machines

MDPI

Article

MomentumNet-CD: Real-Time Collision Detection for Industrial Robots Based on Momentum Observer with Optimized BP Neural Network

Jinhua Ye [1], Yechen Fan [1], Quanjie Kang [1], Xiaohan Liu [1], Haibin Wu [1] and Gengfeng Zheng [2,*]

[1] School of Mechanical Engineering and Automation, Fuzhou University, Fuzhou 350116, China; yejinhua@fzu.edu.cn (J.Y.); 240227014@fzu.edu.cn (Y.F.); 13053496383@163.com (Q.K.); 240227050@fzu.edu.cn (X.L.); wuhb@fzu.edu.cn (H.W.)

[2] Fujian Key Laboratory of Special Intelligent Equipment Safety Measurement and Control, Fujian Special Equipment Inspection and Research Institute, Fuzhou 350008, China

* Correspondence: 18862889747@163.com; Tel.: +86-130-6733-1280

Abstract: The accurate detection and identification of collision states in industrial robot environments is a critically important and challenging task. Deep learning-based methods have been widely applied to collision detection; however, these methods primarily rely on dynamic models and dynamic threshold settings, which are subject to modeling errors and threshold adjustment latency. To address this issue, we propose MomentumNet-CD, a novel collision detection method for industrial robots that leverages backpropagation (BP) neural networks. MomentumNet-CD extracts collision state features through a momentum observer and constructs an observation model using Mahalanobis distance. These features are then processed by an optimized three-layer BP neural network for accurate collision identification. The network is trained using a modified Levenberg–Marquardt algorithm by introducing regularization terms and continuous probability outputs. Furthermore, we developed a comprehensive acquisition system based on the Q8-USB data acquisition card and the QUARC 2.7 real-time control environment. The system integrates key hardware components including a MR-J2S-70A servo driver, ATI six-dimensional force/torque (F/T) sensor, and ISO-U2-P1-F8 isolation transmitter, and the corresponding software module is developed through MATLAB/Simulink R2022b, which achieves the high-frequency real-time acquisition of critical robot joint states. The experimental results show that the MomentumNet-CD method achieves an overall accuracy of 93.65% under five different speed conditions, and the detection delay is only 12.16 ms. Compared with the existing methods, the method shows obvious advantages in terms of the accuracy and response speed of collision detection.

Keywords: industrial robot; momentum observer; Mahalanobis distance; BP neural network; Levenberg–Marquardt algorithm

check for updates

Academic Editor: Dan Zhang

Received: 28 February 2025
Revised: 10 April 2025
Accepted: 15 April 2025
Published: 18 April 2025

Citation: Ye, J.; Fan, Y.; Kang, Q.; Liu, X.; Wu, H.; Zheng, G. MomentumNet-CD: Real-Time Collision Detection for Industrial Robots Based on Momentum Observer with Optimized BP Neural Network. *Machines* **2025**, *13*, 334. https://doi.org/10.3390/machines13040334

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

As industrial automation advances and artificial intelligence technology rapidly develops, the application scenario of industrial robots is undergoing a revolutionary transformation. Industrial robots have evolved from performing simple repetitive tasks to engaging in intelligent and collaborative operations, creating increasing demand for human-robot collaboration in shared workspaces [1]. In this context, robot safety has emerged as a critical performance indicator, in which the research and application of collision detection technology is particularly important [2]. As a core part of the robot safety protection

system, collision detection is not only the last line of defense to protect the operator's safety in case of failure of preventive collision avoidance measures, but also provides an important technical support for subsequent collision inhibition and response control strategies [3]. Advancing collision detection technology offers significant practical benefits and strategic value for improving the safety performance of industrial robots in human–robot collaboration and expanding their application scope [4,5].

1.1. Related Work

Current industrial robot collision detection technologies fall into two main categories: model-free detection methods and model-based detection methods [6–10]. Model-free detection methods distinctively operate without reliance on robot dynamics models and realize the detection function by directly collecting and analyzing the collision-related signals. In this field, scholars have put forward a variety of innovative technical solutions. Yun et al. [11] developed an integrated columnar three-axis strain sensor that accurately measures external forces while maintaining structural rigidity. Their calibration experiments confirmed the stability of this collision detection approach. Wu et al. [12] advanced flexible haptic sensing by developing a sensor that accurately locates collision positions while providing collision force buffering. Min et al. [13] proposed a detection method based on vibration modal features of collision signals, which can show excellent detection accuracy in robot body testing by extracting the intrinsic frequency and vibration modal features.

Model-based collision detection methods offer greater generalizability across different robot systems [14–16]. By constructing a robot kinematics and dynamics model, the method is able to calculate key parameters, such as the momentum and energy of each joint linkage of the robot. Li et al. [17] proposed a new collision detection method based on a robot dynamics model, which improves the sensitivity and reliability of collision detection for collaborative robots by measuring the F/T at the base and avoiding the complex joint friction of the modeled joints. Zhang et al. [18] proposed a collision risk detection model utilizing acceleration discriminant factors to improve detection accuracy and motion efficiency in complex environments.

In the study of obtaining joint residual torques based on robot dynamics modeling, the traditional method requires quadratic differentiation of the joint position information to obtain the acceleration information, but this process is often accompanied by significant noise interference, which seriously affects the estimation accuracy of the residual torques. To address this problem, Zhang et al. [19] proposed an innovative method combining the Kalman filtered external torque observer and the time-varying symmetric threshold function, which achieves an excellent performance of 52.03% improvement in external torque estimation accuracy and 58.06% reduction in detection delay. On this basis, Chen et al. [20] developed a collision detection method based on a second-order sliding mode momentum observer, which demonstrated better collision sensitivity and noise immunity than the traditional first-order momentum observer. Subsequently, Zhao et al. [21] constructed an external torque observer based on generalized momentum deviation, innovatively introduced an end-load compensation mechanism, effectively eliminated the interference of equivalent torques, and realized high-precision collision detection under sensorless conditions.

Recent advances in deep learning have created new opportunities for robot collision detection research [22–26]. Bin et al. [15] innovatively combined the cubic friction force model with a CNN-iTransformer network and proposed a high-precision collision detection method for collaborative robots, which achieved remarkable results in zero-force control and demonstrative reproduction tasks. Subsequently, Niu et al. [27] proposed a detection framework based on continuous wavelet transform–convolutional neural network (CWT-CNN), which successfully achieved a double breakthrough in the efficient use of data

and system generalization capability through systematic wavelet parameter optimization and in-depth transmissibility analysis. On the basis of this research, Park et al. [3] further explored the application scope of the detection method and successfully extended the learning detection method to the six-degree-of-freedom robotic arm system by skillfully applying a one-dimensional convolutional neural network to process the momentum observer residuals.

1.2. Motivation

Industrial robot collision detection in human–robot collaborative environments presents two significant challenges. First, despite the rapid emergence of numerous deep learning-based collision detection methods [3,15,27] in recent years, there remains a lack of accurate and generalized data acquisition systems. The study by [26] employed force sensitive resistor (FSR) sensors merely as switches to record collision timing, but this over-simplified binary method was unable to obtain the magnitude and direction of the collision force. Secondly, existing collision detection methods primarily rely on kinetic models and dynamic threshold settings [17,18], making it difficult to meet the requirements of high reliability and robustness in industrial scenarios due to modeling errors and threshold adjustment lags.

1.3. Main Contributions

- A complete data acquisition system is designed, which integrates both hardware and software. In terms of the hardware architecture, the system consists of five key components: the MR-J2S-70A servo driver (Mitsubishi Electric, Tokyo, Japan), the Q8-USB data acquisition card (Quanser, Markham, ON, Canada), the ATI six-dimensional F/T sensor(ATI Industrial Automation, Apex, NC, USA), the ISO-U2-P1-F8 isolation transmitter(Shenzhen Sunyuan Technology Co., Ltd., Shenzhen, China), and a metal-oxide semiconductor (MOS) tube-triggered switch module (Elecrow, Shenzhen, China). The software part is constructed based on MATLAB/Simulink R2022b and the QUARC 2.7 real-time control development environment, which realizes high-frequency real-time acquisition and provides a reliable hardware foundation for the subsequent neural network training and collision detection experiments.
- A novel collision detection method is proposed, which firstly obtains the collision state features through a momentum observer, followed by the construction of a robust observation model using Mahalanobis distance. Subsequently, the extracted features are fed into a three-layer BP neural network optimized by the Levenberg–Marquardt (LM) algorithm to achieve high-precision collision identification.

1.4. Organization

The rest of the paper is organized as follows: Section 2 describes the design of the data acquisition system, Section 3 describes the modeling of the collision detection system, Section 4 details the optimal implementation of the neural network detector, Section 5 shows the experimental validation results and the comparative analysis with the existing methods, and Section 6 presents the conclusions and outlines directions for future work.

2. CollisionSense DAQ System Design

In order to clearly illustrate the MomentumNet-CD methodology and the Collision-Sense DAQ System proposed in this paper, the overall technical framework is given in Figure 1.

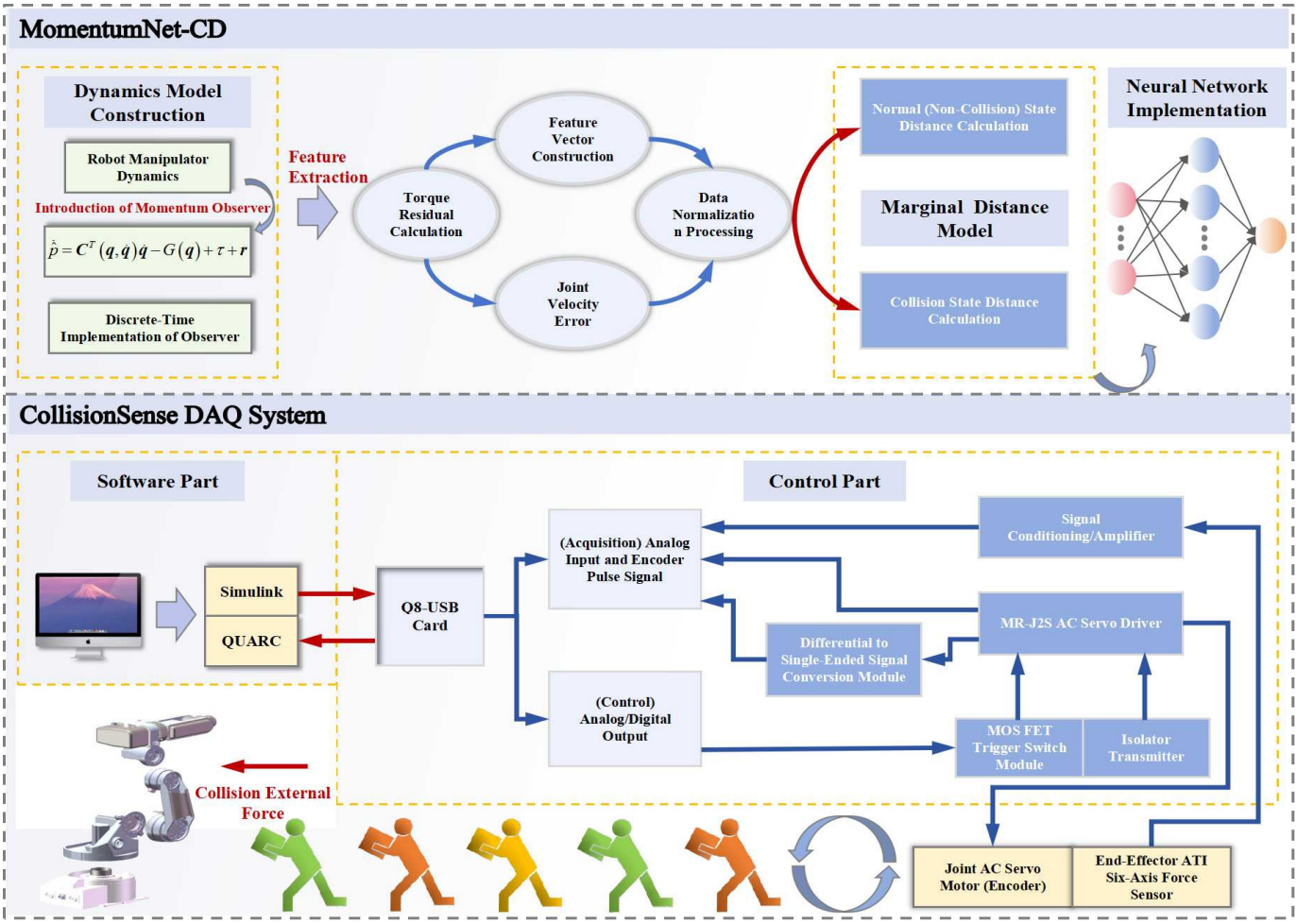


Figure 1. Overall technical framework. The framework includes the MomentumNet-CD method and the CollisionSense DAQ System, which constructs a complete detection architecture through the momentum observer, feature extraction, and Mahalanobis distance observation model, and uses the optimized BP neural network to achieve accurate collision identification. The CollisionSense DAQ System contains two major parts: hardware realization and software design. In terms of hardware, the system includes a control acquisition module. The software design is based on MATLAB/Simulink R2022b and the QUARC 2.7 componentized development environment through the proprietary block diagram module provided by it.

2.1. Hardware Integration Architecture

The hardware architecture of the data acquisition system designed in this paper consists of five key components: the MR-J2S-70A servo driver, the Q8-USB data acquisition card, the ATI six-dimensional F/T sensor, the ISO-U2-P1-F8 isolation transmitter, and the MOS tube-triggered switch module. The system integration of these components provides the hardware foundation for subsequent neural network training and collision detection experiments.

In the servo drive system, the MR-J2S-70A servo driver realizes its core function through three key connectors, as shown in Figure 2a. The CN1A and CN1B connectors are responsible for the input and output of digital signals, the CN3 communication connector handles analog information, such as joint motor current and DC pulses, and the CN2 encoder connector connects with the servomotor encoder to realize the reception of signals. For the needs of robot joint information acquisition and analysis of the change rule during collision in this study, the driver was configured in the position control mode, the resolution was set to 131,072 kpps, and the open collector method was used to input the command

pulse string, as shown in Figure 2c. In addition, the MR-J2S-70A servo driver output motor current and hysteresis pulse information in voltage form through the [MO1, LG] and [MO2, LG] analog channels of the CN3 connector and also output differential pulse signals through the [LA, LAR], [LB, LBR], and [LZ, LZR] pins of the CN1A connector based on the encoder signals calculated and processed. This was used to obtain the motor position and speed data. The specific pin positions, functions and connections are shown in Table 1 and Figure 2d,e.

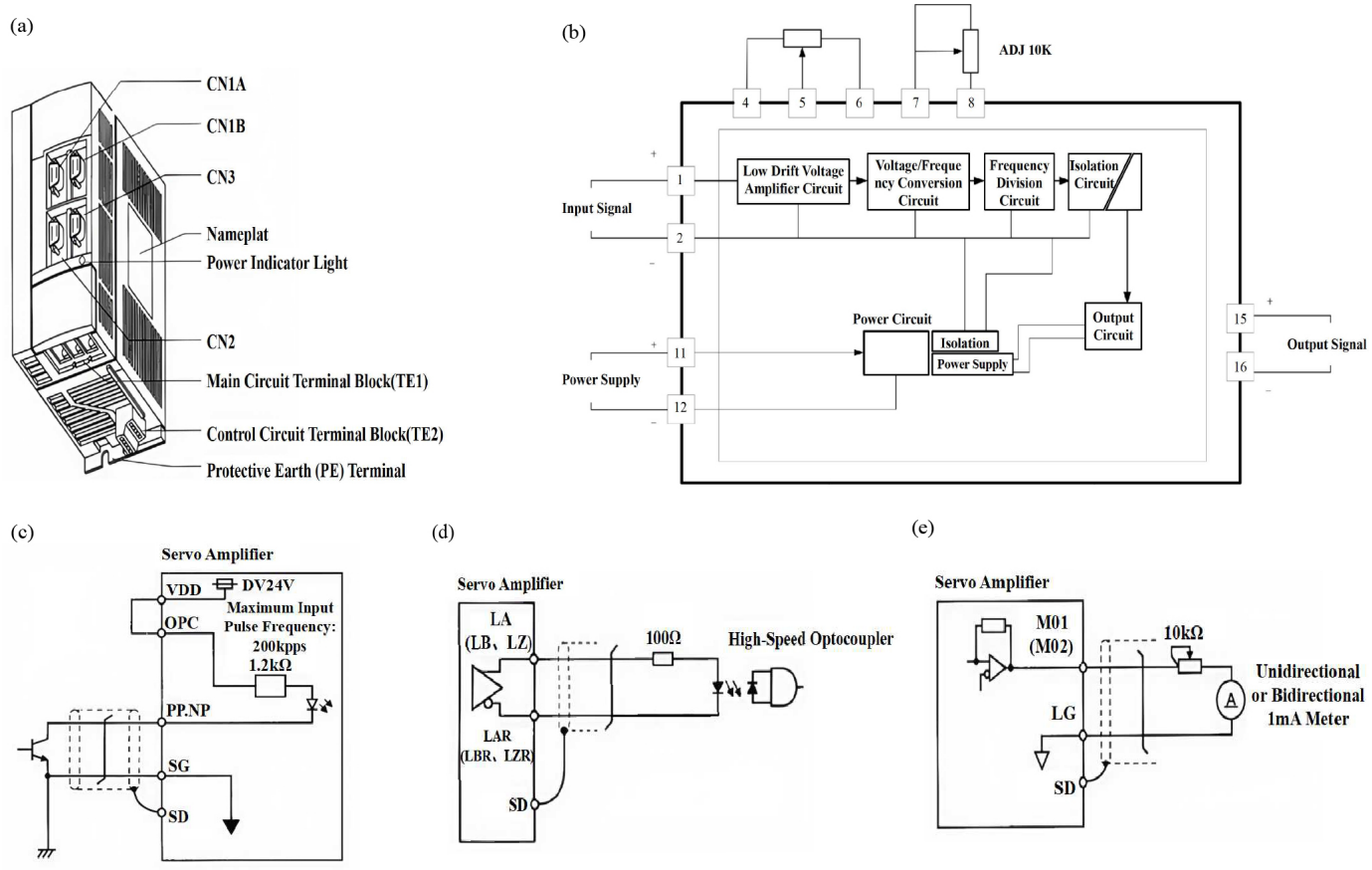


Figure 2. Data acquisition system integration. (**a**) MR-J2S-70A servo driver composition; (**b**) model ISO-U2-P1-F8 isolation transmitter schematic block diagram; (**c**) open collector input method wiring diagram; (**d**) differential drive method encoder pulse output; (**e**) analog output.

Table 1. Function and output range of each output pin of MR-J2S-70A servo driver.

Pinout	Location and Pin Number	Function	Output Range
MO1	CN3 4	Voltage output parameter No. 17 between MO1 and LG	0–8 V
MO2	CN3 14	Voltage output parameter No. 17 between MO2 and LG	0–8 V
LA/LAR	CN1A6/16	Encoder A-phase pulses (differential drive)	-
LB/LBR	CN1A7/17	Encoder B-phase pulse (differential drive)	-
LZ/LZR	CN1A5/15	Encoder Z-phase pulses (differential drive)	-
LG	CN1A 1	Common ground for output interface	-

The data acquisition core utilizes Quanser's Q8-USB card (Tables 2 and 3), which features a USB 2.0 high-speed interface with a wealth of input and output functions. The system is equipped with 8-channel digital input interfaces and 8-channel digital output interfaces that can be used as pulse width modulation (PWM) outputs. At the same time, it integrates 8-channel 16-bit configurable-range analog-to-digital converters and digital-to-

analog converters. Its multi-bit ADC analog input function can effectively collect the servo drive's joint information and six-dimensional sensor force information. The single-ended encoder function supports the collection of joint position and speed information directly at the hardware level, which has lower noise and higher accuracy than the traditional positional differentiation method. The DAC analog output function can be used as the input source of the isolation transmitter and the output source of the corresponding frequency of pulse signals to achieve precise control of the joints and motors.

Table 2. Q8-USB card usage by function.

Function	Goal	Channel Selection	Setting Range
Configurable range 16-bit analog input (ADC)	Collect the joint motor current information output by the AC servo driver and the joint motor detent pulse signal.	[0,1]	0–10 V
	Collect the voltage of the six channels of the ATI six-dimensional F/T sensor.	[2–7]	0–10 V
Configurable range 16-bit digital-to-analog converter (DAC)	The input to the analog-to-pulse frequency conversion module.	[0]	0–10 V
Digital output interface	Control the on/off switching of the MOS field-effect transistor (FET) trigger switch module.	[0]	-
Single-ended encoder input	Collect the position and speed of the joint motor.	[0]	-

Table 3. Hardware component functions and system applications.

Component	Function	Use in the System
MR-J2S-70A servo driver	Control of robot joints, processing of different signals through three connectors (CN1A/B, CN3, CN2).	Output motor current, hysteresis pulse information, and differential pulse signals to provide joint position and velocity data.
Q8-USB data acquisition card	Provides multiple inputs and outputs, including 8 channels of 16-bit ADC/DAC and encoder inputs.	Acquisition of joint information, sensor data, and control of joint motors via analog outputs.
ATI six-dimensional F/T sensor	Measurement of external forces using silicon strain technology with a safety factor of up to 4080% and a transmission rate of 28.5 KHz.	Connection to Q8-USB card via analog output to provide collision force reference data.
ISO-U2-P1-F8 isolation transmitter	Converts analog DC voltage signals into digital pulse frequency signals for triple isolation.	Receives Q8-USB card output, converts and controls robot position.
MOS tube-triggered switch module	Provides high current output (10 A, up to 15 A with heat sink) and handles PWM signals up to 2.5 kHz.	Controlled by the digital output of the Q8-USB card as a switching interface between system components.

To accurately acquire external collision force data, an ATI FT six-dimensional F/T sensor was selected for this paper. The sensor adopts advanced noise-resistant silicon strain technology with excellent durability and a safety factor of up to 4080%. A transmission rate of 28.5 KHz can be realized through the controller. The sensor supports a variety of output interfaces, such as peripheral component interconnect (PCI), analog output, USB, etc., ensuring system compatibility and expandability.

For signal conversion needs, this paper introduces the ISO-U2-P1-F8 isolation transmitter to realize the conversion of analog DC voltage signals into digital pulse frequency signals. The transmitter features a hybrid integrated circuit design (Figure 2b) that provides triple isolation of power supply, input signal, and output signal through advanced process structures and isolation technology. The output range of the pulse signal frequency can

be flexibly set by the precise calibration of the zero and full potentiometers. The system operates over a voltage range of DC4 V–60 V, supports high- and low-level trigger modes, and can process up to 2.5 kHz of PWM signals. In terms of output performance, the module can continuously output 10 A of current at room temperature, with a power output of 600 W. This can be increased to 15 A after equipping it with auxiliary heat dissipation, guaranteeing the stable operation of the system.

2.2. QUARC-Based Control System

This system implements QUARC 2.7, a Quanser Inc. real-time control environment that integrates with Simulink for hardware design and verification. The seamless integration of QUARC and Simulink provides an efficient solution for real-time application development, supporting a variety of target platforms, such as Windows and Linux. Researchers can use its proprietary block diagram module to quickly build system models for online parameter tuning and real-time condition monitoring.

The core strengths of the QUARC environment are its high real-time performance, multi-rate and multi-threaded model support, and the ability to generate multi-objective code from a single Simulink model. These features significantly improve system development efficiency and manageability. In this study, the robot control and data acquisition system was constructed by Simulink, and the generated real-time code was deployed to the Q8-USB data acquisition card by the QUARC Objective Manager.

The system employs a two-layer architecture comprising a PC and a Q8-USB card for data exchange. The Q8-USB card generates voltage signals via its analog output interface, which are then converted into pulse signals by the isolation transmitter to control the robot's position. Simultaneously, the joint motor position, speed, and current information, along with the force sensor data, are fed back into the Q8-USB card through analog inputs and single-ended encoder interfaces, enabling closed-loop control. The acquisition and control components are illustrated in Figure 3a,b. In addition, Table 4 clearly shows the functionality of each module in Simulink.

Table 4. Key modules of Simulink.

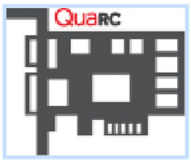
Module	Name	Function
	HIL Initialize	Initializes acquisition card parameters to associate the system with a specific HIL card (Q8-USB card).
	HIL Read Analog	When this block is executed, the analog signal of the specified channel is read.
	HIL Read other	When this block is executed, the encoder pulse signal of the specified channel is read.
	HIL Write Analog	When this block is executed, the corresponding analog signal is written to the specified channel.

Table 4. *Cont.*

Module	Name	Function
simout	To Workspace	Writes input signal data to workspace.

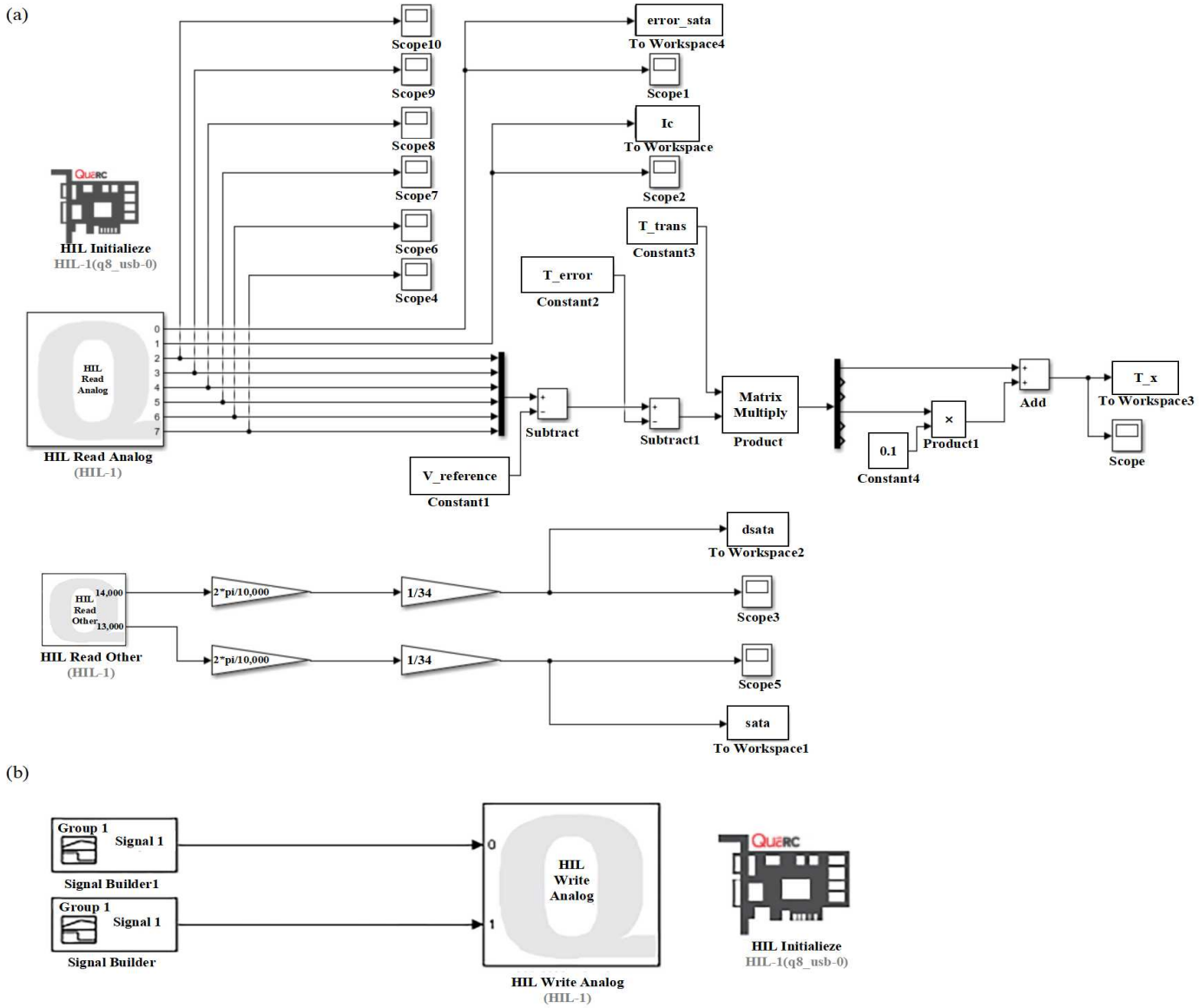


Figure 3. MATLAB/Simulink-based data acquisition and control module. (**a**) Acquisition section; (**b**) control section.

2.3. Sampling Frequency Determination

The choice of sampling frequency is a key consideration when designing a data acquisition system. According to the Nyquist–Shannon sampling theorem [28], to reconstruct a band-limited signal without distortion, the sampling frequency must be at least twice the highest frequency of the signal:

$$f_s \geq 2f_{\max} \tag{1}$$

where f_s is the sampling frequency and $f_{\max}$ is the highest frequency component in the signal. According to Farrow et al. [29], for sampling a finite bandwidth signal, the Nyquist interval can be expressed as follows:

$$d_{rN} \geq \frac{\pi}{Q_{\max}} \tag{2}$$

where $Q_{\max}$ is the maximum frequency of the signal in the frequency domain. The corresponding sampling frequency is $f_s = 2f_{\max} = Q_{\max}/\pi$.

For non-uniform and locally averaged sampling [30], the Nyquist sampling rate is also related to the density δ of sampling points:

$$\varphi_s > 2 \min_{F_e \in \mathcal{F}} \max |(\sigma(L|F_e))| \tag{3}$$

where φ_s is the corner sampling frequency, $\min_{F_e \in \mathcal{F}}$ denotes the minimum among all frequencies F_e, $\sigma(L|F_e)$ denotes the eigen-spectrum of the signal in L-space with respect to frequency F_e, and $\max|(\sigma(L|F_e))|$ is the maximum absolute value of the imaginary part of the eigen-spectrum.

In industrial robot collision detection systems, the dynamic characteristics of joint velocity, torque, and position information are mainly concentrated in the low and medium frequency bands. Referring to previous studies [31], the main mechanical response frequency of a robotic system is usually below 500 Hz. The motor current and hysteresis pulse information output from the MR-J2S-70A servo drive through the CN3 connector is in the frequency range well below 500 Hz. Whereas the ATI six-dimensional force sensors have a transmission rate of up to 28.5 kHz, the ISO-U2-P1-F8 isolation transmitter can handle PWM signals up to 2.5 kHz. Therefore, the sampling frequency of 1 kHz was chosen in this paper not only to satisfy the requirements of the Nyquist–Shannon sampling theorem, but also to be within the range of the actual hardware processing capabilities. This frequency is sufficient to capture the key frequency components of these signals to ensure the accuracy and timeliness of detection.

3. Modeling and Characterization of Collision State Systems

3.1. Dynamics Modeling

We modeled the dynamics of the robot manipulator arm as a theoretical basis for collision detection. For the n degree-of-freedom robot manipulator arm, its complete dynamics equation can be expressed as follows:

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + G(\boldsymbol{q}) = \boldsymbol{\tau} + \boldsymbol{\tau}_t \tag{4}$$

where $\boldsymbol{M}(\boldsymbol{q})$ is the inertia matrix, reflecting the dynamic coupling between the joints; $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is the Coriolis and centrifugal force matrix, characterizing the nonlinear properties during high-speed motion; $G(\boldsymbol{q})$ represents the gravity term effects, directly related to the joint configuration; τ is the control input torque; $\boldsymbol{\tau}_t$ is the torque resulting from external collision; $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$ denote the joint position, velocity, and acceleration vectors, respectively.

To effectively estimate external collision torques, we implement a momentum observer based on the dynamics equations [5]:

$$\dot{\hat{p}} = \boldsymbol{C}^T(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - G(\boldsymbol{q}) + \tau + \boldsymbol{r} \tag{5}$$

where $\dot{\hat{p}}$ denotes the estimated momentum derivative, computed from known control inputs and system states; $\boldsymbol{r}$ is the observation residual vector, which is used to reflect the effect of external torques and is a central metric for collision detection. This observer allows us to make an accurate estimate of the external torque without directly measuring it. Considering the discrete time nature of practical applications, the observer is realized in discrete time in the following form:

$$\boldsymbol{r}_t = \boldsymbol{K}_o \widetilde{p}_t - \boldsymbol{K}_o \sum_{k=1}^{t-1} \left(\boldsymbol{C}^T (\boldsymbol{q}_k, \dot{\boldsymbol{q}}_k) \dot{\boldsymbol{q}}_k - g(\boldsymbol{q}_k) + \tau_k + \boldsymbol{r}_k \right) \tag{6}$$

where r_t denotes the observed residuals at discrete time step t moments, K_o is the observer gain matrix, $\widetilde{p}_t$ denotes the momentum estimation error, and subscript k denotes the index of the discrete time series. With this discrete implementation, the system is able to efficiently update the torque residual estimates during the real-time control loop.

3.2. Collision Feature Extraction

Collision events cause significant changes in the robot state, and in order to effectively capture the collision state characteristics, we define the time derivative of the torque residuals as follows:

$$\gamma_t = \boldsymbol{r}_t - \boldsymbol{r}_{t-1} \tag{7}$$

where γ_t is the amount of change in the residual torque computed at time t, which provides information about the rate of change in the system's dynamical state. r_t and r_{t-1} are the torque residuals at the current and previous time steps, respectively. By comparing the difference between these two measurements, transient characteristics caused by the collision can be captured.

In addition to the moment information, collisions also cause the joint motion to deviate from the expected trajectory. Therefore, we introduce the joint velocity error as a complementary feature:

$$\dot{e}_t = \dot{\boldsymbol{q}}_d - \dot{\boldsymbol{q}} \tag{8}$$

where $\dot{\boldsymbol{q}}_d$ is the desired joint velocity, $\dot{\boldsymbol{q}}$ is the actual joint velocity, and the difference between the two $\dot{e}_t$ can reflect the extent to which the collision interferes with the kinematic state. This feature is particularly important for capturing minor collisions, which may not be evident enough in the torque residuals, but will show up in the velocity error. Considering the above eigenvolumes together, we construct the integrated measurement vector for the first i joint:

$$\boldsymbol{z}_t^{[i]} = \left[\gamma_t^{[i]} \ \dot{e}_t^{[i]} \right]^T \tag{9}$$

Normalization was introduced to remove the effect of the scale and to standardize the data distribution:

$$\widetilde{\boldsymbol{z}}_t^{[i]} = \frac{\boldsymbol{z}_t^{[i]} - \boldsymbol{\mu}_z}{\boldsymbol{\sigma}_z} \tag{10}$$

where $\boldsymbol{\mu}_z$ and $\boldsymbol{\sigma}_z$ represent the mean vector and the standard deviation vector of the historical data, respectively. Through this normalization, the feature components are mapped to similar numerical ranges, which eliminates the impact of the difference in magnitude on the subsequent analysis, and at the same time enhances the numerical stability and generalization ability of the algorithm.

3.3. State of the Art Assessment of Mahalanobis Distance

In order to quantitatively assess the degree of similarity between the current measurement data and the normal state or the collision state, this section constructs a statistical observation model based on the Mahalanobis distance. As a metric that considers the data distribution characteristics, the Mahalanobis distance can effectively address the correlation between features and the weighting of different features [32]. For non-collision states, the distance metric is defined as follows:

$$d_f^{[i]} \left(\boldsymbol{z}_t^{[i]} \right) = \sqrt{ \left(\boldsymbol{z}_t^{[i]} \right)^T \left(\sum\nolimits_f^{[i]} \right)^{-1} \left(\boldsymbol{z}_t^{[i]} \right) } \tag{11}$$

where $d_f^{[i]}\left(z_t^{[i]}\right)$ denotes the Mahalanobis distance metric for the i joint, which is used to quantify the degree of deviation of the current observation from the normal state distribution; $z_t^{[i]}$ is the vector of normalized measurements for the i joint; $\sum_f^{[i]}$ is the covariance matrix of the measurements in the normal operating state, which captures the correlation structure among the features.

Accordingly, for the collision state, the corresponding distance metric is as follows:

$$d_c^{[i]}\left(z_t^{[i]}\right) = \sqrt{\left(z_t^{[i]+} - \mu_c^{[i]}\right)^T \left(\sum_c^{[i]}\right)^{-1} \left(z_t^{[i]+} - \mu_c^{[i]}\right)} \tag{12}$$

where $\left(z_t^{[i]+} - \mu_c^{[i]}\right)^T$ is the difference between the normalized measurement vector and the mean value of the collision state, indicating the deviation of the current observation from the mean value of the collision state. This metric takes into account the mean shift of the collision state $\mu_c^{[i]}$ and the covariance $\sum_c^{[i]}$, which more accurately characterizes the distribution of the measurement data in the collision state, as shown in Figure 4.

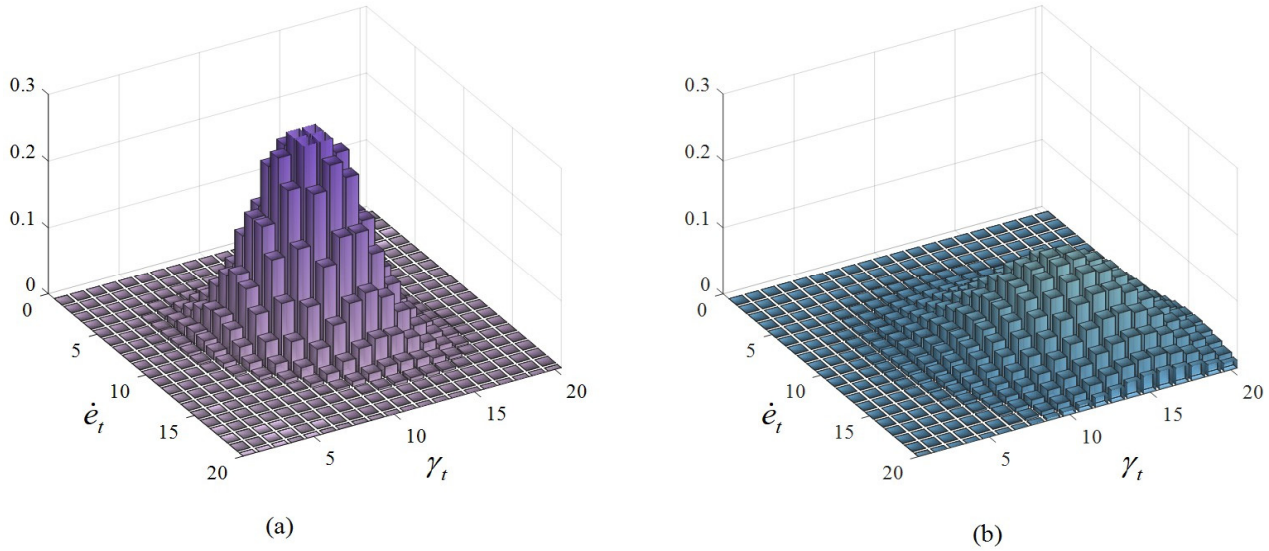


Figure 4. Mahalanobis distance in two different states under characterization γ_t and $\dot{e}_t$ of representation. (**a**) Normal state (no collision) $d_f^{[i]}$ distribution of features; (**b**) collision state $d_c^{[i]}$ distribution of features.

4. Neural Network Detection Optimization and Implementation

4.1. Network Architecture Design

Based on the observation model and feature extraction described above, and considering both the nonlinear nature of the problem and real-time requirements, we constructed a typical three-layer BP network. The output of the hidden layer is computed with the following expression:

$$h = f_1\left(\omega_1 \tilde{z}_t^{[i]} + b_1\right) \tag{13}$$

where ω_1 is the weight matrix connecting the input layer and the implicit layer, which linearly transforms the normalized input eigenvectors $\tilde{z}_t^{[i]}$ into the implicit layer space; b_1 is the implicit layer bias vector, which increases the flexibility of the model; $f_1(\cdot)$ is the activation function of the implicit layer.

The computational expression for the output layer is as follows:

$$y = f_2(\omega_2 h + b_2) \tag{14}$$

where ω_2 is the weight matrix connecting the implicit layer to the output layer, which maps the representation of the implicit layer to the final decision space; b_2 is the bias vector of the output layer, $f_2(\cdot)$ is the activation function of the output layer. Considering the generalization performance of the network, we introduce the error function with a regularization term:

$$E = \frac{1}{2}\sum(\hat{y}_d - y)^2 + \frac{\lambda}{2}\sum\left(\omega_2^2\right) \tag{15}$$

It is worth noting that $\hat{y}_d$ is the desired output, i.e., the collision state of the markers; y is the actual output; λ is the regularization coefficient, which controls the strength of the regularization; ω_2 represents all the weight parameters in the network; $\sum(\hat{y}_d - y)^2$ is the standard error term of the network, which represents the difference between the target output and the actual output; $\frac{\lambda}{2}\sum(\omega_2^2)$ is the regularization term, which is used to penalize excessively large weights and prevent the network from overfitting.

4.2. LM Algorithm Optimization

Neural network training quality critically determines the collision detection system's performance. Considering the dual requirements of convergence speed and stability, we use the improved LM algorithm for network parameter optimization [33]. The LM algorithm balances the stability of gradient descent with Newton method's rapid convergence properties, which is particularly suitable for the training of medium-sized networks. The core update equation of the LM algorithm is given as follows:

$$\omega(k+1) = \omega(k) - \left[J^T J + \mu I\right]^{-1} J^T e \tag{16}$$

This equation embodies the core idea of the LM algorithm, where μ is the damping factor and the key tuning parameter. $\omega(k)$ denotes the weight vector of the network at the *kth* iteration; J is the Jacobi matrix containing the partial derivatives of the error with respect to all weights; e denotes the error vector; and I is the identity matrix. $\left[J^T J + \mu I\right]^{-1}$ is the central part of the LM algorithm by which the updated weight step is computed. This term ensures a balance between convergence and stability. The overall error function of the network using Equation (15) is transformed into a mean squared error form, which ensures the numerical stability of the training process, as shown in Figure 5a. The calculation of the Jacobi matrix is based on the sensitivity of the errors to the weights:

$$J_{ij} = \sigma_E / \sigma_\omega \tag{17}$$

where J_{ij} denotes the element of the Jacobi matrix in row i and column j, which describes the sensitivity of the error to the weights, and σ_E and σ_ω are the standard deviations of the error and the weights, respectively.

The adaptive tuning mechanism of the LM algorithm is as follows:

$$\begin{aligned} if\ E(k+1) < E(k) \quad & \mu = \mu/\beta \\ else \quad & \mu = \mu \cdot \beta \end{aligned} \tag{18}$$

This tuning mechanism dynamically balances the algorithm's ability to explore and utilize through the parameter β. If the error of the current iteration is smaller than the error of the previous iteration, the damping factor μ will be decreased (via μ/β); if the error increases, the damping factor will be increased (via $\mu \cdot \beta$). Based on the trained neural network, we transform the results of the Mahalanobis distance state assessment introduced in Section 3.3 into a continuous probabilistic output:

$$P\left(z_t^{[i]}\right) = y_t^{[i]} \tag{19}$$

This probability value $P\left(z_t^{[i]}\right)$ indicates the likelihood that the first i joint is in a collision state, where $y_t^{[i]}$ is the output value of the neural network. By comparing the evaluation results of the Mahalanobis distance metric, $d_f^{[i]}$ and $d_c^{[i]}$ are transformed by the neural network into continuous probability outputs within the interval $[0, 1]$, as shown in Figure 5b.

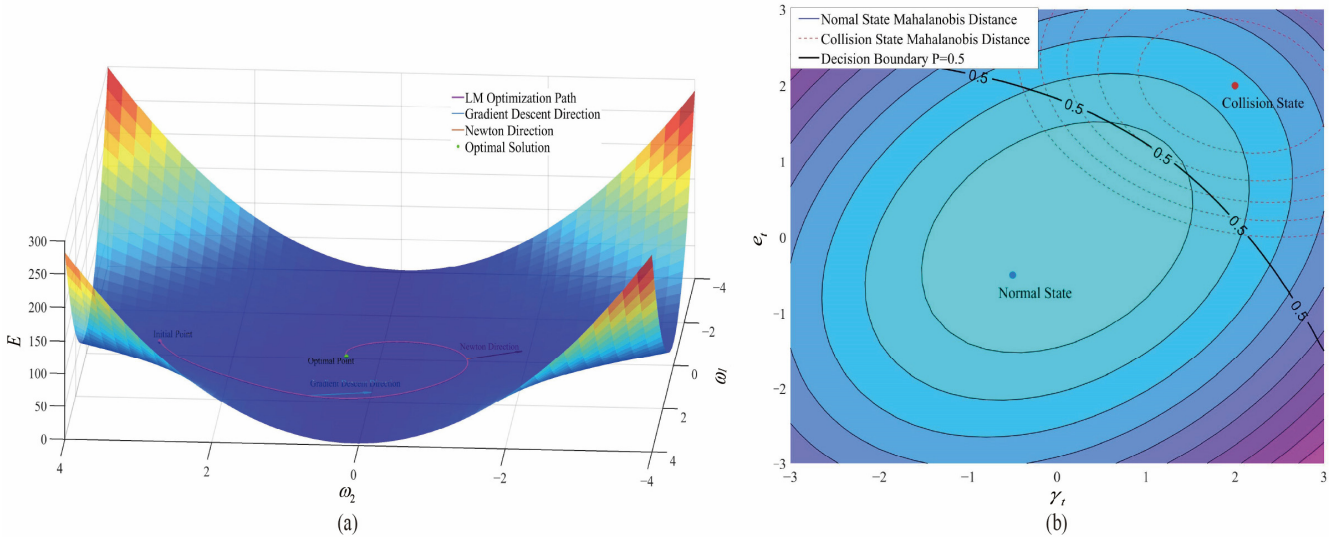


Figure 5. (**a**) Optimization trajectory of the improved LM algorithm on error surfaces showing adaptive damping factor μ tuning. (**b**) Mahalanobis distance field with probabilistic decision boundary showing normal state (blue), collision state (red dashed line), and decision boundary (black).

5. Experimental Research

5.1. Data Acquisition

We conducted experiments on a Rebot-V-6R-650 six-degree-of-freedom industrial robot (Figure 6a) controlled by a computer equipped with an Intel Core i7-12700H processor and 16 GB RAM. This computer was equipped with an Intel Core i7-12700H processor (14 cores, 2.7 GHz) and 16 GB of RAM. The hardware connection and software configuration of the system were completed before data acquisition, including connecting the Q8-USB data acquisition card to the computer via the USB 2.0 interface, connecting the ATI six-dimensional F/T sensor (Figure 6b) to the ADC channel of the Q8-USB card via the analog output port, and connecting the CN3 connector and CN2 encoder connector of the MR-J2S-70A servo drive to the analog input port and the single-ended encoder connector of the Q8-USB card. Figure 7a shows the configuration of the parameters of the hardware part. The device type was x86-64 (Windows 64), and the hardware board was set to "Determined by the code generation system target file". The code generation system target file was quarc_win64.tlc, which included the configuration of the number of bits of various data types.

The software environment was configured in the MATLAB R2022b and QUARC 2.7 development environments, and the acquisition module was constructed for reading the position information of the joint encoder, acquiring the joint current and six-dimensional force sensor data (the Simulink-based simulation module introduced in Section 2.2). Figure 7b demonstrates the solver parameter settings in the simulation interface. The simulation time was configured with a start time of 0.0 and end time of 10. The solver selected was a fixed-step type with automatic solver selection mode, and the fixed step size (basic sam-

pling time) was set to 0.001. During the acquisition process, the system collected the data synchronously at a frequency of 1 kHz (the sampling frequency confirmed in Section 2.3) and monitored the signal quality in real time through the Scope module.

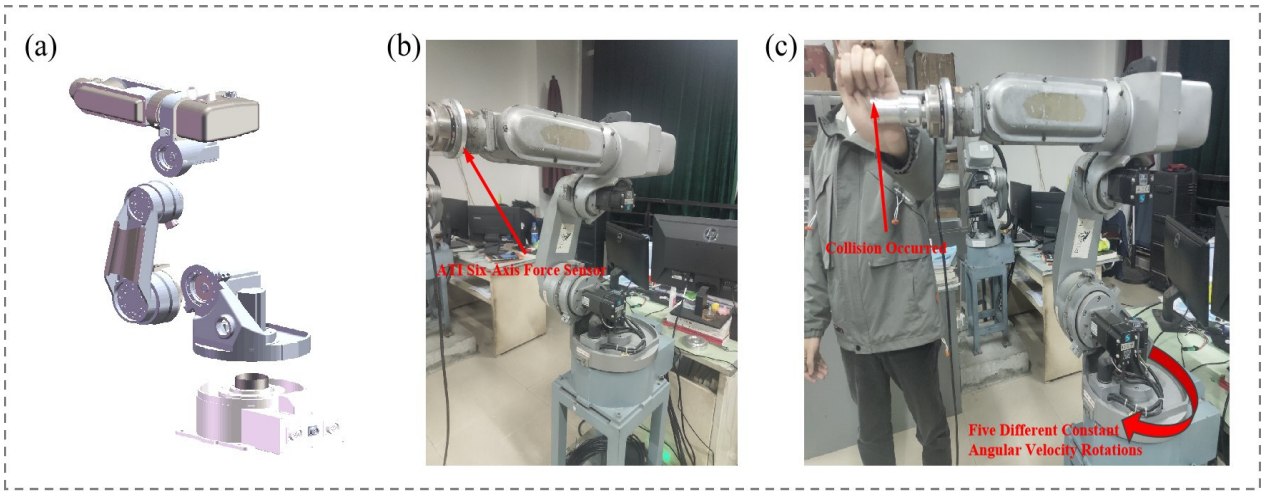


Figure 6. (**a**) Rebot-V-6R-650 six-degrees-of-freedom industrial robot used for data acquisition and experiments; (**b**) ATI six-dimensional F/T sensor; (**c**) experimental scenario of a simulated collision occurring between a human and a robot.

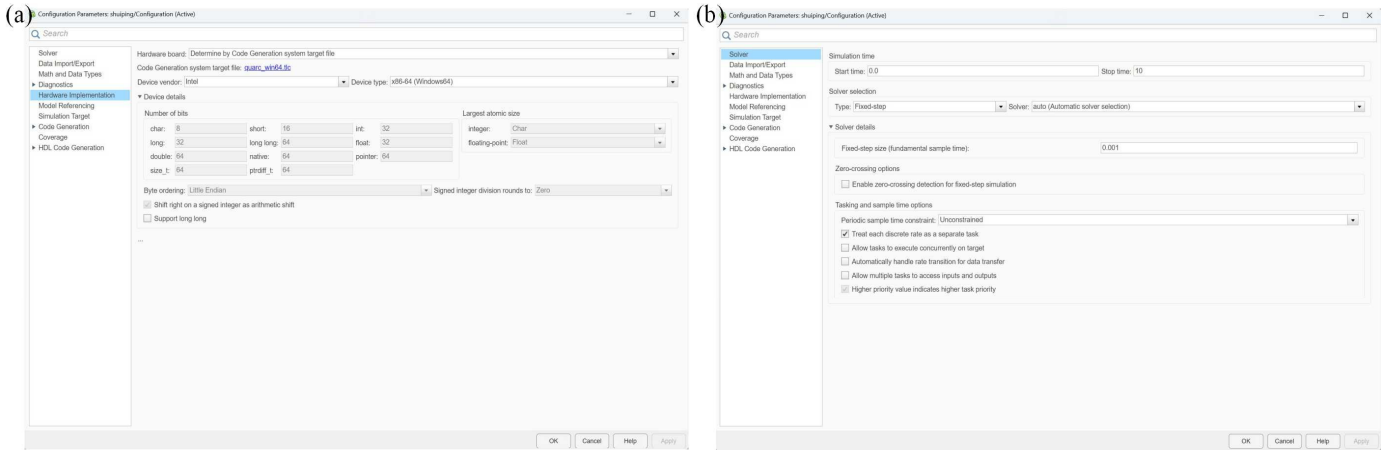


Figure 7. Parameter configuration. (**a**) Parameterization of the hardware section. (**b**) Parameter configuration for simulation.

The general flow of the CollisionSense DAQ system is summarized in Algorithm 1, and the flowchart is in Figure 1 of the general technical framework.

The data collection consisted of two scenarios—normal motion and interference from external forces, corresponding to the robot in the no-force and human collision conditions, respectively. The dataset comprised 13,717 collision samples and 228,610 non-collision samples, capturing comprehensive joint parameters, including velocities, torques, currents, positions, and hysteresis pulses. Throughout the collection process, the robot moved in a position-controlled mode and did not apply a reaction strategy.

Algorithm 1 CollisionSense DAQ System

Require: Q8-USB card, QUARC environment, f_s (sampling frequency), Ntotal (total samples), ratio (collision sample ratio)
Ensure: Complete dataset containing collision and non-collision states

```
function Data_Acquisition()
    Initialize QUARC environment (f_s = 1000)
    Configure Q8-USB card channels:
        analog_inputs ← [0:7]
        encoder ← 0
        analog_output ← 0
        digital_output ← 0
    Set up hardware components:
        servo_driver ← Setup_MR_J2S_70A(position_control, 131,072 kpps)
        ft_sensor ← Setup_ATI_force_sensor()
        iso_transmitter ← Setup_isolation_transmitter (0–10 V, 0–2.5 kHz)
    Create_Simulink_model()
    non_collision_count ← 0
    collision_count ← 0
    non_collision_target ← Ntotal × (1 − ratio)
    collision_target ← Ntotal × ratio
    Set_collision_mode(false)
    Start_data_acquisition()
    while non_collision_count < non_collision_target do
        position ← Read_encoder_position()
        velocity ← Read_encoder_velocity()
        current ← Read_analog_input([0,1])
        hysteresis ← Read_hysteresis_pulse()
        Store_non_collision_data(position, velocity, current, hysteresis)
        non_collision_count ← non_collision_count + 1
    end while
    Set_collision_mode(true)
    while collision_count < collision_target do
        position ← Read_encoder_position()
        velocity ← Read_encoder_velocity()
        current ← Read_analog_input([0,1])
        hysteresis ← Read_hysteresis_pulse()
        ft_data ← Read_analog_input([2–7])
        Store_collision_data(position, velocity, current, hysteresis, ft_data)
        collision_count ← collision_count + 1
    end while
    Stop_data_acquisition()
    Save_dataset('data.mat')
    return dataset
end function
```

5.2. Model Training

The input layer of the network contained 12 neurons corresponding to features such as velocity, current, and hysteresis pulse at each moment; the hidden layer used 20 neurons, and the output layer used 1 neuron, representing the collision detection results. The hidden and output layers used the tansig and purelin activation functions, respectively.

To improve the network performance, the training samples were normalized so that they were distributed in the $[-1, 1]$ interval. During the training process, the LM algorithm dynamically adjusted the Hessian matrix by means of the correction value μ. The weights were updated by the following formula:

$$\Delta\omega = -\left[\boldsymbol{J}^T\boldsymbol{J} + \mu\boldsymbol{I}\right]^{-1}\boldsymbol{J}^T e \tag{20}$$

where $\boldsymbol{J}$ is the Jacobi matrix, e is the network output error, and $\boldsymbol{I}$ is the unit matrix. When the error decreased, the μ value decreased to approximate the Newton method to speed up the convergence speed; when the error increased, the μ value increased to approximate the gradient descent method to ensure training stability. We employed cross-validation, dividing our dataset into training, validation, and testing sets with a 7:2:1 ratio. To prevent overfitting, an early stopping strategy was used to terminate the training when the validation set error did not decrease significantly for six consecutive iterations. After 5000 iterations of training, the detection accuracy of the network on the validation set reached 97.8% and the mean square error converged to 0.0021.

5.3. Experimental Validation

In order to verify the effectiveness of the proposed method (MomentumNet-CD), five sets of systematic experiments were designed and implemented. In each set of experiments, the first joint of the robot was set to run at different constant angular velocities of 10°/s, 12°/s, 15°/s, 20°/s, and 25°/s. During each set of experiments, we randomly applied 900 collision interferences to the end-effector in the robot's motion trajectory to simulate accidental contact situations in real environments. The experimental scenario configuration is shown in Figure 6c, and the experimental results are shown in Table 5 and Figure 8.

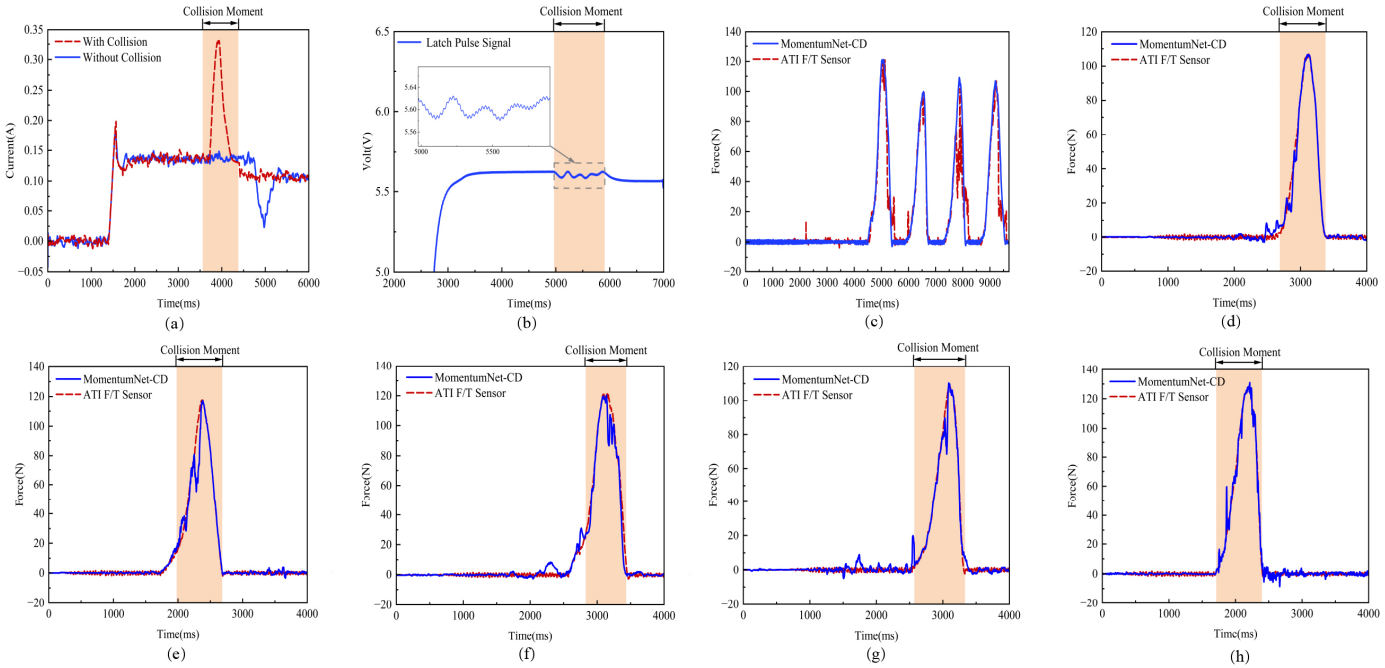


Figure 8. Experimental results (Joint 6). (**a**) Trend of the joint current with and without collision; (**b**) trend of the joint motor hysteresis pulse signal when subjected to collision; (**c**) trend of the comparison between the MomentumNet-CD model-predicted collision force and the actual collision force measured by the ATI F/T sensor recorded randomly during the experiments; (**d**–**h**) joint velocities at 10°/s, 12°/s, 15°/s, 20°/s, and 25°/s, respectively; comparison trends between MomentumNet-CD model predicted collision forces and actual collision forces measured by ATI six-dimensional F/T sensors.

Table 5. Performance comparison of MomentumNet-CD at different velocities.

Joint Velocity	NC	CC	FN	FP	Accuracy/%
10°/s	900	848	52	0	94.26
12°/s	900	847	53	0	94.21
15°/s	900	844	56	0	93.85
20°/s	900	840	60	1	93.34
25°/s	900	833	67	1	92.57
Total	4500	4212	288	2	93.65

Note: NC denotes the total number of collisions, CC denotes the number of correctly detected collision detections, FN denotes the number of false negatives (i.e., the number of undetected collisions), and FP denotes the number of false positives (i.e., the number of falsely detected collisions).

The joint current and motor stall pulse signals exhibited significant fluctuations during the collision events (Figure 8a,b). Notably, even under normal operating conditions without collisions, nonlinear joint friction and linkage inertial forces caused minor signal fluctuations, which are expected in standard operational environments. Considering that robot body sensing signals are inevitably affected by electrical noise, this study employed digital filtering techniques to optimize the response quality while maintaining the system timeliness.

In addition, the proposed MomentumNet-CD algorithm performed excellently in collision force prediction. As seen from the comparison curves (Figure 8c), the model estimates highly match the measured values of the six-dimensional F/T sensor, reflecting the excellent fitting accuracy.

Figure 8d–h present the experimental results under five different speed conditions. All tests successfully and accurately detected end-of-robot collision events. By analyzing the results against the standard reference values provided by the ATI six-dimensional F/T sensors, the predicted results of the MomentumNet-CD algorithm show remarkable consistency with the actual measurements, which ensures a high degree of accuracy in collision detection. Of particular interest is that during the initial phase of the collision, when the external force was gradually increasing, the model exhibited excellent dynamic tracking ability, with the predicted values reaching the collision threshold almost in synchronization with the actual force values, with minimal system delay.

Based on the data analysis in Table 5, the MomentumNet-CD algorithm performed consistently over the speed range from 10°/s to 25°/s, maintaining an accuracy of over 92%, with only a slight decrease with increasing speed (94.26% to 92.57%). The algorithm achieved an overall accuracy of 93.65% across 4500 samples, with 4212 correct detections, 288 missed detections, and only 2 false detections. This demonstrates the algorithm's excellent detection reliability and very low false detection rate in a variable speed environment.

5.4. Comparison with Existing Method

To evaluate the effectiveness of the proposed MomentumNet-CD algorithm in real-world application scenarios, CollisionNet proposed by Heo et al. [26] was selected for comparison in this paper. CollisionNet adopts a deep learning approach to directly process the robot joint signals and output the collision detection results through a one-dimensional convolutional neural network, which has demonstrated good performance in industrial scenarios.

To ensure the comparability of the experiments, we implemented two collision detection methods on the Rebot-V-6R-650 industrial robot platform. In the experiments, we installed FSR sensors on the robot for recording the actual collision time. The test scenarios covered collisions with different contact strengths (from light to strong collision) and multiple locations. Through comparative experiments, we focused on the performance of the two methods in two key performance metrics, DD and FP.

The experimental results in Table 6 are compared and analyzed in Figure 9a. The MomentumNet-CD method proposed in this paper achieved a significant breakthrough in the performance of collision detection, with a detection delay of only 12.16 ms, which was 41.4% shorter than that of CollisionNet's 20.75 ms. In terms of false alarm control, MomentumNet-CD demonstrated excellent stability and maintained high accuracy regardless of the diverse contact strength variations. Figure 9b further verifies the robustness of the method in the face of complex disturbances, such as joint friction, load variation, and environmental vibration.

Table 6. Performance comparison of different methods.

Method	DD	FP
MomentumNet-CD	12.16 ms	0
CollisionNet	20.75 ms	2

Note: DD denotes collision delay, and FP denotes the number of false positives (i.e., the number of falsely detected collisions).

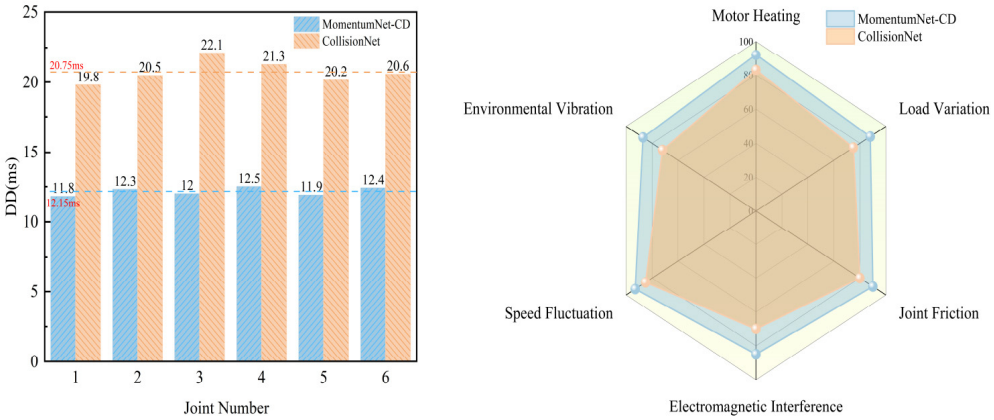


Figure 9. Comparison of MomentumNet-CD and CollisionNet performances. (**a**) Comparison of collision detection delay for each joint; (**b**) comparison under complex environmental interference.

6. Conclusions and Future Work

The experimental results demonstrate that MomentumNet-CD delivers high accuracy and excellent real-time performance; the overall accuracy of collision detection reached 93.65% under five different speed conditions, and the detection delay was only 12.16 ms, which verifies the effectiveness and practical value of the method when compared with the existing CollisionNet algorithm. In addition, the universal acquisition system based on the Q8-USB data acquisition card and the QUARC real-time control environment provides a reliable hardware platform for the experimental validation of collision detection. It is worth noting that MomentumNet-CD ensures robot stability during collision detection. This stability is primarily attributed to the well-designed momentum observer, which accurately estimates external collision torques without interfering with the normal control process, thereby maintaining the stability of the original control strategy during non-collision states. The optimized BP neural network decision-making mechanism adopts continuous probability output instead of the traditional threshold judgment method, which effectively avoids the control discontinuity problem caused by sudden changes of state and further improves the overall stability and reliability.

Our approach shows significant potential for application in human–robot interaction scenarios. For example, in manually guided robot scenarios [34], MomentumNet-CD's low-latency detection provides real-time safety for human–robot collaboration, and its momentum observer-based feature extraction can be combined with an impedance learning framework to

efficiently differentiate between guiding forces and accidental collisions. Future work will focus on integrating variable impedance control strategies to balance safety and efficiency in human–robot interactions, with safety remaining our primary consideration.

Author Contributions: Conceptualization, J.Y. and Y.F.; methodology, J.Y.; software, Y.F. and Q.K.; validation, J.Y. and Y.F.; formal analysis, Q.K.; resources, H.W. and G.Z.; writing—original draft preparation, J.Y.; writing—review and editing, J.Y., Y.F. and X.L.; supervision, H.W. and G.Z.; project administration, G.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the Talent Program of the State Administration for Market Regulation (QNBJ202319) and in part by a research project of the Fujian Provincial Market Supervision Bureau (FJMS2023016).

Data Availability Statement: The data will be made available upon request from the corresponding author.

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Zhang, T.; Ge, P.; Zou, Y.; He, Y. Robot collision detection without external sensors based on time-series analysis. *J. Dyn. Syst. Meas. Control.* **2021**, *143*, 041005. [CrossRef]
2. Bonci, A.; Cheng, P.D.C.; Indri, M.; Nabissi, G.; Sibona, F. Human-robot perception in industrial environments: A survey. *Sensors* **2021**, *21*, 1571. [CrossRef] [PubMed]
3. Park, K.M.; Kim, J.; Park, J.; Park, F.C. Learning-based real-time detection of robot collisions without joint torque sensors. *IEEE Robot. Autom. Lett.* **2020**, *6*, 103–110. [CrossRef]
4. Huang, S.; Gao, M.; Liu, L.; Chen, J.; Zhang, J. Collision detection for cobots: A back-input compensation approach. *IEEE/ASME Trans. Mechatron.* **2022**, *27*, 4951–4962. [CrossRef]
5. Park, J.; Kim, T.; Gu, C.; Kang, Y.; Cheong, J. Dynamic collision estimator for collaborative robots: A dynamic Bayesian network with Markov model for highly reliable collision detection. *Robot. Comput.-Integr. Manuf.* **2024**, *86*, 102692. [CrossRef]
6. Lv, H.; Liu, L.; Gao, Y.; Zhao, S.; Yang, P.; Mu, Z. A compound planning algorithm considering both collision detection and obstacle avoidance for intelligent demolition robots. *Robot. Auton. Syst.* **2024**, *181*, 104781. [CrossRef]
7. Chang, Z.; Chen, H.; Hua, M.; Fu, Q.; Peng, J. A bio-inspired visual collision detection network integrated with dynamic temporal variance feedback regulated by scalable functional countering jitter streaming. *Neural Netw.* **2025**, *182*, 106882. [CrossRef]
8. Montaut, L.; Le Lidec, Q.; Petrik, V.; Sivic, J.; Carpentier, J. GJK++: Leveraging Acceleration Methods for Faster Collision Detection. *IEEE Trans. Robot.* **2024**, *40*, 2564–2581. [CrossRef]
9. Shen, T.; Liu, X.; Dong, Y.; Yang, L.; Yuan, Y. Switched Momentum Dynamics Identification for Robot Collision Detection. *IEEE Trans. Ind. Inform.* **2024**, *20*, 11252–11261. [CrossRef]
10. Xu, T.; Tuo, H.; Fang, Q.; Shan, D.; Jin, H.; Fan, J.; Zhu, Y.; Zhao, J. A novel collision detection method based on current residuals for robots without joint torque sensors: A case study on UR10 robot. *Robot. Comput.-Integr. Manuf.* **2024**, *89*, 102777. [CrossRef]
11. Yun, A.; Lee, W.; Kim, S.; Kim, J.-H.; Yoon, H. Development of a robot arm link system embedded with a three-axis sensor with a simple structure capable of excellent external collision detection. *Sensors* **2022**, *22*, 1222. [CrossRef] [PubMed]
12. Wu, H.; Chen, J.; Su, Y.; Li, Z.; Ye, J. New tactile sensor for position detection based on distributed planar electric field. *Sens. Actuators A Phys.* **2016**, *242*, 146–161. [CrossRef]
13. Min, F.; Wang, G.; Liu, N. Collision Detection and Identification on Robot Manipulators Based on Vibration Analysis. *Sensors* **2019**, *19*, 1080. [CrossRef]
14. Ma, J.; Zhuang, X.; Zi, P.; Zhang, T.; Zhang, W.; Xu, K.; Ding, X. Efficient Collision Detection Algorithm for Space Reconfigurable Integrated Leg-Arm Robot. In Proceedings of the 2024 IEEE 19th Conference on Industrial Electronics and Applications (ICIEA), Kristiansand, Norway, 5–8 August 2024; pp. 1–6.
15. Zhao, B.; Wu, C.; Chang, L.; Jiang, Y.; Sun, R. Research on Zero-Force control and collision detection of deep learning methods in collaborative robots. *Displays* **2025**, *87*, 102969. [CrossRef]
16. Liu, B.; Fu, Z.; Hua, Z.; Zhang, J. Collision detection and fault diagnosis with DMAO-GRU for flow regulating valve. *Meas. Sci. Technol.* **2024**, *36*, 016238. [CrossRef]
17. Li, W.; Han, Y.; Wu, J.; Xiong, Z. Collision detection of robots based on a force/torque sensor at the bedplate. *IEEE/ASME Trans. Mechatron.* **2020**, *25*, 2565–2573. [CrossRef]

18. Zhang, X.; Zhong, Z.; Guan, W.; Pan, M.; Liang, K. Collision-risk assessment model for teleoperation robots considering acceleration. *IEEE Access* **2024**, *12*, 101756–101766. [CrossRef]
19. Zhang, T.; Chen, Y.; Zou, Y. Robot collision detection based on external moment observer. *J. South China Univ. Technol. (Nat. Sci. Ed.)* **2024**, *52*. [CrossRef]
20. Chen, S.; Xiao, H.; Qiu, L.; Bi, Q.; Chen, X. Robotic Flexible Collision Detection Based on Second-Order Sliding-Mode Momentum Observer. In Proceedings of the 2024 10th International Conference on Electrical Engineering, Control and Robotics (EECR), Guangzhou, China, 29–31 March 2024; pp. 1–7.
21. Zhao, P.; Gao, Z.; Liu, X.; Zeng, Y.; Zhou, Y. Collision observation of collaborative robots based on generalized momentum deviation. *Proc. Inst. Mech. Eng. Part C J. Mech. Eng. Sci.* **2024**, *239*, 09544062241299672. [CrossRef]
22. Lu, S.; Xu, Z.; Wang, B. Human-robot collision detection based on the improved camshift algorithm and bounding box. *Int. J. Control. Autom. Syst.* **2022**, *20*, 3347–3360. [CrossRef]
23. Kim, D.; Lim, D.; Park, J. Transferable collision detection learning for collaborative manipulator using versatile modularized neural network. *IEEE Trans. Robot.* **2021**, *38*, 2426–2445. [CrossRef]
24. Fan, T.; Long, P.; Liu, W.; Pan, J. Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios. *Int. J. Robot. Res.* **2020**, *39*, 856–892. [CrossRef]
25. Sharkawy, A.N.; Koustoumpardis, P.N.; Aspragathos, N. Human-robot collisions detection for safe human-robot interaction using one multi-input-output neural network. *Soft Comput.* **2020**, *24*, 6687–6719. [CrossRef]
26. Heo, Y.J.; Kim, D.; Lee, W.; Kim, H.; Park, J.; Chung, W.K. Collision detection for industrial collaborative robots: A deep learning approach. *IEEE Robot. Autom. Lett.* **2019**, *4*, 740–746. [CrossRef]
27. Niu, Z.; Hassan, T.; Boushaki, M.N.; Werghi, N.; Hussain, I. Continuous Wavelet Network for Efficient and Transferable Collision Detection in Collaborative Robots. *IEEE Trans. Syst. Man Cybern. Syst.* **2024**, *55*, 2046–2061. [CrossRef]
28. Zeng, Z.; Liu, J.; Yuan, Y. A generalized Nyquist-Shannon sampling theorem using the Koopman operator. *IEEE Trans. Signal Process.* **2024**, *72*, 3595–3610. [CrossRef]
29. Farrow, C.L.; Shaw, M.; Kim, H.; Juhás, P.; Billinge, S.J. Nyquist-Shannon sampling theorem applied to refinements of the atomic pair distribution function. *Phys. Rev. B—Condens. Matter Mater. Phys.* **2011**, *84*, 134105. [CrossRef]
30. Song, Z.; Liu, B.; Pang, Y.; Hou, C.; Li, X. An improved Nyquist–Shannon irregular sampling theorem from local averages. *IEEE Trans. Inf. Theory* **2012**, *58*, 6093–6100. [CrossRef]
31. Liu, S.; Wu, C.; Liang, L.; Zhao, B.; Sun, R. Research on Vibration Suppression Methods for Industrial Robot Time-Lag Filtering. *Machines* **2024**, *12*, 250. [CrossRef]
32. De Maesschalck, R.; Jouan-Rimbaud, D.; Massart, D.L. The mahalanobis distance. *Chemom. Intell. Lab. Syst.* **2000**, *50*, 1–18. [CrossRef]
33. Moré, J.J. The Levenberg-Marquardt algorithm: Implementation and theory. In *Numerical Analysis, Proceedings of the Biennial Conference Held at Dundee, Dundee, UK, 28 June–1 July 1977*; Springer: Berlin/Heidelberg, 2006; pp. 105–116.
34. Xing, X.; Burdet, E.; Si, W.; Yang, C.; Li, Y. Impedance learning for human-guided robots in contact with unknown environments. *IEEE Trans. Robot.* **2023**, *39*, 3705–3721. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.