Robotica (2024), 1–18
doi:10.1017/S0263574724001905

CAMBRIDGE UNIVERSITY PRESS

RESEARCH ARTICLE

Robot hybrid inverse dynamics model compensation method based on the BLL residual prediction algorithm

Yong Tao[1], Shuo Chen[1,2], Haitao Liu[1], Jiahao Wan[1], Hongxing Wei[1] and Tianmiao Wang[1]

[1]School of Mechanical Engineering and Automation, Beihang University, Beijing 100191, China
[2]School of Large Aircraft Engineering, Beihang University, Beijing 100191, China
Corresponding author: Yong Tao; Email: taoy@buaa.edu.cn

Received: 20 June 2024; **Revised:** 23 September 2024; **Accepted:** 9 October 2024

Keywords: residual compensation; inverse dynamics model; model compensation method; industrial robot; LSTM; ensemble learning

Abstract

The inverse dynamics model of an industrial robot can predict and control the robot's motion and torque output, improving its motion accuracy, efficiency, and adaptability. However, the existing inverse rigid body dynamics models still have some unmodelled residuals, and their calculation results differ significantly from the actual industrial robot conditions. The bootstrap aggregating (bagging) algorithm is combined with a long short-term memory network, the linear layer is introduced as the network optimization layer, and a compensation method of hybrid inverse dynamics model for robots based on the BLL residual prediction algorithm is proposed to meet the above needs. The BLL residual prediction algorithm framework is presented. Based on the rigid body inverse dynamics of the Newton–Euler method, the BLL residual prediction network is used to perform error compensation on the inverse dynamics model of the Franka robot. The experimental results show that the hybrid inverse dynamics model based on the BLL residual prediction algorithm can reduce the average residuals of the robot joint torque from 0.5651 N·m to 0.1096 N·m, which improves the accuracy of the inverse dynamics model compared with those of the rigid body inverse dynamics model. This study lays the foundation for performing more accurate operation tasks using industrial robots.

1. Introduction

The inverse dynamics model of industrial robots plays a key role in improving robot control accuracy and operation efficiency. These inverse dynamics models allow robots to perform complex tasks more accurately and effectively by predicting and calculating the forces and torques required. In high-performance manufacturing, automated assembly, and fine operation, the application of inverse dynamics models is particularly important. However, these inverse dynamics models still face challenges when dealing with complex variables and uncertainties in the actual industrial environment. For example, environmental noise, joint friction, dynamic load change, and uncertain factors such as flexibility and joint clearance of the robotic arm may lead to inaccurate inverse dynamics model predictions, affecting industrial robot performance. Considering these problems and urgent needs, academia and industry are actively researching improving the accuracy and computational efficiency of inverse dynamics models to better meet the needs of modern industry for high-precision and high-efficiency industrial robots.

The uncertainty residuals of the inverse dynamics model originated from influencing factors such as the greater influence of the motor by temperature, the flexible connection of the harmonic reducer, and the greater influence of temperature on nonlinear friction. Therefore, Gao et al. [1] proposed that an accurate and reasonable friction model is important for studying the inverse dynamics mode. Iskandar et al. [2] proposed a collaborative joint friction model of industrial robots that comprehensively considered the effects of speed, temperature, and load torque. Callar [3] proposed a physically excited friction

model with a parametric description of a nonlinear dependence of temperature and velocity as well as the dependence on external loads.

The friction model alone is insufficient; the model still has many gaps, which cause uncertain residuals. Fitting models such as neural networks can provide efficient data-driven optimization with a small deviation. Romeres et al. [4] developed a learning method for an inverse dynamics model of a long short-term memory (LSTM) network based on time complexity. Seeger et al. [5] discussed an online robot inverse dynamics modelling algorithm. Dalla et al. [6] used the Gaussian process (GP) framework to model the inverse dynamics model. Michael et al. [7] proposed a method for forward dynamics estimation from data-driven inverse dynamics learning, which derived the physical dynamics equations of rigid body dynamics (RBD) by learning the inverse dynamics model and estimating each dynamics component from it. However, these purely data-driven inverse kinetics models usually have low data efficiency, cannot guarantee the stability of the extrapolation method, and can achieve good performance only near the training data, which prevents them from being used in safety-critical applications in semistructured environments.

Hybrid models inspired by physics-inspired neural networks, such as deep Lagrangian networks (DeLaNs) [8] or Hamiltonian networks [9], ensure physically reasonable dynamic models and thus save energy. These physics-inspired neural networks incorporate the system dynamics structure into the neural network architecture, are limited to modelling conservative forces, and capture all nonconservative effects using an additional neural network. The residual hybrid model combines the rigid body model with an additional residual model to learn the residuals of the RBD model [10]. A GP or a neural network (NN) can be used for the residual model. Nguyen-Tuong and Eters [11] used the GP model to learn the residuals of the inverse dynamics model. Several recent works have combined standard multilayer perceptron NNs with rigid body models [12–15]. Similar methods have been applied to physics-inspired NNs to capture nonconservative forces. Our previous work successfully applied different recurrent NNs to the black-box inverse dynamics model [16]. The LSTM algorithm was applied to train the residual mixture model, and the results were better than those of GP.

Industrial robot is a complex nonlinear system with multiple inputs and multiple outputs, with time-varying, strongly coupled, nonlinear, and other complex dynamics, thus making the high-performance control of the robot very complex and difficult [17]. There are many control methods for robotic arms nowadays; for example, An et al. proposed a sparrow search algorithm [18] based on the Levy flight operator for self-correcting robot controller parameters to improve the accuracy of motion trajectory. Chen et al. proposed a target tracking control system based on a bionic closed-loop central pattern generator to achieve accurate target tracking [19]. Wu et al. formulated a systematic approach for the elastic dynamics analysis of tandem robot manipulators [20]. Chen et al. proposed a segmented dynamic modelling approach to construct propulsion and lift models for the beaver-like tail of a robot. Combining fluid dynamics and material mechanics, a theory of sectorial flexible dynamics was developed [21]. Model-based predictive control (MPC) algorithms [22] provide better results with less accurate models avoiding the need for dynamic modelling of disturbances. However, traditional Proportion Integration Differentiation (PID) control as well as MPC control can hardly meet the demand of high speed and high-precision control in modern industry, and feedforward control based on the inverse dynamics model is an effective way to improve the control performance of robots. However, this control method has the problem of model imprecision. The actual industrial robot system has parameter uncertainty, non-parametric uncertainty, external environment interference, etc. It is difficult to establish an accurate inverse dynamics model of the robot, and the inaccuracy of the model will lead to a decrease in the control performance. In this paper, under the assumption that the robot is an ideal rigid body and elastic deformation is not taken into account, we carry out a research to address the above problems and propose a hybrid inverse dynamics model compensation method for industrial robots oriented to the feedforward control method of robots. However, only the LSTM algorithm is still insufficient for the compensation accuracy of the inverse dynamics of industrial robots, and there is an urgent need to propose new algorithms to further improve the compensation accuracy. Therefore, this paper proposes a combination of bootstrap aggregating (bagging) algorithm and LSTM network and at the same time introduces the linear layer

https://doi.org/10.1017/S0263574724001905 Published online by Cambridge University Press

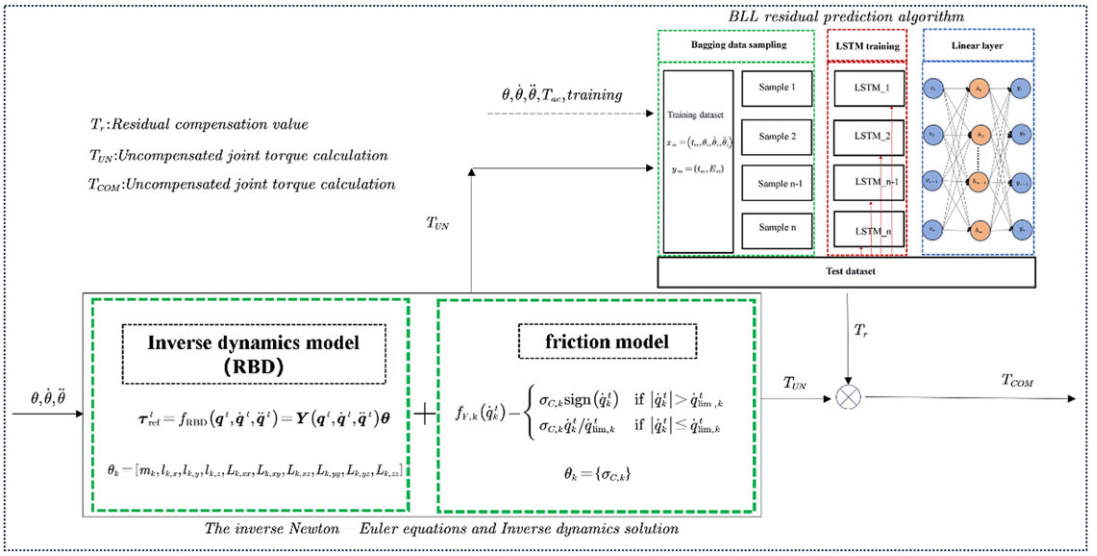

Figure 1. *Flow chart of the hybrid inverse dynamics residual compensation algorithm.*

as the network optimization layer. The BLL residual prediction algorithm (BLL algorithm for short) is constructed to improve the accuracy of the residual model of industrial robots.

First, the rigid body inverse dynamics model and the hybrid inverse dynamics model based on the compensation of the residual prediction algorithm are introduced. Then, the composition and principle of the BLL algorithm are introduced. Finally, experiments are conducted based on the public dataset of the Franka Emika Panda robot to validate the average residual value of each joint of the hybrid inverse dynamics model of the robot decreased from 0.5651 N · m to 0.1096 N · m, which improved the calculation accuracy of the robot inverse dynamics model. Compared with the LSTM and GP algorithms in this field, the proposed BLL algorithm improves the mean squared error (MSE), root mean square residual error (RMSE), mean absolute error (MAE), R2, and the reduction rate of the mean value of the residuals after compensation, validating the effectiveness of the proposed algorithm. Figure 1 shows the flow chart of the model compensation technique.

2. Robot inverse dynamics model

2.1. Body inverse dynamics model

The primary purpose of industrial robot dynamics is to achieve real-time control and an accurate dynamic model; accurate dynamic parameters are the key. Robot inverse dynamics modelling methods include Lagrangian methods, recursive Newton–Euler methods, symbolic computational methods, and methods based on a mixture of symbolic and numerical methods. Among these, the Newton–Euler [23] method has a relatively fast computational speed, especially when using a recursive form, which is well suited for robot control systems and is computationally efficient [24], so the Newton–Euler method is used here, with the following expression:

$$\tau = M(q)\ddot{q} + H(q,\dot{q}) + G(q) + \xi \tag{2.a}$$

q, $\dot{q}$, $\ddot{q}$ are the vectors of the joint position, velocity, and acceleration, respectively, and n represents robot degrees of freedom (DOF). τ is the joint torque vector, $M(q) \in R^{n\times n}$ is the inertia matrix, $H(q,\dot{q}) \in R^{n}$ is the vector of the centrifugal force and the Coriolis force, $G(q) \in R^{n}$ is the gravitational torque or force, and $\xi \in R^{n}$ is the offset torque, which represents the dynamic uncertainty influencing factor. The nonlinear friction factor is one of the factors affecting the high-precision movement of a robot [25]. The

https://doi.org/10.1017/S0263574724001905 Published online by Cambridge University Press

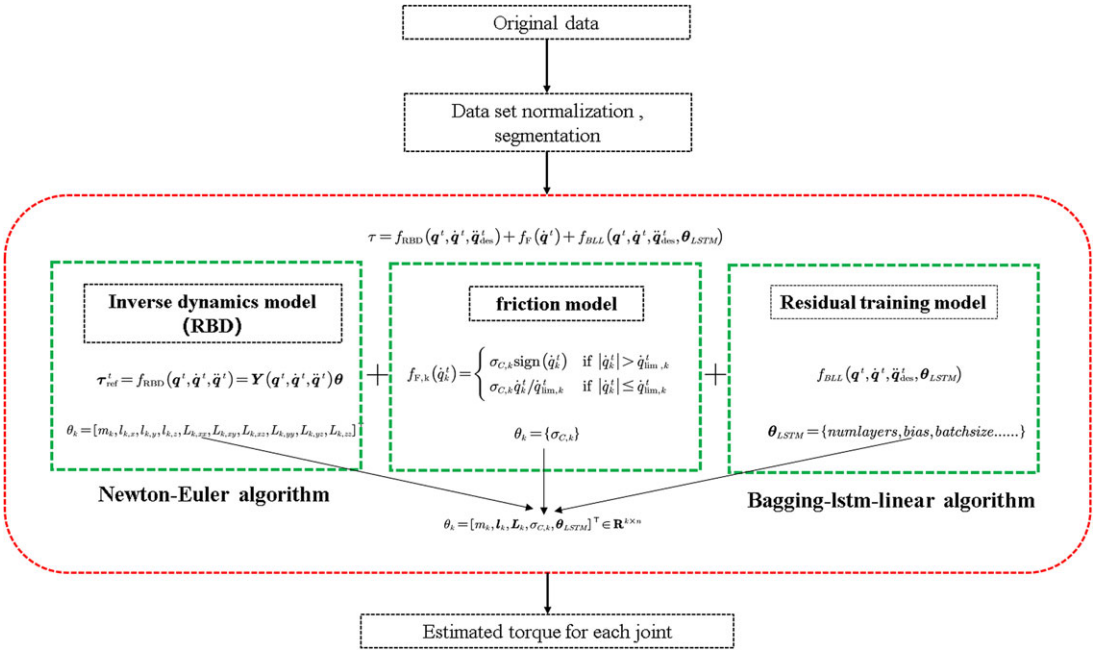

Figure 2. *Two-frame diagram of the hybrid inverse dynamics model based on the BLL algorithm compensation.*

dynamic identification model of industrial robots usually uses the viscosity and Coulomb friction model to represent the effect of friction on robot dynamics:

$$\xi = f_v\dot{q} + f_c sign(\dot{q}) + \epsilon(q, \dot{q}, \ddot{q}) \tag{2.b}$$

where f_v, f_c are the viscosity and the Coulomb friction coefficient, respectively, and $(\mathrm{q}, \dot{\mathrm{q}}, \ddot{\mathrm{q}}) \in \mathrm{R}^{\mathrm{n}}$ represents the factors not modelled in the physical kinetics model. Predicting the unmodelled part is the focus of this paper in Sections 3 and 4 and is described in detail.

2.2. Hybrid inverse dynamics model compensated based on the BLL residual prediction algorithm

Based on Formula (2.a), this paper proposes a hybrid architecture that couples the RBD model, friction model, and residual error prediction algorithm. The BLL algorithm models the residual errors of the RBD model and accounts for the uncaptured effects, such as the flexibility of the robot. The architecture proposed in this paper is composed of the following three parts:

$$\tau = f_{RBD}(q^t, \dot{q}^t, \ddot{q}^t_{des}) + f_F(\dot{q}^t) + f_{BLL}(q^t, \dot{q}^t, \ddot{q}^t_{des}, \theta_{BLL}) \tag{2.c}$$

f_{RBD} is a partial rigid body model using the motion equations in Eq. (2.a). $f_{\mathrm{F}}f_{BLL}$ is a residual prediction model that captures some observable effects, such as joint and link flexibility and more complex friction effects.

This paper proposes a hybrid inverse dynamics model framework, as shown in Fig. 2. First, the Newton–Euler algorithm is used for inverse dynamics joint torque calculations. Then, the values of the friction model are calculated. Finally, the calculated joint torques are used as the BLL network inputs to train the BLL network model. The network-predicted values are compared with those of the RBD model. The friction models are combined to obtain a hybrid inverse dynamics model. In this paper, the main objective is to predict the residual error.

https://doi.org/10.1017/S0263574724001905 Published online by Cambridge University Press

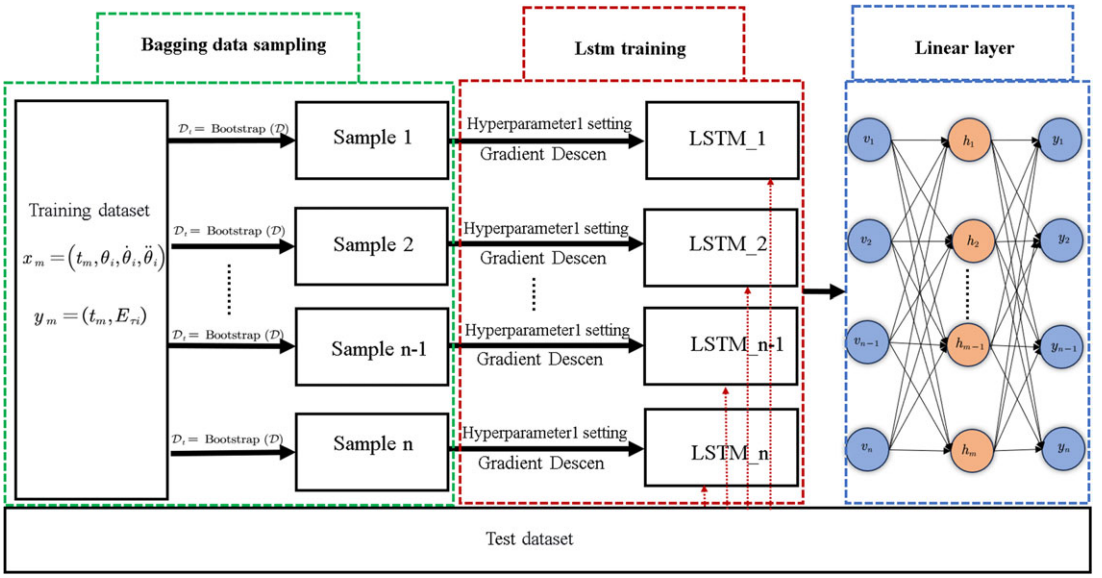

Figure 3. *Flow chart of the BLL algorithm.*

3. Principle of the BLL residual prediction algorithm

The BLL residual prediction algorithm proposed in this paper comprises three layers: a sample data sampling layer based on ensemble learning bagging, a network LSTM prediction layer, and a linear optimization fitting layer to predict the residual joint torque values (Fig. 3).

Bootstrap aggregating (bagging) was first proposed by Leo Breiman in 1996 [26]. The bagging algorithm can be combined with other classification and regression algorithms to improve the accuracy and stability and reduce the variance in the results simultaneously to avoid overfitting.

LSTM NNs are a special type of recurrent neural network (RNN) [27]. During the training of the original RNN, with prolonged training time and an increase in the number of network layers, the problem of gradient explosion or gradient disappearance is prone to occur, and the RNN cannot process longer sequence data and thus cannot obtain long-distance data. LSTM was proposed to solve this problem.

The linear layer, also called the fully connected layer, is a basic network layer structure in deep learning [28]. A linear layer usually contains an input matrix, a weight matrix, and a bias vector.

In this paper, the bagging algorithm is first used to sample n groups, and an independent LSTM network is used for training. At this time, the learning rate of the fully connected layer is set to 0, and no training is performed. The trained n groups of LSTM network models are validated using the same validation set, and n sets of validation results are obtained. These results still have a residual error ϵ from the actual value. To further optimize the residuals, the n-group LSTM validation set results are averaged and combined with the true values y_m of the original validation set to form a new training set, which is used to train the subsequent fully connected layer to optimize the residuals. The learning rate of the fully connected layer is changed to 0.01. Then, a unified whole is validated with the test dataset after the training is complete. The validation is performed using the same validation set to validate the n-group model and perform residual optimization.

Bagging generates different training subsets based on random sampling, such that some data in each subset are not sampled. In this way, more data changes can be introduced, which can increase the difference between models and improve the diversity of the ensemble model. In addition, as a deep learning network, LSTM has many parameters that require training and is prone to overfitting the training data. Through bagging, since each learner is trained on a different subset, overfitting can be reduced, and the generalization ability of the model can be improved. Finally, the linear layer is trained and optimized based on the LSTM prediction result to reduce the residual error further.

https://doi.org/10.1017/S0263574724001905 Published online by Cambridge University Press

Table I. *Model-related mathematical symbols.*

Mathematics symbol	Meaning
f_t	Forget gate
i_t	Input gate
$\tilde{C}_t$	update vector
C_t	Cell status
o_t	Output gate
h_t	Cell status output
x_t	Input value

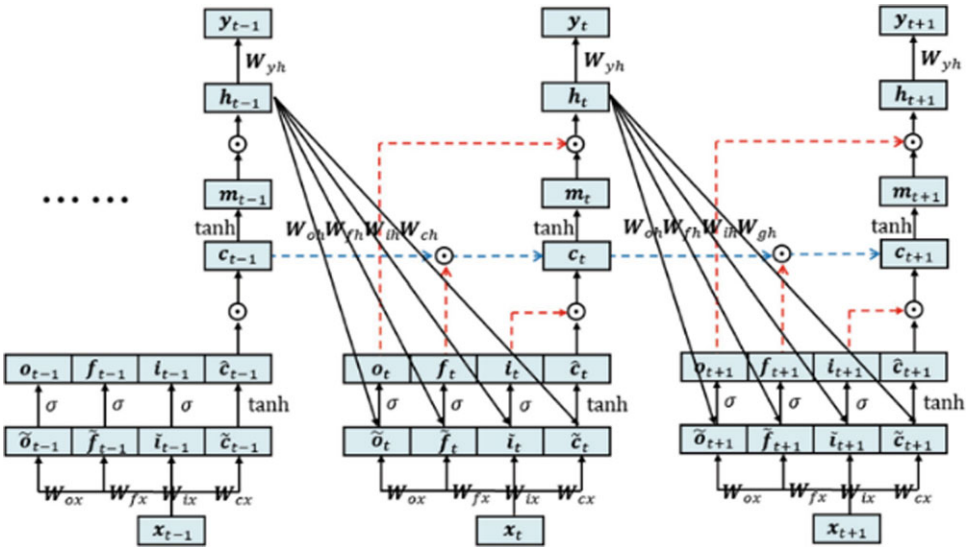

Figure 4. *Principle of LSTM [29].*

In this paper, three algorithms, namely, bagging, LSTM, and linear, are coupled, and the mathematical mapping formula corresponding to the BLL algorithm is created according to the coupling mechanism. The relevant mathematical symbols are shown in Table I.

Bagging-based random sampling:

$$\mathcal{D}_t = \text{Bootstrap}(\mathcal{D}) \tag{3.a}$$

Next, the LSTM algorithm makes independent predictions. Figure 4 shows the principle of LSTM.

The underlying logic of LSTM is to selectively retain and forget historical information [30] based on various gating weight matrices. The specific formulas are as follows:

Forget gate:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{3.b}$$

Input gate:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{3.c}$$

Update vector:

$$\tilde{C}_t = tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{3.d}$$

https://doi.org/10.1017/S0263574724001905 Published online by Cambridge University Press

Cell status update:

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{3.e}$$

Output gate:

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{3.f}$$

Cell state output:

$$h_t = o_t * tanh(C_t) \tag{3.g}$$

Output value:

$$y_t = h_t \cdot W_{yh} \tag{3.h}$$

After n groups of LSTM networks are independently trained, the average value of the n models is taken at each moment after validation is performed using test_data:

$$\overline{y_t} = \sum_{1}^{n} y_t / n \tag{3.i}$$

Assuming there are K time points in the validation set, the average vector predicted by LSTM is:

$$\overline{Y}_{k\times 1} = [\overline{y}_1, \overline{y}_2 \cdots \overline{y}_k]^T \tag{3.j}$$

The validation results of the test_data and actual result are used as the input of the linear layer to train a NN with a total of three layers, including input, output, and linear hidden layers. The linear weight matrix is:

$$W^1_{m\times k} = \begin{bmatrix} W_{11} & \cdots & W_{1k} \\ \vdots & \ddots & \vdots \\ W_{m1} & \cdots & W_{mk} \end{bmatrix} \tag{3.k}$$

The linear hidden layer is:

$$\mathrm{h}_0 = relu\left(W^1_{m\times k}\overline{Y}_{k\times 1} + \mathrm{b}_1\right) \tag{3.l}$$

The output result of the full connection is:

$$OUT = relu\left(W^2_{k\times m}\mathrm{h}_{\cdot 0} + \mathrm{b}_2\right) \tag{3.m}$$

4. Residual model validation

4.1. Dataset preprocessing

The dynamic parameters and public datasets of the Franka Emika Panda robot used in this paper are from the "Dynamic Identification of the Franka Emika Panda Robot With Retrieval of Feasible Parameters Using Penalty-Based Optimization" provided by Claudio Gaz et al. [31].

In this paper, the angular acceleration is calculated by derivation based on the time, angular displacement, and angular velocity of the seven joints in the dataset. The inverse Newton–Euler algorithm [32] was used to calculate the torque variation in each joint of the rigid body inverse dynamics model (2.a).

In this paper, the data collected at 12.9 S–13.2 S are selected as the results. The input layer of the inverse Newton–Euler algorithm has seven channels: the joint angle, joint velocity, and joint acceleration of the seven DOF of the Franka robot; the output is seven channels, which are the calculated torques of the seven joints of the Franka robot. The calculation results of the inverse Newton–Euler algorithm are compared with the actual acquisition results, as shown in Fig. 5.

The residual data of subtracting the calculated and actual torque are shown in Fig. 6.

The above figure shows a large gap exists between the calculated and actual results. The model training process and results are described in detail in Section 4.2.

https://doi.org/10.1017/S0263574724001905 Published online by Cambridge University Press

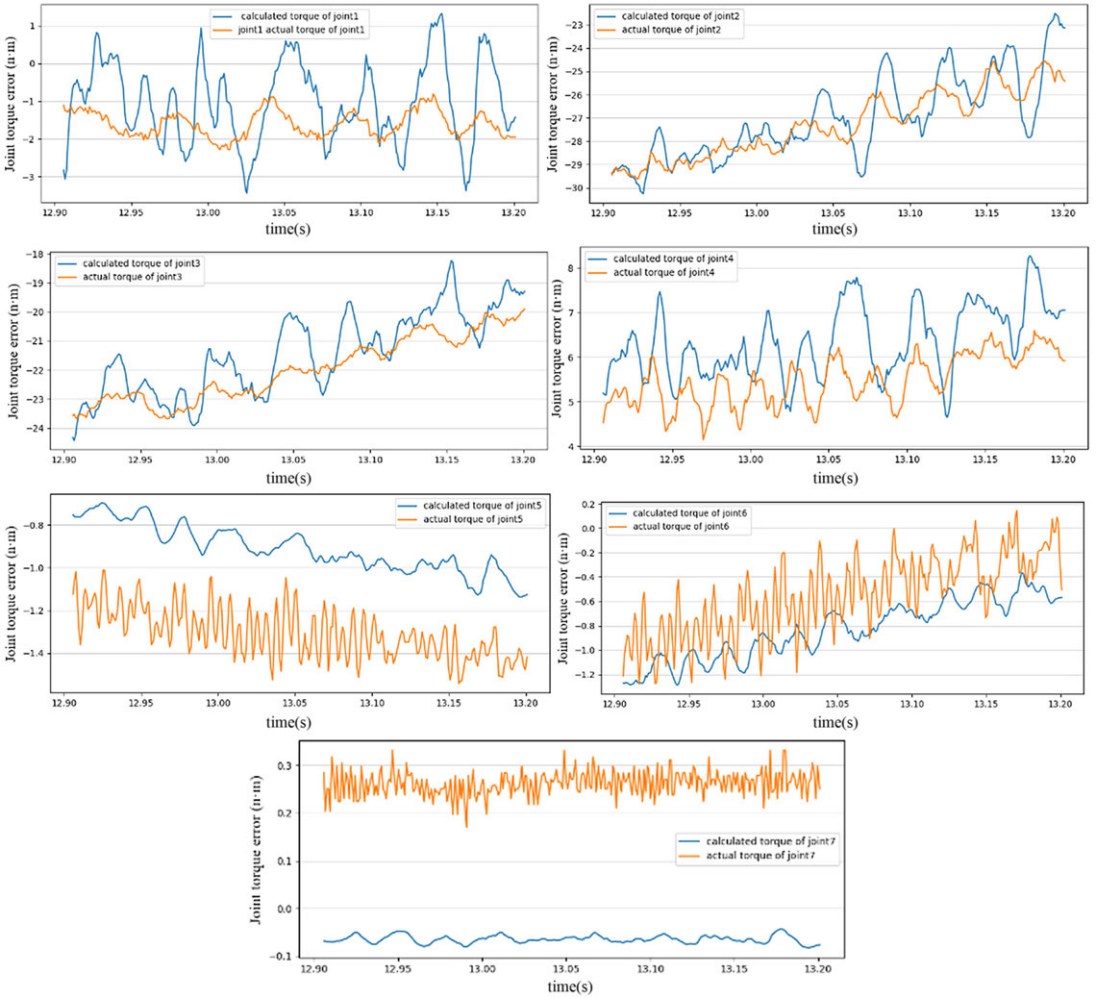

Figure 5. *Comparison of the calculated and actual torques for joints 1–7.*

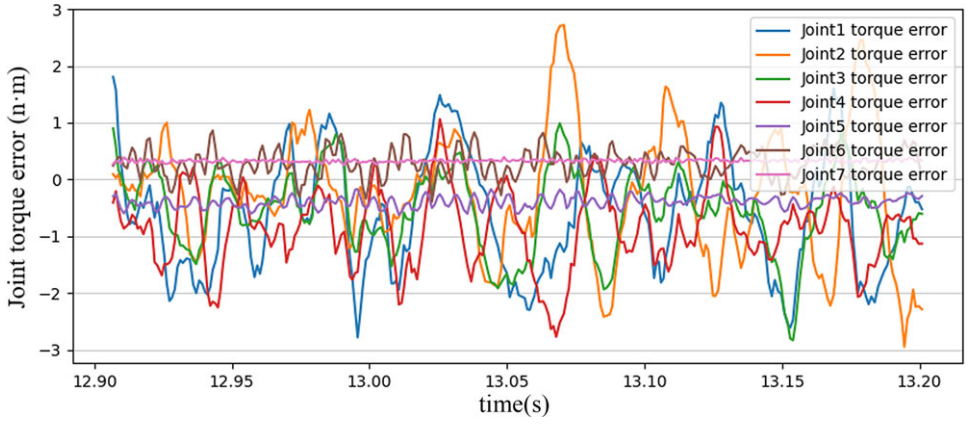

Figure 6. *Residual values of the seven robot joints.*

https://doi.org/10.1017/S0263574724001905 Published online by Cambridge University Press

Table II. *Loss training variation in the seven robot joints.*

	Joint 1	**Joint 2**	**Joint 3**	**Joint 4**	**Joint 5**	**Joint 6**	**Joint 7**
Conv_epoch	86	82	51	49	118	90	39
Loss_max	0.038	0.051	0.034	0.048	0.19	0.095	0.37
Loss_min	0.002	0.0018	0.0019	0.002	0.008	0.005	0.02
De_proportion	94.7%	96.4%	94.4%	95.8%	95.7%	94.7%	94.6%

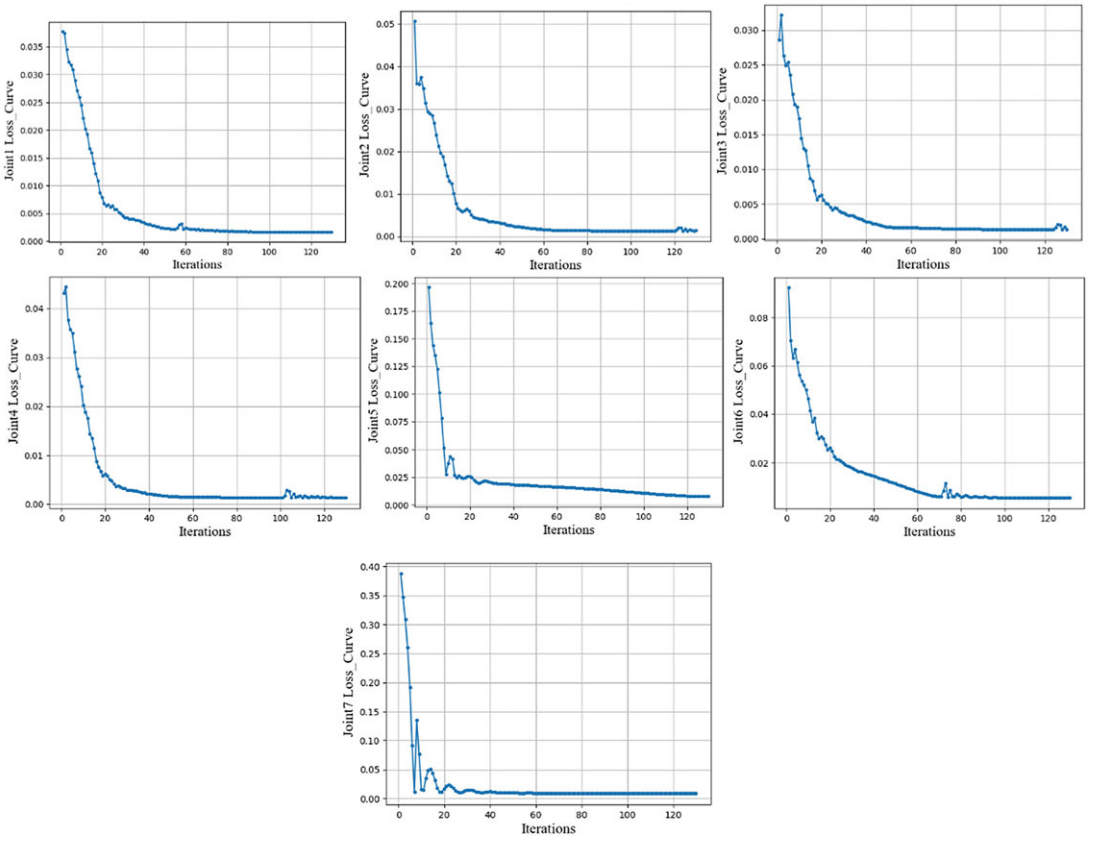

Figure 7. *Joints loss function curves of seven joints.*

4.2. Analysis of the model training set results

The hardware conditions of this model were an Intel i7-13700H CPU, 16 GB of memory, an NVIDIA RTX4060 graphics card with 8 GB of video memory, an operating system of Windows 11, and environmental configurations of Anaconda Python 3.9, PyTorch-Cuda 11.8, and PyTorch 2.1.0.

The input layer of the BLL algorithm is the torque residual of a single joint, which is the difference between the calculated torque and the actual torque of the RBD of the robot; the output layer of the BLL is a single channel, which is the predicted value of the torque residual of a single joint. The figure below shows the decline curve of the loss function when the BLL algorithm trains seven joint data points alone.

The fitness value function decreases as the number of iterations increases. After 800 iterative training tests, most converge within 130 iterations. Therefore, in this paper, the number of epochs is adjusted to 130. The training loss is shown in Fig. 7. The training data are presented in Table II.

Table II shows that the training results of the decrease rate of the fitness function of the model training were between 94.4% and 96.4%. Additionally, the model was able to fit the data points better. Next, the BLL parameter settings for training are introduced (Table III).

https://doi.org/10.1017/S0263574724001905 Published online by Cambridge University Press

Table III. *BLL training parameters.*

Parameter	Symbol	Optimization value
	LSTM_hidden_dim	32
BLL hidden layer dimension	Linear_hidden_dim1	100
	Linear_hidden_dim2	50
	Linear_hidden_dim3	25
BLL input dimension	input_dim	1
BLL output dimension	output_dim	1
BLL Learning rate	Learning_rate	0.01
Number of BLL iterations	num_epochs	130
Batch size for BLL	batch_size	32
BLL activation function	Activation	ReLU

In Table III, the dimensions of the BLL hidden layer include the original LSTM and fully connected layers. Therefore, the hidden dimension needs to be divided into LSTM_hidden_dim and linear_hidden_dim1-linear_hidden_dim3. The effect of a batch size of approximately 32 is the best. The following is the evaluation of the test set and the model comparison.

4.3. Results analysis on the model test set

To verify the effectiveness and improvement effect of the proposed BLL network, 290 verification points were arbitrarily selected within the range of 25,000 datasets to verify the training set. The comparison results of the actual residuals with the BLL-predicted residuals and the comparison between the actual residuals and model-compensated residuals are shown in Figs. 9 and 10, respectively.

The BLL network proposed in this paper uses four indicators: the MSE, the RMSE, the MAE, and the R2 score. We conducted a test set analysis [33]. The definitions and descriptions of these four indicators are given as follows:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2 \tag{4.a}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2} = \sqrt{MSE} \tag{4.b}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \hat{y}_i\right| \tag{4.c}$$

$$R2 = 1 - \frac{\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2}{\sum_{i=1}^{n}\left(y_i - \bar{y}_i\right)^2} = 1 - \frac{\frac{\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2}{n}}{\frac{\sum_{i=1}^{n}\left(y_i - \bar{y}_i\right)^2}{n}} = 1 - \frac{REMSE_|}{\text{Var}} \tag{4.d}$$

where y_i represents the true value of the dataset, $\hat{y}_i$ represents the predicted value of the dataset, $\bar{y}_i$ represents the mean value of the predicted value of the dataset, and n represents the number of datasets and the variance in the data. R2 represents the determination coefficient of the model. The best score is 1.0, indicating that the model predicts the true value perfectly. It can also be negative because the model can vary arbitrarily; that is, there is no mapping fit between the predicted and real data. Table IV shows the MSE, RMSE, MAR, and R2 values.

The determination coefficient of R2 of the predicted values of the specific residuals is shown in Fig. 8. R2 is approximately 0.84–0.98. The predicted values (blue dots) are closely distributed around the true

https://doi.org/10.1017/S0263574724001905 Published online by Cambridge University Press

Table IV. *Values of MSE, RMSE, MAE, and R2.*

	Joint 1	**Joint 2**	**Joint 3**	**Joint 4**	**Joint 5**	**Joint 6**	**Joint 7**
MSE	0.0432	0.0652	0.0291	0.0335	0.0046	0.0072	0.0010
RMSE	0.2078	0.2554	0.1705	0.1830	0.0678	0.0848	0.0316
MAE	0.1770	0.1453	0.1453	0.1391	0.0530	0.0649	0.0212
R2	0.9601	0.9455	0.9436	0.9656	0.9208	0.8428	0.9867

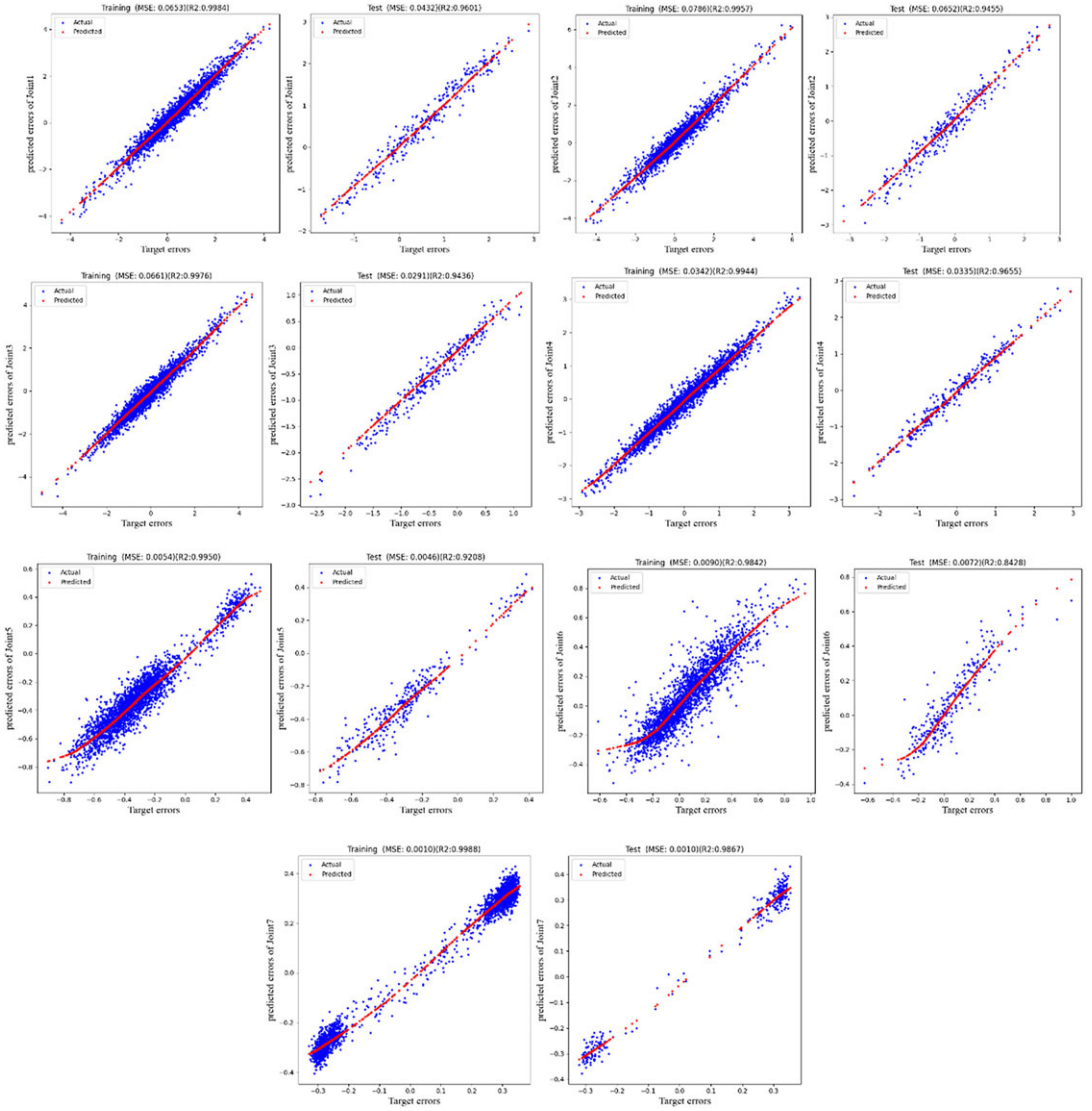

Figure 8. *Training and testing of joints 1–7.*

value (red line), with R2 close to 1, indicating that the predicted values have a high correlation with the actual value and a high fitting accuracy.

During the BLL network training, the training and iterations were performed in a single dimension, so the characteristics of each dimension were inconsistent. In addition, the accuracy of the robot end was affected by nonlinear factors such as the accuracy of the dataset and the environmental conditions,

https://doi.org/10.1017/S0263574724001905 Published online by Cambridge University Press

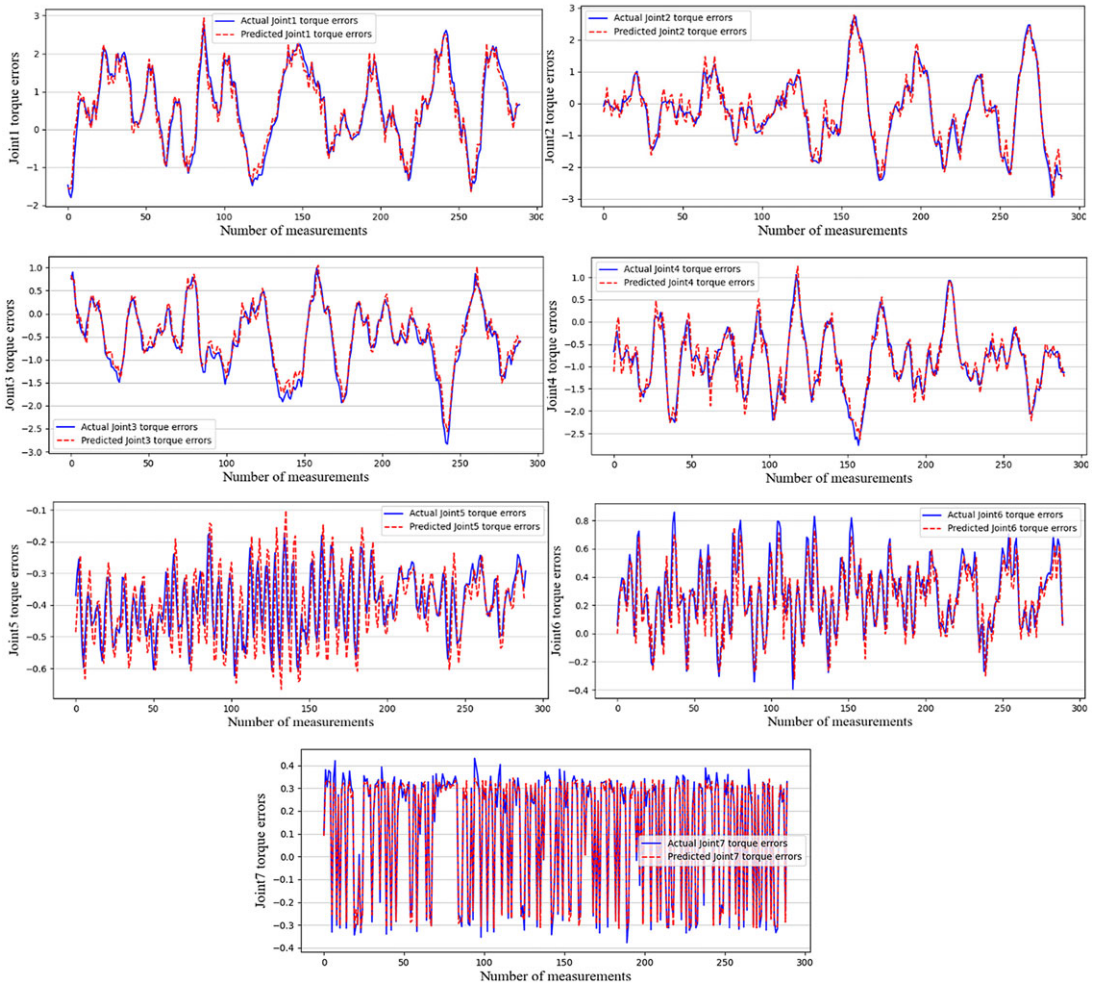

Figure 9. *Comparison of the actual residuals and the BLL-predicted residuals between joints1 and 7.*

which caused some differences in the effect of residual compensation of the end-accuracy of the robots, such as joint 6.

Table IV shows that the MSE, RMSE, and MAE of the predicted values of the joint robot residuals are all close to zero. Therefore, the proposed machine learning model is adaptable to the prediction of the residuals of the inverse dynamics model.

As Figs. 9 and 10 show, before compensation, the absolute values of the Joint1 residuals were approximately evenly distributed at [4.0374, 0.0006]; the absolute values of the Joint2 residuals were distributed at [5.4627, 0.0121]; the absolute values of the Joint3 residuals were distributed at [4.3725, 0.0066]; the absolute values of the Joint4 residuals were distributed at [2.9014, 0.0109]; the absolute values of the Joint5 residuals were distributed at [0.7850, 0.0039]; the absolute values of the Joint6 residuals were distributed at [0.6646, 0.0014]; and the absolute values of the Joint7 residuals were evenly distributed at [0.4305, 0.0088]. Using the BLL algorithm-based residual compensation method proposed in this paper, the residual errors of the seven robot joints are distributed at approximately 0, the fluctuations near $\pm 0.2N \cdot m$, and the fluctuation range is very small, indicating that the compensated accuracy has high stability and can improve the operating accuracy of the robot.

The results of the static statistical analysis of the robot inverse dynamics model before and after residual compensation are shown in Table V. Compared with the precompensation, the model residual of the seven robot joints was reduced [62.76–92.72%] to good effect.

https://doi.org/10.1017/S0263574724001905 Published online by Cambridge University Press

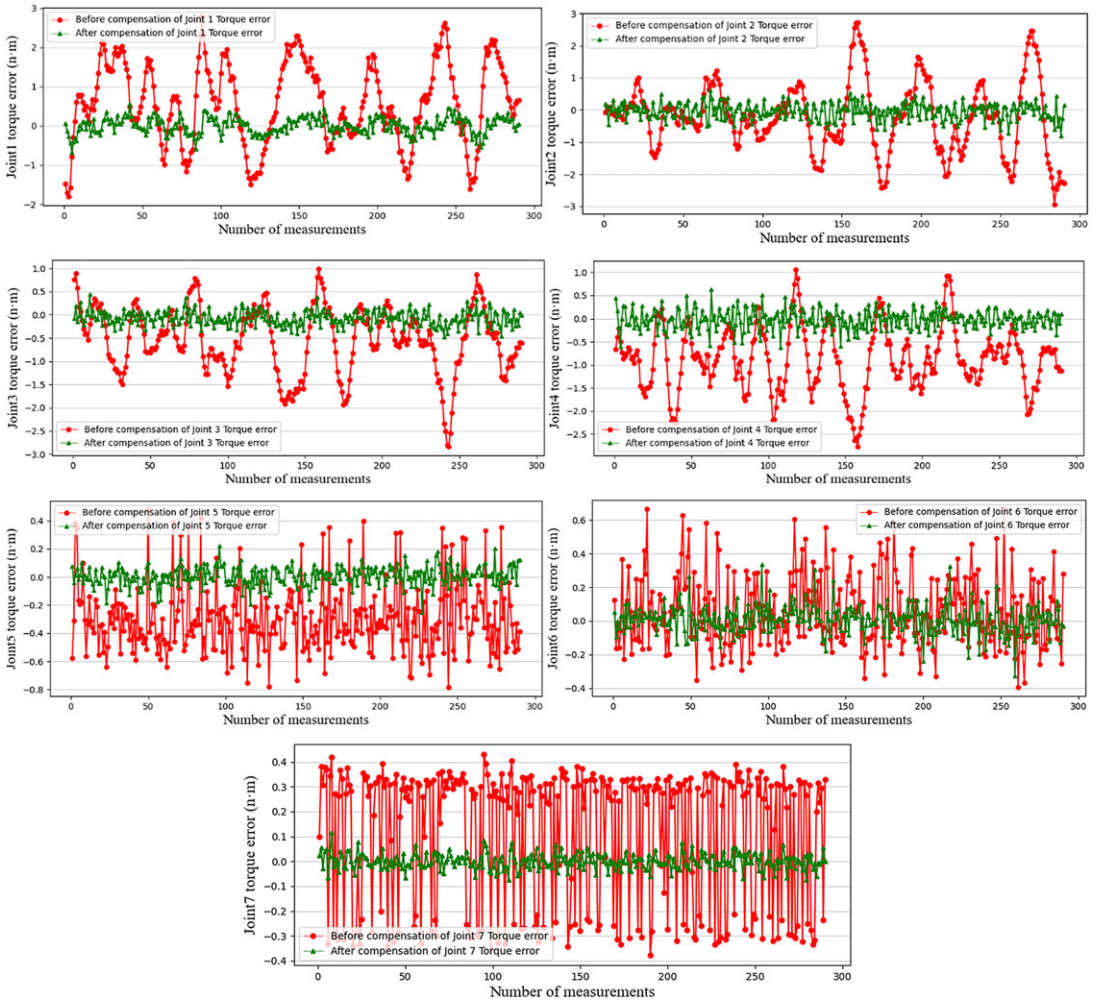

Figure 10. *Comparison of the actual residual error and the model-compensated residual error of joints 1–7.*

The robot dataset is based on Franka Emika Panda, a highly collaborative robot. The torque residual range is much smaller than that of traditional heavy-duty industrial robots. Therefore, using LSTM or GP [34] for feature extraction, model training, and optimization is more difficult. In this paper, the BLL algorithm is proposed to predict and compensate for the residuals of the Franka Emika Panda robot's inverse dynamics model, and a joint residual mapping model and a hybrid inverse dynamics model of the industrial robot are established. Figure 11 shows the seven joints reproducing the actual torques after rearranging the errors to compensate for them in the order in which they were collected. Note that the range and accuracy of the vertical axis are different for each figure. Therefore, the seven joints cannot be compared. The results show that the reproduction results of the 7 joints were better, and the residual compensation model was effective.

4.4. Results analysis on the model test set

To validate the test results, the method in this paper was compared with the LSTM algorithm [35] and the GP algorithm [36], which are commonly used in previous studies. The results are shown in Tables VI and VII. After offline compensation, the algorithm proposed in this paper outperforms the LSTM and GP algorithms in terms of the MSE, RMSE, MAE, and R2. Compared with those of the LSTM, the

https://doi.org/10.1017/S0263574724001905 Published online by Cambridge University Press

Table V. *Analysis of the positional errors.*

Joint error (N · m)		Error range	Percent improvement
Joint 1 error (N · m)	before	[4.0374, 0.0006]	81.29%
	after	[0.6190, 0.0009]	
Joint 2 error (N · m)	before	[5.4627,0.0121]	77.65%
	after	[0.7132,0.0013]	
Joint 3 error (N · m)	before	[4.3725,0.0066]	82.76%
	after	[0.4685,0.0009]	
Joint 4 error (N · m)	before	[2.9014,0.0109]	82.60%
	after	[0.6077,0.0015]	
Joint 5 error (N · m)	before	[0.7850,0.0039]	84.96%
	after	[0.2604,8.5e-05]	
Joint 6 error (N · m)	before	[0.6646,0.0014]	62.76%
	after	[0.3443,0.0003]	
Joint 7 error (N · m)	before	[0.4305,0.0088]	92.72%
	after	[0.0738,1.4e-05]	

Table VI. *Values of MSE, RMSE, MAE, and R2.*

Evaluation	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6	Joint 7
BLL_MSE	0.0432	0.0652	0.0291	0.0335	0.0046	0.0072	0.0010
LSTM_MSE	0.0497	0.0655	0.0320	0.0497	0.0086	0.0214	0.00067
GP_MSE	0.4988	0.5689	0.2228	0.5281	0.0744	0.0996	0.0255
BLL_RMSE	0.2078	0.2554	0.1705	0.1830	0.0678	0.0848	0.0316
LSTM_RMSE	0.2230	0.2560	0.1788	0.2229	0.093	0.1465	0.0260
GP_RMSE	0.7062	0.7542	0.4720	0.7267	0.2728	0.3156	0.1599
BLL_MAE	0.1770	0.1878	0.1453	0.1391	0.0530	0.0649	0.0212
LSTM_MAE	0.2042	0.2091	0.2069	0.1750	0.0730	0.1130	0.0254
GP_MAE	0.4605	0.4593	0.3224	0.4861	0.1915	0.2343	0.1132
BLL_R2	0.9601	0.9455	0.9436	0.9656	0.9208	0.8428	0.9867
LSTM_R2	0.9534	0.9252	0.9379	0.8959	0.8801	0.6314	0.9578
GP_R2	0.5332	0.5247	0.5682	0.5268	0.5123	0.5564	0.5621

mean values of the seven joints increase with respect to the MSE by 19.23%, RMSE by 12.68%, MAE by 21.69%, R2 by 5.84%, and the average residual reduction rate by 21.48%. Compared with those of the GP, the average increase in MSE is 90.89%, *t* in RMSE is 70.63%, in MAE is 65.23%, in R2 is 42.37%, and the residual reduction rate is 65.23%. Therefore, compared with the LSTM and GP methods, the RMSE, MAE, R2, and reduction rate of the average residual in the robot inverse dynamics model based on the BLL method proposed in this paper increased.

5. Conclusions

In order to improve the residual prediction accuracy of inverse dynamics models for industrial robots, this paper combines the bootstrap aggregating (bagging) algorithm with LSTM network, introduces linear layer as the network optimization layer, and proposes a method based on the BLL residual prediction algorithm, which is introduced as a network optimization layer. Additionally, hybrid inverse dynamics model compensation method for robots based on BLL residual prediction algorithm is proposed, and the framework of BLL residual prediction algorithm is given. The BLL residual prediction algorithm

https://doi.org/10.1017/S0263574724001905 Published online by Cambridge University Press

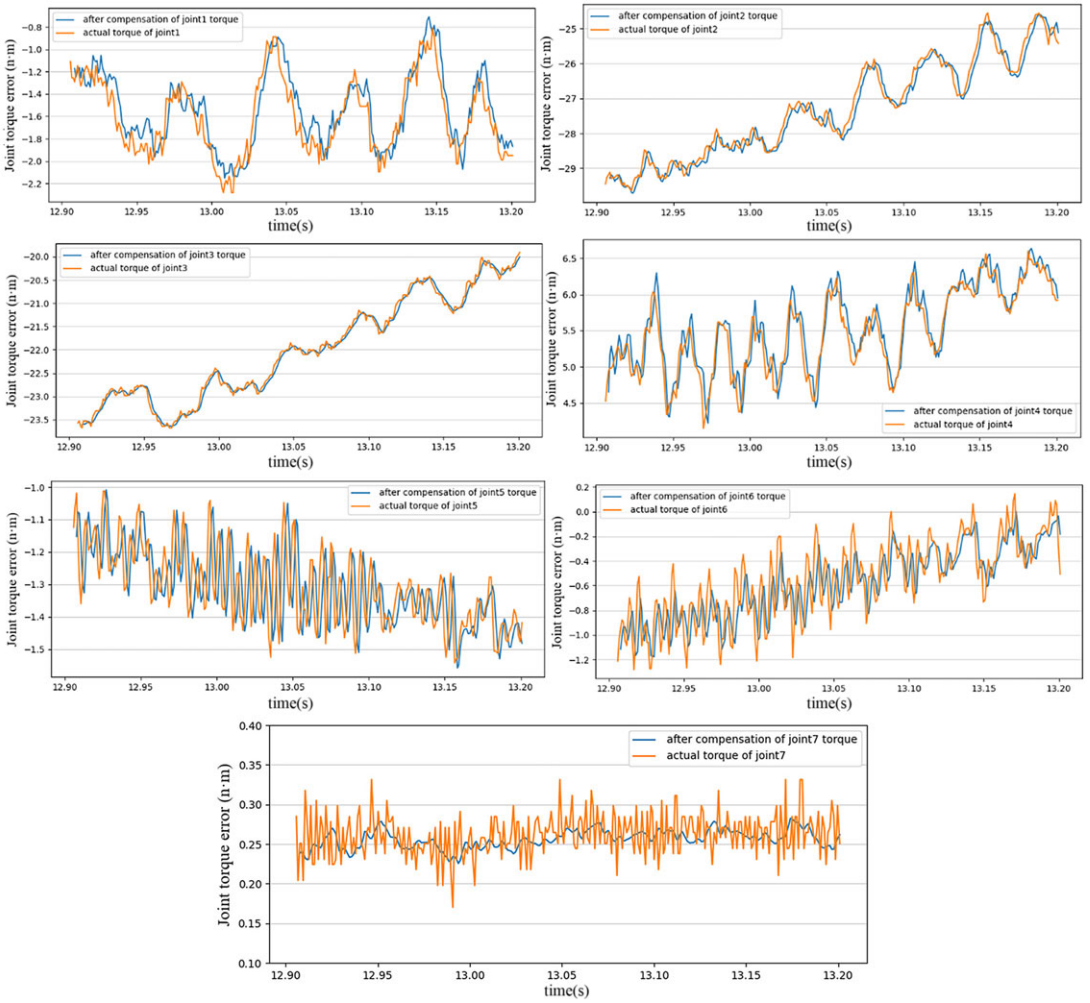

Figure 11. *Comparison of the actual torque of joints 1–7 and the calculated torque after model compensation.*

can achieve accurate prediction of residuals, and the algorithm can accurately estimate the residual value under the current value, which provides a prerequisite for the establishment of a hybrid inverse dynamics model.

To verify the effectiveness of the BLL residual model-based compensation method, we compared it with the rigid body inverse dynamics model. The torque residual of each robot joint decreased from 0.5651 N · m to 0.1096 N.m on average for the hybrid inverse dynamics model based on the BLL residual prediction algorithm. The calculation accuracy of the robot inverse dynamics model is improved, which is beneficial for aiding the robot to perform manipulation tasks more accurately. The BLL algorithm proposed in this paper was compared and analysed with the LSTM and GP algorithms in this field, and the BLL algorithm proposed in this paper achieved improvements in MSE, RMSE, MAE, R2, and the reduction rate of average residuals after compensation, which validated the effectiveness of the proposed algorithm in this paper. The study laid the foundation for accurately modelling the inverse dynamics of industrial robots.

https://doi.org/10.1017/S0263574724001905 Published online by Cambridge University Press

Table VII. *Values of MSE, RMSE, MAE, and R2 comparison of values after error compensation of the BLL, LSTM, and GP algorithm.*

Error (N m)	Max	Min	Average
Uncompensated joint 1	4.0374	0.0006	0.9464
LSTMs	0.9323	0.0028	0.2042
GP	2.199	7.3e-12	0.4605
BLL	0.6190	0.0009	0.1770
Uncompensated joint 2	5.4627	0.0121	0.8404
LSTMs	1.0800	0.0011	0.2069
GP	2.9480	7.6 e-12	0.4592
BLL	0.7132	0.0013	0.1878
Uncompensated joint 3	4.3725	0.0066	0.8429
LSTMs	0.9004	0.0027	0.2069
GP	1.5329	6.5e-12	0.3224
BLL	0.4685	0.0009	0.1453
Uncompensated joint 4	2.9014	0.0109	0.7996
LSTMs	0.6605	0.0015	0.1750
GP	2.2541	2.0e-12	0.4861
BLL	0.6077	0.0007	0.1391
Uncompensated joint 5	0.7850	0.0039	0.3525
LSTMs	0.3194	0.0001	0.0730
GP	0.6510	7.7e-12	0.1914
BLL	0.2604	8.5e-05	0.0530
Uncompensated joint 6	0.6646	0.0014	0.1743
LSTMs	0.6068	0.0002	0.1130
GP	0.6042	1.4e-12	0.2342
BLL	0.3443	0.0003	0.0649
Uncompensated joint 7	0.4305	0.0088	0.2911
LSTMs	0.1178	2.10e-05	0.0254
GP	0.3859	5.9e-13	0.1132
BLL	0.0738	1.4e-05	0.0212

Author contribution. Conceptualization, Y.T. and S.C.; methodology, Y.T.; software, H.L. and S.C.; validation, J.W.; formal analysis, S.C.; investigation, H.W.; resources, H.L.; data curation, S.C.; writing – original draft preparation, S.C.; writing – review and editing, S.C.; visualization, S.C.; supervision, J.W.; project administration, Y.T.; funding acquisition, Y.T. All authors have read and agreed to the published version of the manuscript.

Financial support. This research was funded by the Ministry of Industry and Information Technology of the People's Republic of China, National Key Research and Development Plan "Intelligent Robot" Project No. 2022YFB4700400.

Competing interests. The authors declare no competing interests.

References

[1] L. Gao, J. Yuan, Z. Han, S. Wang and N. Wang, "A Friction Model with Velocity, Temperature and Load Torque Effects for Collaborative Industrial Robot joints," In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE (2017) pp. 3027–3032. doi: 10.1109/IROS.2017.8206141.

[2] M. Iskandar and S. Wolf, "Dynamic Friction Model with Thermal and Load Dependency: Modelling, Compensation and External Force Estimation[C]," In: *2019 International Conference on Robotics and Automation (ICRA), IEEE*, IEEE (2019) pp. 7367–7373.

https://doi.org/10.1017/S0263574724001905 Published online by Cambridge University Press

[3] T. C. Çallar and S. Böttger, "Hybrid learning of time-series inverse dynamics models for locally isotropic robot motion[J]," *IEEE Robot Autom Lett* **8**(2), 1061–1068 (2022

[4] D. Romeres, M. Zorzi, R. Camoriano, S. Traversaro and A. Chiuso, "Derivative-free online learning of inverse dynamics models[J]," *IEEE Trans Contr Syst Technol* **28**(3), 816–830 (2019).

[5] M. Seeger, "Gaussian processes for machine learning[J]," *Int J Neural Syst* **14**(02), 69–106 (2004).

[6] A. D. Libera, G. Giacomuzzo, R. Carli, D. Nikovski and D. Romeres, Forwards Dynamics Estimation from Data-Driven Inverse Dynamics Learning. arXiv e-prints, 2023: arXiv: 2307.05093.

[7] M. Lutter, K. Listmann and J. Peters, "Deep Lagrangian Networks for End-to-End Learning of Energybased Control for Underactuated Systems," In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE (2019) pp. 7718–7725.

[8] S. Greydanus, M. Dzamba and J. Yosinski, "Hamiltonian neural networks," *Adv Neur Inf Process Syst* **32**, 1–5 (2019)

[9] J. K. Gupta, K. Menda, Z. Manchester and M. Kochenderfer, "Structured Mechanical Models for Robot Learning and Control," In: *Learning for Dynamics and Control*, (PMLR, 2020) pp. 328–337.

[10] F. Meier, D. Kappler, N. Ratliff and S. Schaal, "Towards Robust Online Inverse Dynamics Learning," In: *In. 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE (2016) pp. 4034–4039.

[11] D. Nguyen-Tuong and J. Peters, "Using Model Knowledge for Learning Inverse Dynamics," In: *2010 IEEE International Conference on Robotics and Automation*, IEEE (2010) pp. 2677–2682.

[12] Y. Wang and Y. Liu, "Adaptive output-feedback tracking for nonlinear systems with unknown control direction and generic inverse dynamics[J]," *Sci China Inf Sci* **65**(8), 182204 (2022). doi: 10.1007/s11432-020-3207-3.

[13] D. Kappler, F. Meier, N. Ratliff and S. Schaal, "A New Data Source for Inverse Dynamics Learning," In: *In. 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE (2017) pp. 4723–4730.

[14] X. Zhao, J. Wang, L. Liu, "Research on the inverse dynamics of the planar 3-RRR rigid-flexible coupling parallel robot[J]," *J Mach Des* **36**(11), 8 (2019). doi: CNKI: SUN: JXSJ.0.2019-11-002

[15] P. Sun, Z. Shao, Y. Qu, Y. Guan and J. Tan, "Inverse Dynamics Modelling of Robotic Manipulator with Hierarchical Recurrent Network[C]," In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE (2019) pp. 751–756.

[16] V. Shaj, P. Becker, D. Buchler, H. Pandya, C. J. T. Niels van Duijkeren, M. Hanheide and G. Neumann, "Action-Conditional Recurrent Kalman Networks for Forwards and Inverse Dynamics Learning," In: *Conference on Robot Learning*, (2020) pp. 1–5.

[17] M. W. Spong, "An historical perspective on the control of robotic manipulators[J]," *Annu Rev Control Robot Auton Syst* **5**(1), 1–31 (2022).

[18] B. An, J. Chen, H. Sun, M. Yin, Z. Song, C. Zhuang, C. Chang and M. Liu, "Optimization of fracture reduction robot controller based on improved sparrow algorithm[J]," *Biomimetic Intell Robot* **3**(4), 100120 (2023).

[19] G. Chen, Y. Xu, X. Yang, H. Hu, H. Cheng, L. Zhu, J. Zhang, J. Shi and X. Chai, "Target tracking control of a bionic mantis shrimp robot with closed-loop central pattern generators[J]," *Ocean Eng* **297**, 116963 (2024).

[20] G. Wu, On the elastodynamics of a five-axis lightweight an-thropomorphic robotic arm[J], (2021).

[21] G. Chen, Y. Xu, Z. Wang, J. Tu, H. Hu, C. Chen, Y. Xu, X. Chai, J. Zhang and J. Shi, "Dynamic tail modeling and motion analysis of a beaver-like robot[J]," *Nonlinear Dynam* **112**(9), 6859–6875 (2024).

[22] P. Tatjewski, "Effective nonlinear predictive and CTC-PID control of rigid manipulators[J]," *J Autom Mobile Robot Intell Syst* **18**(2), 1–16 (2024).

[23] D. Zhang, Y. Xu, J. Yao and Y. Zhao, "Inverse dynamic analysis of novel 5-DOF hybrid manipulator [J]," *Trans Chinese Soc Agric Mach* **48**(9), 8 (2017). doi: 10.6041/j.issn.1000-1298.2017.09.049 .

[24] N. Yilmaz, J. Y. Wu, P. Kazanzides and U. Tumerdem, "Neural Network based Inverse Dynamics Identification and External Force Estimation on the da Vinci Research Kit[C]," In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE (2020) pp. 1387–1393.

[25] M. Wang and M. Wang, "Dynamic modeling and performance evaluation of a new five-DOF hybrid robot [J]," *J Mech Eng* **59**(9), 63–75 (2023). doi: 10.3901/JME.2023.09.063.

[26] L. Breiman, "Bagging predictors [J]," *Mach Learn* **24**(2), 123–140 (1996).

[27] A. Graves, "Long Short-Term Memory [J]," In: *Supervised Sequence Labelling with Recurrent NNs*, (Springer, 2012) pp. 37–45.

[28] H. Abdi, "A neural network primer [J]," *J Biol Syst* **2**(03), 247–281 (1994).

[29] H. Zhao, Z. Lai, H. Leung and X. Zhang, "Neural-Network-Based Feature Learning: Recurrent Neural Network[J]," In: *Feature Learning and Understanding: Algorithms and Applications*, (Springer, 2020) pp. 253–275.

[30] N. Liu, L. Li, B. Hao, L. Yang, T. Hu, T. Xue and S. Wang, "Modelling and simulation of robot inverse dynamics using LSTM-based deep learning algorithm for smart cities and factories[J]," *IEEE Access* **7**, 173989–173998 (2019).

[31] C. Gaz, M. Cognetti, A. Oliva, P. Robuffo Giordano and A. De Luca, "Dynamic identification of the Franka Emika Panda robot with retrieval of feasible parameters using penalty-based optimization," *IEEE Robot Autom Lett* **4**(4), 4147–4154 (2019). doi: 10.1109/lra.2019.

[32] M. Lutter, *A Differentiable Newton–Euler Algorithm for Real-World Robotics[M]//Inductive Biases in Machine Learning for Robotics and Control* (Springer Nature Switzerland, Cham, 2023) pp. 9–34.

[33] M. W. Spong, S. Hutchinson and M. Vidyasagar, *Robot modeling and control[M]* (John Wiley & Sons, USA 2020).

[34] S. Rezaei-Shoshtari, D. Meger and I. Sharf, "Cascaded Gaussian Processes for Data-Efficient Robot Dynamics Learning[C]," In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (IEEE, 2019) pp. 6871–6877.

https://doi.org/10.1017/S0263574724001905 Published online by Cambridge University Press

[35] X. U. Zheng, Z. Gong, W. Huo-Ming, H. Zhi-Cheng, Y. Wen-Lin, L. Ji-min, W. Jian and G. Xing, "Error compensation of collaborative robot dynamics based on deep recurrent neural network [J]," *Chinese J Eng* **43**(7), 995–1002 (2021).
[36] E. Rueckert, M. Nakatenus, S. Tosatto and J. Peters, "Learning Inverse Dynamics Models at o (n) Time with LSTM networks[C]," In: *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, IEEE (2017) pp. 811–816.

Cite this article: Y. Tao, S. Chen, H. Liu, J. Wan, H. Wei and T. Wang, "Robot hybrid inverse dynamics model compensation method based on the BLL residual prediction algorithm", Robotica. https://doi.org/10.1017/S0263574724001905