*Robotica* (2025), **43**, pp. 1027–1042
doi:10.1017/S0263574725000086

CAMBRIDGE UNIVERSITY PRESS

RESEARCH ARTICLE

# A novel joint external torque estimate model of the lightweight robot's joint based on a BP neural network

Tao Zhang[1], Haochong Li[1], Yongping Shi[1,2], Lei Wang[1], Xuanchen Zhang[3], Jun Zhang[3] and Huapeng Wu[2]

[1]School of Mechanical Engineering, Hefei University of Technology, Hefei, China
[2]Lappeenranta University of Technology (LUT), Lappeenranta, Finland
[3]Institute of Plasma Physics Chinese Academy of Sciences (ASIPP), Chinese Academy of Sciences, Hefei, China
**Corresponding authors:** Haochong Li; Email: haochongli2023@163.com, Yongping Shi; Email: yongpingshi123@gmail.com

**Received:** 19 July 2024; **Revised:** 11 November 2024; **Accepted:** 6 January 2025; **First published online:** 4 February 2025

**Keywords:** robot joint external torque; momentum observer; backpropagation neural network

## Abstract

The safety of human-collaborative operations with robots depends on monitoring the external torque of the robot, in which there are toque sensor-based and torque sensor-free methods. Economically, the classic method for estimating joint external torque is the first-order momentum observer (MOB) based on a physic model without torque sensors. However, uncertainties in the dynamic model, which encompasses parameters identification error and joint friction, affect the torque estimation accuracy. To address this issue, this paper proposes using the backpropagation neural network (BPNN) method to estimate joint external torque without the delicate physical model by utilizing the powerful machine learning ability to handle the uncertainties of the MOB method and improve the accuracy of torque estimation. Using data obtained from the torque sensor to train the BPNN to build up a digital torque model, the trained BPNN can perceive force in practical applications without relying on the torque sensor. In the end, by contrast to the classic first-order MOB, the result demonstrates that BPNN achieves higher estimation accuracy compared to the MOB.

## 1. Introduction

Robots are playing an important role in industrial applications with modern manufacturing development. The scenario of modern manufacturing tasks is more delicate and numerical, thus the man-robot collaboration is necessary. As shown in Figure 1, lightweight robots are a typical device in man-robot collaboration surroundings and have been widely used in assembly [1], polishing [2], rehabilitation [3, 4], and other occasions with the advantages of compact structure, lightweight, stable work, and so on. Man would engage in the above scenarios so that the collision detection function of a lightweight robot is necessary to ensure the safety of humankind. Traditionally, such as in industrial robots or normal robots, the collision detection function is unequipped so that they would integrate the torque or force sensor to sense the external surroundings, which is expensive because of the costs. However, the torque or force generated by the payloads or outer force is relative to the robot state message such as joint current so that the current loop can be deeply studied replacing the torque sensor to decrease the cost. In lightweight robot scenarios, when using the force or current loop as control mode, its dynamic model should be established with some external torque sensor mathematical model to identify the force or torque to improve its performance in collaborative tasks such as collision detection. Hence, in this technical roadmap, the real-time measurement and delicate model of the robot's joint external torque are vital for a well-behaved lightweight robot study.

There are some mainstream roadmaps in this research field. Foremost, using a force/torque sensor [5, 6] mounted at the end effector or base or installing torque sensors at each joint can obtain the robot

© The Author(s), 2025. Published by Cambridge University Press.

https://doi.org/10.1017/S0263574725000086 Published online by Cambridge University Press

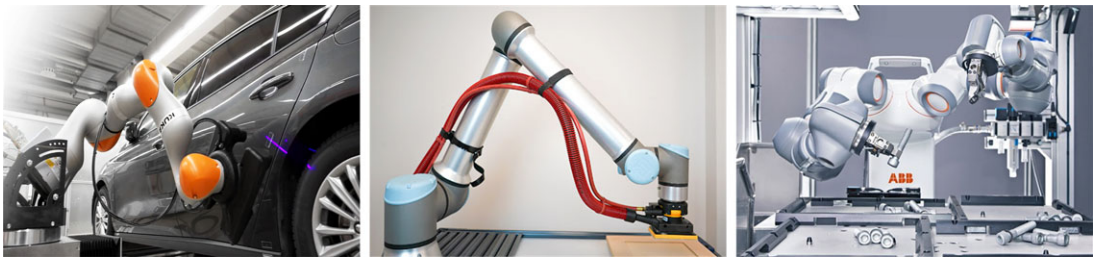


***Figure 1.*** *Light-robot application in industrial scenery.*



***Figure 2.*** *Diverse applications of joint modules.*

torque directly. For instance, installing a six-axis force/torque sensor on the end effector can detect external torque directly on the end joint [7], while it cannot detect torque on the other part of the robot, such as force/torque on the arm. Under consideration of this issue, some researchers installed a six-axis force/torque sensor on the robot base to detect external force/torque on the entire robot so that the force/torque on the end joint can be estimated simultaneously [8]. However, the payload of the torque/force sensor limits the size and weight of the robot equipped on the sensor, moreover, the big payload of this kind of sensor would cost too much for the entire device and the algorithm based on the torque sensor would be more difficult. Another solution is artificial robot skin for robots [9–11], but large-area skin patches would be expensive. To reduce the cost of external torque detection, using existing sensors in the robot such as joint encoders and current sensors/loops to design an algorithm identifying external torque without an external force/torque sensor, which is efficient, highly integrated, and cheap. Among the methods without external force/torque sensors, momentum observer (MOB) [12, 13] based on the dynamic model is widely used, which can avoid introducing acceleration terms with noise and delay [14]. However, the identification of dynamic parameters and numerical differentiation of position can lead to the model's uncertainties, which affect the estimation accuracy of external torque, because these changes can be regarded as external disturbances, which are difficult to decouple [15]. Data-driven machine learning [16, 17] and statistical algorithms have a big advantage in handling model uncertainties by using an optimal and well-trained model, which can effectively identify and understand complex correlations within data when the precise physical model is hard to build.

This paper introduces an external torque estimation algorithm based on the BP Neural Network (BPNN). Before applying the BPNN model, it is trained using measurable joint state data and torque sensor data. Once trained, the BPNN can accurately estimate external torque based solely on joint state data, eliminating the need for costly torque sensors. Furthermore, this approach also eliminates the reliance on a physical model, reducing the impact of model uncertainties on torque estimation. This algorithm is meaningful for the current lightweight robots modular design and some issues also should be mentioned as follows: Firstly, the joint can be assembled into different robots such as lightweight robots, medical robots, and so on according to its workspace, flexibility, application areas, and payload requirements. For example, TECH robot in Figure 2, a Chinese robot brand adopt the modular design to assemble the different robots. This modular design allows each joint to operate independently, making it straightforward to apply the external torque estimation method, designed for individual joints, to the

https://doi.org/10.1017/S0263574725000086 Published online by Cambridge University Press

entire system. Secondly, each of the joints is different in choosing the different power motor, reducer, mechanism, and so on, and the joints should be tested by the method before integrating into the whole arm to ensure its validation. Lastly, the modular design method makes it possible to achieve the part of the design of the robot becoming a standard product (the robot joint modules) and can be manufactured in standardization administration.

Section 1 introduces the background and related work to relative research. The external torque estimation based on the physic model is introduced by the order of the robot dynamic model, the parameter identification method in Section 2, and the author summarized the first-order MOB algorithm which is the foundation of the robot dynamic model in the appendix for better understanding the method mentioned in this article. Further, a physic-free model for estimating external torque is in Section 3. Section 4 designs an experimental platform to verify the above two methods and the parameter identification results based on this experiment are analyzed to compare the physics-based model and physic-free model in Section 5. Finally, Section 6 summarizes this work and proposes future work for torque estimation technology.

## 2. External torque estimation based on physical model

Among the physics-based model research algorithms for estimating external torque in robots, the MOB is the one with the best performance [18]. The classic first-order MOB always needs two assumptions [19]: the dynamic model is perfectly known and the friction torque is negligible or known. Therefore, this section will focus on obtaining the robot's dynamic model (including the friction model), primarily addressing the aspects of modeling and parameter identification. Additionally, the appendix introduces some foundations of the MOB algorithm based on the physical dynamic model for a better understanding of the method of this article. At the end of the paper, a comparison is made between the performance of the MOB algorithm and the BPNN algorithm (physic-free model) in external torque estimation.

### *2.1. Physical dynamic model*

Use the Lagrangian approach to establish the robot dynamic model.

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

$$\tau = K_t I \tag{2}$$

where $M(q)$, $C(q,\dot{q})$, and $G(q)$ represent the inertia matrix, the Coriolis matrix, and the gravity matrix, respectively, $q$, $\dot{q}$, and $\ddot{q}$ represent the joint position, joint velocity, and joint acceleration, respectively, $\tau_e$ is the external torque applied at the end of the joint, $\tau$ is the drive torque of joint module which is proportion of drive current according to Eq. (2), and $K_t$ is the torque coefficient.

For a lightweight robot, the whole body can be regarded as an integration of each joint module. Hence, the single joint module's model and its control are the foundation of the whole robot's performance. For the joint module, a test platform is established as shown in Figure 3 which includes the motor, reducer, coupling, torque sensor, and load which can simulate different external work conditions.

Considering joint friction, the dynamic model in Figure 3 is represented by Eq. (3). Since the motor rotates around a fixed horizontal axis, both the Coriolis matrix $C(q,\dot{q})$ and the gravity matrix $G(q)$ in Eq. (1) are zero. The inertia matrix $M(q)$ has only one term, $I_{zz}$, which represents the system's moment of inertia about the z-axis.

$$\tau + \tau_e = I_{zz}\ddot{q} + \tau_f \tag{3}$$

$$\tau_f = f_c \mathrm{sgn}(\dot{q}) + f_v \dot{q} \tag{4}$$

where $\tau_f$ is the friction torque, described by the classical Coulomb-viscous friction model, as shown in Eq. (4), and $f_c$ and $f_v$ represent the coefficient of Coulomb friction and viscous friction, respectively.

https://doi.org/10.1017/S0263574725000086 Published online by Cambridge University Press

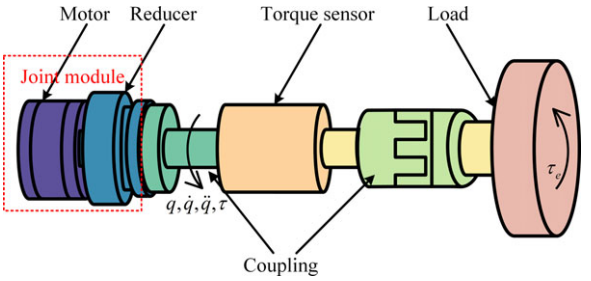



***Figure 3.*** *Schematic diagram of a single joint system.*

However, some issues should be mentioned. The Coulomb-Viscous friction model is only suitable for describing simple frictional behaviors. However, in practical applications, friction forces are often more complex. The Coulomb-Viscous model has limitations when addressing these complexities in unstable or nonlinear friction phenomena. For instance, the friction force exhibits significant variation across different velocity ranges, the Stribeck effect [20]. It's difficult for the traditional Coulomb friction model to accurately capture this nonlinear characteristic, especially in low speeds or static states, where friction forces may exhibit dramatic changes.

### *2.2. Parameter identification of physical dynamic model*

The single joint module dynamic model which is described as Eq. (3) is linear so that its dynamic parameters can be identified using the least squares method (LSM) directly. The core idea of LSM is to minimize the sum of squared residuals between the actual observed values and the model estimates to seek optimal parameters fitting model. To promote the efficiency of identification, the joint mechanism must be operated without a load, in other words, $\tau_e$ is zero. The dynamic model can be further represented in Eq. (5) according to Eq. (3) and Eq. (4).

$$\tau = Y\left(\dot{q}, \ddot{q}\right) P \tag{5}$$

where $Y(\dot{q}, \ddot{q})$ is the measurement matrix, specifically $[\ddot{q}, \mathrm{sgn}(\dot{q}), \dot{q}]$, and $P$ is the parameter identification matrix, specifically $[I_{zz}, f_c, f_v]^T$.

Then use Eq. (6) to identify parameter identification matrix.

$$P = \left[Y^T Y\right]^{-1} Y^T \tau \tag{6}$$

The parameter identification procedure uses the robot's trajectory experiments to form the datasets. Hence, the excitation trajectory design would influence the accuracy of robot dynamic parameter identification [21]. The sampling datasets furtherly form the coefficient matrix $Y(\dot{q}, \ddot{q})$. While the condition number of coefficient matrix is large or ill-conditioned condition number, the dynamic model would be numerical sensitive, which means tiny input's changes (robot state variables' change) would affect the output's accuracy (torque's accuracy in Eq. (5). Therefore, the excitation trajectory should be designed with its coefficient matrix having a small condition number, specifically a well-conditioned condition number to reduce errors and uncertainties caused. Additionally, the excitation trajectory should be periodic and smooth for repeatable robot motion so that the Fourier series [22], composed of sine and cosine functions with the periodic feature and continuous higher-order derivatives is designed to meet the above requirements. In this article, a third-order Fourier series, as shown in Eq. (7), is chosen as the excitation trajectory for dynamic parameters identification.

$$q(t) = a + \sum_{n=1}^{3} \left[ \frac{b_n}{\omega_f n} \sin\left(\omega_f n t\right) - \frac{c_n}{\omega_f n} \cos\left(\omega_f n t\right) \right] \tag{7}$$

where $t$ is time, $a$ is a constant, $b_n$ and $c_n$ are the Fourier coefficients, $\omega_f$ is the angular frequency.

https://doi.org/10.1017/S0263574725000086 Published online by Cambridge University Press

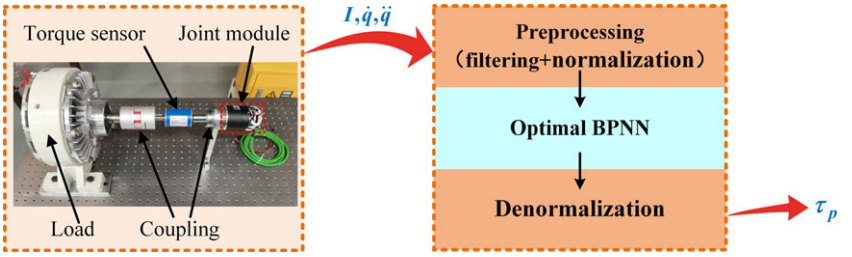



***Figure 4.*** *Estimating external torque model by BPNN.*

To ensure that the joint mechanism adheres to its hardware's physical motion limitations while executing the trajectory, and robot's smooth start or stop, the boundary condition of the excitation trajectory is shown in Eq. (8).

$$\begin{cases} \dot{q}\,(t) \le \dot{q}_{max} \\ q(t_0) = q(t_f) = 0 \\ \dot{q}(t_0) = \dot{q}(t_f) = 0 \\ \ddot{q}(t_0) = \ddot{q}(t_f) = 0 \end{cases} \tag{8}$$

where $\dot{q}_{max}$ is the upper limit of the joint angular velocity, $t_0$ is the initial time of the trajectory, and $t_f$ is the final time of the trajectory.

## 3. External torque estimation based on physic-free model

The appendix introduces a physics-based model external torque estimation method according to the classical MOB mechanism, which has a complicated mathematical model and is sensitive to the excitation trajectory. Thus, there is an intuition to find a physic-free model to build up an external torque estimation. According to the previous study, the dynamic model encompassing parameters identification error and joint friction has uncertainties that is nonlinear, so an intelligent algorithm may have a better fitting feature. Through the analysis of the mechanics and physics of the robot's joint, the external torque is relative to some joint's state such as current, velocity, and acceleration. In addition, this kind of joint is periodic so the joint value would be less relative to the joint torque. Following the mechanism of Figure 3, a torque estimation prototype is built up, which is shown on the left of Figure 4. Through the PLC controller, the robot joint state such as current, velocity, and acceleration can be sampled, and these datasets can form the input of the physic-free model, which estimates the external torque. In this article, a BPNN model is designed to connect the external torque and robot joint state by its powerful nonlinear predictive through supervised machine learning, whose core idea is shown in Figure 4. BPNN is a multilayer feedforward neural network trained by an error back propagation mechanism, whose core structures are data processing, BPNN design, and its training methods. The following section focuses on these aspects of study and design serving our research objects.

### *3.1. Data processing of physic-free model*

To improve the data quantity of our external torque estimation BPNN model, data preprocessing is necessary to filter some error values such as outers and normalize data to promote the training performance. The specific filtering method is described in Section 4.2. Besides, the data preprocessing is performed using linear normalization as shown in Eq. (9) and its denormalization is shown in Eq. (10).

$$y^* = \frac{y - y_{min}}{y_{max} - y_{min}} \tag{9}$$

https://doi.org/10.1017/S0263574725000086 Published online by Cambridge University Press

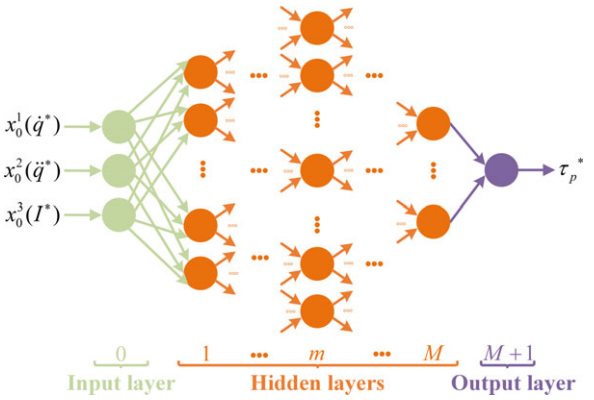



***Figure 5.*** *BP neural network of Physic-Free Model.*

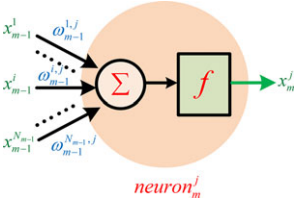



***Figure 6.*** *The principle of neuron's mechanism.*

where $y$ is the original data, $y_{max}$ is the maximum value of $y$, $y_{min}$ is the minimum value of $y$, and $y^*$ is the preprocessed data.

$$y = (y_{max} - y_{min}) * y^* + y_{min} \tag{10}$$

### *3.2. BPNN structure in physic-free model*

The BP neural network of physic-free model is made of an input layer, several hidden layers, and an output layer as Figure 5 shows. The output layer and input layer's neurons are determined by the dimension of input data (three dimensions which are the velocity, acceleration, and current) and output data (one dimension which is the external torque) respectively.

In the hidden layers, every neuron has its inputs and outputs. For a neuron, each of the inputs is summarized together by the linear combination and furtherly functioned by a nonlinear transformation to output its result as Eq. (11) shows. The principle of the $i^{th}$ neuron in the $m^{th}$ layer is designed in Figure 6.

$$x_m^j = f_m\left(\sum_{i=1}^{N_{m-1}} \omega_m^{i,j} x_{m-1}^i\right), j \in [1, 2, \ldots, N_m] \tag{11}$$

where $x_m^j$ is the output of the $neuron_m^j$, $f_m$ is the activation function of $m^{th}$ layer, which is always a nonlinear function. $\omega_m^{i,j}$ is the weight between $neuron_m^j$ and $neuron_{m-1}^i$, and all weights of the neural network are shown in Eq. (12).

$$\omega = [\omega_1, \cdots \omega_m, \cdots, \omega_{M+1}], \omega_m = \left[\omega_m^{1,1}, \cdots, \omega_m^{i,j}, \cdots, \omega_m^{N_{m-1},N_m}\right]^T \tag{12}$$

While the neuron principle is determined, the number of hidden layers and the number of neurons in each layer need to be designed. The more complex the hidden layers, the stronger the network's fitting ability, but excessively complex hidden layers may lead to overfitting, reducing the generalization ability of the network [23]. The design of the hidden layers mostly relies on empirical knowledge and can refer to Eq. (13).

$$N_m = \sqrt{N_{in} + N_{out}} + d \tag{13}$$

https://doi.org/10.1017/S0263574725000086 Published online by Cambridge University Press

where $N_m$, $N_{in}$, and $N_{out}$ represent the number of neurons in the hidden layers, input layer, and output layer, respectively, $d$ is a constant between 0 and 10. After determining the structure of a network, training is continuously needed to obtain the optimal parameters of the network.

### *3.3. BPNN training in physic-free model*

To verify the model in Section 3.2, 85% of data sets with a size of $N_{tr}$ are used for network training, and the other part with a size of $N_{val}$ is used for verifying the network performance, which are the training set and the validation set, respectively. The training process of BPNN includes two parts: forward propagation and backward propagation, and the aim of the training is to optimize the initial weights until the optimal solution is solved continuously. In our model, all of data are processed before which are $[I^*, \dot{q}^*, \ddot{q}^*]^T$ to $\tau_p{}^*$. For the forward propagation, each set of data inputs into the network, passes through the hidden layer in order, and finally goes through the output layer to obtain the estimated result $\tau_p{}^*$ according to Eq. (11) and Eq. (14).

$$\tau_p{}^* = f_{M+1}\left(\sum_{i=1}^{N_M} \omega_{M+1}^{i,1} x_M^i\right) \tag{14}$$

where $\tau_p{}^*$ is the output of the output layer neuron. For the convenience of subsequent tasks, the mathematical operation $\Phi$ is determined for reshaping $\omega$ into a column vector $\Omega$ of length $n$ in Eq. (15).

$$\Omega = \Phi\left(\omega\right) = [\omega_1{}^T, \cdots \omega_m{}^T, \cdots, \omega_{M+1}{}^T]^T \tag{15}$$

The actual $\tau$ is processed by normalization to nominal $\tau^*$ in this model. Furtherly, an error model is designed in Eq. (16). To drive model training, the initial weights $\Omega^0$ are randomly generated within the range of 0 to 1, and the loss function based on output error is designed in Eq. (17).

$$e_r\left(\Omega\right) = \tau^*\left(r\right) - \tau_p{}^*\left(\Omega, x_0^1\left(r\right), x_0^2\left(r\right), x_0^3\left(r\right)\right) \tag{16}$$

$$L\left(\Omega\right) = \frac{1}{N_{tr}} \sum_{r=1}^{N_{tr}} \left[e_r\left(\Omega\right)\right]^2 \tag{17}$$

where $e_r(\Omega)$ is the error between the actual value $\tau^*$ and the estimated value $\tau_p^*$ of the $r^{th}$ sample, $L(\Omega)$ is the loss function, which uses the mean square error.

Using Levenberg-Marquardt algorithm [24] for backpropagation to optimize loss problem, the optimal solution $\Omega^{opt}$ can be calculated as Eq. (18) shows. There are some different optimization algorithms such as gradient descent, Newton-Euler, and LM which are shown in Eq. (19), Eq. (20), and Eq. (22) with the weights updating condition $L(\Omega^{k+1}) < L(\Omega^k)$.

$$\Omega^{opt} = argmin\{L\left(\Omega\right)\} \tag{18}$$

$$\Omega^{k+1} = \Omega^k - pg\left(\Omega^k\right) \tag{19}$$

$$\Omega^{k+1} = \Omega^k - H(\Omega^k)^{-1} g\left(\Omega^k\right) \tag{20}$$

$$H\left(\Omega^k\right) = \begin{bmatrix} \frac{\partial^2 L\left(\Omega^k\right)}{\partial(\Omega_1^k)^2} & \frac{\partial^2 L\left(\Omega^k\right)}{\partial \Omega_1^k \partial \Omega_2^k} & \cdots & \frac{\partial^2 L\left(\Omega^k\right)}{\partial \Omega_1^k \partial \Omega_n^k} \\ \frac{\partial^2 L\left(\Omega^k\right)}{\partial \Omega_2^k \partial \Omega_1^k} & \frac{\partial^2 L\left(\Omega^k\right)}{\partial(\Omega_2^k)^2} & \cdots & \frac{\partial^2 L\left(\Omega^k\right)}{\partial \Omega_2^k \partial \Omega_n^k} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 L\left(\Omega^k\right)}{\partial \Omega_n^k \partial \Omega_1^k} & \frac{\partial^2 L\left(\Omega^k\right)}{\partial \Omega_n^k \partial \Omega_2^k} & \cdots & \frac{\partial^2 L\left(\Omega^k\right)}{\partial(\Omega_n^k)^2} \end{bmatrix} \tag{21}$$

https://doi.org/10.1017/S0263574725000086 Published online by Cambridge University Press

where $\Omega^{k+1}$ is the specific value of the parameters in the neural network after the iteration $(k+1)^{th}$, $g(\Omega^k)$ is the search direction of $L(\Omega^k)$ at point $\Omega^k$, $p$ is the step size, $H(\Omega^k)$ is the Hessian matrix of $L(\Omega^k)$ at point $\Omega^k$.

Reasonable step size $p$ is vital for nonlinear problem convergence. Generally, a big $p$ could hinder convergence and vice versa [25]. For backpropagation, there are Newton's method and the LM algorithm. Newton's method improves convergence speed by using the Hessian matrix of the loss function to determine the step size. However, the calculation of the Hessian matrix requires computing the second derivative of the loss function, which incurs substantial computational costs. However, the Hessian matrix might not be positive definite, which could cause the algorithm's instability. Therefore, Newton's method is not typically used as the neural network backpropagation algorithm. The LM algorithm combines the advantages of gradient descent and Newton's method, improving convergence speed and ensuring stability. Its core idea is to adjust the step size by approximating the Hessian matrix, which makes the LM algorithm have better search performance and quickly converge to the optimal solution. Therefore, this paper chooses the LM algorithm for neural network backpropagation as Eq. (22) to Eq. (25) show.

$$\Omega^{k+1} = \Omega^k - \left[J^T\left(\Omega^k\right) J\left(\Omega^k\right) + \mu U\right]^{-1}\left[J^T\left(\Omega^k\right) e\left(\Omega^k\right)\right] \tag{22}$$

$$J\left(\Omega^k\right) = \begin{bmatrix} \frac{\partial e_1\left(\Omega^k\right)}{\partial \Omega_1^k} & \frac{\partial e_1\left(\Omega^k\right)}{\partial \Omega_2^k} & \cdots & \frac{\partial e_1\left(\Omega^k\right)}{\partial \Omega_n^k} \\ \frac{\partial e_2\left(\Omega^k\right)}{\partial \Omega_1^k} & \frac{\partial e_2\left(\Omega^k\right)}{\partial \Omega_2^k} & \cdots & \frac{\partial e_2\left(\Omega^k\right)}{\partial \Omega_n^k} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial e_{N_{tr}}\left(\Omega^k\right)}{\partial \Omega_1^k} & \frac{\partial e_{N_{tr}}\left(\Omega^k\right)}{\partial \Omega_2^k} & \cdots & \frac{\partial e_{N_{tr}}\left(\Omega^k\right)}{\partial \Omega_n^k} \end{bmatrix} \tag{23}$$

$$\mu = \mu \cdot \beta \tag{24}$$

$$\mu = \frac{\mu}{\beta} \tag{25}$$

where $J(\Omega^k)$ is the Jacobian matrix, $e(\Omega^k)$ is an $n$-dimensional column vector composed of whole $e_r(\Omega^k)$, $\mu$ (MU) is a damping factor, which is divided by the constant $\beta$ when the performance of the neural network increases, otherwise $\mu$ will be multiplied by $\beta$, $U$ is an identity matrix, adding $\mu U$ is to ensure that the matrix is positively definite.

Neural networks use forward propagation and backward propagation to adjust their parameters. In practical training, it is essential to establish stopping criteria by consideration of iterations, efficiency, synchronicity, overfitting, and so on. In engineering applications, there are five criteria as follows.

1. $k \geq k_{target}$
2. $L(\Omega^k) \leq L_{target}$
3. $|g(\Omega^k)| \leq g_{target}$
4. $\xi \geq \xi_{target}$
5. $\mu \geq \mu_{target}$

where $k$ is the iteration, setting the maximum iteration $k_{target}$ to prevent training from continuing indefinitely. The neural network training approaches the optimum when $L(\Omega^k)$ achieves sufficiently small, the gradient magnitude $|g(\Omega^k)|$ achieves sufficiently small, or $\mu$ achieves sufficiently large. Additionally, $\xi$ (VAL CHECK) is the number of consecutive increases in validation set error, which is used for preventing overfitting for the neural network. When $\xi$ continues to rise $\xi_{target}$ times, it indicates a decline in the performance of the neural network, so the training should be stopped.

https://doi.org/10.1017/S0263574725000086 Published online by Cambridge University Press

***Table I.*** *Characteristics of experimental platform.*

| Devices | Characteristics | Descriptions |
|---|---|---|
| Joint module | – | Model: TE Connectivity RJSII14 |
| | Rate velocity $(^\circ/s)$ | 178.2 |
| | Rate torque $(N \cdot m)$ | 30.3 |
| Torque sensors | – | Model: NCTE FC-S2000 |
| | Sensitivity $(mV/N \cdot m)$ | 26.38 |
| | Range $(V)$ | 0.5∼4.5 (2.5 $V$ equals to approx. 0 $N \cdot m$) |
| Analog input terminal | – | Model: EL3064 |
| | Signal voltage $(V)$ | 0∼10 |
| Powder brake | – | Model: PB-10 |
| Powder brake's Tension Controller | – | Model: KTC800A |
| | Out voltage DC $(V)$ | 0∼24 |

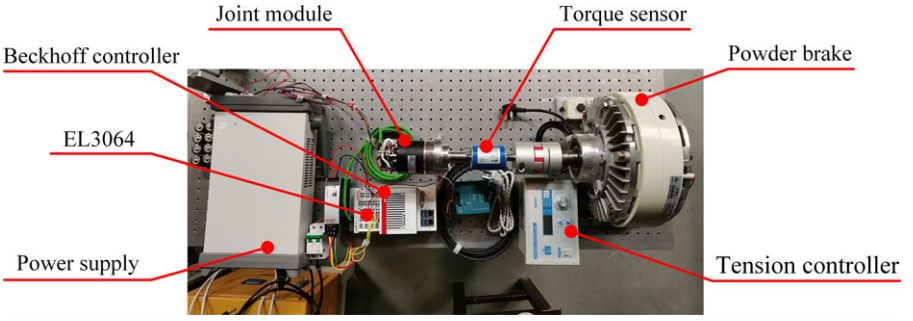



***Figure 7.*** *Experiment platform.*

## 4. Experiment

### *4.1. Experimental platform*

A joint's external torque testing platform is furnished as shown in Figure 7 to compare the physics-based model and physic-free model, which mainly consists of three parts: joint module, torque sensor, and powder brake. The joint module consists of a motor, harmonic reducer, and hardware to drive the actuator motion and monitor state variables, and the powder brake supplies a stable payload to simulate actual external torque, and the torque sensor records the value of different real-time external torque. Specifically, the experimental platform consists of six devices and its information as shown in Table I, in which there is a joint module (integrates a harmonic reducer, frameless torque motor, incremental encoder, and servo driver), Beckhoff controller and EL3064 (drives the actuator and receives the external torque information from the sensor), NCTE torque sensors, the powder brake and its tension controller (supplies the stable external torque) and power supply.

### *4.2. Experimental principle*

Figure 8 represents the experimental principle: the PC sends control commands following the excitation trajectory to the Beckhoff controller to drive the joint module motion. Then powder brake will work to generate different external torques by changing its control voltage signal. Meanwhile, the corresponding external torque value can be measured by the torque sensor, which is the analog voltage signal that can be converted into a digital signal through EL3064 and then transmitted into an actual physical quantity by Eq. (26). The joint data $I, \dot{q}, \ddot{q}$ in Beckhoff controller and external torque $\tau_e$ are collected at a sampling frequency of 1000 $Hz$ into the PC simultaneously. Subsequently, design a second-order low-pass Butterworth filter [26] with a cutoff frequency of 10 $Hz$, whose frequency response characteristics in

https://doi.org/10.1017/S0263574725000086 Published online by Cambridge University Press

***Table II.*** *Excitation trajectory parameters.*

| **Excitation trajectory** | **a** | $b_1$ | $c_1$ | $b_2$ | $c_2$ | $b_3$ | $c_3$ |
|---|---|---|---|---|---|---|---|
| Identification trajectory | 0.5583 | 0.1549 | 0.2530 | 0.3537 | 0.4531 | −0.5087 | −0.3864 |
| Validation trajectory | 0.2760 | 0.0475 | 0.1179 | 0.1894 | 0.2471 | −0.2369 | −0.2040 |

***Table III.*** *Identified dynamic parameters of external torque estimation based on physical model.*

| **Identified dynamic parameters** | $I_{zz}$ | $f_c$ | $f_v$ |
|---|---|---|---|
| Value | 0.1738 | 1.8869 | 2.9286 |

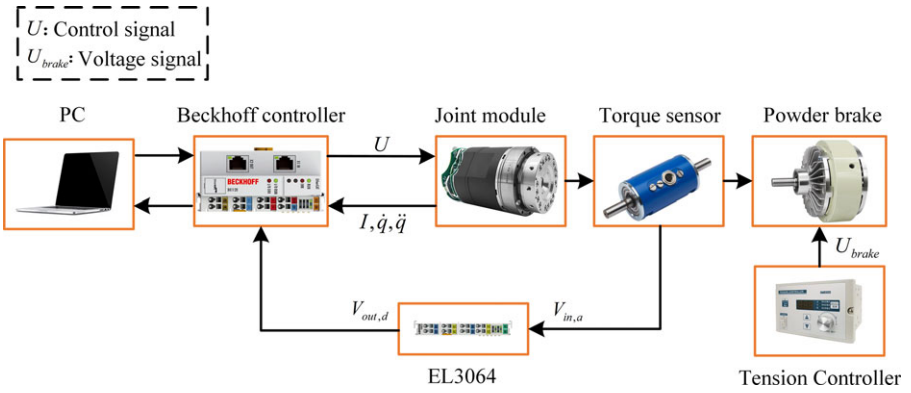



***Figure 8.*** *Experimental schematic diagram.*

the z-domain can be represented by the transfer function in Eq. (27), where the coefficients are obtained using the MATLAB "butter" function [27]. Then, apply this transfer function to filter those data both in forwards and backwards procedure. The filtered data can be further used for experimental result analysis.

$$\tau_e = \frac{\left(\frac{2.5}{2^{13}} * V_{out,d} - 2.5\right) * 1000}{26.38} \tag{26}$$

$$H(z) = \frac{9.4469 \times 10^{-4} + 0.0019z^{-1} + 9.4469 \times 10^{-4}z^{-2}}{1 - 1.9112z^{-1} + 0.9150z^{-2}} \tag{27}$$

## 5. Experimental results

### *5.1. Experimental results of parameter identification*

Before comparing the physics-based model and the physic-free model of torque estimation, the physical model method as well as physics-based model should be tested to show its performance. Hence, excitation trajectories are designed to identify the parameters by the MOB, and the coefficients and position figure of the identification trajectory and the validation trajectory are shown in Table II and Figure 9, respectively, which are with well-conditioned condition numbers and joint velocity limits $\dot{q}_{max}$ of $\frac{\pi}{3}/s$ and $\frac{\pi}{6}/s$ according to the Eq. (7) and Eq. (8).

Using the identification trajectory, and collecting data sets, the identified dynamic parameters can be calculated by Eq. (6) and the results are shown in Table III.

Then these identified dynamic parameters can construct the model and use the identification trajectory and validation trajectory to test its accuracy with the Eq. (2). The results are shown in Figure 10. From Figure 10, it's obvious that the estimation accuracy is better in the identification trajectory

https://doi.org/10.1017/S0263574725000086 Published online by Cambridge University Press

***Table IV.*** *The error evaluation of different trajectories by identified dynamic model.*

| Different Trajectories | RMSE (N · m) | R-squared |
|---|---|---|
| Identification trajectory | 0.3675 | 0.9900 |
| Validation trajectory | 0. 5064 | 0.9722 |

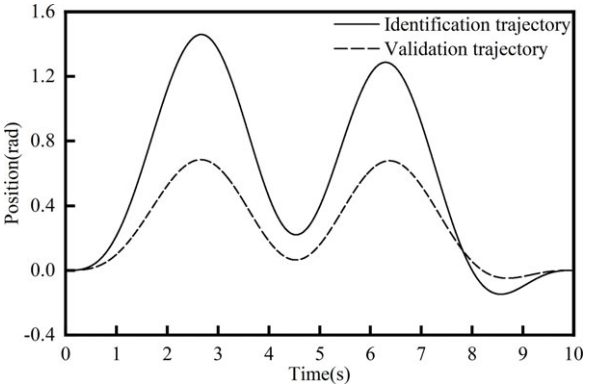



***Figure 9.*** *Excitation trajectory with the parameters in Table II.*

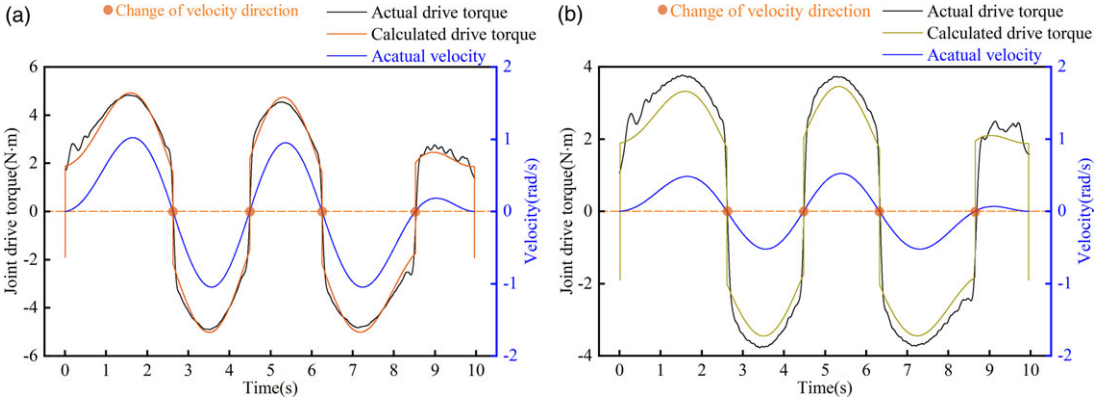



***Figure 10.*** *Joint driving torque estimation in different excitation trajectory. (a) Identification trajectory and (b) validation trajectory.*

compared to the validation trajectory. The reason is that the parameter identification is based on the identification trajectory. Moreover, the fitting procedure shows that drive torque estimation by the identified dynamic model has a bad generalization ability, proving that some lightweight robots' drag-and-drop teaching sometimes behaves badly when the operators drag different joints with different forces.

Furthermore, the errors in different excitation trajectories are compared in Figure 11. The results show that the huge impulse errors always exist around the change of velocity direction, in which the velocity is around zero, as well as around the orange point zone in Figure 11. Table IV shows different evaluation way with the RSME and R-squared, which reveals that the identified dynamic model in identification trajectory has better performance with smaller RSME value and larger R-squared value.

### *5.2. Comparison of external torque estimation with two model*

To improve the model generalization ability by the BPNN method, it is crucial to establish a diverse training set, collected by many different scenarios and variations. In our experiments, there are six control

https://doi.org/10.1017/S0263574725000086 Published online by Cambridge University Press

***Table V.*** *The parameters of external torque estimation based on physic-free model.*

| **Item** | $N_{in}$ | $N_{out}$ | $M$ | $N_1$ | $N_2$ | $k_{target}$ | $L_{target}$ | $g_{target}$ | $\mu_{target}$ |
|---|---|---|---|---|---|---|---|---|---|
| – | 3 | 1 | 2 | 5 | 8 | 1000 | $10^{-5}$ | $10^{-7}$ | $10^{10}$ |
| Item | $\mu_{initial}$ | $\beta$ | $\xi_{target}$ | $f_1(x)$ | | $f_2(x)$ | | $f_3(x)$ | |
| – | $10^{-3}$ | 10 | 6 | $\frac{2}{1+e^{-2x}}-1$ | | $\frac{2}{1+e^{-2x}}-1$ | | $x$ | |

***Table VI.*** *The error evaluation by MOB model and BPNN model with the same identification trajectory.*

| **Model** | **RMSE ($N \cdot m$)** | **R-squared** |
|---|---|---|
| MOB | 2.0232 | 0.9509 |
| BPNN | 0.3707 | 0.9983 |

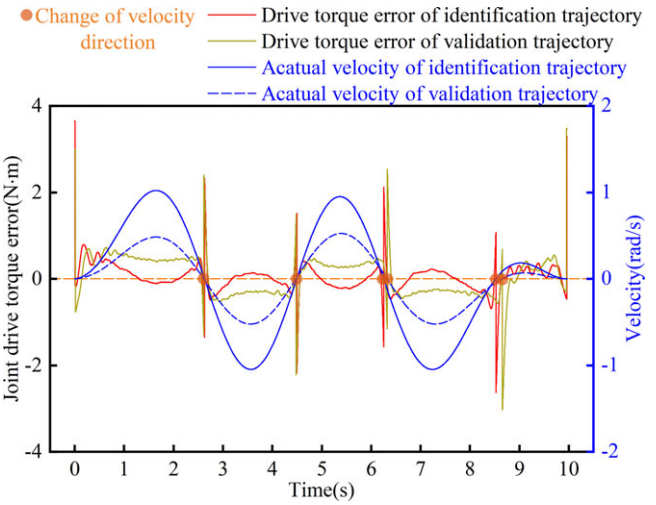



***Figure 11.*** *Estimation errors of different trajectory by identified dynamic model.*

groups with different voltages ($1V$, $2V$, $3V$, $4V$, $5V$, $6V$) of powder brake, corresponding to different external torque for the joint module from small to large. Then select the identification trajectory in Figure 9 as the excitation trajectory and use the control groups with $1V$, $2V$, $3V5V$, and $6V$ as training sets with $4V$ as a testing set to build up the BPNN model. Through the training, the model parameters are shown in Table V.

Compare physics-based model (MOB model) and physic-free model (BPNN model) with the identification trajectory with some payload ($4V$ of powder brake) to verify the accuracy. In this MOB model, the proportion coefficient $k_p$ is 10. The estimated torque by different model is shown in Figure 12, in which the BPNN model has better fitting ability compared to the MOB model.

Furthermore, the error of different model is shown in Figure 13, which also shows that the huge impulse errors are around the change of velocity direction (orange zone in Figure 13). Table VI shows different evaluation way with the RSME and R-squared, which reveals that the BPNN model has better performance with smaller RSME value and larger R-squared value.

## 6. Conclusion and discussion

From the experiment's results, it's obvious that the BPNN model has better generalization ability compared to the MOB model. Meanwhile, the BPNN model is more accurate in estimating external torque.

https://doi.org/10.1017/S0263574725000086 Published online by Cambridge University Press

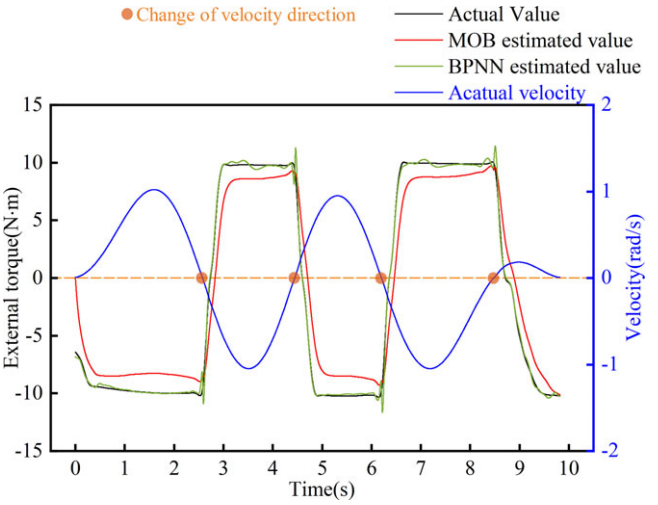



***Figure 12.*** *Estimated torque by MOB model and BPNN model with same identification trajectory.*

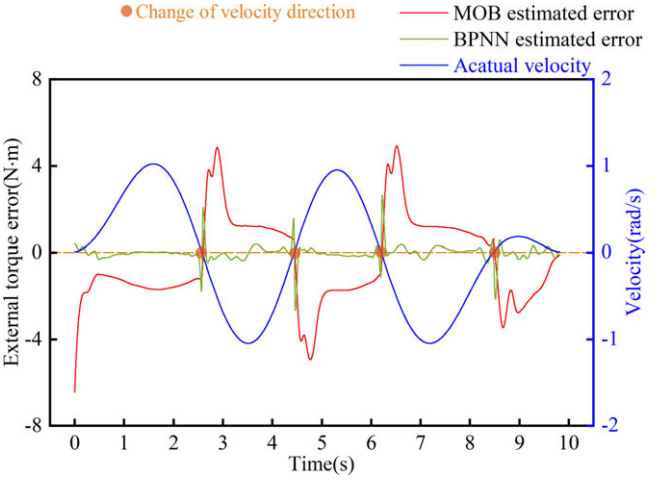



***Figure 13.*** *Estimated torque error by MOB model and BPNN model with same identification trajectory.*

From this study, the reasons for that are origin from the parameter identification errors in the dynamic model, the vibration of the first-order MOB's position derivative, the MOB model's overfitting in identification trajectory (in other words, the worse generalization ability of the MOB model in validation trajectory) and so on. However, this experiment also has some shortcomings at the beginning of joint module motion or its velocity is nearly zero, it's hard to distinguish whether the joint is stopping or moving at a very small velocity. If in the stop state, the dynamics of the robot are invalidated because the torque is equal to the payload. Hence the MOB model cannot be used for estimating. Besides, in the very short duration time between stops to an obvious motion of the joint module (usually 0.1 s in this experiment every single time), friction shows obvious nonlinear characteristics compared to the motion state. However, this period is short and happens rarely so the training physic-free model or BPNN model in this period behaves badly for lacking sufficient data. Hence, the BPNN model also has some impulse errors around the change of velocity directions. Future work will focus on the distinguishment of torque under the robot's different states so that a more delicate model can be built to calculate the external torque.

Generally, this paper introduces an external torque estimation algorithm based on the BPNN to reduce the estimation error, which is a physic-free model without torque sensors, as well as an economical way.

https://doi.org/10.1017/S0263574725000086 Published online by Cambridge University Press

This method avoids uncertainties by the physical model, which encompasses parameters identification error and joint friction, and enhances the accuracy of the estimation. Compared to the classical physical dynamics, the first-order MOB, the BPNN model for external torque estimation also improves the whole accuracy, mostly especially near the velocity direction change areas. The external torque estimation based on BPNN will be further studied in the more-axis robot to solve much more collaborative scenarios.

## 7. Future work

In the future, the research will focus on the adaptability of the BPNN model for lightweight robotic systems and the further impact of nonlinear friction on its performance especially in the same series of joint modules. Additionally, it's vital to improve the precision of the BPNN model sustainably in estimating joint external torque, especially the influence of nonlinear friction at a very low joint velocity, which enhances the model's performance across a range of motion states.

**Author contributions.** Tao Zhang and Huapeng Wu conceived and designed the study. Haochong Li and Yongping Shi conducted data gathering. Xuanchen Zhang and Jun Zhang performed statistical analyses. Tao Zhang and Haochong Li wrote the article.

**Financial support.** This project is funded by the National Natural Science Foundation of China (No. 12105070), the Anhui Provincial Key Research and Development Project (No. 2022e07020027), and the Fundamental Research Funds for the Central Universities (NO. JZ2024HGTB0221). Thanks to all members who helped with this research.

**Competing interests.** The authors declare no competing interests exist.

**Ethical approval.** None

## References

[1] M. Metzner, S. Leurer, A. Handwerker, E. Karlidag, A. Blank, F. Hefner and J. Franke, "High-precision assembly of electronic devices with lightweight robots through sensor-guided insertion," *Proc. CIRP* **97**, 337–341 (2021).
[2] S. Husmann, S. Stemmler, S. Hähnel, S. Vogelgesang, D. Abel and T. Bergs, "Model predictive force control in grinding based on a lightweight robot," *IFAC-PapersOnLine* **52**(13), 1779–1784 (2019).
[3] Y. Liu, X. Li, A. Zhu, Z. Zheng and H. Zhu, "Design and evaluation of a surface electromyography-controlled lightweight upper arm exoskeleton rehabilitation robot," *Int. J. Adv. Robot. Syst.* **18**(3) (2021).
[4] W. Li, K. Liu, C. Li, Z. Sun, S. Liu and J. Gu, "Development and evaluation of a wearable lower limb rehabilitation robot," *J. Bionic Eng.* **19**(3), 688–699 (2022).
[5] M. Y. Cao, S. Laws and F. R.y Baena, "Six-axis force/torque sensors for robotics applications: A review," *IEEE Sensors J.* **21**(24), 27238–27251 (2021).
[6] U. Kim, D.-H. Lee, Y. B. Kim, D.-Y. Seok and H. R. Choi, "A novel six-axis force/torque sensor for robotic applications," *IEEE-ASME Trans. Mech.* **22**(3), 1381–1391 (2017).
[7] Y. B. Kim, D.-Y. Seok, S. Y. Lee, J. Kim, G. Kang, U. Kim and H. R. Choi, "6-axis force/torque sensor with a novel autonomous weight compensating capability for robotic applications," *IEEE Robot. Autom. Lett.* **5**(4), 6686–6693 (2020).
[8] Z. Wang, L. Liu, W. Yan, J. He, B. Cui and Z. Li, "Research on collision point identification based on six-axis force/torque sensor," *J. Sensors* **2020**, 1–13 (2020).
[9] G. Cheng, E. Dean-Leon, F. Bergner, J. R. G. Olvera, Q. Leboutet and P. Mittendorfer, "A comprehensive realization of robot skin: Sensors, sensing, control, and applications," *Proc. IEEE* **107**(10), 2034–2051 (2019).
[10] Z. Ye, G. Pang, K. Xu, Z. Hou, H. Lv, Y. Shen and G. Yang, "Soft robot skin with conformal adaptability for on-body tactile perception of collaborative robots," *IEEE Robot. Autom. Lett.* **7**(2), 5127–5134 (2022).
[11] C. Zhong, S. Zhao, Y. Liu, Z. Li, Z. Kan and Y. Feng, "A flexible wearable e-skin sensing system for robotic teleoperation," *Robotica* **41**(3), 1025–1038 (2023).
[12] L. Han, W. Xu, B. Li and P. Kang, "Collision detection and coordinated compliance control for a dual-arm robot without force/torque sensing based on momentum observer," *IEEE-ASME Trans. Mech.* **24**(5), 2261–2272 (2019).
[13] A. De Luca and R. Mattone. Sensorless Robot Collision Detection and Hybrid Force/Motion Control. **In:** *Proceedings of the 2005 IEEE International Conference on Robotics and Automation* (IEEE, 2005), pp. 999–1004.
[14] S.-D. Lee and J.-B. Song, "Sensorless collision detection based on friction model for a robot manipulator," *Int. J. Precis. Eng. Manuf.* **17**(1), 11–17 (2016).
[15] N. Briquet-Kerestedjian, M. Makarov, P. Rodriguez-Ayerbe and M. Grossard, Quantifying the uncertainties-induced errors in robot impact detection methods (2016).

https://doi.org/10.1017/S0263574725000086 Published online by Cambridge University Press

[16] K. M. Park, J. Kim, J. Park and F. C. Park, "Learning-based real-time detection of robot collisions without joint torque sensors," *IEEE Robot. Autom. Lett.* **6**(1), 103–110 (2021).
[17] A. Parvaresh and S. A. A. Moosavian, "Dynamics and path tracking of continuum robotic arms using data-driven identification tools," *Robotica* **40**(4), 1098–1124 (2022).
[18] S. Haddadin, A. De Luca and A. Albu-Schaeffer, "Robot collisions: A survey on detection, isolation, and identification," *IEEE Trans. Robot.* **33**(6), 1292–1312 (2017).
[19] G. Garofalo, N. Mansfeld, J. Jankowski and C. Ott, Sliding mode momentum observers for estimation of external torques and joint acceleration (2019).
[20] K. Liu, C. Wei, D. Ni, Z. Lu, Z. Zhang and Y. Li, Stribeck friction model identification based on genetic algorithm (2022).
[21] J. Swevers, C. Ganseman, D. B. Tukel, J. d. Schutter and H. V. Brussel, "Optimal robot excitation and identification," *IEEE Trans. Robot. Autom.* **13**(5), 730–740 (1997).
[22] K.-J. Park, "Fourier-based optimal excitation trajectories for the dynamic identification of robots," *Robotica* **24**(5), 625–633 (2006).
[23] S. Karsoliya, "Approximating number of hidden layer neurons in multiple hidden layer BPNN architecture," *Int. J. Eng. Trends Technol.* **3**(6), 714–717 (2012).
[24] M. T. Hagan and M. B. Menhaj, "Training feedforward networks with the Marquardt algorithm," *IEEE Trans. Neural Netw.* **5**(6), 989–993 (1994).
[25] M. T. Hagan, H. B. Demuth and M. Beale, *Neural Network Design* (PWS Publishing Co, 1997).
[26] J. G. Proakis, "Digital Signal Processing: Principles," **In:** *Algorithms, and Applications*. vol. **4** (Pearson Education India, 2007).
[27] MathWorks. MathWorks Documentation [EB/OL]. [2024-11-08]. https://ww2.mathworks.cn/help/dsp/ref/butter.html?searchHighlight=butter&s_tid=srchtitle_support_results_2_butter.

## Appendix A. Momentum observer of physical dynamic model

MOB is a classic algorithm used for external torque estimation in robotics. The external torque can be calculated by inverse dynamics described in Eq. (1), but the inaccurate acceleration term of the dynamic model, which is the real-time second derivative of the joint position causing the large noise and delay, can lead to the wrong estimates. The first-order MOB as shown in Figure A1 only requires the velocity, a first derivative of the joint position, and drive torque of the joint module as inputs to compute the joint external torque. The generalized momentum of the robot is defined in Eq. (3).

$$M(q)\,\ddot{q} + C(q,\dot{q})\,\dot{q} + G(q) = \tau + \tau_e - \tau_f \tag{A1}$$

$$\tau_f = f_c \mathrm{sgn}(\dot{q}) + f_v \dot{q} \tag{A2}$$

where $M(q)$, $C(q,\dot{q})$, and $G(q)$ are represent the inertia matrix, the Coriolis matrix, and the gravity matrix, respectively, $q$, $\dot{q}$, and $\ddot{q}$ are represent the joint position, joint velocity, and joint acceleration, respectively, $\tau_e$ is the external torque applied at the end of the joint, $\tau$ is the drive torque of joint module, $\tau_f$ is the friction torque, described by the classical Coulomb-viscous friction model, as shown in Eq. (2), and $f_c$ and $f_v$ represent the coefficient of Coulomb friction and viscous friction, respectively.

$$p = M(q)\,\dot{q} \tag{A3}$$

In the robot dynamics model, $M(q)$ is a positive definite symmetric matrix, and the matrix $\dot{M}(q) - 2C(q,\dot{q})$ is a skew-symmetric matrix. This can be expressed mathematically as follows

$$\dot{M}(q) = C(q,\dot{q}) + C^T(q,\dot{q}) \tag{A4}$$

The time derivative of Eq. (1) can be obtained using Eq. (1) and Eq. (4) as follows

$$\dot{p} = \dot{M}(q,\dot{q})\,\dot{q} + M(q,\dot{q})\,\ddot{q} = \tau + \tau_e - \tau_f \tag{A5}$$

Define the residual term $r$ to detect external torque.

$$\hat{\tau}_e = r \in \mathrm{R}^n, \dot{r} = 0 \tag{A6}$$

$$\dot{\hat{p}} = \tau + r - \tau_f \tag{A7}$$

https://doi.org/10.1017/S0263574725000086 Published online by Cambridge University Press

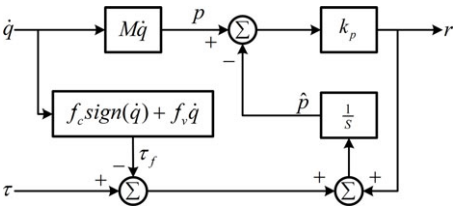



***Figure A1.*** *First-order MOB model.*

$$r = k_p \left[ p(t) - \int_0^t \left( \tau + r - \tau_f \right) dt - p(0) \right] \tag{A8}$$

where $p(0)$ and $p(t)$ are the initial value and final value of the generalized momentum, respectively, and $k_p$ is a proportion coefficient that is always greater than 0. Substitute Eq. (5) into Eq. (8) and differentiate both sides concerning time.

$$\dot{r} = k_p(\tau_e - r) \tag{A9}$$

Apply the Laplace transform to Eq. (9) results in

$$\tau_e(s) = \frac{s + k_p}{k_p} r(s) \tag{A10}$$

It is evident that $k_p$ is sufficiently large and $r$ serves as a qualified observation of the external torque $\tau_e$.

**Cite this article:** T. Zhang, H. Li, Y. Shi, L. Wang, X. Zhang, J. Zhang and H. Wu (2025). "A novel joint external torque estimate model of the lightweight robot's joint based on a BP neural network", Robotica **43**, 1027–1042. https://doi.org/10.1017/S0263574725000086

https://doi.org/10.1017/S0263574725000086 Published online by Cambridge University Press