Hindawi
Complexity
Volume 2020, Article ID 9053715, 11 pages
https://doi.org/10.1155/2020/9053715

Research Article

Semiparametric Deep Learning Manipulator Inverse Dynamics Modeling Method for Smart City and Industrial Applications

Nan Liu,[1,2] Liangyu Li,[1] Bing Hao,[3] Liusong Yang,[3] Tonghai Hu,[3] Tao Xue,[2] Shoujun Wang ,[2] and Xingmao Shao[2]

[1]*School of Mechanical Engineering, Tianjin Polytechnic University, Tianjin 300387, China*
[2]*National Demonstration Center for Experimental Mechanical and Electrical Engineering Education, Tianjin University of Technology, Tianjin 300384, China*
[3]*CITIC Heavy Industries Co., Ltd., Luoyang 471003, China*

Correspondence should be addressed to Shoujun Wang; wangshoujun@tjut.edu.cn

Received 14 April 2020; Revised 25 May 2020; Accepted 2 June 2020; Published 30 June 2020

Guest Editor: Zhihan Lv

In smart cities and factories, robotic applications require high accuracy and security, which depends on precise inverse dynamics modeling. However, the physical modeling methods cannot include the nondeterministic factors of the manipulator, such as flexibility, joint clearance, and friction. In this paper, the Semiparametric Deep Learning (SDL) method is proposed to model robot inverse dynamics. SDL is a type of deep learning framework, designed for optimal inference, combining the Rigid Body Dynamics (RBD) model and Nonparametric Deep Learning (NDL) model. The SDL model takes advantage of the global characteristics of classic RBD and the powerful fitting capabilities of the deep learning approach. Moreover, the parametric and nonparametric parts of the SDL model can be optimized at the same time instead of being optimized separately. The proposed method is validated using experiments, performed on a UR5 robotic platform. The results show that the performance of SDL model is better than that of RBD model and NDL model. SDL can always provide relatively accurate joint torque prediction, even when the RBD or NDL model is not accurate.

1. Introduction

Smart cities and factories contain "intelligent" things that can autonomously and collaboratively enhance the quality of living and working conditions, save human lives, and act as a sustainable resource ecosystem. To implement these advanced collaborative technologies, such as drones, robots, artificial intelligence, and Internet of Things, it is required to increase the "intelligence" of smart cities and factories, by improving the connectivity, energy efficiency, and quality of services [1]. There have been many excellent application cases, such as [2–4]. Particularly, intelligent robotic platforms are a technology, increasingly used, in smart cities and factories, where the constantly changing applications scenarios also place higher demands in robot control. Specifically, in motion control systems, there is a time delay in the transmission of feedback information, making smooth motion impossible to achieve by feedback control alone. Therefore, feedforward control becomes particularly important. In robotics, feedforward control usually refers to model-based control, involving the dynamics of the robotic platform. The accuracy of such dynamical models is critical to the development of control laws that are compliant, energy efficient, and safe [5].

There are two major approaches for modeling robot dynamics: parametric and nonparametric. Parametric approaches rely on parameterized Newtonian physics models of the robot dynamics. Common methods for physics-based dynamics modeling can be found in the literature. These methods require the mechanical parameters of the rigid bodies, composing the robot, to be identified [6–9] and then employed in model-based control and state estimation schemes [10]. The advantage of these models is that they represent a global and unique relationship between the joint

trajectory $(q, \dot{q}, \ddot{q})$ and the torques τ_{RBD}. This type of inverse dynamics model can be computed efficiently and employed in real time. Thus, a great deal of prior knowledge is acquired, without the need of data. For example, it is well known that robots are subject to gravitational forces, viscous forces, and joint constraints, making it wasteful to have to go through a laborious data-gathering and machine learning process, to discover these well-known constraints. The disadvantage of parametric models is that they are only crude idealizations of the actual system dynamics, such as rigidity of links or a simple analytical form of friction, which may not be accurate in real systems. In the case of traditional industrial robots, these unmodeled dynamics can often be ignored. However, for modern robotic platforms, these omittances and simplifications result in significant control inefficiencies.

Alternatively, the model can be obtained from experimental data, using machine learning techniques, resulting in a nonparametric model. Nonparametric methods, based on algorithms such as Support Vector Regression (SVR) [11–13], Neural Network (NN) [14–16], Local Weighted Projection Regression (LWPR) [17–19], Independent Joint Learning (IJL) [20–22], or Gaussian Processes Regression (GPR) [23–27], can model dynamics by extrapolating the input-output relationship directly from the available data. If a suitable kernel function or learning architecture is selected, then the nonparametric model is a universal approximator which can account for the dynamics factors, not considered by the parametric model. Therefore, nonparametric methods can be more flexible to use and are powerful in capturing higher order nonlinearities, resulting in faster model approximation and higher learning accuracy. When learning inverse dynamics, the nonparametric methods will approximate a function describing the relationship $q, \dot{q}, \ddot{q} \longrightarrow \tau$, including all nonlinearities encoded by the sampled data.

Nonparametric methods attempt to learn the model from scratch and, thus, do not make use of any knowledge available from analytical robotics. Nevertheless, nonparametric learning methods also exhibit several drawbacks. First, very large amounts of data are necessary for obtaining a sufficiently accurate model and predictions on the entire input space [28]. Second, since nonparametric models rely on local neighborhood training data to make predictions, they do not generalize well to unexplored state regions, where little or no training data are available. Covering the entire state space becomes exponentially harder, as the complexity and number of degrees of freedom in the robot system increase. Thus, if only small and relatively poor data sets are available, nonparametric models will not be able to generalize well for unknown data. Third, it is indeed wasteful to have to go through a laborious data-gathering and machine learning process to discover such well-known prior knowledge as Rigid Body Dynamics.

Thus, it appears quite desirable to combine the benefits of parametric and nonparametric approaches to improve on the aforementioned issues. However, doing so, in an efficient way, is not trivial. A reasonable approach would be to first fit a parametric model and then fit a nonparametric model to the errors made by the parametric model. Nguyen-Tuong et al. [29] present a learning technique which combines prior knowledge about the physical structure of the mechanical system and learning from available data using Gaussian Process Regression (GPR) [30]. Similar approaches are presented in [20] and [31]. In [32], an incremental semiparametric robot dynamics learning scheme, based on Locally Weighted Projection Regression (LWPR), initialized using a linearized parametric model, is presented [33]. However, this approach uses a fixed parametric model that is not updated, as new data become available. Moreover, LWPR has been shown to underperform with respect to other methods (e.g., [34]). These semiparametric methods, as described above, could not benefit from simultaneous optimization of parametric and nonparametric models. Instead, the nonparametric model is applied, after parametric identification, which may result in a suboptimal model. In addition, as far as it can be known, there is no semiparametric method based on deep learning methods.

Deep learning is a new approach in machine learning, which has been widely applied in smart cities and factories [35]. Deep learning has turned out to be very good at discovering intricate structures in high-dimensional data and is therefore applicable to many domains of science, business, and government. Since it requires very little engineering by hand, it can easily take advantage of increased amount of available computation resources and data [36]. In this work, a method that is based on deep learning and semiparametric approach is presented. The method is formalized in the framework of what is called Semiparametric Deep Learning (SDL), designed for optimal inference using combinations of parametric RBD and Nonparametric Deep Learning models. Key properties of this method are (1) appropriate deep learning frame for a semiparametric approach and (2) features that can be optimized simultaneously for parametric and nonparametric models. The proposed method is validated using experiments performed on a UR5 robot. The article is organized as follows. In Section 2, a complete description of the proposed Semiparametric Deep Learning framework is introduced. Section 3 presents the validation of the proposed method on the UR5 robotic platform. Finally, Section 4 summarizes the content of the presented work.

2. Methodology

The parametric modeling method and NDL, as the basis of the SDL method, have been elaborated in previous research publications [37, 38]. Therefore, this section only briefly reviews the above two methods, while it analyzes the SDL modeling method, proposed in this paper.

2.1. Parametric Robot Dynamics Model. It is well known that the robot dynamics can be modeled according to the following [39]:

$$\tau(q, \dot{q}, \ddot{q}) = M(q)\ddot{q} + C(q, \dot{q}) + G(q) + \varepsilon(q, \dot{q}, \ddot{q}), \qquad (1)$$

in which $q, \dot{q}, \ddot{q} \in \mathbb{R}^{m\times 1}$ are joint positions, velocities, and accelerations of the robot, respectively, $\tau \in \mathbb{R}^{m\times 1}$ denotes the joint torques, $M(q)$ is the generalized inertia matrix of the robot, $C(q, \dot{q})$ are the Coriolis and centripetal forces, and $G(q)$ is gravity. As shown in equation (1), the robot dynamics equation contains Rigid Body Dynamics (RBD) model:

$$\tau_{\mathrm{RBD}} = M(q)\ddot{q} + C(q, \dot{q}) + G(q). \tag{2}$$

The model errors $\varepsilon(q, \dot{q}, \ddot{q})$ are caused by unmodeled dynamics (e.g., hydraulic tubes, actuator dynamics, and flexibility and dynamics of the cable drives), ideal-joint assumptions (e.g., no friction and clearance), and inaccuracies in the RBD model parameters. The RBD model of a manipulator is well known to be linear regarding the parameters β [39], i.e.,

$$\tau_{\mathrm{RBD}} = \varphi(q, \dot{q}, \ddot{q})\beta, \tag{3}$$

in which φ is a matrix containing nonlinear functions of joint angles, velocities, and accelerations, often called basis functions. Modeling the robot dynamics, using the RBD model in equation (3), requires the identification of the dynamics parameters β. For the 6 Degree-of-Freedom (DoF) UR5 robot, for example, 60 dynamics parameters are to be identified (for each DoF, there are 10 parameters that could ideally be obtained directly from the CAD data).

2.2. Nonparametric Deep Learning Model

2.2.1. NDL Formulation. The inverse dynamic model, in model-based control, is described as the mapping from joint positions, velocities, and accelerations to torques, as shown in equation (1). The aim of nonparametric learning model is to predict the torque value of the i^{th} joint, $\tau_i \in \mathbb{R}$, as the response of the query point at $(q, \dot{q}, \ddot{q}) \in \mathbb{R}^{3m\times 1}$, by using the given n training data, $\{q[j], \dot{q}[j], \ddot{q}[j], \tau_i[j]\}_{j=1}^{n}$, in which $q[j], \dot{q}[j], \ddot{q}[j] \in \mathbb{R}^{m\times 1}$ and $\tau_i[j] \in \mathbb{R}$. Since the problem can be considered as a supervised learning problem, any supervised learning technique can be employed for the learning process, as shown in

$$\tau(q, \dot{q}, \ddot{q}) \sim \mathcal{DL}(q, \dot{q}, \ddot{q}). \tag{4}$$

Deep learning methods are representation-learning methods with multiple levels of representation, obtained by composing simple but nonlinear modules, each transforming the representation at one level (starting with the raw input) into a representation at a higher, slightly more abstract level. With the composition of enough such transformations, very complex robotic dynamics functions can be learned [36, 40]. Therefore, the application cases of deep learning are too numerous, such as [41–45].

The sequential nature of manipulator inverse dynamics suggests that, to predict the joint torque, it is important to model the relationship among sequential data points [39]. RNNs, a type of deep learning network, can be seen as very deep feedforward network, where all the layers share the same weights. Although their main purpose is to determine long-term dependencies, theoretical and empirical evidence shows that it is difficult to learn to store information for very long time [46]. LSTM networks have subsequently proved to be more effective than conventional RNNs, especially when there are several layers at each time step [48], enabling an entire speech recognition system that goes all the way from acoustics to the sequence of characters in the transcription. LSTM networks or related forms of gated units are also currently used for the encoder and decoder networks, performing very well in machine translation [47–49]. Moreover, studies presented in [38] and [50] confirmed the validity of applying LSTM to the prediction of manipulator inverse dynamics. Therefore, in this section, the LSTM network is proposed as the nonparametric learning technique for modeling the inverse dynamics of manipulator.

2.2.2. NDL Model Architecture. In this paper, the proposed architecture of the Nonparametric Deep Learning network has one input layer, one LSTM layer, one full-connected layer, one dropout layer, and one output layer, as shown in Figure 1. Only 1 LSTM layer is used here, because it has been verified in our previous studies that, with the same number of neurons, the fewer the layers, the better the prediction performance [38].

The input layer has 18 neurons (manipulator's 6 joint positions, 6 velocities, and 6 accelerations, as shown in Figure 2).

The state activation functions of LSTM cells are set to "tanh," while the gate activation functions are set to "sigmoid." The input weights are initialized according to the Glorot initializer. The forget gate bias is initially set to 1 and the remaining biases are set to 0. The training algorithm adopts back-propagation through time.

2.3. Proposed Semiparametric Deep Learning (SDL) Model.

In this section, the proposed semiparametric model, based on deep learning and RBD, is described in detail. The method is formalized in the framework of the so-called Semiparametric Deep Learning (SDL), which can be used to predict the joint torque of a robotic arm more accurately. First, formulation of SDL is introduced in Section 2.3.1, while the specific model architecture of SDL is described in Section 2.3.2.

2.3.1. SDL Formulation. Using the Nonparametric Deep Learning (DL) framework, the robot dynamics can be modeled by $\tau \sim \mathcal{DL}(q, \dot{q}, \ddot{q})$, in which $q, \dot{q}, \ddot{q}$ is the input and τ is the output of the Deep Learning Model. Consequently, the DL model does not make use of any prior knowledge, which allows reproducing arbitrary functions. One way to include the RBD model, as shown in equation (5), is to set the τ_{RBD} as input to the DL model. This approach is equivalent to a semiparametric model:

$$\begin{aligned}\tau(q, \dot{q}, \ddot{q}) &\sim \mathcal{SDL}(q, \dot{q}, \ddot{q}) \sim \mathcal{DL}(\tau_{RBD}, q, \dot{q}, \ddot{q}) \\ &\sim \mathcal{DL}(\varphi(q, \dot{q}, \ddot{q})\beta, q, \dot{q}, \ddot{q}).\end{aligned} \tag{5}$$

The resulting dynamics model, as described in equation (5), is a semiparametric frame which consists of a parametric

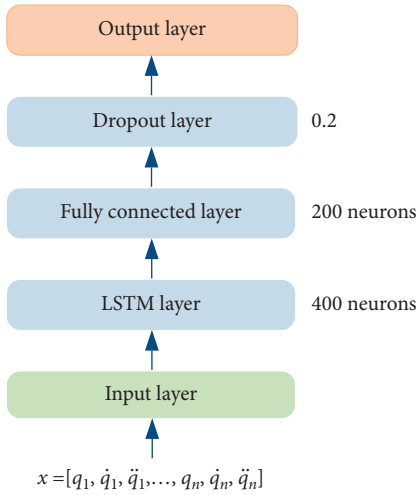

Figure 1: Proposed deep learning architecture.

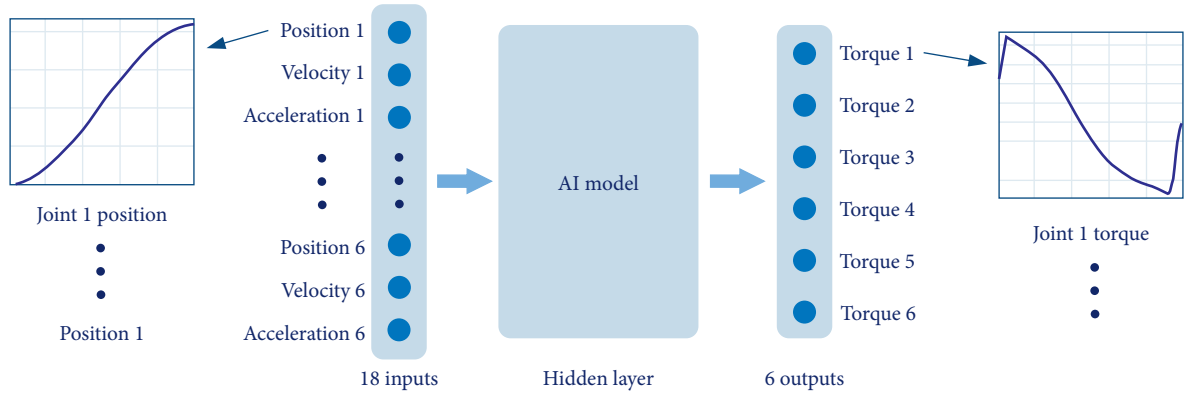

Figure 2: Inverse dynamics learning architecture of manipulator with 21 input data sets.

part, i.e., the RBD model. When comparing equation (5) to the robot dynamics in equation (1), it becomes evident that the main purpose of the nonparametric term is absorbing the unmodeled dynamics $\varepsilon(q, \dot{q}, \ddot{q})$. In order to approximate the unmodeled dynamics with an appropriate DL, a model can be used, such as DNN or LSTM. Key properties of this semiparametric learning method are (1) employment of appropriate deep learning frame as nonparametric part and (2) features that can be optimized simultaneously for parametric and nonparametric model; i.e., the value of τ_{RBD} will also be updated by weight changes as the learning process evolves.

If the RBD model perfectly describes the robot dynamics, the error $\varepsilon(q, \dot{q}, \ddot{q})$ in equation (1) will disappear and the prediction will depend only on the RBD part; i.e., it is very easy to train deep learning networks. Equation (5) also shows that if the query point is far away from the training data, the resulting torque prediction will mainly depend on the RBD part. This property is important, as the complete state space can never be completely included using finite (and possibly small) training data sets. If the robot moves to the regions of the state space, not considered by the sampled data (i.e., the learned nonparametric models may not generalize well in these state space regions), the torque prediction will rely on the parametric RBD part.

2.3.2. SDL Model Architecture. The effectiveness of the NDL model has been verified in previous work presented in [40]. The SDL model proposed in this article adds an RBD term to the NDL architecture. Its specific model architecture is shown in Figure 3. The input of the SDL architecture is still $[q, \dot{q}, \ddot{q}]$. However, unlike NDL, the input passes through the RBD term, forming the new input vector $[\tau_{\mathrm{RBD}}, q, \dot{q}, \ddot{q}]$. Next, the new input vector enters the LSTM hidden layer. The advantage of this architecture is that the parametric and nonparametric parts can be optimized simultaneously; i.e., the weight of τ_{RBD} in the feature vector will also be updated during network training. The grid search method is used to optimize the hyperparameters of the SDL model.

3. Evaluation

The proposed SDL method will be verified on a collaborative robot UR5, while the torque prediction results will be compared to the ones provided by NDL methods. The prediction performance for training and for generating

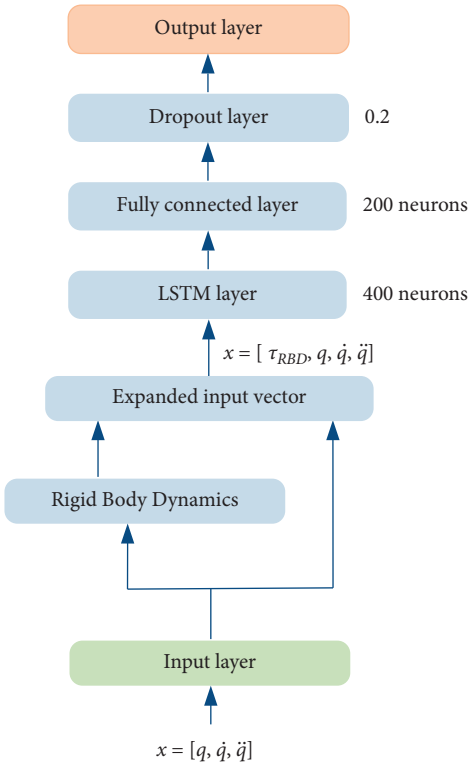

Figure 3: Proposed deep learning architecture.

predictions in rhythmic motor tasks is evaluated. The joint angles, the joint velocities, the joint accelerations, and the joint torques are recorded using a GUI (Graphical User Interface), i.e., PolyScope, making it easy to program the robot to move the tool along a desired trajectory path.

3.1. Experimental Setup. UR5 is a 6-DoF collaboration robot with extruded aluminum tubes and joints. It has six rotary joints, and its structure is shown in Figure 4. The UR5 robot has a joint rotation range of $[-2\pi, 2\pi]$ (rads) and a joint acceleration range of $[0, \pi]$ (rads/s^2). The UR5 robot is very popular in the robot research field.

According to the robot rotation angle and installation restrictions, the robot workspace is selected to be a hemisphere with an approximate radius of 850 mm above the installation plane. The range of joints motion is shown in Table 1. In order to best approximate the actual working situation of the robot, within the selected robot workspace, 1000 points are randomly selected. According to the actual use requirements of the robot, the joint running speed range is [0.8–2] rads/s, while the acceleration range is [1–1.8] rads/s.

The robot is ordered to run in a rhythmic way, according to the set trajectory. The joint position, speed, and servo motor current data, along the robot trajectory, is delivered from the robot controller at a frequency of 100 Hz. Since the UR5 robot is not equipped with a torque sensor, the measured torque is obtained indirectly through the motor current, at each joint. The relationship between torque and current is as follows [51]:

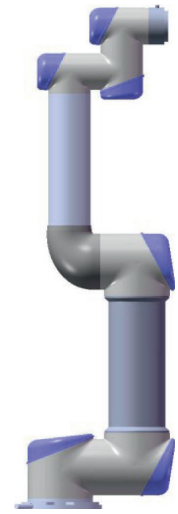

Figure 4: UR5 robot structure.

$$\tau_i = N_i k_i I_i, \tag{6}$$

in which N_i is the gear ratio, $N_i = 101, i = 1, \cdots, 6$; k_i motor constant, $k_i = 0.125$ Nm/A, $i = 1, \cdots, 3$, $k_i = 0.0922$ Nm/A, $i = 4, \cdots, 6$; and I_i is the motor current (A).

During the actual operation, the robot is affected by noise, making the sampling data fluctuate. In that case, there will be large fluctuations and ripples in the actual measured current, which seriously affects the accuracy of the torque prediction. The average data method can increase the signal-to-noise ratio of the data [52], reduce the influence of noise, and improve the prediction accuracy. The average joint position $\dot{q}$ can be expressed as follows:

$$\overline{q}(k) = \frac{1}{M} \sum_{m=1}^{M} q_m(k), \tag{7}$$

in which M is the number of running trajectories, $q_m(k)$ is the k-th sampling point of one running trajectory, and $\overline{q}(k)$ is the position after M times of averaging. The same method is used to deal with speed and current. The zero-phase low-pass Butterworth filter (forward and reverse IIR Butterworth filters) with a cutoff low-pass frequency of 1 Hz is used to process the averaged position and velocity $(\overline{q}, \overline{\dot{q}})$. Acceleration is obtained by the central difference method [53]. Data processing is important for the accuracy of parameter identification, making the above processing very appropriate, as it avoids large deviations in identification.

A total of 1,000 groups (a total of 108,008) of valid samples were obtained. According to the ratio of 80% to 20%, they were divided into training and testing sets, serving in K-fold cross-validation. In addition, the test data is ensured to be sufficiently different from the training data, highlighting the generalization ability of the learned models. The above sample set is used to train and test the NDL model and SDL model, respectively, while it also analyzes and compares the prediction performance.

Table 1: Rotation angle range of each joint in the selected robot workspace.

Joint #	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6
Range of motion (deg)	[−180, 180]	[−180, 0]	[−130, 130]	[−180, 0]	[−180, 180]	[−180, 180]

Table 2: Prediction results of RBD, NDL, and SDL models.

		Axis 1	Axis 2	Axis 3	Axis 4	Axis 5	Axis 6	All axes
Cross 1	RBD	2.2899	2.3417	2.3683	0.6523	0.5889	0.5290	1.7024
	NDL	2.6107	2.7715	1.8227	0.3418	0.4487	0.4577	1.7486
	SDL	1.8310	1.9460	1.3504	0.3450	0.4220	0.4524	1.2560
Cross 2	RBD	2.3316	2.3162	2.2893	0.6642	0.6498	0.5525	1.6937
	NDL	2.3342	2.5439	2.5428	0.3864	0.5162	0.4722	1.7807
	SDL	1.6721	1.8492	1.4528	0.3435	0.4633	0.4654	1.2162
Cross 3	RBD	2.4861	2.6723	2.2126	0.6417	0.5924	0.3582	1.7846
	NDL	2.2477	2.4820	1.1689	0.4081	0.4487	0.3977	1.4779
	SDL	1.4031	1.5533	0.9656	0.3294	0.4022	0.3428	0.9748
Cross 4	RBD	2.7325	2.7323	2.5119	0.6625	0.6386	0.3708	1.9246
	NDL	2.3239	2.1714	1.6374	0.3745	0.4245	0.4280	1.4889
	SDL	1.4009	1.6816	1.1613	0.3435	0.4399	0.3696	1.0478
Cross 5	RBD	2.8011	2.7946	2.2741	0.6326	0.5795	0.3620	1.9015
	NDL	1.6768	2.2989	1.6690	0.4346	0.3893	0.3393	1.3746
	SDL	1.5326	1.6847	1.0004	0.3385	0.3775	0.3058	1.0439
Cross average	RBD	2.5282	2.5714	2.3313	0.6506	0.6098	0.4345	1.8014
	NDL	2.2387	2.4535	1.7682	0.3891	0.4455	0.4190	1.5741
	SDL	1.5679	1.7430	1.1861	0.3400	0.4210	0.3872	1.1077

All input and output data are normalized to match the consistency of the learning model. After the prediction of inverse dynamics, the real value is restored. The normalized equation is as follows:

$$x_n = \frac{x_r - x_{\min}}{x_{\max} - x_{\min}}, \tag{8}$$

in which x_n represents the normalized value; x_r denotes the real value; and $x_{\min}$ and $x_{\max}$ are the minimum and maximum real values, respectively.

The performance of manipulator inverse dynamics predictions is evaluated using Root Mean Square Error (RMSE), which is defined as follows:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(p_i - y_i)^2}, \tag{9}$$

in which p_i and y_i represent the i-th predicted value and real value, respectively, and N is the total number of the data sets.

4. Results

The training and prediction in this paper were performed with MATLAB 2019a, using an ordinary personal computer. Computer hardware has a high influence on training time. In this work, the models are trained on a CPU with a clock speed of 2.7 GHz. The structure and hyperparameters of the NDL and SDL models were initially set according to previous work, while the final settings were determined based on the fivefold cross-validation method.

The prediction results of the RBD, NDL, and SDL models are listed in Table 2. The data in the table shows the RMSE of the RBD, NDL, and SDL models, on different robot axes and under different cross situations. Based on Table 2, the following conclusions apply:

(1) The results of "cross average" (Figure 5) indicate that the prediction accuracy of SDL is generally more accurate than the one of NDL and NDL is generally more accurate than RBD. Also, the prediction accuracy of the SDL model for the first 3 axes of the robot is significantly improved.

(2) The results of "all axes" (Figure 6) indicate that the prediction accuracy of the NDL model is sometimes better than the one of the RBD model (cross 3, 4, 5), while sometimes it is worse (cross 1, 2). However, the NDL model is better than the RBD model (mean) in general, while the prediction accuracy of the SDL model is always better than in the case of the other two models.

(3) All the data in Table 2 shows that the semiparametric models are able to combine the strengths of both models, i.e., the parametric RBD model and the Nonparametric Deep Learning model. The prediction accuracy of the SDL model is always better than that of the RBD and NDL models.

RMSE is not sufficient to fully represent the performance of torque prediction, because the range of torque variation for each joint of the robot varies greatly. Therefore, the ratio of cross average RMSE to the range of measured torque values was used to further analyze the predictive

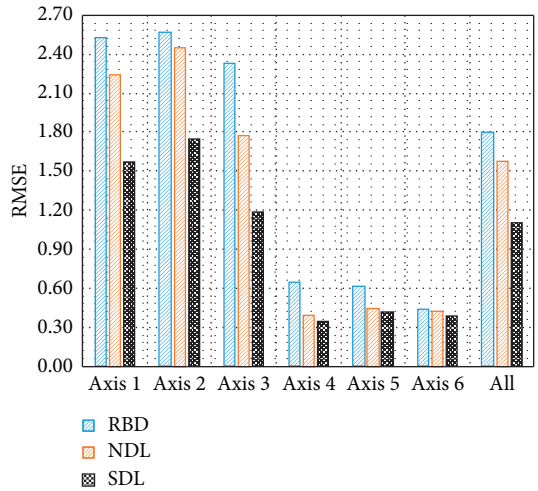

Figure 5: Cross average prediction results.

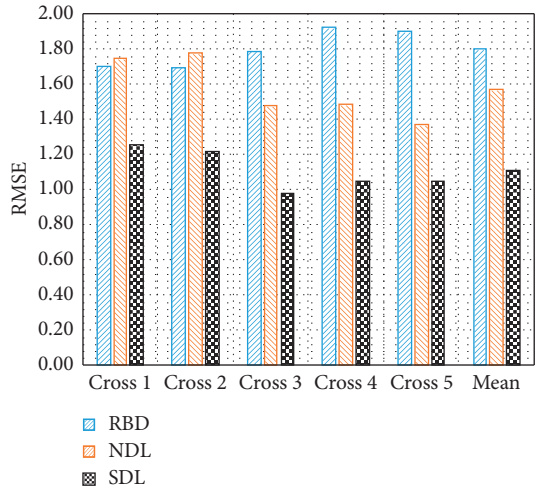

Figure 6: All axes prediction results.

Table 3: Comparative analysis of prediction performance of different joint forces.

		Axis 1	Axis 2	Axis 3	Axis 4	Axis 5	Axis 6
Cross average RMSE	RBD	2.5282	2.5714	2.3313	0.6506	0.6098	0.4345
	NDL	2.2387	2.4535	1.7682	0.3891	0.4455	0.4190
	SDL	1.5679	1.7430	1.1861	0.3400	0.4210	0.3872
Range of measured torque		42.1816	98.1207	49.3103	5.7766	4.4944	6.6976
Ratio	RBD	0.0599	0.0262	0.0473	0.1126	0.1357	0.0649
	NDL	0.0531	0.0250	0.0359	0.0674	0.0991	0.0626
	SDL	0.0372	0.0178	0.0241	0.0589	0.0937	0.0578

performance of different joints. As shown in Table 3, the torque prediction performance of the first three (elbow) joints is better than that of the last three (wrist) joints.

Part of the measured values of each joint torque, RBD predicted value, NDL predicted value, and SDL predicted value are all plotted in Figure 7. Figures 7(a)–7(f) are the predicted moments of axis 1 to axis 6, respectively; Figures 7(g)–7(l) are the predicted errors of moments of axis 1 to axis 6, respectively; the purple curve represents the measured torque value, the blue represents the calculated torque value using RBD method, the red represents the DL predicted torque value, and the yellow represents the SDL predicted torque value. Figure 7 shows that the SDL model combines the advantages of the RBD model and the NDL model. For example, as shown by the red dashed box in Figure 7, when the torque prediction error of the RBD is

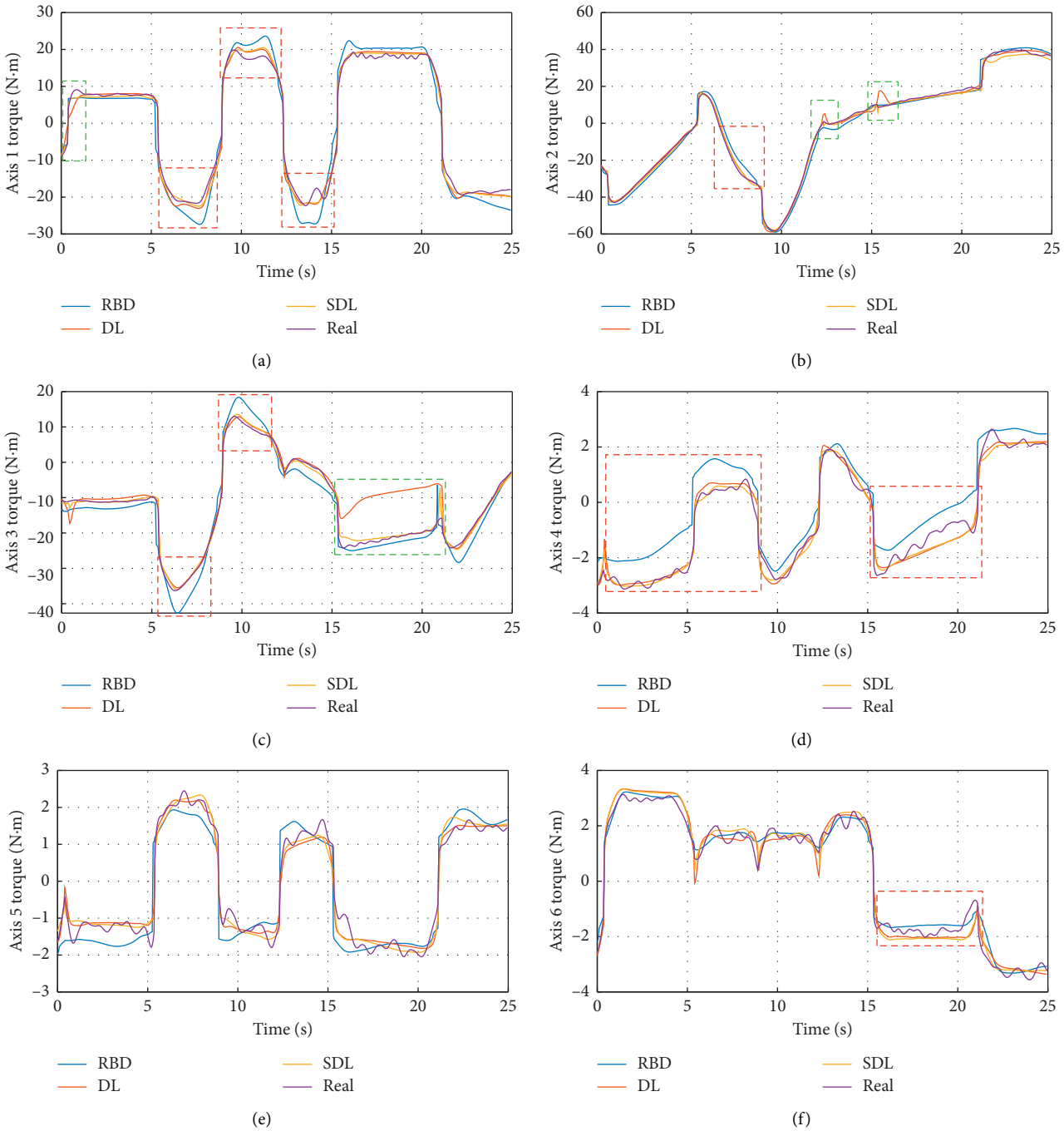

Figure 7: Continued.

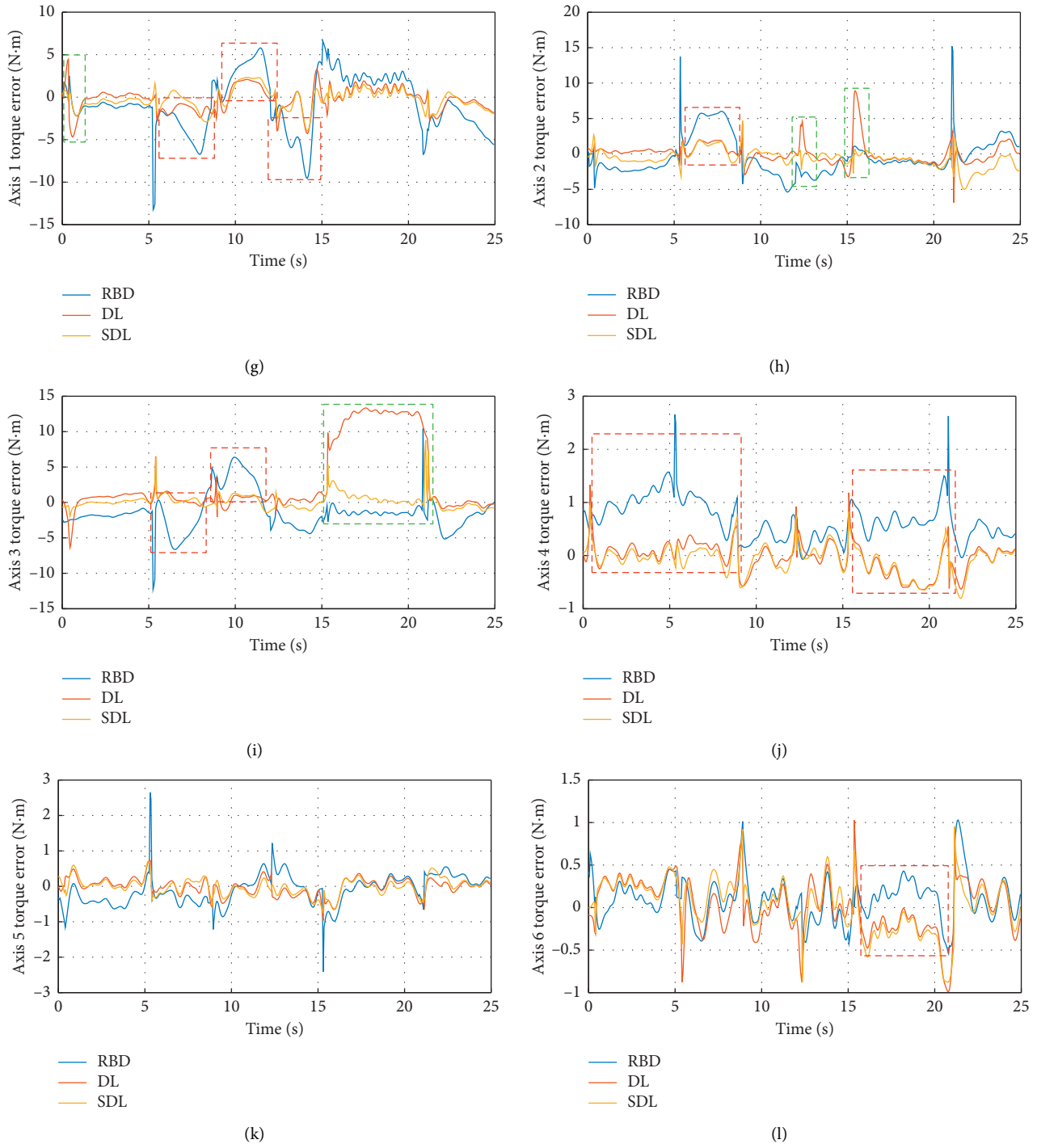

Figure 7: Joint torque prediction comparison curves. (a) Axis 1 torque. (g) Axis 1 torque error. (b) Axis 2 torque. (h) Axis 2 torque error. (c) Axis 3 torque. (i) Axis 3 torque error. (d) Axis 4 torque. (j) Axis 4 torque error. (e) Axis 5 torque. (k) Axis 5 torque error. (f) Axis 6 torque. (l) Axis 6 torque error.

large, the nonparametric part works, which greatly improves the prediction accuracy of the SDL model. As another example, shown by the green dotted box in Figure 7, when the learned nonparametric models do not generalize well to the state space regions, the torque prediction will rely on the parametric RBD part.

5. Conclusion

In this work, Semiparametric Deep Learning (SDL) method is proposed to model robot inverse dynamics, for smart city and industrial applications. The SDL model takes advantage of the global characteristics of classic RBD and the powerful

fitting capabilities of deep learning methods. Moreover, SDL model can be optimized simultaneously for the parametric and nonparametric model, instead of separate optimizations. The results on the UR5 robot show that the SDL models provide higher accuracy and better generalization, compared to RBD and NDL. The essence of the SDL model is to fully utilize the a priori information encoded in the parameterized model, overcome the limitations of the NDL method, and always show good learning performance. As far as future work is concerned, the flexible factors in dynamics will be considered as addition to the SDL model, in order to improve the accuracy prediction performance.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Key Research and Development Program of China (no. 2017YFB1302100).

References

[1] J. Jiang, F. Lin, J. Fan et al., "A destination prediction network based on spatiotemporal data for bike-sharing," *Complexity*, vol. 2019, Article ID 7643905, 14 pages, 2019.

[2] G. Li, L. Zhang, Y. Sun, and J. Kong, "Towards the sEMG hand: internet of things sensors and haptic feedback application," *Multimedia Tools and Applications*, vol. 78, no. 21, pp. 29765–29782, 2019.

[3] Y. He, G. Li, Y. Liao et al., "Gesture recognition based on an improved local sparse representation classification algorithm," *Cluster Computing*, vol. 22, no. S5, pp. 10935–10946, 2019.

[4] J. Qi, G. Jiang, G. Li, Y. Sun, and B. Tao, "Intelligent human-computer interaction based on surface EMG gesture recognition," *IEEE Access*, vol. 7, pp. 61378–61387, 2019.

[5] B. Siciliano and O. Khatib, *Dynamics, Springer Handbook of Robotics*, Springer, Berlin, Germany, 2008.

[6] K. Yamane, "Practical Kinematic and Dynamic Calibration Methods For Force-Controlled Humanoid Robots," in *Proceedings of 11th IEEE-RAS International Conference on Humanoid Robots*, ICMLBled, Slovenia, pp. 269–275, 2011.

[7] S. Traversaro, A. D. Prete, R. Muradore, L. Natale, and F. Nori, "Inertial Parameter Identification Including Friction and Motor Dynamics," in *Proceedings of 13th IEEE-RAS International Conference on Humanoid Robots*, pp. 68–73, Atlanta, GA, USA, 2013.

[8] Y. Ogawa, G. Venture, and C. Ott, "Dynamic Parameters Identification Of A Humanoid Robot Using Joint Torque Sensors and/or Contact Forces," in *Proceedings of 14th IEEE-RAS International Conference on Humanoid Robots*, pp. 457–462, Atlanta, GA, USA, 2014.

[9] J. Hollerbach, W. Khalil, and M. Gautier, *"Model Identification," Springer Handbook of Robotics*, Springer, Berlin, Germany, 2008.

[10] R. Camoriano, S. Traversaro, L. Rosasco et al., "Incremental semiparametric inverse dynamics learning," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, Paris, France, pp. 544–550, 2016.

[11] Y.-P. Zhao, B. Li, Y.-B. Li, and K.-K. Wang, "Householder transformation based sparse least squares support vector regression," *Neurocomputing*, vol. 161, pp. 243–253, 2015.

[12] D. Nguyen-Tuong, B. Scholkopf, and J. Peters, "Sparse online model learning for robot control with support vector regression," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, Madrid, Spain, pp. 3121–3126, 2009.

[13] Y. Choi, S.-Y. Cheong, and N. Schweighofer, "Local online support vector regression for learning control," in *Proceedings of the 2007 IEEE International Symposium on Computational Intelligence in Robotics and Automation*, IEEE, Jacksonville, FL, USA, pp. 13–18, June 2007.

[14] H. Mori, Y. Ohama, N. Fukumura, and Y. Uno, "Learning of real robot's inverse dynamics by a forward-propagation learning rule," *Electrical Engineering in Japan*, vol. 161, no. 4, pp. 38–48, 2007.

[15] N. Ishibashi and Y. Maeda, "Learning of inverse-dynamics for SCARA robot," in *Proceedings of the SICE Annual Conference*, pp. 1300–1303, Takamatsu, Japan, September 2011.

[16] O. Ken and M. Yutaka, "Learning of inverse-dynamics and inverse-kinematics for two-link SCARA robot using neural networks," in *Proceedings of the SICE Annual Conference*, pp. 1031–1034, Takamatsu, Japan, September 2011.

[17] S. Vijayakumar, A. D'Souza, and S. Schaal, "Incremental online learning in high dimensions," *Neural Computation*, vol. 17, no. 12, pp. 2602–2634, 2005.

[18] S. Vijayakumar and S. Schaal, "Fast and efficient incremental learning for high-dimensional movement systems," in *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 2, IEEE, Montreal, Canada, pp. 1894–1899, 2000.

[19] J. S. d. l. Cruz, *Learning Inverse Dynamics for Robot Manipulator Control*, University of Waterloo, Waterloo, Canada, 2011.

[20] T. T. Um, M. S. Park, and J. Park, ""Independent Joint Learning: a novel task-to-task transfer learning scheme for robot models," 2014," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, Paris, France, pp. 5679–5684, 2014.

[21] K. Caluwaerts and J. J. Steil, "Independent joint learning in practice: local error estimates to improve inverse dynamics control," in *Proceedings of the 2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, IEEE, Seoul, South Korea, pp. 643–650, November 2015.

[22] Z. Shareef, P. Mohammadi, and J. Steil, "Improving the inverse dynamics model of the KUKA LWR IV+ using independent joint learning∗∗Z. Shareef received funding from the German federal ministry of education and research (BMBF) within the leading-edge cluster competition. P. Mohammadi received funding from the European community's horizon 2020 robotics program ICT-23-2014 under grant agreement 644727—CogIMon," *IFAC-PapersOnLine*, vol. 49, no. 21, pp. 507–512, 2016.

[23] K. Chai, *Multi-task Learning with Gaussian Processes*, Institute for Adaptive and Neural Computation, School of Informatics, University of Edinburgh, Edinburgh, Scotland, 2010.

[24] M. Deisenroth, J. Peters, and C. Rasmussen, "Approximate dynamic programming with Gaussian processes," in

Proceedings of the American Control Conference, pp. 4480–4485, Denver, Colorado, 2008.

[25] M. Deisenroth and C. R. Pilco, "A model-based and data-efficient approach to policy search," in *Proceedings of the Twenty Eighth International Conference on Machine Learning*, ICML, Bellevue, WA, USA, 2011.

[26] D. Nguyen-Tuong and J. Peters, "Local Gaussian process regression for real-time model-based robot control," in *Proceedings of the International Conference on Intelligent Robots and Systems*, pp. 380–385, Macau, China, 2008.

[27] A. Rottmann and W. Burgard, "Learning non-stationary system dynamics online using Gaussian processes," *Pattern Recognition of Lecture Notes in Computer Science*, vol. 6376, pp. 192–201, Springer, Berlin, Germany, 2010.

[28] D. Nguyen-Tuong, J. Peters, and M. Seeger, "Computed torque control with nonparametric regression models," in *Proceedings of the 2008 American Control Conference*, ACC, Seattle, DC, USA, 2008.

[29] D. Nguyen-Tuong and J. Peters, "Using model knowledge for learning inverse dynamics," in *Proceedings of the 2010 IEEE International Conference on Robotics and Automation*, pp. 2677–2682, Anchorage, AK, USA, May 2010.

[30] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*, MIT Press, Cambridge, MS, USA, 2006.

[31] T. Wu and J. Movellan, *Semi-Parametric Gaussian Process for Robot System Identification*, IROS, Bengaluru, India, 2012.

[32] J. Sun de la Cruz, D. Kulic, W. Owen, E. Calisgan, and E. Croft, "On-line dynamic model learning for manipulator control," *IFAC Robot Control*, vol. 10, no. 1, pp. 869–874, 2012.

[33] S. Vijayakumar and S. Schaal, "Locally weighted projection regression," in *Incremental Real Time Learning in High Dimensional Space." ICML*, P. Langley, Ed., pp. 1079–1086, Morgan Kaufmann, Burlington, MS, USA, 2000.

[34] A. Gijsberts and G. Metta, "Incremental Learning Of Robot Dynamics Using Random Features," in *Proceedings of the 2011 IEEE International Conference on Robotics and Automation*, pp. 951–956, Shanghai, China, May 2011.

[35] L. Zhang, C. P. Lim, and J. Han, "Complex deep learning and evolutionary computing models in computer vision," *Complexity*, vol. 2019, Article ID 1671340, 2 pages, 2019.

[36] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[37] N. Liu, L. Li, B. Hao et al., "Inverse dynamic modeling and simulation of multiple-degree-of-freedom heavy-duty hydraulic manipulator," *International Journal of Mechatronics and Applied Mechanics*, vol. 2, no. 6, pp. 87–96, 2019.

[38] N. Liu, L. Li, B. Hao et al., "Modeling and simulation of robot inverse dynamics using LSTM-based deep learning algorithm for smart cities and factories," *IEEE Access*, vol. 7, pp. 173989–173998, 2019.

[39] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Dynamics and Control*, John Wiley & Sons, New York. NJ, USA, 2006.

[40] A. Hernandez-Blanco, B. Herrera-Flores, D. Tomas et al., "A systematic review of deep learning approaches to educational data mining," *Complexity*, vol. 2019, Article ID 1306039, 22 pages, 2019.

[41] G. Li, J. Li, Z. Ju, Y. Sun, and J. Kong, "A novel feature extraction method for machine learning based on surface electromyography from healthy brain," *Neural Computing and Applications*, vol. 31, no. 12, pp. 9013–9022, 2019.

[42] W. Cheng, Y. Sun, G. Li, G. Jiang, and H. Liu, "Jointly network: a network based on CNN and RBM for gesture recognition," *Neural Computing and Applications*, vol. 31, no. S1, pp. 309–323, 2019.

[43] G. Li, D. Jiang, Y. Zhou, G. Jiang, J. Kong, and G. Manogaran, "Human lesion detection method based on image information and brain signal," *IEEE Access*, vol. 7, pp. 11533–11542, 2019.

[44] D. Jiang, G. Li, Y. Sun, J. Kong, and B. Tao, "Gesture recognition based on skeletonization algorithm and CNN with ASL database," *Multimedia Tools and Applications*, vol. 78, no. 21, pp. 29953–29970, 2019.

[45] F. P. An, "Pedestrian re-recognition algorithm based on optimization deep learning-sequence memory model," *Complexity*, 2019.

[46] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994.

[47] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *Advances in Neural Information Processing Systems*, pp. 3104–3112, 2014.

[48] K. Cho, B. Van Merrienboer, C. Gulcehre et al., "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Proccedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 1724–1734, Doha, Qatar, October 2014.

[49] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proceedings of the 3rd International Conference on Learning Representations*, ICLR, San Diego, CA, USA, 2015.

[50] E. Rueckert, M. Nakatenus, S. Tosatto et al., "Learning Inverse Dynamics Models in on Time with LSTM Networks," 2017," in *Proceedings of the IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pp. 811–816, Santa Monica, CA, USA, November 2017.

[51] N. Kovincic, A. Müller, H. Gattringer et al., "Dynamic parameter identification of the universal robots UR5," in *Proceedings of the ARW & OAGM Workshop*, pp. 44–53, Steyr, Austrian, May 2019.

[52] S. Jiang, M. Jiang, Y. Cao et al., "A typical dynamic parameter identification method of 6-degree-of-freedom industrial robot," *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 231, no. 9, pp. 740–752, 2017.

[53] J. Jin and N. Gans, "Parameter identification for industrial robots with a fast and robust trajectory design approach," *Robotics and Computer-Integrated Manufacturing*, vol. 31, pp. 21–29, 2015.