# Optimized Impedance Adaptation of Robot Manipulator Interacting With Unknown Environment

Xing Liu, *Member, IEEE*, Shuzhi Sam Ge, *Fellow, IEEE*, Fei Zhao, *Member, IEEE*, and Xuesong Mei

***Abstract*— In this brief, impedance adaptation is investigated for robots interacting with unknown environments, subject to unknown environment dynamics and position parameters. A cost function that measures the tracking error and interaction force is defined, and a complete state-space function considering the desired trajectory, environment dynamics, and position parameters is presented. The unknown environment dynamics and the unobservable environment position lead to unknown part of the system function as well as the system states in the complete system model. To this end, the output feedback adaptive dynamic programming (OPFB ADP) method is selected to realize the optimized impedance adaptation. Moreover, when considering the environment position/trajectory, the optimal impedance solution is difficult to obtain due to the existence of the arbitrary endpoint. An adaptive dynamic programming algorithm considering the trajectory tracking problem with arbitrary endpoint is proposed to deal with the problem. The convergence speed is accelerated by adding a discount factor. The validity of the proposed method is verified through simulation and experimental studies.**

***Index Terms*— Adaptive dynamic programming (ADP), optimized impedance adaptation, output feedback, robot–environment interaction, unknown environment.**

## I. Introduction

IN SOCIAL applications such as elderly care, health care, and human–robot collaboration, the dynamics parameters and trajectories of the environments are typically unknown to the robots. Therefore, the control of interaction between robots and environments is essential and there also exist many challenges.

Impedance control regulates the interaction force and the relative motions between the robot and the environment. As such, the robot is governed to comply with the interaction force exerted by the environment and guarantee the safety. In the early research works of impedance control, a desired passive impedance model is usually prescribed, which is too conservative [1]. For robot–environment interaction, other than the robot itself, the environment should also be considered to obtain the desired impedance parameters. As such, a fixed prescribed impedance model does not suffice in many applications and variable impedance control is necessary due to the varying environments [2]. To deal with this, adaptive impedance control [3] and iterative learning impedance control [4] have been studied. For adaptive impedance control method, efficient robot–environment interaction with better performance can be achieved, whereas it is not the optimal solution to the interactive system. For an iterative learning method, it needs repetitive operations of the robots and is somewhat tedious too. Because of these reasons, adaptive optimal control method is more promising and will be focused in this brief. Here, we develop an approach based on adaptive optimal control.

For impedance learning and adaptation, optimization plays an important role because the control objective of impedance control includes both the force regulation and trajectory tracking and, usually, it is the tradeoff of these two objectives [5]. This mimics the human beings' adaptation of force and impedance as a concurrent minimization of instability, motion error, and metabolic cost in muscle space [6]. In the previous literature, the linear quadratic regulator (LQR) is selected to determine the impedance parameters, but the environment dynamics needs to be known [7]. For practical applications, this always does not hold. The adaptive dynamic programming (ADP) method has been widely studied in the previous literature [8]–[11] to realize optimal control of the systems with unknown dynamics. Several publications utilized the ADP method for impedance control of robot manipulator [5], [12], [13]. In our previous work [5], adaptive optimal control or ADP method is selected to acquire the optimized impedance model when the robot interacts with an unknown environment. This method is more promising because it does not need repetitive motions of the robot manipulator. As a result, ADP will be utilized in this brief. However, generally, the above-mentioned work does not take the complete environment parameters, including the dynamics parameters and the positions/trajectories into consideration. This will be the focus of this brief.

In robot–environment or human–robot interaction scenarios, the position of the environment or the virtual desired trajectory generated by the human central nervous system is usually unknown to the robot or very expensive to measure. As such, not all the states of the complete robot–environment interactive system are observable. This is very common in

Manuscript received July 2, 2019; revised December 26, 2019; accepted January 30, 2020. Manuscript received in final form January 31, 2020. This work was supported in part by the National Natural Science Foundation of China under Grant 91748208, in part by the China Scholarship Council under Grant 201706280378, and in part by the Department of Science and Technology of Shaanxi Province under Grant 2017ZDL-G-3-1. Recommended by Associate Editor A. Serrani. *(Corresponding author: Xing Liu.)*

Xing Liu, Fei Zhao, and Xuesong Mei are with the State Key Laboratory for Manufacturing Systems Engineering, Xi'an Jiaotong University, Xi'an 710049, China, also with the Shaanxi Key Laboratory of Intelligent Robots, Xi'an Jiaotong University, Xi'an 710049, China, and also with the School of Mechanical Engineering, Xi'an Jiaotong University, Xi'an 710049, China (e-mail: liuxing1990.ok@163.com; ztzhao@mail.xjtu.edu.cn; xsmei@xjtu.edu.cn).

Shuzhi Sam Ge is with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore, 117576, and also with the Institute for Future, Qingdao University, Qingdao 266071, China (e-mail: samge@nus.edu.sg).

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

Digital Object Identifier 10.1109/TCST.2020.2971944

real applications. In the previous literature, output feedback ADP (OPFB ADP) is widely used for solving this kind of problems [10], [11], [14]. Therefore, the OPFB ADP method is chosen in this brief to determine the optimized impedance parameters in case of unknown environment dynamics as well as positions or trajectories.

From the earlier analysis, we would like to summary our contributions as follows.

1) The optimal impedance adaptation is investigated for robots interacting with environments with unknown dynamics and position parameters. A complete state-space function considering the robot desired trajectory, environment dynamics, and position parameters is presented to address the optimal impedance control problem. There exists an unknown part of the system function as well as the system states in the complete system model since the environment dynamics are unknown and the environment position is unobservable. As such, the OPFB ADP method is selected to realize the optimal control of the complete robot–environment interaction system.
2) For the environment positions/trajectories with arbitrary starting and ending points, the optimal impedance solution is difficult to obtain due to the existence of the final desired position. In this brief, we develop the adaptive optimal control for trajectory tracking with an arbitrary endpoint to address the problem.
3) For better implementation of the proposed algorithm, some improvements are made to the OPFB ADP method. To achieve a faster convergence speed of the proposed algorithm, a discount factor is added. The reason for using a discount factor is also proved.

The rest of this brief is organized as follows. In Section II, the dynamics of the robot and environment are described, and impedance control and the objective of this brief are discussed. In Section III, ADP using output feedback considering the trajectory tracking problem is derived. The implementation procedure is described in Section IV. In Section V, a complete robot–environment interaction model is established. In Section VI, simulations and experimental studies of the above robot–environment interaction problems are made. Then, the conclusion of this brief is given in Section VIII.

## II. Problem Formulation

### A. System Description

The system under study includes a rigid robot manipulator and an environment, where the end-effector of the robot manipulator physically interacts with the environment.

The dynamics of the robot manipulator with $n^\dagger$ DOFs in the joint space are given by

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau + J^T(q)F \tag{1}$$

where $q$, $\dot{q}$, and $\ddot{q}$ denote the position, velocity, and acceleration of the robot joint, respectively, $M(q) \in \mathbb{R}^{n^\dagger \times n^\dagger}$ denotes the inertia matrix, $C(q,\dot{q}) \in \mathbb{R}^{n^\dagger \times n^\dagger}$ is the centrifugal-Coriolis matrix, $G(q) \in \mathbb{R}^{n^\dagger}$ denotes the gravitational term, $\tau \in \mathbb{R}^{n^\dagger}$ is the input torque vector, and $F(t) \in \mathbb{R}^{n^\dagger}$ denotes the interaction force between the robot and the environment, and $J^T(q)$ denotes the transpose of the Jacobian matrix.

The other part of the interaction system is the environment. The mass–damping–stiffness environment is considered in this brief. The environment dynamics are described by the following model:

$$M_e\ddot{x} + C_e\dot{x} + G_e(x - x_e) = -F \tag{2}$$

where $x$, $\dot{x}$, and $\ddot{x}$ denote the position, velocity, and acceleration of the Cartesian space along the direction of the force, respectively, $M_e$, $C_e$, and $G_e$ are mass, damping, and stiffness matrices of the environment models, respectively, and $x_e$ is the position of the contact object. In this brief, we assume that the dynamics parameters of the environments are unknown constants and the position is unknown and varying.

From the perspective of the robot manipulator

$$M_d\ddot{x} + C_d\dot{x} + C_{d1}\dot{x} + G_{d1}(x - x_d) = F \tag{3}$$

where $M_d$, $C_d$, and $G_d$ denote the desired mass, damping, and stiffness parameters of the robot manipulator, respectively, and $x_d$ denotes the desired trajectory of the robot end-effector.

Sum up (2) and (3) and obtain

$$(M_d + M_e)\ddot{x} + C_d\dot{x} + f + C_e\dot{x} + G_e(x - x_e) = 0 \tag{4}$$

where $f = C_{d1}\dot{x} + G_{d1}(x - x_d)$ denotes the noninertial interaction force [15], which we want to minimize.

*Remark 1:* From (2), it can be seen that the complete parameters of an environment include the dynamics parameters $M_e$, $C_e$, and $G_e$ and positions $x_e$. When a robot manipulator physically interacts with objects, e.g., the workpiece, $x_e$ represents the contour or position of the object, whereas $x_e$ represents the virtual desired trajectory of the central nervous system for human–robot interaction. In previous research of robot–environment interaction, a few works consider the complete environment parameters. This motivates our research herein.

### B. Impedance Control

In the two-loop impedance control framework [5], the outer loop generates the virtual desired trajectory according to the interaction force $f$ and the impedance model $Z(\cdot)$. Taking the environment position parameter into consideration, the desired impedance model in the Cartesian space can be written as

$$f = Z(x_0, x_d, x_e) \tag{5}$$

where $x_d$ is the desired trajectory and $x_0$ is the virtual desired trajectory in the Cartesian space. Then, by using the robot kinematics, the virtual desired trajectory in the joint space is obtained and will be tracked in the inner loop. For convenience, we assume that there exists a perfect inner-loop trajectory tracking controller, i.e., $q(t) = q_d(t)$ and $x(t) = x_0(t)$. As such, the desired impedance model becomes

$$f = Z(x, x_d, x_e). \tag{6}$$

In particular, for impedance regulation, the impedance model $Z(\cdot)$ is very important in realizing optimal robot–environment interaction with specified performance index. However, the solution of an optimal impedance model is nontrivial when subject to unknown environment dynamics. Similar to our previous work [5], the ADP method is chosen to resolve the optimal impedance regulation problems.

## III. Adaptive Optimal Control With Output Feedback

### A. Dynamic Programming for Discretized System

*1) Without Considering the Trajectory Tracking Subsystem:* The continuous-time state-space function can be written as

$$\dot{\xi} = A\xi + Bu \quad y = C\xi \tag{7}$$

where $\xi \in \mathbb{R}^n$ is the state, $u \in \mathbb{R}^m$ is the control input, and $y \in \mathbb{R}^p$ is the measured output. Assume throughout that $(A, B)$ is controllable and $(A, C)$ is observable.

A discretized model of (7) using the zero-order hold method is obtained by taking periodic sampling

$$\xi_{k+1} = A_d\xi_k + B_d u_k \quad y_k = C\xi_k \tag{8}$$

where $A_d = e^{Ah}, B_d = (\int_0^h e^{A\tau}\mathrm{d}\tau)B$, and $h > 0$ is a small sampling period. Given a stabilizing control policy $u_k = \mu(\xi_k)$, associated with the system performance index

$$V^\mu(\xi_k) = \sum_{i=k}^{\infty}\left(y_i^T Q_d y_i + u_i^T R_d u_i\right) \tag{9}$$

with weighting matrices $Q_d = Qh \geq 0$, $R_d = Rh > 0$, and $(A_d, \sqrt{Q_d}C)$ being observable and $(A_d, B_d)$ controllable.

For the LQR case, the value function is quadratic such that

$$V^\mu(x_k) = \xi_k^T P_d \xi_k. \tag{10}$$

with matrix $P_d \in \mathbb{R}^{n\times n}$. Substituting this into (9), the LQR Bellman equation is obtained as follows:

$$\xi_k^T P_d \xi_k = y_k^T Q_d y_k + u_k^T R_d u_k + \xi_{k+1}^T P_d \xi_{k+1}. \tag{11}$$

According to [10], the optimal control can be derived as

$$u_k = -\left(R_d + B_d^T P_d B_d\right)^{-1} B_d^T P_d A_d \xi_k = -K_d^* \xi_k \tag{12}$$

where $P_d$ is also the solution of the following Riccati equation:

$$\begin{aligned}0 = {}& A_d^T P_d A_d - P_d + C^T Q_d c\\ &- A_d^T P_d B_d\left(R_d + B_d^T P_d B_d\right)^{-1} B_d^T P_d A_d.\end{aligned} \tag{13}$$

*2) Considering the Trajectory Tracking Subsystem:* Considering the trajectory tracking subsystem, the state-space function in (7) can be rewritten as follows [14]:

$$\dot{\xi} = A\xi + Bu + Gr \quad y = C\xi \tag{14}$$

where $G \in \mathbb{R}^{n\times q}$ denotes the input matrix for the final desired position $r$.

After discretization, the state-space function in (14) is given by

$$\xi_{k+1} = A_d\xi_k + B_d u_k + G_d r_k \quad y_k = C\xi_k \tag{15}$$

where $G_d = (\int_0^h e^{A\tau} d\tau)G$.

Inserting (15) into (11), one has

$$\begin{aligned}\xi_k^T P \xi_k = {}& y_k^T Q_d y_k + u_k^T R_d u_k + (A_d\xi_k + B_d u_k + G_d r_k)^T\\ &\times P_d(A_d\xi_k + B_d u_k + G_d r_k).\end{aligned} \tag{16}$$

Similar to (12), the minimizing control is given by

$$\begin{aligned}u_k &= -\left(R_d + B_d^T P_d B_d\right)^{-1} B_d^T P_d(A_d\xi_k + G_d r_k)\\ &= -\left(R_d + B_d^T P_d B_d\right)^{-1} B_d^T P_d A_d \xi_k\\ &\quad - \left(R_d + B_d^T P_d B_d\right)^{-1} B_d^T P_d G_d r_k.\end{aligned} \tag{17}$$

In (17), also because of the existence of $r_k$, the optimal feedback gain $K_d$ cannot be solved directly. As a result, the Riccati equation (13) may not be derived as in Section II.

Substituting $u_k$ in (17) into (16) obtains

$$\begin{aligned}&\xi_k^T P_d \xi_k\\ &\quad = y_k^T Q_d y_k + u_k^T R_d u_k + (A_d\xi_k + B_d u_k + G_d r_k)^T\\ &\qquad \times P_d(A_d\xi_k + B_d u_k + G_d r_k)\\ &\quad = \xi_k^T C^T Q_d c \xi_k\\ &\qquad + (-T_d A_d \xi_k - T_d G_d r_k)^T R_d(-T_d A_d\xi_k - T_d G_d r_k)\\ &\qquad + ((A_d - B_d T_d A_d)\xi_k + (G_d - B_d T_d G_d)r_k)^T P_d\\ &\qquad \times((A_d - B_d T_d A_d)\xi_k + (G_d - B_d T_d G_d)r_k))\end{aligned} \tag{18}$$

where $T_d = (R_d + B_d^T P_d B_d)^{-1} B_d^T P_d$.

Define $\theta_k = [\xi_k; r_k]$, and then, (18) can be rewritten as follows:

$$\begin{aligned}&\theta_k^T M^T P_d M \theta_k\\ &\quad = \theta_k^T M^T C^T Q_d C M \theta_k\\ &\qquad + \theta_k^T[-T_d A_d \ \ - T_d G_d]^T R_d[-T_d A_d \ \ - T_d G_d]\theta_k\\ &\qquad + \theta_k^T[A_d - B_d T_d A_d \ \ G_d - B_d T_d G_d]^T\\ &\qquad \times P_d[A_d - B_d T_d A_d \ \ G_d - B_d T_d G_d]\theta_k\end{aligned} \tag{19}$$

where $\xi_k = M\theta_k$, with $M$ a constant matrix.

Then, eliminate the variable $\theta_k$ in (19) to obtain

$$\begin{aligned}M^T P_d M = {}& M^T C^T Q_d C M\\ &+ [-T_d A_d \ \ - T_d G_d]^T R_d[-T_d A_d \ \ - T_d G_d]\\ &+ [A_d - B_d T_d A_d \ \ G_d - B_d T_d G_d]^T P_d\\ &\times [A_d - B_d T_d A_d \ \ G_d - B_d T_d G_d].\end{aligned} \tag{20}$$

Based on (20), the kernel matrix $P_d$ can be computed.

### B. VI Algorithm

It is well known that the optimal control can be determined in real time using temporal difference (TD) reinforcement learning (RL) methods [10], which solve online for the value that makes the following Bellman TD error small:

$$e_k = -\xi_k^T P_d \xi_k + y_k^T Q_d y_k + u_k^T R_d u_k + \xi_{k+1}^T P_d \xi_{k+1}. \tag{21}$$

There are two types of RL algorithms, which can be used to iteratively approximate the optimal value and control policy, i.e., policy iteration (PI) and value iteration (VI) algorithms. The VI algorithm does not need an initial control policy. As such, it is chosen in this brief to deal with the optimized impedance control problem. It is noteworthy that this method requires measurement of the full state $\xi_k \in \mathbb{R}^n$.

Consider the following equation:

$$V^*(\xi_k) = \min_{u_k} \left(y_k^T Q_d y_k + u_k^T R_d u_k + V^*(\xi_{k+1})\right) \tag{22}$$

which can be solved using a contraction map by successive approximation as in [10].

## C. ADP Design for Output-Feedback Control

*1) State Reconstruction:* On the basis of the discretized system model (15), the state $\xi_k$ can be reconstructed as follows [10], [11]:

$$\begin{aligned}\xi_k &= M_y \bar{y}_{k-1,k-N} + M_u \bar{u}_{k-1,k-N} + M_r \bar{r}_{k-1,k-N}\\ &= [M_u \ M_y \ M_r] \begin{bmatrix} \bar{u}_{k-1,k-N} \\ \bar{y}_{k-1,k-N} \\ \bar{r}_{k-1,k-N} \end{bmatrix} = \Theta \bar{z}_{k-1,k-N}\end{aligned} \tag{23}$$

where $\bar{z}_{k-1,k-N} = [\bar{u}_{k-1,k-N}^T, \bar{y}_{k-1,k-N}^T, \bar{r}_{k-1,k-N}^T]^T \in \mathbb{R}^l, l = N(m+p+q), \Theta = [M_u, M_y, M_r]$, and $M_u$, $M_y$, and $M_r$ are the parameter matrices, the derivation, and computation details for these matrices can be seen in [10], respectively.

*2) ADP Design for Output-Feedback Control:* Now, we will express the value function in terms of the inputs and outputs. According to (10) and (23), one has

$$\begin{aligned}V^\mu(\xi_k) &= \bar{z}_{k-1,k-N}^T \begin{bmatrix} M_u^T \\ M_y^T \\ M_r^T \end{bmatrix} P_d [M_u \ M_y \ M_r] \bar{z}_{k-1,k-N}\\ &\equiv \bar{z}_{k-1,k-N}^T \bar{P}_d \bar{z}_{k-1,k-N}\end{aligned} \tag{24}$$

where $\bar{P}_d = \Theta^T P_d \Theta \in \mathbb{R}^{(m+p+q)N \times (m+p+q)N}$.

Then, we can write Bellman's equation (11) in terms of the observed data as

$$\begin{aligned}\bar{z}_{k-1,k-N}^T \bar{P}_d \bar{z}_{k-1,k-N} &= y_k^T Q_d y_k + u_k^T R_d u_k\\ &\quad + \bar{z}_{k,k-N+1}^T \bar{P}_d \bar{z}_{k,k-N+1}.\end{aligned} \tag{25}$$

Based on (25), write the TD error (21) in terms of the inputs and outputs as

$$\begin{aligned}e_k = -\bar{z}_{k-1,k-N}^T \bar{P}_d \bar{z}_{k-1,k-N} + y_k^T Q_d y_k + u_k^T R_d u_k\\ + \bar{z}_{k,k-N+1}^T \bar{P}_d \bar{z}_{k,k-N+1}.\end{aligned} \tag{26}$$

Using this TD error, the value update step of the VI algorithm based on the Bellman TD error (21) can be equivalently performed using only the measured data.

The policy improvement step may be written in terms of the observed data as

$$\begin{aligned}\mu(\xi_k) &= \arg\min_{u_k} \left(y_k^T Q_d y_k + u_k^T R_d u_k + x_{k+1}^T P_d x_{k+1}\right)\\ &= \arg\min_{u_k} \left(y_k^T Q_d y_k + u_k^T R_d u_k + \bar{z}_{k,k-N+1}^T \bar{P}_d \bar{z}_{k,k-N+1}\right).\end{aligned} \tag{27}$$

Partition $\bar{z}_{k,k-N+1}^T \bar{P}_d \bar{z}_{k,k-N+1}$ as

$$\bar{z}_{k,k-N+1}^T \bar{P}_d \bar{z}_{k,k-N+1} = \begin{bmatrix} u_k \\ \bar{u}_{k-1,k-N+1} \\ \bar{y}_{k,k-N+1} \\ \bar{r}_{k,k-N+1} \end{bmatrix}^T \begin{bmatrix} p_0 & p_u & p_y & p_r \\ p_u^T & P_{22} & P_{23} & P_{24} \\ p_y^T & P_{32} & P_{33} & P_{34} \\ p_r^T & P_{42} & P_{43} & P_{44} \end{bmatrix} \begin{bmatrix} u_k \\ \bar{u}_{k-1,k-N+1} \\ \bar{y}_{k,k-N+1} \\ \bar{r}_{k,k-N+1} \end{bmatrix}. \tag{28}$$

One has $p_0 \in R^{m\times m}$, $p_u \in \mathbb{R}^{m\times m(N-1)}$, $p_y \in \mathbb{R}^{m\times pn}$, and $p_r \in \mathbb{R}^{m\times qN}$.

Then, similar to [10], differentiating with respect to $u_k$ to perform the minimization in (27) yields

$$\begin{aligned}u_k = -(R_d + p_0)^{-1}(p_u \bar{u}_{k-1,k-N+1} + p_y \bar{y}_{k,k-N+1}\\ + p_r \bar{r}_{k,k-N+1}).\end{aligned} \tag{29}$$

Similar to the work of [10], we also propose the following VI algorithm using OPFB.

**Algorithm 1** IV Algorithm Using OPFB

```
Select an initial control policy $\mu_k^0 = \mu^0$. Then, for $j = 0, 1, \ldots$, perform until convergence.
1. **Value Update**:
    $$\bar{z}_{k-1,k-N}^T \bar{P}_d^{j+1} \bar{z}_{k-1,k-N} = y_k^T Q_d y_k + (u_k^j)^T R_d u_k^j + \bar{z}_{k,k-N+1}^T \bar{P}_d^j \bar{z}_{k,k-N+1}. \quad (30)$$
2. **Policy Improvement**:
    $$u_k^{j+1} = \mu^{j+1}(x_k) = -(R_d + p_0^{j+1})^{-1}(p_u^{j+1} \bar{u}_{k-1,k-N+1} + p_y^{j+1} \bar{y}_{k,k-N+1} + p_r^{j+1} \bar{r}_{k,k-N+1}). \quad (31)$$
where $p_0^{j+1}$ is part of the matrix $\bar{P}_d^{j+1}$.
```

## D. Convergence Speed of the Proposed Algorithm

For an optimized impedance control problem, the convergence speed of the algorithm is very important. Among these methods to improve the convergence speed, adding a discount factor to the controller is very effective and convenient [8], [16].

*Theorem 1:* Let $\delta^j$, $j = 1, 2, 3, \ldots$ be the maximum difference between $V^j(x_k)$ and $V^*(\xi_k)$. When adding a discount factor $\gamma$ to the VI algorithm, the convergence rate of the algorithm to the optimal value $P^*$ is

$$\delta^{j+1} \le \gamma \delta^j. \tag{32}$$

*Proof:* When adding a discount factor to the VI algorithm, we have

$$V^{j+1}(\xi_k) = y_k^T Q_d y_k + \left(u_k^j\right)^T R_d u_k^j + \gamma V^j(\xi_{k+1}). \tag{33}$$

Assume that $V^*(\xi_k)$ is the optimal return value at states $\xi_k$ and the difference between the optimal return value and

$V^j(\xi_k)$ is bounded by $\delta^j$, i.e., $|V^*(\xi_k) - V^j(\xi_k)| \le \delta^j$. It can also be written as follows:

$$V^*(\xi_k) - \delta^j \le V^j(\xi_k) \le V^*(\xi_k) + \delta^j. \quad (34)$$

Then, (33) can also be written as

$$\begin{aligned} V^{j+1}(\xi_k) &= y_k^T Q_d y_k + \left(u_k^j\right)^T R_d u_k^j + \gamma V^j(\xi_{k+1}) \\ &\le y_k^T Q_d y_k + \left(u_k^j\right)^T R_d u_k^j + \gamma (V^*(\xi_{k+1}) + \delta^j) \\ &= y_k^T Q_d y_k + \left(u_k^j\right)^T R_d u_k^j + \gamma V^*(\xi_{k+1}) + \gamma \delta^j \\ &= V^*(\xi_k) + \gamma \delta^j. \end{aligned} \quad (35)$$

In the same way, we can also prove that

$$V^{j+1}(\xi_k) - V^*(\xi_k) \ge -\gamma \delta^j. \quad (36)$$

From (35) and (36), we can get

$$\delta^{j+1} \equiv \max|V^{j+1}(\xi_k) - V^*(\xi_k)| \le \gamma \delta^j. \quad (37)$$

□

Since $0 \le \gamma < 1$, we can draw the conclusion that the existence of the discount factor $\gamma$ will accelerate the convergence speed of the algorithm.

After adding a discount factor, the LQR Bellman equation (25) becomes

$$\begin{aligned} \bar{z}_{k-1,k-N}^T \bar{P}_d \bar{z}_{k-1,k-N} &= y_k^T Q_d y_k + u_k^T R_d u_k \\ &\quad + \gamma \bar{z}_{k,k-N+1}^T \bar{P}_d \bar{z}_{k,k-N+1}. \end{aligned} \quad (38)$$

Then, from [26]–[28], we can achieve the optimal control law as follows:

$$\begin{aligned} u_k = -(R_d/\gamma + p_0)^{-1}(p_u \bar{u}_{k-1,k-N+1} &+ p_y \bar{y}_{k,k-N+1} \\ &+ p_r \bar{r}_{k,k-N+1}). \end{aligned} \quad (39)$$

## IV. Implementation of the Proposed Algorithm

How to effectively implement the proposed algorithm is a key issue. Generally, there are two ways for learning, i.e., online method and offline method. The offline learning method is more suitable for robot–environment interaction scenarios to guarantee the safety and it is chosen in this brief. The implementation procedure is shown as follows.

1) *Step 1:* When the robot manipulator runs for the first time, apply $u_k = -(R_d/\gamma + p_0^0)^{-1}(p_u^0 \bar{u}_{k-1,k-N+1} + p_y^0 \bar{y}_{k,k-N+1}) + p_r^0 \bar{r}_{k,k-N+1}) + e_k$ as the input to the system, where $e_k$ is the exploration noise. Collect the input data $u_k$ and output data $y_k$.
2) *Step 2:* Based on the collected data, employ the improved VI algorithm 2 to iteratively compute the optimal value function and control policy. The stop criterion is $| \bar{P}_d^{j+1} - \bar{P}_d^j | < \epsilon$, with $\epsilon$ a small predefined threshold.
3) *Step 3:* After learning, the optimized $\bar{P}_d$ is obtained and the corresponding control policy is utilized in the system.

**Algorithm 2** Improved VI Algorithm Using OPFB

Select an initial control policy $\mu_k^0 = \mu^0$. Then, for $j = 0, 1, \ldots,$ perform until convergence.

1. **Value Update**:

$$\begin{aligned} \bar{z}_{k-1,k-N}^T \bar{P}_d^{j+1} \bar{z}_{k-1,k-N} &= y_k^T Q_d y_k + (u_k^j)^T R_d u_k^j \\ &+ \gamma \bar{z}_{k,k-N+1}^T \bar{P}_d^{j+1} \bar{z}_{k,k-N+1}. \end{aligned} \quad (40)$$

2. **Policy Improvement**:

$$\begin{aligned} u_k^{j+1} = -(R_d/\gamma + p_0^{j+1})^{-1}(p_u^{j+1} \bar{u}_{k-1,k-N+1} \\ + p_y^{j+1} \bar{y}_{k,k-N+1} + p_r^{j+1} \bar{r}_{k,k-N+1}). \end{aligned} \quad (41)$$

## V. Impedance Adaptation

The mass–damping–stiffness environment described by (2) is considered in this section. By taking the environment dynamics and position parameters into consideration, we can develop an impedance adaptation algorithm to realize the optimized robot–environment interaction. The first step is to determine the cost function for minimization. Similar to [17], the following cost function is given:

$$\Gamma = \int_t^\infty [\dot{x}^T Q_1 \dot{x} + (x - x_d)^T Q_2 (x - x_d) + f^T R f] \mathrm{d}\tau \quad (42)$$

where $Q_1 \in \mathbb{R}$ and $Q_2 \in \mathbb{R}$ are the weights of the velocity and the trajectory tracking error, respectively, and $R \in \mathbb{R}$ is the weight of the interaction force.

Considering the dynamics of the desired trajectory and the environment position, the state variable of the system can be described as follows:

$$\xi = \left[\dot{x}^T, x^T, z_1^T, z_2^T\right]^T \quad (43)$$

where $z_1 \in \mathbb{R}^{n^\dagger}$ is the state of the following system:

$$\begin{cases} \dot{z}_1 = U_1 z_1 \\ x_d = V_1 z_1 \end{cases} \quad (44)$$

where $U_1$ and $V_1$ are two known matrices. The linear system (44) is to determine the desired trajectory $x_d$, which is able to generate a large variety of desired trajectories, e.g., polynomial functions of time of any order [5].

Similarly, $z_2 \in \mathbb{R}^n$ is the state of the following system:

$$\begin{cases} \dot{z}_2 = U_2 z_2 \\ x_e = V_2 z_2 \end{cases} \quad (45)$$

where $U_2$ and $V_2$ are two unknown matrices. The linear system (45) is utilized to determine the varying position of the contact object $x_e$.

Then, according to (42) and (44), we have

$$\begin{aligned} \Gamma &= \int_t^\infty \left[\dot{x}^T Q_1 \dot{x} + \left[x^T x_d^T\right] \begin{bmatrix} Q_2 & -Q_2 \\ -Q_2 & Q_2 \end{bmatrix} \begin{bmatrix} x \\ x_d \end{bmatrix} + f^T R f \right] \mathrm{d}\tau \\ &= \int_t^\infty [y^T Q y + f^T R f] \mathrm{d}\tau \end{aligned} \quad (46)$$

where $Q = \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2' \end{bmatrix}$ with $Q_2' = \begin{bmatrix} Q_2 & -Q_2 V_1 \\ -V_1^T Q_2 & V_1^T Q_2 V_1 \end{bmatrix}$.

Considering the defined state variable (43) and taking the interaction force $f$ in (42) as the "system input" to the robot dynamics, we rewrite (2) as the following state-space form:

$$\dot{\xi} = A\xi + Bf + Gr, \; y = C\xi \tag{47}$$

where

$$A = \begin{bmatrix} -M_1^{-1}C_e & -M_1^{-1}G_e & 0 & M_1^{-1}G_eV_2 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & U_1 & 0 \\ 0 & 0 & 0 & U_2 \end{bmatrix}$$

$$B = \begin{bmatrix} -M_1^{-1} & 0 & 0 & 0 \end{bmatrix}^T, \quad G = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^T$$

$$M_1 = M_d + M_e, \quad r = [r_d \; r_e]^T,$$

and $r_d$ and $r_e$ denote the final desired positions of the robot desired trajectory and the environment contour/trajectory, respectively.

Note that $A$ and $B$ include the environment dynamics and they are unknown. Because the state variable $z_2$ is unmeasurable, the output matrix $C$ is given by

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \tag{48}$$

The state feedback law can be obtained as follows such that the cost function (42) is minimized:

$$f = -K_d^*\xi \tag{49}$$

where $K_d^*$ is the optimal feedback gain.

## VI. Simulation Study

### A. Simulation Conditions

In this section, we consider a robot manipulator with two revolute joints physically interacting with the environment. The simulation is implemented in MATLAB.

The initial position of the robot end-effector in the Cartesian space is $\begin{bmatrix}0.2 \; 0\end{bmatrix}^T$ m. For simplicity, assume that the environment and the robot end-effector are in the same position. It is also assumed that the environment just exerts force along the $x$-direction and the $y^\dagger$-direction is interaction-free. Thus, we just focus on the $x$-direction. The final positions of the robot and environment are set to 0.4 and 0.3 m, respectively, that is to say, $r = \begin{bmatrix}0.4 \; 0.3\end{bmatrix}^T$ m. The environment positions/trajectories between the starting point and the endpoint are unknown.

According to the adaptation procedure in Section IV, three steps are included in a simulation. In the simulation, the sampling interval is set to 1 ms. In Step 1, the exploration noise is added to the initial control input to guarantee the PE conditions. In our simulation, the exploration noise is chosen as $e_k = \Sigma_{\omega=1}^{10}(0.06/\omega)\sin(k\omega)$ and added in this step. Optimal impedance learning is conducted in the second step and it stops when the stopping criterion is satisfied. Then, the optimized impedance model is acquired and used in the third step.

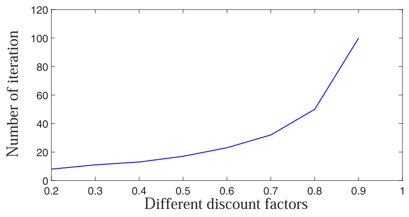

Fig. 1. Convergence speed of the system with four states using different $\gamma$.

For simulation, the following environment is considered in Section VI-B, i.e., $\ddot{x} + 10\dot{x} + 100(x - x_e) = -F$. The position of the environment in the Cartesian space is given by (45), and $U_2$ and $V_2$ are specified as $-1$ and 1, respectively. The robot desired trajectory in the Cartesian space is given by (44) with $U_1$ and $V_1$ specified as $-1$ and 1, respectively.

### B. Simulation Results

For the cost function (46), the parameters $Q_1$, $Q_2$, and $R$ are set to 1, 100 000, and 1, respectively.

The stopping criterion is $| \bar{P}_d^{j+1} - \bar{P}_d^j | < 0.0001$.

To verify the effects of discount factors on the convergence speed, different discount factors have been chosen to implement the iteration process under the same criterion. Fig. 1 clearly shows that with the increase of $\gamma$, the convergence speed will be lowered, which is consistent with Theorem 1. The discount factor $\gamma$ selected here is 0.9.

All the elements of the initial matrix $\bar{P}_d$ are set as 0.0001. After the iteration process of 100 steps, the convergence of the parameters of $\bar{P}_d$, such as $p_0$, $p_u$, $p_y$, and $p_r$, is acquired. Fig. 2 shows the convergence of $p_0$, $p_{u1}$, $p_{u2}$, and $p_{u3}$ of $\bar{P}_d$ to the approximate values. Fig. 3 shows the convergence of $p_{y0}$, $p_{y1}$, $p_{y2}$, and $p_{y3}$ of $\bar{P}_d$ to the approximate values. Fig. 4 shows the convergence of $p_{r0}$, $p_{r1}$, $p_{r2}$, and $p_{r3}$ of $\bar{P}_d$ to the approximate values.

The optimized values of $\bar{P}_d$ are shown as follows:

$$\begin{aligned} p_0^* &= 1.51 \times 10^{-7} \\ p_{u1}^* &= 5.22 \times 10^{-7}, \quad p_{u2}^* = 7.03 \times 10^{-7}, \quad p_{u3}^* = 4.93 \times 10^{-7} \\ p_{y0}^* &= 10^{-3} \times [0.4269, -2.021, 2.090] \\ p_{y1}^* &= 10^{-3} \times [0.121, -2.063, 2.050] \\ p_{y2}^* &= 10^{-3} \times [-0.188, -2.067, 2.047] \\ p_{y3}^* &= 10^{-3} \times [-0.500, -2.066, 2.049] \\ p_{r0}^* &= 10^{-5} \times [7.9, 5.9], \quad p_{r1}^* = 10^{-6} \times [-4.7, -3.5] \\ p_{r2}^* &= 10^{-5} \times [-1.31, -0.98] \\ p_{r3}^* &= 10^{-5} \times [-1.34, -1.00]. \end{aligned}$$

Then, the optimized control policy can be easily derived as

$$\begin{aligned} f_k = &-\left(R_d/\gamma + p_0^*\right)^{-1} \\ &\times \big(p_{u1}^* f_{k-1} + p_{u2}^* f_{k-2} + p_{u3}^* f_{k-3} \\ &\quad + p_{y0}^* y_k + p_{y1}^* y_{k-1} + p_{y2}^* y_{k-2} + p_{y3}^* y_{k-3} \\ &\quad + p_{r0}^* r + p_{r1}^* r + p_{r2}^* r + p_{r3}^* r\big). \end{aligned}$$

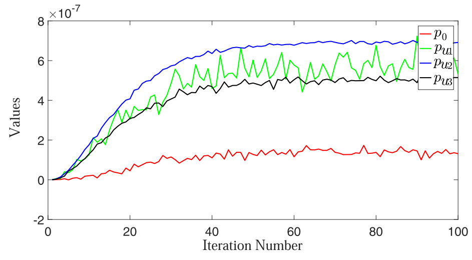

Fig. 2. Convergence of parameters $p_0$ and $p_u$.

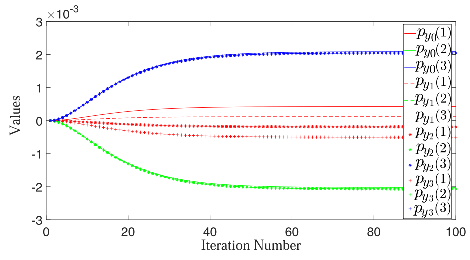

Fig. 3. Convergence of parameters $p_{y0}$, $p_{y1}$, $p_{y2}$, and $p_{y3}$.

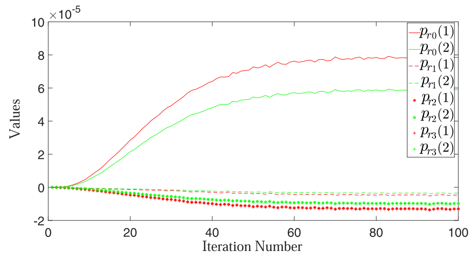

Fig. 4. Convergence of parameters $p_{r0}$, $p_{r1}$, $p_{r2}$, and $p_{r3}$.

According to (24), the optimized kernel matrix $P_d$ is obtained from $\bar{P}_d$ as follows:

$$P_d = \begin{bmatrix} 0.1354 & 7.9573 & -7.8129 & -0.1822 \\ 7.9573 & 984.2995 & -969.1406 & -20.3742 \\ -7.8129 & -969.1406 & 954.2370 & 20.0551 \\ -0.1822 & -20.3742 & 20.0551 & 0.4260 \end{bmatrix}. \tag{50}$$

For comparison, according to (20), using the nominal system function, the numerical solution of the optimized kernel matrix can be computed as follows:

$$P_d^* = \begin{bmatrix} 0.1209 & 7.9740 & -8.0702 & 0.1677 \\ 7.9740 & 984.2743 & -982.8410 & 7.3345 \\ -8.0702 & -982.8410 & 981.6063 & -7.5074 \\ 0.1677 & 7.3345 & -7.5074 & 0.3121 \end{bmatrix}. \tag{51}$$

It can be seen from the above-mentioned two matrices that the optimized kernel matrix obtained from the method in this brief is very close to the numerical solution. This proves the validity of the presented algorithm in this brief well.

### C. Discussion

In conclusion, it has been shown that the proposed method can be adopted to accurately obtain the desired impedance model subject to unknown environment dynamics and position parameters. The improvement of the VI algorithm using OPFB ADP has also been proved to be effective. The discount factor accelerates the convergence speed very well.

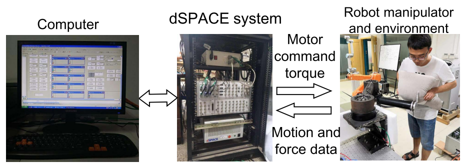

Fig. 5. Two-DOF robot manipulator and the dSPACE control system.

TABLE I
PARAMETERS OF THE TWO-DOF ROBOT MANIPULATOR

| Parameter | Description | Value |
|---|---|---|
| $m_{r1}, m_{r2}$ | Mass of joints 1 and 2 | 7.6489 kg |
| $I_{r1}, I_{r2}$ | Inertia of joints 1 and 2 | 2.2 kgm$^2$ |
| $l_1$ | Length of link 1 | 0.51 m |
| $m_{l1}$ | Mass of link 1 | 8.879 kg |
| $I_{l1}$ | Inertia of link 1 | 0.993 kgm$^2$ |
| $l_2$ | Length of link 2 | 0.4655 m |
| $m_{l2}$ | Mass of link 2 | 1.587 kg |
| $I_{l2}$ | Inertia of link 2 | 0.1507 kgm$^2$ |

## VII. Experimental Study

### A. Experimental Setup

To verify the effectiveness of the proposed method, the robot–environment interaction experiments using the presented algorithm are carried out on a two-DOF robot manipulator, which is shown in Fig. 5 with the dynamics parameters shown in Table I. The dSPACE control platform, in particular, the DS1005 board, is employed as the control system of the robot manipulator. MATLAB/Simulink is used to design and implement the presented algorithm in this brief. The execution and sampling interval for our experiments are chosen as 1 ms. An ATI mini-45 force/torque sensor is mounted at the end-effector of the robot arm. The environment is a stuffed toy with a deformable surface. The initial position of the robot arm in the Cartesian space is $[0.562\ 0]^T$ m. The robots desired task trajectory is from $[0.562\ 0]^T$ m to $[0.491\ 0.159]^T$ m. The force exerted by the environment is only along the $X$-axis, so the robot arm along the $Y$-axis is interaction-free. The mass parameter of the environment is almost equal to zero, and the damping and stiffness parameters are equal to 20 Ns/m and 40 N/m, respectively, which are unknown constant values.

For the cost function (46), the parameters $Q_1$, $Q_2$, and $R$ are set to 10, 1 000 000, and 500, respectively. The stopping criterion is $|\ \bar{P}_d^{j+1} - \bar{P}_d^{j}\ | < 0.01$. It can be seen from Fig. 1 that lower discount factor will lead faster convergence speed. The discount factor $\gamma$ selected here is 0.2.

### B. Experimental Results

The initial value of $p_0$ is set as $-1.5$. The initial value of $p_{y0}$ is set as $[-200\ -1000\ 1000]$. All the other elements of the initial matrix $\bar{P}_d$ is set as 0. Using these parameters as the initial control policy to conduct experiments and collect the data. Then, the data are trained to obtain the optimal control policy. After the iteration process of 21 steps, the convergence of the

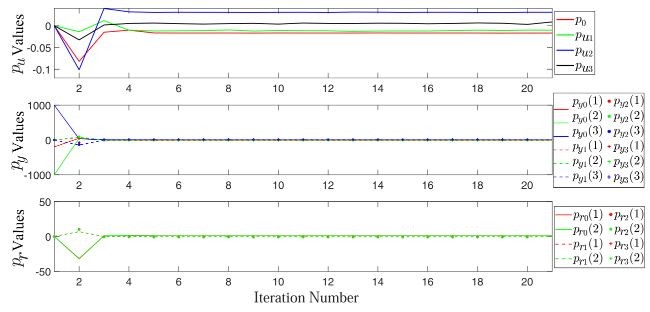
Fig. 6. Convergence of parameters of $\bar{P}_d$ matrix.

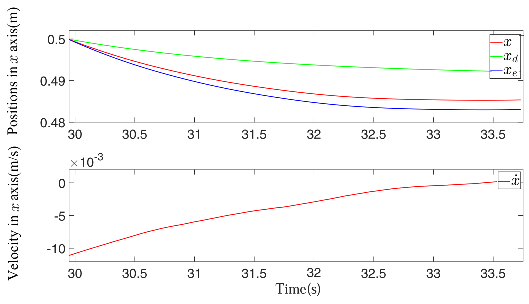
Fig. 7. System states of robot interaction with damping–stiffness environment.

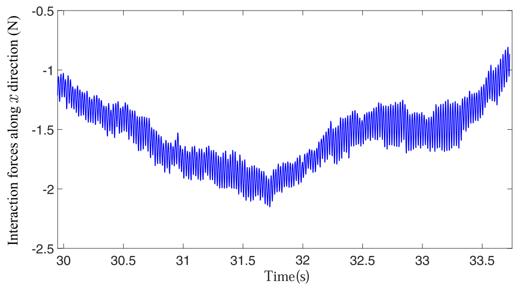
Fig. 8. Interaction forces between the robot and the environment.

parameters of $\bar{P}_d$, such as $p_0$, $p_u$, $p_y$, and $p_r$, is acquired. Fig. 6 shows the convergence of $p_0$, $p_u$, $p_y$, and $p_r$ of $\bar{P}_d$ to the approximate values.

The optimized values of $\bar{P}_d$ are shown as follows:

$$\begin{aligned}
p_0^* &= -0.016863\\
p_{u1}^* &= -0.010768, \quad p_{u2}^* = 0.030369, \quad p_{u3}^* = 0.0048352\\
p_{y0}^* &= [-3.61713, -2.53239, 4.06675]\\
p_{y1}^* &= [-9.92584, -4.70294, 2.26692]\\
p_{y2}^* &= [3.15024, -5.35799, 6.48609]\\
p_{y3}^* &= [6.12401, -5.34676, 5.12428]\\
p_{r0}^* &= [1.86735, 1.83673], \quad p_{r1}^* = [-0.22167, -0.21804]\\
p_{r2}^* &= [-0.84663, -0.83273], \quad p_{r3}^* = [-0.83098, -0.81733].
\end{aligned}$$

After training, the optimized control policy is utilized to control the robot–environment interaction in this brief. Fig. 7 shows the trajectories of the system sates. Fig. 8 shows the interaction force between the robot and the environment.

## C. Discussion

First, from Fig. 6, it can be seen that the training process can converge quickly when using the experimental data. Second, in Fig. 7, the actual trajectory $x$ is between the robot desired trajectory $x_d$ and the environment position/trajectory $x_e$. The actual trajectory $x$ is a tradeoff between $x_d$ and $x_e$. According to different cost functions, e.g., when the $Q_2$ parameter in the cost function (46) is larger, the robot actual trajectory will be closer to the desired trajectory and the interaction force will also be larger. When the $R$ parameter in the cost function (46) is larger, the robot actual trajectory will be closer to the environment position/trajectory and the interaction force will be smaller.

In the past literature [5] regarding the optimized robot–environment interaction problem, the environment position/trajectory is assumed to be constant. In many cases, this is not true. When the robot manipulator interacts with the objects with curved surfaces or with human beings with variable equilibrium point trajectory, the varying position or trajectory of the environments must be considered to obtain the optimized robot–environment interaction performance. The experimental results show that the presented algorithm can adapt to varying environment positions/trajectories. This is an important contribution of this brief.

## VIII. Conclusion

In this brief, impedance adaptation has been developed to realize the optimized robot–environment interaction subject to environments with unknown dynamics and position parameters. A complete state-space function considering the robot desired trajectory, environment dynamics, and position parameters was presented to address the optimized impedance control problem. The ADP method with output feedback was employed as the fundamental of the proposed method to solve the problem of unknown environment positions. For faster convergence speed, a discount factor was added in the algorithm and this method has also been proved. This brief has taken a step in realizing the optimized robot–environment interaction when considering the complete environment characteristics. The simulation results verified the correctness of the proposed algorithm and the experimental results further proved that the presented method can solve the robot–environment interaction problems in real world.

## References

[1] S. P. Buerger and N. Hogan, "Complementary stability and loop shaping for improved human–robot interaction," *IEEE Trans. Robot.*, vol. 23, no. 2, pp. 232–244, Apr. 2007.

[2] T. Tsumugiwa, R. Yokogawa, and K. Hara, "Variable impedance control based on estimation of human arm stiffness for human-robot cooperative calligraphic task," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 1, Jun. 2003, pp. 644–650.

[3] F. Ficuciello, L. Villani, and B. Siciliano, "Variable impedance control of redundant manipulators for intuitive human–robot physical interaction," *IEEE Trans. Robot.*, vol. 31, no. 4, pp. 850–863, Aug. 2015.

[4] Y. Li and S. S. Ge, "Impedance learning for robots interacting with unknown environments," *IEEE Trans. Control Syst. Technol.*, vol. 22, no. 4, pp. 1422–1432, Jul. 2014.

[5] S. S. Ge, Y. Li, and C. Wang, "Impedance adaptation for optimal robot–environment interaction," *Int. J. Control*, vol. 87, no. 2, pp. 249–263, Feb. 2014.

[6] D. W. Franklin *et al.*, "CNS learns stable, accurate, and efficient movements using a simple algorithm," *J. Neurosci.*, vol. 28, no. 44, pp. 11165–11173, Oct. 2008.

[7] M. Matinfar and K. Hashtrudi-Zaad, *Optimization-based Robot Compliance Control: Geometric and Linear Quadratic Approaches*. Newbury Park, CA, USA: Sage, 2005.

[8] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *J. Artif. Intell. Res.*, vol. 4, no. 1, pp. 237–285, Jan. 1996.

[9] D. Vrabie, O. Pastravanu, M. Abu-Khalaf, and F. Lewis, "Adaptive optimal control for continuous-time linear systems based on policy iteration," *Automatica*, vol. 45, no. 2, pp. 477–484, Feb. 2009.

[10] F. L. Lewis and K. G. Vamvoudakis, "Reinforcement learning for partially observable dynamic processes: Adaptive dynamic programming using measured output data," *IEEE Trans. Syst., Man, Cybern. B. Cybern.*, vol. 41, no. 1, pp. 14–25, Feb. 2011.

[11] W. Gao, Y. Jiang, Z.-P. Jiang, and T. Chai, "Output-feedback adaptive optimal control of interconnected systems based on robust adaptive dynamic programming," *Automatica*, vol. 72, pp. 37–45, Oct. 2016.

[12] B. Kim, J. Park, S. Park, and S. Kang, "Impedance learning for robotic contact tasks using natural actor-critic algorithm," *IEEE Trans. Syst., Man, Cybern. B. Cybern.*, vol. 40, no. 2, pp. 433–443, Apr. 2010.

[13] C. Wang, Y. Li, S. S. Ge, and T. H. Lee, "Optimal critic learning for robot control in time-varying environments," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 10, pp. 2301–2310, Oct. 2015.

[14] L. M. Zhu, H. Modares, G. O. Peen, F. L. Lewis, and B. Yue, "Adaptive suboptimal output-feedback control for linear systems using integral reinforcement learning," *IEEE Trans. Control Syst. Technol.*, vol. 23, no. 1, pp. 264–273, Jan. 2015.

[15] Y. Li, S. S. Ge, and C. Yang, "Impedance control for multi-point human-robot interaction," in *Proc. 8th Asian Control Conf. (ASCC)*, May 2011, pp. 1187–1192.

[16] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1238–1274, 2013.

[17] C. Yang and E. Burdet, "A model of reference trajectory adaptation for Interaction with objects of arbitrary shape and impedance," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2011, pp. 4121–4126.