e}\dot{\boldsymbol{x}}_{\boldsymbol{e}}$. Combining it with (4), we have

$$\dot{V} = \boldsymbol{e_x}^T\boldsymbol{K_e}\dot{\boldsymbol{x}}_{\boldsymbol{d}} - \boldsymbol{e_x}^T\boldsymbol{K_e}\boldsymbol{J_e}\dot{\boldsymbol{\theta}}_{\boldsymbol{d}}. \tag{54}$$

where $\boldsymbol{\theta} = \boldsymbol{\theta_d}$ is assumed. With the desired joint velocity $\dot{\boldsymbol{\theta}}_{\boldsymbol{d}}$ given in (21), the above equation becomes

$$\dot{V} = \boldsymbol{e_x}^T\boldsymbol{K_e}\dot{\boldsymbol{x}}_{\boldsymbol{d}} - \boldsymbol{e_x}^T\boldsymbol{K_e}\boldsymbol{J_e}\Big[\boldsymbol{J}_{\boldsymbol{e}}^{\dagger}(\dot{\boldsymbol{x}}_d + \boldsymbol{K_e}\boldsymbol{e_x}) + \Big(\boldsymbol{I} - \boldsymbol{J}_{\boldsymbol{e}}^{\dagger}\boldsymbol{J_e}\Big)[\alpha z_o + (1-\alpha)z_r]\Big]. \tag{55}$$

Noting that $\boldsymbol{J_e}\boldsymbol{J}_{\boldsymbol{e}}^{\dagger}$ results in an identity matrix, we could eventually arrive at $\dot{V} = -\boldsymbol{e_x}^T\boldsymbol{K_e}\boldsymbol{J_e}\boldsymbol{J}_{\boldsymbol{e}}^{\dagger}\boldsymbol{K_e}\boldsymbol{e_x} \leq 0$ which implies $\boldsymbol{e_x}$ asymptotically converge to zero.

Acknowledgment

The authors would like to thank Professor Angelo Cangelosi of Plymouth University for the technical support provided.

References

[1] M. Ferre, R. Aracil, and M. Navas, "Stereoscopic video images for telerobotic applications," *J. Robot. Syst.*, vol. 22, no. 3, pp. 131–146, Mar. 2005.

[2] R. Oboe and P. Fiorini, "A design and control environment for Internet-based telerobotics," *Int. J. Robot. Res.*, vol. 17, no. 4 pp. 433–449, 1998.

[3] H. Boessenkool, D. A. Abbink, C. J. M. Heemskerk, and F. C. T. van der Helm, "Haptic shared control improves tele-operated task performance towards performance in direct control," in *Proc. IEEE World Haptics Conf. (WHC)*, Istanbul, Turkey, 2011, pp. 433–438.

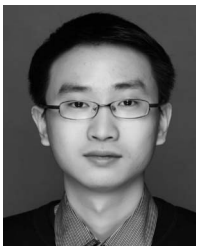
[4] H. Boessenkool, D. A. Abbink, C. J. M. Heemskerk, F. C. T. van der Helm, and J. G. W. Wildenbeest, "A task-specific analysis of the benefit of haptic shared control during telemanipulation," *IEEE Trans. Haptics*, vol. 6, no. 1, pp. 2–12, Mar. 2013.

[5] J. Kofman, X. Wu, T. J. Luu, and S. Verma, "Teleoperation of a robot manipulator using a vision-based human-robot interface," *IEEE Trans. Ind. Electron.*, vol. 52, no. 5, pp. 1206–1219, Oct. 2005.

[6] A. A. Maciejewski and C. A. Klein, "Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments," *Int. J. Robot. Res.*, vol. 4, no. 3, pp. 109–117, 1985.

[7] L. Zlajpah and B. Nemec, "Kinematic control algorithms for on-line obstacle avoidance for redundant manipulators," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, vol. 2. Lausanne, Switzerland, 2002, pp. 1898–1903.

[8] Z. Peng, G. Wen, A. Rahmani, and Y. Yu, "Distributed consensus-based formation control for multiple nonholonomic mobile robots with a specified reference trajectory," *Int. J. Syst. Sci.*, vol. 46, no. 8, pp. 1447–1457, 2015.
[9] G. Wen, Z. Peng, A. Rahmani, and Y. Yu, "Distributed leader-following consensus for second-order multi-agent systems with nonlinear inherent dynamics," *Int. J. Syst. Sci.*, vol. 45, no. 9, pp. 1892–1901, 2014.
[10] Z. Peng, S. Yang, G. Wen, A. Rahmani, and Y. Yu, "Adaptive distributed formation control for multiple nonholonomic wheeled mobile robots," *Neurocomputing*, vol. 173, pp. 1485–1494, Jan. 2016.
[11] Z. Peng, G. Wen, A. Rahmani, and Y. Yu, "Leader–follower formation control of nonholonomic mobile robots based on a bioinspired neurodynamic based approach," *Robot. Auton. Syst.*, vol. 61, no. 9, pp. 988–996, 2013.
[12] L. Cheng, L.-G. Hou, M. Tan, and W. J. Zhang, "Tracking control of a closed-chain five-bar robot with two degrees of freedom by integration of an approximation-based approach and mechanical design," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 5, pp. 1470–1479, Oct. 2012.
[13] A. Smith *et al.*, "Novel hybrid adaptive controller for manipulation in complex perturbation environments," *PLoS One*, vol. 10, no. 6, 2015, Art. no. e0129281.
[14] B. S. Park, J. B. Park, and Y. H. Choi, "Adaptive formation control of electrically driven nonholonomic mobile robots with limited information," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 41, no. 4, pp. 1061–1075, Aug. 2011.
[15] Z. G. Hou, L. Cheng, and M. Tan, "Decentralized robust adaptive control for the multiagent system consensus problem using neural networks," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 3, pp. 636–647, Jun. 2009.
[16] L. Cheng, Z.-G. Hou, and M. Tan, "Adaptive neural network tracking control for manipulators with uncertain kinematics, dynamics and actuator model," *Automatica*, vol. 45, no. 10, pp. 2312–2318, 2009.
[17] J.-Q. Huang and F. L. Lewis, "Neural-network predictive control for nonlinear dynamic systems with time-delay," *IEEE Trans. Neural Netw.*, vol. 14, no. 2, pp. 377–389, Mar. 2003.
[18] F. Lu, J.-Q. Huang, C.-S. Ji, D.-D. Zhang, and H.-B. Jiao, "Gas path online fault diagnostics using a nonlinear integrated model for gas turbine engines," *Int. J. Turbo Jet Engines*, vol. 31, no. 3, pp. 261–275, 2014.
[19] H. Modares, F. L. Lewis, and M.-B. Naghibi-Sistani, "Adaptive optimal control of unknown constrained-input systems using policy iteration and neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 10, pp. 1513–1525, Oct. 2013.
[20] X. Ren, A. B. Rad, and F. L. Lewis, "Neural network-based compensation control of robot manipulators with unknown dynamics," in *Proc. Amer. Control Conf. (ACC)*, New York, NY, USA, Jul. 2007, pp. 13–18.
[21] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 2. St. Louis, MO, USA, Mar. 1985, pp. 500–505.
[22] Z. Ju, C. Yang, Z. Li, L. Cheng, and H. Ma, "Teleoperation of humanoid Baxter robot using haptic feedback," in *Proc. Int. Conf. Multisensor Fusion Inf. Integr. Intell. Syst. (MFI)*, Beijing, China, 2014, pp. 1–6.
[23] C. Yang *et al.*, "Visual servoing control of Baxter robot arms with obstacle avoidance using kinematic redundancy," in *Proc. 8th Int. Conf. Intell. Robot. Appl. (ICIRA)*, Portsmouth, U.K., Aug. 2015, pp. 568–580.
[24] X. Wang, C. Yang, H. Ma, and L. Cheng, "Shared control for teleoperation enhanced by autonomous obstacle avoidance of robot manipulator," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Hamburg, Germany, 2015, pp. 4575–4580.
[25] Z. Ju, C. Yang, and H. Ma, "Kinematics modeling and experimental verification of Baxter robot," in *Proc. 33rd Chin. Control Conf. (CCC)*, Nanjing, China, 2014, pp. 8518–8523.
[26] X. Wang, C. Yang, Z. Ju, H. Ma, and M. Fu, "Robot manipulator self-identification for surrounding obstacle detection," *Multimedia Tools Appl.*, pp. 1–26, 2016, doi: 10.1007/s11042-016-3275-8.
[27] J. Park and I. W. Sandberg, "Universal approximation using radial-basis-function networks," *Neural Comput.*, vol. 3, no. 2, pp. 246–257, 1991.
[28] C. Yang, Z. Li, R. Cui, and B. Xu, "Neural network-based motion control of an underactuated wheeled inverted pendulum model," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 11, pp. 2004–2016, Nov. 2014.
[29] F. L. Lewis, A. Yesildirak, and S. Jagannathan, *Neural Network Control of Robot Manipulators and Nonlinear Systems*. Bristol, PA, USA: Taylor and Francis, 1998.
[30] C. P. Bechlioulis and G. A. Rovithakis, "Prescribed performance adaptive control for multi-input multi-output affine in the control nonlinear systems," *IEEE Trans. Autom. Control*, vol. 55, no. 5, pp. 1220–1226, May 2010.



Chenguang Yang (S'07–M'10–SM'16) 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 was a Post-Doctoral Fellow with Imperial College London, London, U.K. He is a Senior Lecturer with the Zienkiewicz Centre for Computational Engineering, Swansea University, Swansea, U.K. His current research interests include robotics, automation, and computational intelligence.

He was a recipient of the 2011 IEEE TRANSACTIONS ON ROBOTICS Best Paper Award.

Xinyu Wang received the bachelor's degree from the Beijing Institute of Technology, Beijing, China, in 2011, where he is currently pursuing the Ph.D. degree with the School of Automation.

He was a visiting student in U.K. from 2014 to 2015. His current research interests include robotics and autonomous vehicle.

Long Cheng (SM'14) received the B.S. (Hons.) degree in control engineering from Nankai University, Tianjin, China, in 2004, and the Ph.D. (Hons.) degree in control theory and control engineering from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2009.

In 2010, he was a Post-Doctoral Research Fellow with the Department of Mechanical Engineering, University of Saskatchewan, Saskatoon, SK, Canada, for seven months. From 2010 to 2011, he was a Post-Doctoral Research Fellow with the Mechanical and Industrial Engineering Department, Northeastern University, Boston, MA, USA. From 2013 to 2014, he was a Visiting Scholar with the Electrical and Computer Engineering Department, University of California at Riverside, Riverside, CA, USA. He is currently a Professor with the Laboratory of Complex Systems and Intelligent Science, Institute of Automation, Chinese Academy of Sciences. He has published over 50 technical papers in peer-refereed journals and prestigious conference proceedings. His current research interests include intelligent control of smart materials, coordination of multiagent systems, neural networks and their applications to robotics.

Dr. Cheng is an Editorial Board Member of *Neural Processing Letters*, *Neurocomputing*, and the *International Journal of Systems Science*.

Hongbin Ma (M'12) received the bachelor's degree from Zhengzhou University, Zhengzhou, China, in 2001, and the Ph.D. degree from the Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing, China, in 2006.

From 2006 to 2009, he served as a Research Scientist with the Temasek Laboratories, Naitonal University of Singapore, Singapore. He has been a Professor with the School of Automation, Beijing Institute of Technology, Beijing, since 2009. His current research interests include adaptation, learning and recognition, especially applications in robots, self-driving cars, and unmanned aerial vehicle (UAV).

Prof. Ma was a recipient of the 13th Huo Ying Dong Young Teacher Award for Higher Education in 2012.