applied sciences

MDPI

Article

Application of External Torque Observer and Virtual Force Sensor for a 6-DOF Robot

Chung-Wen Hung * and Guan-Yu Jiang

Department of Electrical Engineering, National Yunlin University of Science and Technology, 123 University Road, Section 3, Yunlin 64002, Taiwan; m11012045@yuntech.edu.tw
* Correspondence: wenhung@yuntech.edu.tw

Abstract: A personal-computer-based and a Raspberry Pi single-board computer-based virtual force sensor with EtherCAT communication for a six-axis robotic arm are proposed in this paper. Both traditional mathematical modeling and machine learning techniques are used in the establishment of the dynamic model of the robotic arm. Thanks to the high updating rate of EtherCAT, the machine learning-based dynamic model on a personal computer achieved an average correlation coefficient between the estimated torque and the actual torque feedback from the motor driver of about 0.99. The dynamic model created using traditional mathematical modeling and the Raspberry Pi single-board computer demonstrates an approximate correlation coefficient of 0.988 between the estimated torque and the actual torque. The external torque observer is established by calculating the difference between the actual torque and the estimated torque, and the virtual force sensor converts the externally applied torques calculated for each axis to the end effector of the robotic arm. When detecting external forces applied to the end effector, the virtual force sensor demonstrates a correlation coefficient of 0.75 and a Root Mean Square Error of 12.93 N, proving its fundamental competence for force measurement. In this paper, both the external torque observer and the virtual force control are applied to applications related to sensing external forces of the robotic arm. The external torque observer is utilized in the safety collision detection mechanism. Based on experimental results, the system can halt the motion of the robotic arm using the minimum external force that the human body can endure, thereby ensuring the operator's safety. The virtual force control is utilized to implement a position and force hybrid controller. The experimental results demonstrate that, under identical control conditions, the position and force hybrid controller established by the Raspberry Pi single-board computer achieves superior control outcomes in a constant force control scenario with a pressure of 40 N. The average absolute error is 9.62 N, and the root mean square error is 11.16 N when compared to the target pressure. From the analysis of the results, it can be concluded that the Raspberry Pi system implemented in this paper can achieve a higher control command update rate compared to personal computers. As a result, it can provide greater control benefits in position and force hybrid control.

Keywords: robotic arm; Raspberry Pi; machine learning; safety collision detection; hybrid position/force control; EtherCAT

Citation: Hung, C.-W.; Jiang, G.-Y. Application of External Torque Observer and Virtual Force Sensor for a 6-DOF Robot. *Appl. Sci.* **2023**, *13*, 10917. https://doi.org/10.3390/app131910917

Academic Editors: Renato Vidoni, Andrea Giusti and Lorenzo Scalera

Received: 9 September 2023
Revised: 27 September 2023
Accepted: 28 September 2023
Published: 2 October 2023

1. Introduction

With the rise of the automation industry and Industry 4.0, robotic arms have become widely utilized in highly repetitive, dangerous, and high-precision tasks. They not only reduce labor costs but also improve production efficiency and maintain product processing quality. With the advancement of the automation industry, the introduction of tactile sensing into industrial robots will become one of the key developments after machine vision. Applications of tactile sensing include safety collision detection mechanisms during human–robot collaboration and the control of external forces applied to workpieces by equipment to improve processing accuracy.

Research and applications of robotic arms mostly revolve around utilizing external attachments such as cameras, pressure sensors, or force sensors to make judgments and decisions, perform object recognition and gripping, plan processing procedures, assemble processing components, and polish processed parts. Industrial computers are typically used to control robotic arms by sending control commands to motor drivers and receiving current information from the motors. The current position and speed of the arm can be determined through feedback from the motors. Manufacturers typically use current meters to measure the torque of each axis of the arm. However, the precise external contact force cannot be directly measured. For measuring the contact force of the end effector of a robotic arm, six-axis force sensors are commonly used. However, the price of six-axis force sensors is usually 30% to 50% of the price of the robotic arm. Additionally, industrial computers are generally more expensive than computers used by the general public. As a result, the overall cost becomes a significant burden. It is important to reduce the higher costs associated with industrial computers.

Using contact force sensors will increase production costs, while establishing a virtual force sensor based on feedback from the robotic arm. It is a cost-effective method. However, precise mechanical structure parameters are required, such as the weight, center of gravity, rotational inertia, and friction coefficient of each axis. However, suppliers of robotic arms often cannot or are unwilling to provide accurate parameters. Therefore, this paper proposes the use of machine learning to estimate torque and create a virtual force sensor. By comparing it with a virtual force sensor established using parameters provided by manufacturers, experiments were designed to verify the feasibility of safety collision detection and hybrid position/force control using the virtual force sensor.

The price of industrial computers is generally higher than that of computers used by the general public, which can impose a significant financial burden. In order to reduce the high cost of industrial computers, this paper proposes the use of a Raspberry Pi as an upper-level controller, in addition to utilizing personal computers with Windows systems. The Raspberry Pi has advantages such as improved compatibility and a lower price. Therefore, this paper chooses to implement a Real-Time Kernel on the Raspberry Pi. This approach not only fulfills the requirements of EtherCAT motion control for the upper-level controller, but also effectively reduces the cost of the upper-level controller.

2. Related Research

This paper aims to establish a position and force hybrid controller for the arm. To achieve this, an accurate dynamic model of the mechanical arm needs to be established. Traditional dynamic models are established using Lagrangian Mechanics or Newton–Euler Equations [1–6]. The accuracy of identifying unknown parameters is ensured through the design of excitation trajectories. Among the many parameter identification methods, the weighted least squares method and the extended Kalman filtering method are two commonly used methods. Ref. [1] compares these two methods and states that the parameter estimation of the two methods is very similar. However, the extended Kalman filtering algorithm has a longer convergence time and is highly sensitive to initial conditions. Refs. [2,3] propose parameterization and optimization criteria for excitation trajectories, which ensure the average nature of time-domain data and effectively measure noise. Ref. [4] introduces the artificial bee colony algorithm for estimating unknown parameters and verifies the dynamic model's ability to estimate the torque of the mechanical arm. The results show that the correlation coefficient between the predicted torque of the first axis and the actual torque is 0.9272. Ref. [5] obtains the best approximate solution using the newly established unknown parameter approximation method and compares it with the least squares method. The results show that it is superior to the least squares estimation method. Ref. [6] improves the cuckoo search algorithm for parameter identification by incorporating chaotic operators and emotional operators, which help the algorithm escape from local optimal solutions. Through the literature mentioned above, it has been confirmed that it is possible

to establish a traditional dynamic mathematical model by estimating unknown parameters using algorithms.

In addition to the traditional method of establishing dynamic models using mathematical equations, mentioned above, there has been a recent trend in establishing inverse dynamic models through machine learning with the advancement of artificial intelligence. It is stated that both traditional mathematical equations and machine learning can be used to establish inverse dynamics and achieve a certain degree of torque prediction. Compared to the traditional mathematical method, establishing a dynamic model requires a significant amount of knowledge, whereas machine learning relies on a sufficient amount of relevant information for modeling [7]. The proposal suggests utilizing physics simulation and deep learning techniques for the dynamic control of a robotic arm. This method does not require deriving the aforementioned traditional dynamic mathematical equations. The numerical data simulated by the physics engine are utilized as training data. It has been experimentally demonstrated that neural networks can accurately predict the torque of each joint, with a dynamic error controlled within 10% [8]. This meets the requirements for basic robotic arm dynamic control. XGBoost is a machine learning algorithm that has been developed in recent years. It is a framework based on decision trees. XGBoost is characterized by its fast speed, ability to handle large amounts of data, high flexibility, and support for regression functions. It is commonly used in data analysis and other situations [9]. This paper will use the aforementioned XGBoost to establish an inverse dynamic model of a six-axis robotic arm. Through machine learning, the goal is to analyze the nonlinear terms that cannot be solved by traditional mathematical methods when calculating inverse dynamics. This analysis aims to improve the accuracy of torque prediction. It will also compare the prediction results of the machine learning methods and the traditional mathematical methods mentioned above.

In recent years, there has been a shift in the application of robotic arm control towards emphasizing human–machine interaction and collaboration. Alongside simple position control, force control has gradually emerged as a prominent trend in robotic arm research [10–12]. Force sensors are used to measure the contact force at the end of the arm, and they are also utilized to compensate for and fine-tune the position commands. For trajectory tracking, another method is proposed in [13] to adjust the parameters of the internal position control loop based on sensing the force at the end of contact. The experimental results demonstrate that this method can ensure a stable device response. The virtual force sensor utilizes the inverse dynamics equation of the robotic arm to calculate the torque caused by external forces on each axis. It then converts this torque into the end contact force and torque of the arm in Cartesian coordinates using the Jacobian matrix. Using a Kalman filter to reduce torque noise from the driver is advantageous for estimating the external interference torque of each axis. This directly impacts the accuracy of the virtual force sensor [14–16]. Ref. [17] proposes that temperature influences frictional force. Experimental results demonstrate that the frictional torque produced by a preheated arm is reduced by approximately 33% during high-speed motion [18]. Taking into account the temperature effect mentioned earlier, a temperature effect model is incorporated into the friction term in the inverse dynamics to establish the virtual force sensor. When implementing polishing force control, the root mean square error of the external contact force is reduced from around 27.4 N to 21.4 N. After reviewing the relevant literature on force control and the virtual force sensor mentioned earlier, this paper will employ both traditional mathematical methods and machine learning techniques to compute the external disturbance torque derived from the inverse dynamics model in order to establish a virtual force sensor. This will be used to estimate the external contact force and perform position and force hybrid control. Additionally, the differences between the two methods will be compared.

To achieve real-time control of the robot, the EtherCAT system is used with its transmission rate of 100 Mbps to enable the real-time control of the robotic arm [19,20]. The experimental results show that using EtherCAT as the robot control system can provide a data exchange sampling rate of 10 kHz and a motion control position command control

frequency of 1 kHz [21]. Using the EtherCAT control software TwinCAT developed by Beckhoff and ESC as the master controller, and the PIC24H developed by Microchip as the application controller, we verified the feasibility of CANopen over EtherCAT (CoE) and the experimental results demonstrate that CoE is suitable for data collection and motion control [22]. Using the EtherCAT master module IgH, based on the Linux system, to implement the EtherCAT distributed clock can lead to resource preemption problems due to the real-time nature of EtherCAT tasks. As a solution, the Linux system is equipped with a real-time patch to enable the real-time processing of EtherCAT interrupt events. The experimental results show that the synchronization time of the Linux real-time operating system is more accurate than that of the Windows platform, with a standard deviation of approximately 0.236 ns [23]. In the experiment, the real-time performance of the Linux system with RTAI and Xenomai was compared. The results showed that both were superior to using the Linux system alone. It was also found that Xenomai had slightly lower performance than RTAI, but it had a superior architecture, better platform compatibility, and was more conducive to system portability [24]. Therefore, this paper will use EtherCAT for controlling a six-axis robotic arm. TwinCAT will be used on the personal computer, while Xenomai with IgH will be installed on the Raspberry Pi 4B Linux system as the main control platform.

3. Materials and Methods

3.1. System Architecture

3.1.1. Personal-Computer-Based

Figure 1 is the overall system architecture of using a personal computer with the Windows operating system as the computing platform. The communication of peripheral equipment, calculation of kinematics, establishment of the virtual force sensor, and the motion control are calculated on the platform. Robotics are responsible for calculating the posture of the robotic arm. The trajectory planning algorithm can generate smooth movement trajectories based on target points. The dynamic model calculation is performed using Python for machine learning. The dynamic model can calculate the ideal torque for each axis in the current posture based on the feedback of the arm's position and velocity. The results calculated through the dynamic model can be differentiated from the feedback motor torque of each axis to obtain the external torque of each axis. The external torque value can be converted into a virtual force value on the operating surface of the arm's end through the virtual force sensor algorithm. Finally, applications related to force can be performed based on the calculation results. The communication component utilizes EtherCAT to transmit commands and data for each axis of the six-axis robotic arm. In this architecture, the communication cycle is set to 2 ms due to the limitations of the Windows system's underlying clock.

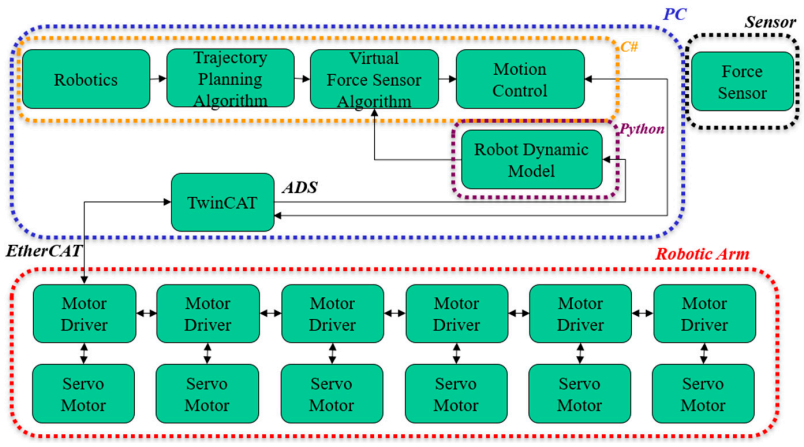


Figure 1. Architecture of the personal-computer-based system.

3.1.2. Raspberry Pi-Based

The system architecture which utilizes Raspberry Pi as the computing platform, based on the Linux system environment, is depicted in Figure 2. The user interface was created using the PyQt5 library in Python, while the algorithm program was implemented using the C programming language. Different functions of the program are assigned to independent cores, and data sharing is performed through Shared Memory, as shown in Figure 2. This reduces resource contention issues and improves the performance of real-time tasks. The commands and data transmission of the six-axis robotic arm are completed through IgH EtherCAT. Compared to the Windows system, the Linux system has a more precise system clock, allowing the motion control command update rate to be set at 4 kHz. Considering the computational complexity of the dynamic algorithm, the computation cycle needs to be approximately 0.8 ms. Therefore, the communication cycle is set to 1ms, which corresponds to a command update rate of 1 kHz.

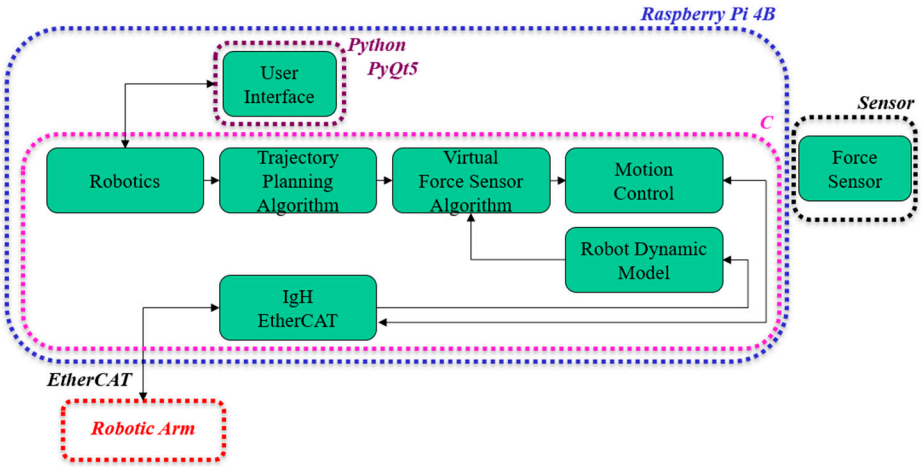


Figure 2. Architecture of Raspberry Pi-based system.

3.2. Machine Learning-Based Robot Dynamic Model

3.2.1. Robot Dynamic Model

The dynamic equations primarily calculate the torque exerted on the motor driver based on dynamic parameters, such as the position, velocity, and acceleration of the robotic arm. The commonly used methods for derivation are the Euler–Lagrange equation and the Newton–Euler equation. This paper utilizes Lagrange's equation of motion to solve the dynamics, as depicted in Equation (1). In this equation, τ is the calculated ideal torque, q is the angle of each axis of the motor, $\dot{q}$ is the angular velocity, $\ddot{q}$ is the angular acceleration, M represents the inertia matrix of the robotic arm, C represents the torque vector due to Coriolis force, G represents the torque vector due to gravity force, and F represents the torque vector due to friction force.

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

Building a mathematical model for dynamics requires several internal parameters of the robotic arm, including the mass of each axis, the position of the center of mass relative to each axis coordinate system, and the inertia tensor matrix. Most of the information mentioned above consists of parameters considered confidential by the manufacturer. Therefore, regression analysis is commonly used in the literature for system identification. By designing various stimulation trajectories, it aims to analyze unknown parameters, which necessitates intricate calculations and results in low accuracy. The robotic arm used in this paper has internal parameters provided by the manufacturer that are relevant to the study. Therefore, both the dynamics equation and machine learning techniques can be employed to establish the dynamic model of the robotic arm. A comparison will be conducted between the two approaches.

3.2.2. Robot Friction Model

Through the mathematical equations of dynamics, the ideal torque of the arm can be calculated based on its current posture and velocity. However, friction is an inevitable factor in real environments. Therefore, an additional mathematical equation for friction needs to be established in order to accurately estimate the torque of each axis. The method for establishing the friction equation is to collect the actual torque of each axis during constant velocity motion and calculate it with the estimated values calculated by the dynamic model. Then, based on the calculated friction, a trend chart related to velocity can be drawn, such as in Figure 3. According to the trend chart, there is a correlation between the velocity of each axis and the friction value. By analyzing the relationship between the velocity and friction of each axis motor in both the forward and reverse directions, it is possible to deduce the linear equation for the friction of each axis motor. It was found that the friction levels in the forward and reverse directions of the motor were not equal.

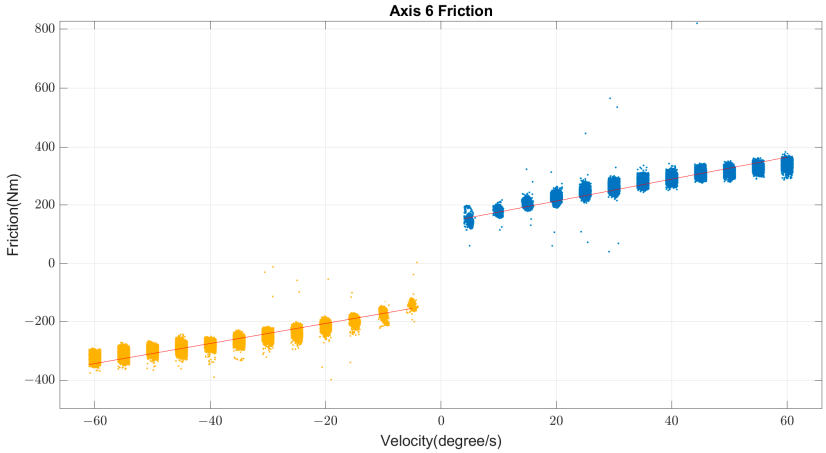


Figure 3. Illustration of the frictional force regression curve.

This paper aims to convert the approximate straight lines obtained from Figure 3 into mathematical formulas. These formulas will then be used as transformation formulas for calculating friction force and velocity. By combining them with the dynamics equation, it is possible to estimate the torque on each axis, including the friction force.

3.2.3. Machine Learning Model

In addition to traditional mathematical methods, machine learning is another option for obtaining the complete parameters of the arm, especially when the internal parameters are known. Due to the poor accuracy of estimating internal parameters through parameter identification, there has been a gradual emergence in the domestic and foreign literature of the use of machine learning methods to address the issue of nonlinear terms in inverse dynamics, which traditional mathematical methods struggle to solve. It is hoped that by using a large amount of training data, the model can produce better results. This paper will utilize Extreme Gradient Boosting (XGBoost) to develop inverse dynamics models and compare the time consumption and accuracy of torque estimation.

XGBoost is a model based on the improvement of Gradient Boosting Decision Tree (GBDT) [9]. Both XGBoost and GBDT belong to the ensemble learning algorithm known as Boosting. It utilizes multiple classifiers to make predictions on the dataset and combines the results of these classifiers through iterative calculations to obtain the final output. As both are boosting models, XGBoost inherits the advantages of GBDT. GBDT finds the optimal solution in the previous training and continues to split new trees based on this, thereby improving the overall accuracy of the model. The difference between XGBoost and GBDT is that GBDT uses the negative gradient of the loss function to construct the tree for the current round, while XGBoost utilizes the second-order approximation of the loss function to expedite the descent of the loss function and achieve a faster iteration

speed. XGBoost also organizes all features before training and stores each feature point in separate memory blocks. This allows for the simultaneous traversal of leaf nodes and data features for splitting points when building a new tree. As a result, parallelization is achieved, leading to improved training speed.

Due to the multi-link nature of the six-axis robotic arm, the torque output of each motor in the arm's axes is influenced by the angle and angular velocity of any given axis. Therefore, the dynamic model established in this paper, as shown in Figure 4, consists of six independent models for each axis, with 12 inputs and 1 output. The inputs for the robotic arm are the angles and angular velocities of each axis' motors, while the output is the corresponding torque of the motor for each axis.

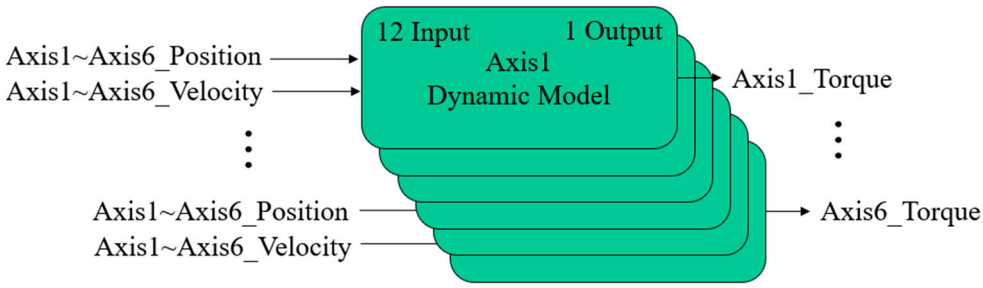


Figure 4. Architecture of the dynamic model based on machine learning.

3.2.4. Dataset Collection

From the inverse dynamics equation described in the previous section, it can be seen that the torques of the six-axis robotic arm are interdependent. Therefore, in order to establish a machine learning model, it is necessary to collect the dynamic information of each axis, including the position, angular velocity, angular acceleration, and motor output torque of the arm during trajectory motion. The angular acceleration is not directly obtainable data and needs to be obtained by differentiating the angular velocity. In order to incorporate diversity in the training data of trajectories, this paper will utilize a random trajectory generation algorithm to enable random arm movements within the workspace. By adjusting the trajectory movement time to increase velocity and acceleration changes, the training data will become more varied. Finally, the high data exchange rate feature of the EtherCAT communication protocol can be utilized to extract the dynamic parameters of each axis.

The algorithm will generate the new target point within the work at random and plan the trajectories through the cubic polynomials in the joint space, to make the dataset represent the features in each posture and motion adequately. It is important to check whether the operation is in working space for all the points in the trajectory after designing the training trajectory to ensure that the arm will not collide during the movement.

From the inverse dynamics equation described in the previous section, it can be seen that the torques of the six-axis robotic arm are interdependent. Therefore, the position, angular velocity, and angular acceleration of each axis are important parameters for training the dynamic model. In order to align the trajectory with the diversity of training data, the movement time for each segment of the trajectory was set to 3 s, 5 s, and 8 s. This experiment collected a total of 20 million data points for the angle and angular velocity of each axis.

3.2.5. Hyperparameter Optimization

The inputs of the model are the angle and angular velocity of each axis, and the definition of the loss function is Mean Square Error (MSE), as shown in Equation (2). Standardization is used to preprocess the input data, and the best hyperparameters are searched through Bayesian Optimization to reduce the time spent in Grid Search. The

range of hyperparameters is shown as Table 1, and this paper tests and analyzes the results of different hyperparameter combinations.

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

Table 1. The range of hyperparameters.

Hyperparameter	Range
Max_depth	1~15
Eta	0.2, 0.1, 0.05, 0.01
Min_child_weight	0~6
Subsample	0.1~1
Colsample_bytree	0.1~1
Gamma	0~1

The top 3 model hyperparameter combinations after completing Bayesian optimization are shown in Table 2. In Table 2, the minimum MSE value was 0.00158. Therefore, this paper chose this set of hyperparameter combinations to train the dynamic model.

Table 2. Bayesian optimization results.

Item\Rank	1	2	3
Max_depth	10	10	8
Eta	0.01	0.1	0.1
Min_child_weight	0	5	0
Subsample	0.79	0.92	0.69
Colsample_bytree	0.73	0.96	0.45
Gamma	0.08	0.31	0.63
MSE	0.00158	0.00161	0.00164

3.3. Virtual Force Sensor

The external toque observer is established by the dynamic model for each axis, $\tau_{a\ 6*1}$ is the actual torque produced by the driver, $\tau_{pred\ 6*1}$ is the torque of the arm when there is no load estimated by the model, and the external torque $\tau_{e\ 6*1}$ is obtained by the external toque observer. After $\tau_{e\ 6*1}$ has been obtained, $\tau_{e\ 6*1}$ is converted into the external force applied to the endpoint of the robotic arm through the inverse Jacobian transpose matrix $(\mathrm{J}')^{-1}$ and the virtual force sensor for the robotic arm is established in this way. Figure 5 is the flowchart of the virtual force sensor algorithm.

$$\tau_{e\ 6*1} = \tau_{a\ 6*1} - \tau_{pred\ 6*1} \tag{3}$$

$$Force_{6*1} = \begin{bmatrix} f_x \\ f_y \\ f_z \\ f_\varnothing \\ f_\theta \\ f_\varphi \end{bmatrix} = (\mathrm{J}')^{-1} \begin{bmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \\ \tau_4 \\ \tau_5 \\ \tau_6 \end{bmatrix} \tag{4}$$

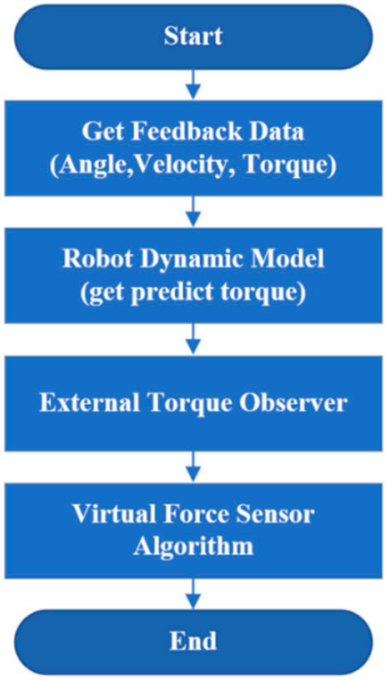


Figure 5. Flowchart of the virtual force sensor algorithm.

3.4. Hybrid Position/Force Control Architecture

The architecture of the proposed position and force hybrid controller is shown in Figure 6. Position control was achieved by specifying a target position and executing Cartesian-space trajectory planning. The calculated trajectory points are coordinate values, so they need to be converted into motor angle commands using inverse kinematics. After obtaining the motor trajectory commands, motion is carried out using Cyclic Synchronous Position (CSP) control mode. The position control, velocity control, and torque control in Figure 6 are all internal loops provided by the motor driver. The actual position control error will be compensated in the next command cycle through error compensation mechanisms. The force control component compares the feedback torque from each axis motor with the estimated torque from the dynamic model and calculates the external torque. A virtual force sensor is established to calculate the estimated end force and target force, using the principles of the virtual force sensor. The force error is then converted into variations in each joint of the robotic arm for force control.

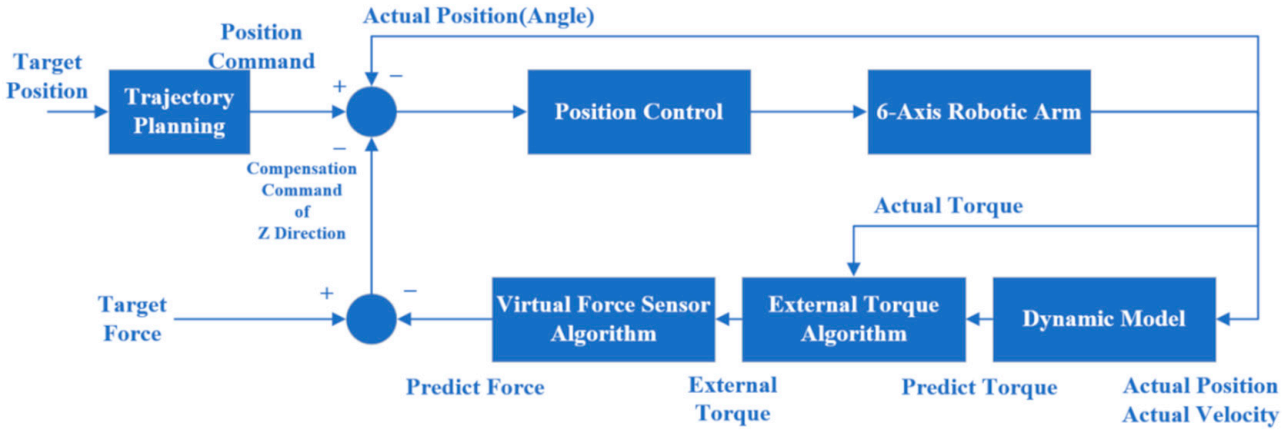


Figure 6. Architecture of hybrid position/force control.

4. Experiment and Results

After establishing the virtual force sensor, this paper applied the virtual force sensor to external contact force detection and force control. In the application of external contact force detection, this paper designed a safety collision detection experiment to verify the feasibility of applying the virtual force sensor to detect the external contact force. In the application of the force control, the hybrid position/force control was applied to the grind experiment of constant force control to verify that the virtual force sensor can be applied to the constant force control.

4.1. Verification of the Robot Dynamic Model Based on PC

In this paper, the best hyperparameter combination of XGBoost was selected for model establishing, and to predict and verify the test set for the model. A total of 100 random motion trajectories were used as a test set for evaluating the best model in this experiment. To verify the precision of the best model, Root Mean Square Error, as shown in Equation (5), Mean Absolute Error, as shown in Equation (6), and correlation coefficient γ of prediction torque τ_{pi} and actual toque τ_i were applied to estimate the model, as shown in Equation (7). In Equation (7), $\bar{\tau} = \frac{1}{n}\sum_{i=1}^{N} \tau_i$ and $\bar{\tau}_p = \frac{1}{n}\sum_{i=1}^{N} \tau_{pi}$.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\tau_i - \tau_{pi})^2} \tag{5}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{(\tau_i - \tau_{pi})}{\tau_i}\right| \tag{6}$$

$$r = \frac{\sum_{i=1}^{N}\left(\tau_i - \bar{\tau}\right)\left(\tau_{pi} - \bar{\tau}_p\right)}{\sqrt{\sum_{i=1}^{N}\left(\tau_i - \bar{\tau}\right)^2 \sum_{i=1}^{N}\left(\tau_{pi} - \bar{\tau}_p\right)^2}} \tag{7}$$

The analysis of the dynamic model based on XGBoost model for each axis is shown in Table 3. As seen in Table 3, the correlation coefficient of prediction torque and actual toque was about 0.99 and the model took about 1ms to predict the torque of the six-axis. Figure 7 compares the predict torque (estimation torque) and the actual torque (feedback torques) for each axis.

Table 3. Analysis of the dynamic model.

	MAE (Nm)	RMSE (Nm)	γ
Axis 1	15.5489	23.9275	0.9859
Axis 2	6.7931	12.4564	0.9876
Axis 3	4.2617	6.3248	0.9872
Axis 4	1.1564	1.6451	0.9923
Axis 5	0.5816	0.8051	0.9906
Axis 6	0.1695	0.2280	0.9967

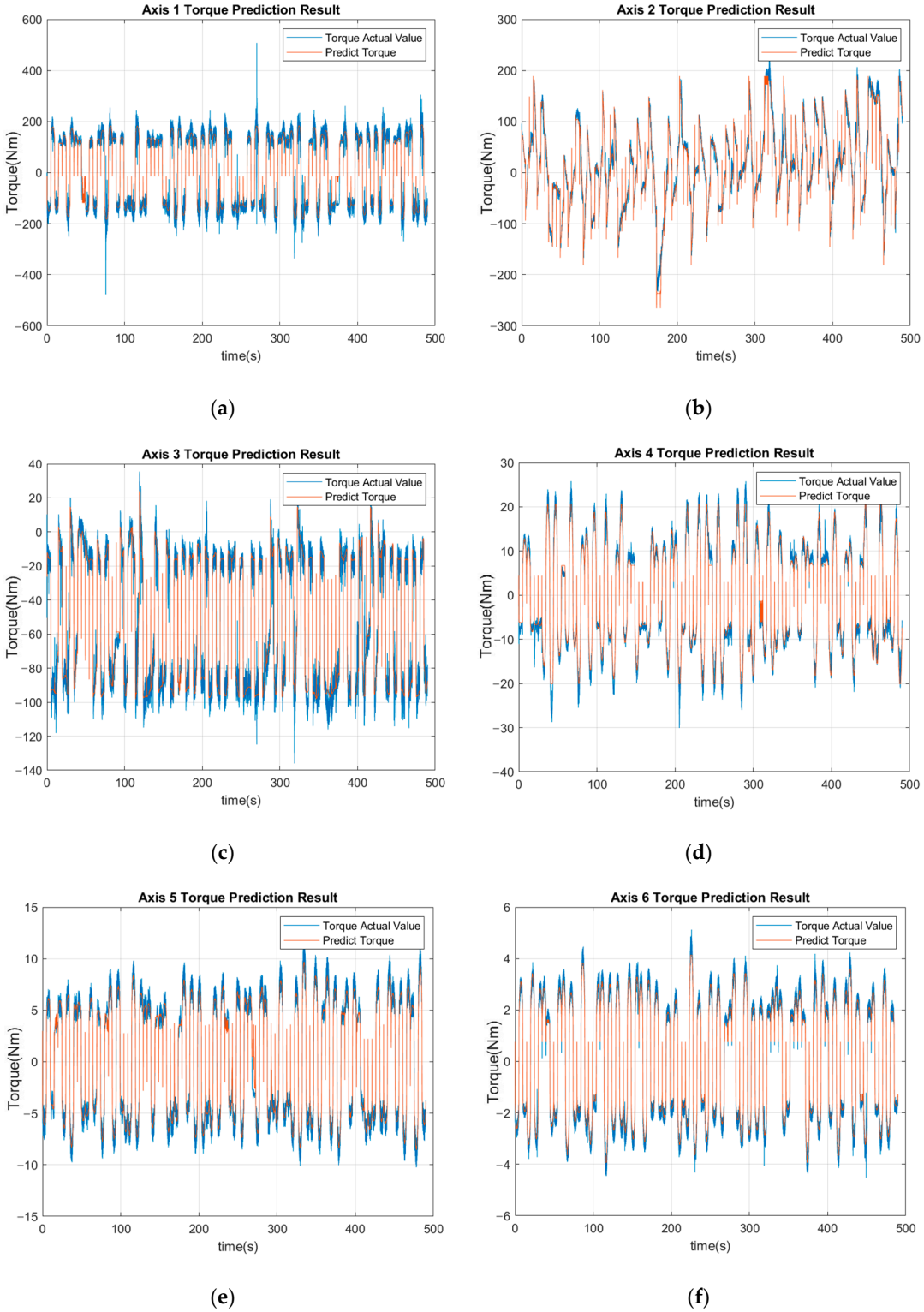


Figure 7. Comparison of the feedback torques and estimation (prediction) torques: (**a**) Axis 1; (**b**) Axis 2; (**c**) Axis 3; (**d**) Axis 4; (**e**) Axis 5; (**f**) Axis 6.

4.2. Verification of the Robot Dynamic Model Based on Raspberry Pi

This section will validate the dynamic model established using Raspberry Pi. Due to the lower computational power of Raspberry Pi compared to personal computers, there may be insufficient computational power when running the AI model. Therefore, the dynamic model of the Raspberry Pi system was conducted using a pure mathematical model,

and the Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and correlation coefficient between actual torque and estimated torque were evaluated. Figure 8 compares the estimated torque and actual torque during the actual test. By organizing the results obtained from Figure 8, the evaluation table in Table 4 was created. It can be observed that, with the exception of the correlation coefficient of the third axis, which was lower than 0.98, the correlation coefficients of the remaining axes were all higher than 0.988. Although the accuracy was lower than using the XGBoost model on a personal computer, the results were still within an acceptable range.

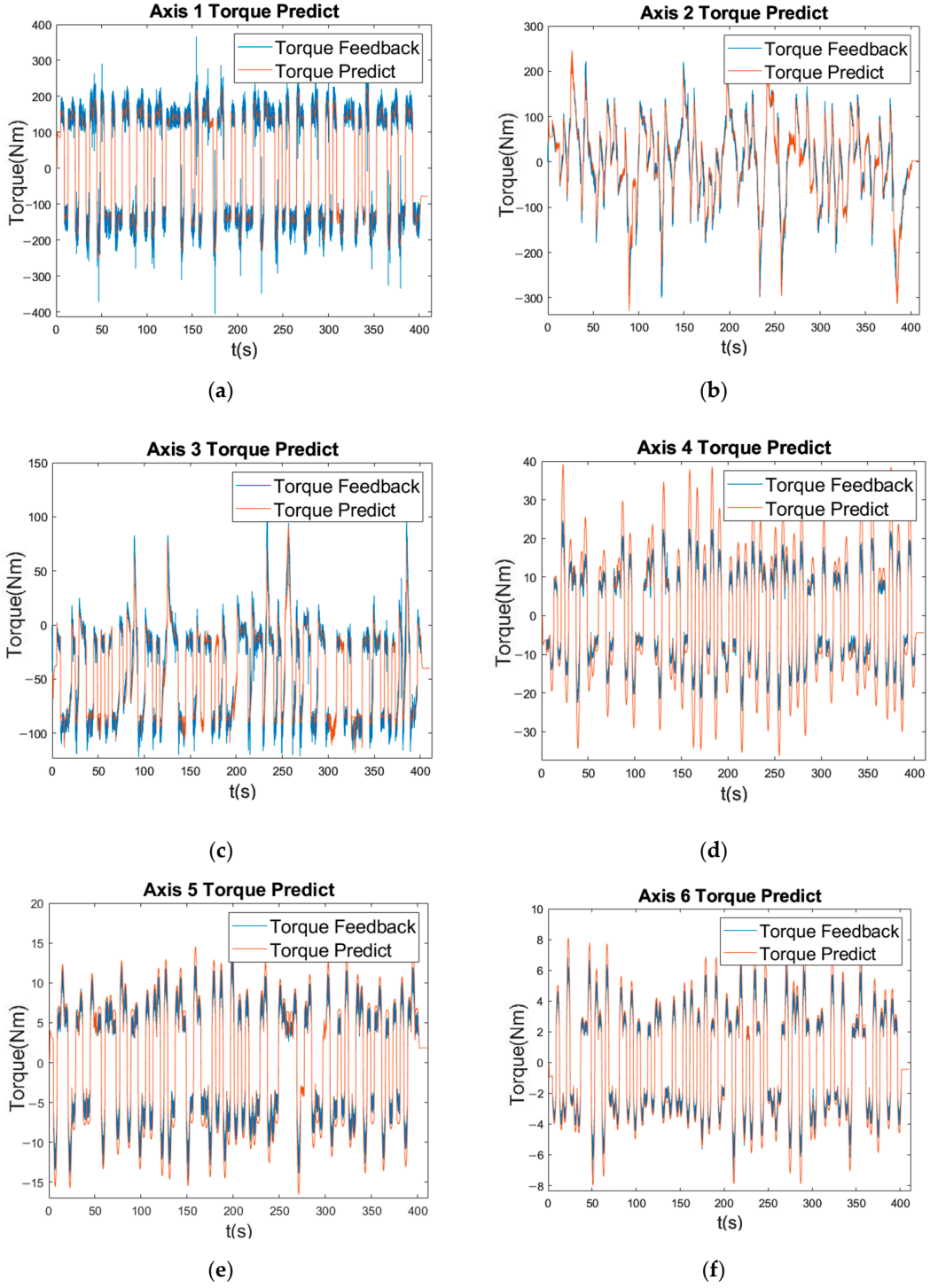


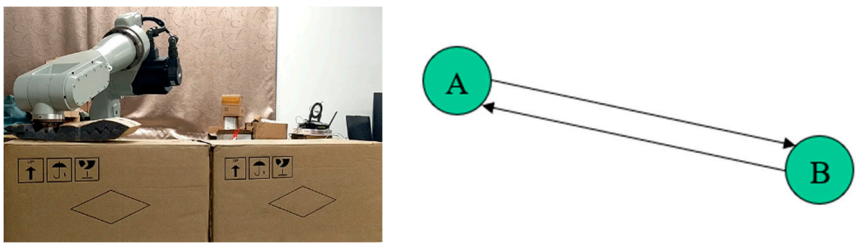
Figure 8. Comparison of the feedback torques and estimation (prediction) torques: (**a**) Axis 1; (**b**) Axis 2; (**c**) Axis 3; (**d**) Axis 4; (**e**) Axis 5; (**f**) Axis 6.

Table 4. Analysis of the dynamic model.

	MAE (Nm)	RMSE (Nm)	γ
Axis 1	14.7371	19.9986	0.9915
Axis 2	9.4744	14.1449	0.9895
Axis 3	6.2997	9.6843	0.9770
Axis 4	5.0730	6.2901	0.9894
Axis 5	1.8219	2.0906	0.9940
Axis 6	0.6542	0.8158	0.9964

4.3. Verification of Virtual Force Sensor

To verify the virtual force sensor, this paper analyzed the external force exerted by the robotic arm through the experiment of grind motion, such as in Figure 9. The experiment plan was to grind from point A to point B in Figure 9b on the XY plane after touching the platform. And the trajectory was planned at the speed of 100 mm/s. After the virtual force sensor was established through the external torque observer, the MAE, RMSE, and correlation coefficients of the virtual force sensor and the force sensor were evaluated in the above experiment. The analysis of the virtual force senor is shown in Table 5. From Table 5, it can be seen that the correlation coefficient of X was approximately 0.71, the correlation coefficient of Y was approximately 0.73, the mean absolute errors (MAE) of X and Y were both around 20 N, the correlation coefficient of Z was approximately 0.75, the MAE was around 9.37 N, and the root mean square error (RMSE) was around 12.93N. Furthermore, Figure 10 shows that the errors and noise generated during the estimation of the external force in the Z direction were relatively small. This is advantageous for our application in pressure sensing and control. Therefore, this paper conducted experiments using the virtual force sensor of the robotic arm to explore its potential for pressure control-related applications and to verify its applicability in the field of force control.



(**a**) (**b**)

Figure 9. (**a**) Grinding experiment; (**b**) motion trajectory of the grinding experiment.

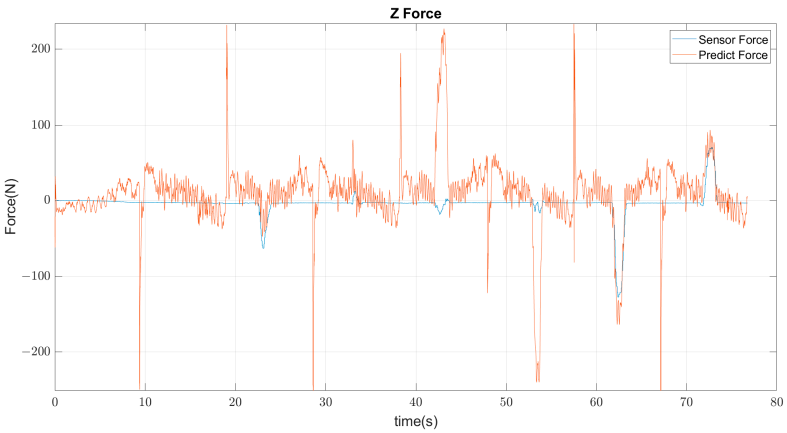


Figure 10. Comparison of the external force between senor and virtual sensor in the Z direction.

Table 5. Analysis of the virtual force sensor.

	MAE (N)	RMSE (N)	γ
F_x	20.0838	24.6242	0.7107
F_y	20.8401	30.0805	0.7325
F_z	9.3788	12.9381	0.7542

Based on the above arguments, the system could not accurately estimate the external contact force of 20N or lower because the virtual force sensor based on the dynamic model was limited by the accuracy of the model, the noise of the driver and the influence of the inaccurate estimation of the static state. Therefore, this system is only suitable for detecting external contact force and lower precision force control applications.

4.4. Safety Collision Detection

In the safety collision detection experiment, the external force was detected through the virtual force sensor, and the movement of robotic arm would stop when the external force exceeded the safe range of external force of the human body to ensure the safety of the human when operating the equipment. In Table 6, BG/BGIA Risk Assessment Recommendations According to Machinery Directive: Design of Workplaces with Collaborative Robots [25] suggest the minimum external force and pressure that the body region can withstand.

Table 6. The minimum external force and pressure that the body region can withstand.

	Skull	Face	Neck (Slides)	Neck (Front)	Shoulder	Chest	Arm	Leg
CSF (N)	130	65	145	35	210	140	160	140
IMF (N)	175	90	190	35	250	210	220	170
PSP (N/cm^2)	30	20	50	10	70	45	50	45
CC (N/mm)	150	75	50	10	35	25	40	60

Clamping/squeezing force (CSF) is the external force affecting the body region at more than 0.5s; impact force (IMF) is the difference between the maximum force and the forces before and after the maximum when it is more than 5 N over a time interval of 0.5 s or less; pressure/surface pressing (PSP) is the pressure generated under the contact area of the collision; and compression constant (CC) is the deformation constant of each region of the body.

According to Table 6, the smallest external force that the front side of the human neck can withstand is about 35 N. Therefore, the safety collision detection mechanism of the experiment was defined through the table. The flowchart of the safety collision detection algorithm is shown in Figure 11; when the external force is higher than the threshold for 10 ms, it will be considered as a collision and the movement of the robotic arm will be stopped.

Figure 12 shows the experiment of collision detection with safety control. According to the result, the external force was detected at 5.11 s and the system judged that the collision occurred at 5.31 s. The safety collision mechanism was triggered within 10 ms after the external force occurred, and the movement stopped immediately. The result shows that the system conformed to the safety specifications suggested by [25] and the virtual force sensor can be applied in safe collision detection.

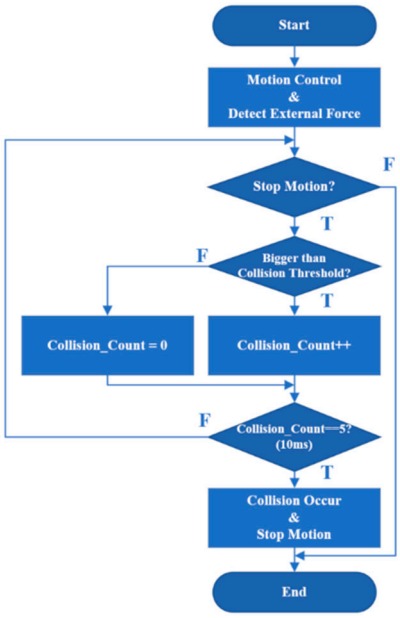
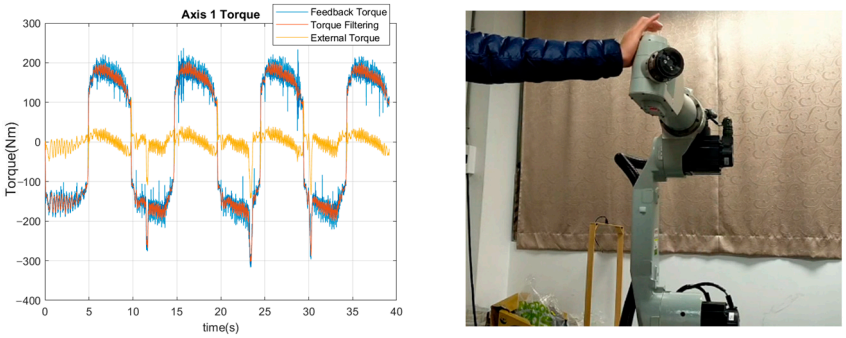


Figure 11. Flowchart of the safety collision detection algorithm.

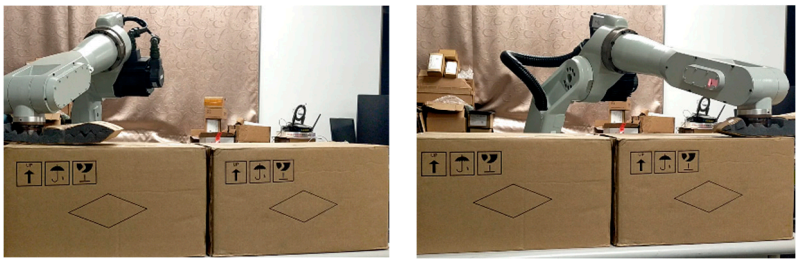


(**a**) (**b**)

Figure 12. (**a**) Results of collision detection with safety control; (**b**) safe collision detection.

4.5. Hybrid Position/Force Control Experiment Result

In the experiment on force control, the hybrid position/force control was applied to the grinding motion and the position and constant force were controlled during the grinding motion. The experiment planned to grind on the XY plane after touching the platform (see Figure 13). The trajectory was planned at the speed of 100 mm/s, the position control was performed on the X and Y directions and the F_z applied to the Z direction was controlled by a constant force controller.



(**a**) (**b**)

Figure 13. (**a**) Contact situation; (**b**) contact and force control.

This experiment validated the feasibility of the hybrid controller by utilizing a virtual force sensor to execute the mixed control of position and force. The experiment planned to validate the hybrid controller through a reciprocating grinding experiment. The experiment first compared the results of controlling pressure under the same target using a personal computer (see Figure 14) and a Raspberry Pi (see Figure 15). The MAE (Mean Absolute Error) for the personal computer was 9.2406 N, and the RMSE (Root Mean Square Error) was 14.6020 N. The mean absolute error (MAE) for the Raspberry Pi was 9.6231 N, and the root mean square error (RMSE) was 11.1620 N. It can be seen that compared to the personal computer, the results of the Raspberry Pi control had a more impressive RMSE. This is because the Raspberry Pi system established in this paper could provide a higher control command update rate compared to the personal computer, resulting in higher control efficiency. Therefore, this paper continued to use the Raspberry Pi to control under different pressure conditions and evaluate the results.

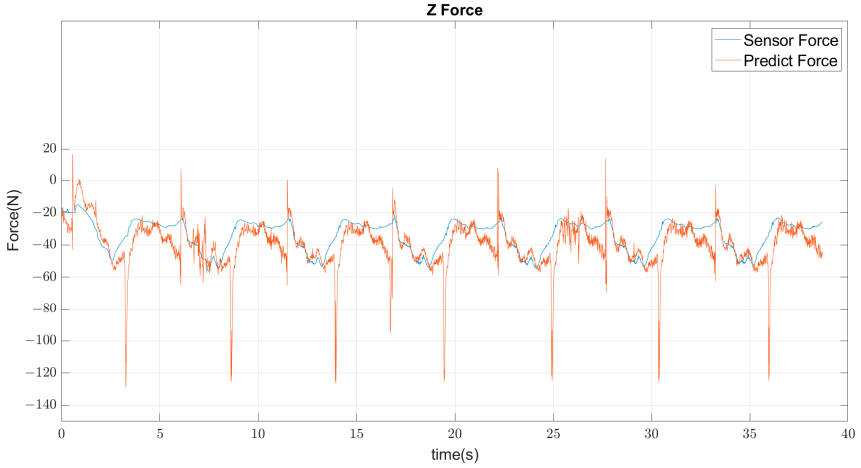


Figure 14. Results of the contact force control experiment on PC.

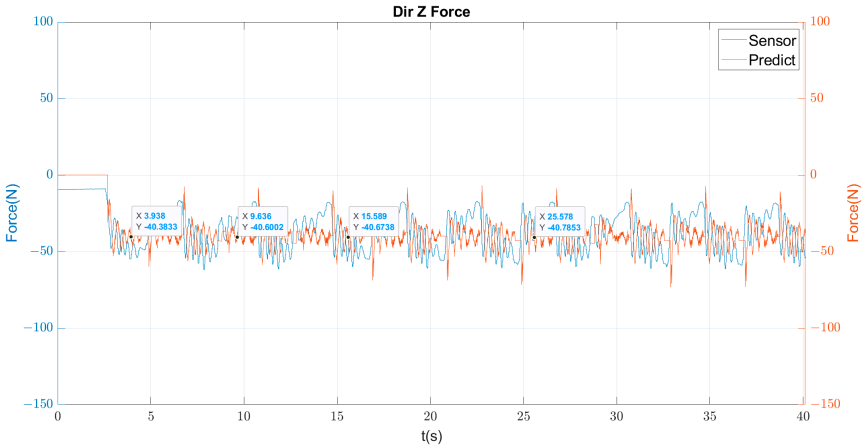


Figure 15. Results of the contact force control experiment on Raspberry Pi.

The external forces applied to the Z direction F_z = 20, 30, 40 were compared in this experiment. In this experiment, the force sensor and the target constant force MAE and RMSE were evaluated for the force controller part, and the RMSE of the X and Y directions with or without the position controller were evaluated for the position controller part.

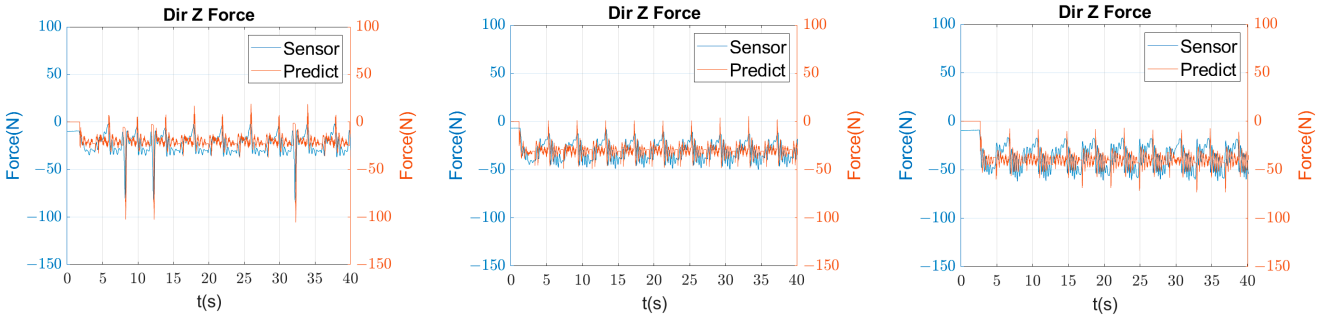
Table 7 gives an evaluation of the effect of Raspberry Pi controlling different compressive pressures. It can be seen that the experimental results generated for 30 N and 40 N are superior to those of the other two groups, with more accurate virtual force values and better control force. From the experimental results, it can be seen that although the control accuracy of the force controller is limited by the accuracy of the virtual force value,

this experimental result can still meet the low-precision force control requirements for different forces.

Table 7. Comparison of the force sensor and virtual force sensor.

	MAE (N)	RMSE (N)
$F_z = 20$	11.1700	16.2204
$F_z = 30$	9.0563	10.5321
$F_z = 40$	9.6231	11.1620

Figure 16 shows the results of the contact force control experiment, and we can see that when the robotic arm moved to the turning point and the speed approached zero, the virtual force sensor was inaccurate, as in the previous experiment. Therefore, the force controller proposed in this system avoids the noise of inaccurate prediction at the turning point.



(**a**) (**b**) (**c**)

Figure 16. Results of the contact force control experiment: desired forces of 20 N (**a**), 30 N (**b**) and 40 N (**c**).

Based on the above experiments, the results generated by applying this system to Raspberry Pi were better. The reason is that the Raspberry Pi system established in this paper could provide a higher control command update rate compared to a personal computer. It also achieved higher control efficiency and verified the feasibility of applying this system to lower precision force control.

5. Conclusions

This article utilized personal computers and Raspberry Pi to create a virtual force sensor for a six-axis robotic arm. It detected external contact forces using a virtual force sensor for applications related to force control. The experiment confirmed that this system can establish an external torque observer and virtual force sensor for a six-axis robotic arm using a dynamic model with both the Raspberry Pi and personal computers. It was applied for force detection, and collision detection mechanisms were implemented using an external torque observer. The position and force hybrid control framework was achieved using the virtual force sensor.

In terms of the dynamic model, this article compared the torque estimation effects of traditional mathematical models and machine learning models using personal computers and Raspberry Pi. The experimental results demonstrate that the correlation coefficient between the torque estimated by the dynamic model established using the mathematical model on the Raspberry Pi system and the actual torque was approximately 0.985. This value is higher than the correlation coefficient of the dynamic model established using the traditional mathematical model [4]. These findings provide evidence for the feasibility of torque estimation using the mathematical model. The correlation coefficient between the torque estimated by the machine learning model established using personal computers and

the actual torque was approximately 0.988. This is higher than the correlation coefficient of the dynamic model established through the mathematical model on the Raspberry Pi system. This result demonstrates the feasibility of establishing a dynamic model through machine learning. The external torque observer and virtual force sensor were established through machine learning, and the accuracy of the virtual force sensor was affected by motor driver noise and model accuracy. The correlation coefficient of the estimated contact force was approximately 0.75, and the RMSE was approximately 13 N, which is better than the results obtained in [17,18].

Through the collision detection experiment of the robotic arm, it was verified that the external torque observer established through the dynamic model in this system could stop the motion of the robotic arm within the recommended safety range in [25], improving the safety of human–robot collaboration.

In the position and force hybrid control experiment, the control results of using the machine learning model to establish the virtual force sensor in personal computers and using the mathematical model to establish the virtual force sensor in Raspberry Pi were compared. From the experimental results, it can be observed that when the controlled contact force was 40 N, the utilization of the machine learning model on personal computers yielded control results with a Mean Absolute Error (MAE) of approximately 9.24 N and a Root Mean Square Error (RMSE) of approximately 14.60 N. On the other hand, when using Raspberry Pi, the MAE was approximately 9.62 N and the RMSE was approximately 11.16 N. It can be concluded from the experimental results that the position and force hybrid controller established using Raspberry Pi could achieve superior control outcomes. In subsequent fixed force control with different target contact forces, it was found that this system had better performance when controlling a fixed force of 30 N and 40 N. The accuracy of fixed force control not only depends on the estimation accuracy of the dynamic model but also on the command update rate of the controller. The feasibility of applying the position and force hybrid control framework proposed in this paper to lower-precision force control domains was also verified.

This paper successfully developed a virtual force control for a six-axis robotic arm using both personal computers and Raspberry Pi. It also established a position and force hybrid control framework. However, we were unable to achieve high-precision force control applications. Therefore, the following improvements are proposed based on the results of this paper: It has been observed that the friction force in the low-speed range differed from that during normal arm movement. This is because the system calculates the friction force equation by comparing the dynamic equation with the actual torque. Therefore, in the future, when establishing the dynamic model, a low-speed model can be added to improve the overall prediction accuracy. This paper utilized Raspberry Pi with Linux and real-time systems to establish an EtherCAT upper control platform with a higher command update rate. However, due to the limitations of Raspberry Pi's specifications, it was not possible to apply the machine learning model used on personal computers to Raspberry Pi. Therefore, in the future, the complexity of the machine learning algorithm could be reduced by adjusting the machine learning model, enabling Raspberry Pi to handle the computational load of the entire system.

Author Contributions: C.-W.H. and G.-Y.J. initiated and developed ideas for this research, developed the presented novel methods, derived relevant formulations, and performed performance analyses of simulation and experimental results. G.-Y.J. wrote the paper draft under the guidance of C.-W.H., who finalized the paper. All authors have read and agreed to the published version of the manuscript.

Funding: This study was partly supported by the Ministry of Science and Technology, Taiwan, under Contract NSTC 111-2622-E-224-013, 111-2221-E-224-052, 112-2221-E-150-027, and IRIS "Intelligent Recognition Industry Service Research Center" from The Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by the Ministry of Education (MOE) in Taiwan.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Gautier, M.; Poignet, P. Extended Kalman filtering and weighted least squares dynamic identification of robot. *Control. Eng. Pract.* **2001**, *9*, 1361–1372. [CrossRef]
2. Swevers, J.; Ganseman, C.; Tukel, D.B.; de Schutter, J.; Van Brussel, H. Optimal robot excitation and identification. *IEEE Trans. Robot. Autom.* **1997**, *13*, 730–740. [CrossRef]
3. Swevers, J.; Verdonck, W.; De Schutter, J. Dynamic Model Identification for Industrial Robots. *IEEE Control Syst.* **2007**, *27*, 58–71.
4. Ding, L.; Wu, H.; Yao, Y.; Yang, Y. Dynamic Model Identification for 6-DOF Industrial Robots. *J. Robot.* **2015**, *2015*, 9. [CrossRef]
5. Olsen, M.; Petersen, H. A new method for estimating parameters of a dynamic robot model. *IEEE Trans. Robot. Autom.* **2001**, *17*, 95–100. [CrossRef]
6. Ding, L.; Li, X.; Li, Q.; Chao, Y. Nonlinear friction and dynamical identification for a robot manipulator with improved cuckoo search algorithm. *J. Robot.* **2018**, *2018*, 1–10. [CrossRef]
7. Bargsten, V.; de Gea Fernandez, J.; Kassahun, Y. Experimental Robot Inverse Dynamics Identification Using Classical and Machine Learning Techniques. In Proceedings of the ISR 2016: 47st International Symposium on Robotics, Munich, Germany, 21–22 June 2016; pp. 1–6.
8. Liang, B.; Li, T.; Chen, Z.; Wang, Y.; Liao, Y. Robot Arm Dynamics Control Based on Deep Learning and Physical Simulation. In Proceedings of the 2018 37th Chinese Control Conference (CCC), Wuhan, China, 25–27 July 2018; pp. 2921–2925.
9. Chen, T.; Guestrin, C. XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 785–794.
10. Lange, F.; Bertleff, W.; Suppa, M. Force and trajectory control of industrial robots in stiff contact. In Proceedings of the 2013 IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, 6–10 May 2013; pp. 2927–2934.
11. Gan, Y.; Duan, J.; Chen, M.; Dai, X. Multi-robot trajectory planning and position/force coordination control in complex welding tasks. *Appl. Sci.* **2019**, *9*, 924. [CrossRef]
12. Lange, F.; Jehle, C.; Suppa, M.; Hirzinger, G. Revised force control using a compliant sensor with a position controlled robot. In Proceedings of the 2012 IEEE International Conference on Robotics and Automation, St Paul, MI, USA, 14–18 May 2012; pp. 1532–1537.
13. Murakami, K.; Ishimoto, K.; Senoo, T.; Ishikawa, M. Robot Hand Interaction Using Plastic Deformation Control with Inner Position Loop. In Proceedings of the 2020 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), Boston, MA, USA, 6–9 July 2020; pp. 1748–1753.
14. Wahrburg, A.; Morara, E.; Cesari, G.; Matthias, B.; Ding, H. Cartesian contact force estimation for robotic manipulators using Kalman filters and the generalized momentum. In Proceedings of the 2015 IEEE International Conference on Automation Science and Engineering (CASE), Gothenburg, Sweden, 24–28 August 2015; pp. 1230–1235.
15. Zeng, F.; Xiao, J.; Liu, H. Force/Torque Sensorless Compliant Control Strategy for Assembly Tasks Using a 6-DOF Collaborative Robot. *IEEE Access* **2019**, *7*, 108795–108805. [CrossRef]
16. Wahrburg, A.; Bos, J.; Listmann, K.D.; Dai, F.; Matthias, B.; Ding, H. Motor-Current-Based Estimation of Cartesian Contact Forces and Torques for Robotic Manipulators and Its Application to Force Control. *IEEE Trans. Autom. Sci. Eng.* **2018**, *15*, 879–886. [CrossRef]
17. Simoni, L.; Beschi, M.; Legnani, G.; Visioli, A. Friction modeling with temperature effects for industrial robot manipulators. In Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 28 September–October 2015; pp. 3524–3529.
18. Simoni, L.; Villagrossi, E.; Beschi, M.; Marini, A.; Pedrocchi, N.; Tosatti, L.M.; Legnani, G.; Visioli, A. On the use of a temperature based friction model for a virtual force sensor in industrial robot manipulators. In Proceedings of the 2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Limassol, Cyprus, 1–12 September 2017; pp. 1–6.
19. Liu, Z.; Liu, N.; Zhang, T.; Cui, L.; Li, H. EtherCAT based robot modular joint controller. In Proceedings of the 2015 IEEE International Conference on Information and Automation, Lijiang, China, 8–10 August 2015; pp. 1708–1713.
20. Jung, I.K.; Lim, S. An EtherCAT based control system for human-robot cooperation. In Proceedings of the 2011 16th International Conference on Methods & Models in Automation & Robotics, Miedzyzdroje, Poland, 22–25 August 2011; pp. 341–344.
21. Jung, I.-K.; Lim, S. An EtherCAT based real-time centralized soft robot motion controller. In Proceedings of the 2012 International Symposium on Instrumentation & Measurement, Sensor Network and Automation (IMSNA), Harbin, China, 18–20 May 2012; Volume 1, pp. 117–120.
22. Chen, Y.; Chen, H.; Zhang, M.; Li, Y. The relevant research of CoE protocol in EtherCAT Industrial Ethernet. In Proceedings of the 2010 IEEE International Conference on Intelligent Computing and Intelligent Systems, Xiamen, China, 29–31 October 2010; pp. 67–70.
23. Yi, H.C.; Choi, J.Y. Performance analysis of Linux-based EtherCAT DC synchronization. In Proceedings of the 2015 IEEE International Conference on Advanced Intelligent Mechatronics (AIM), Busan, Republic of Korea, 7–11 July 2015; pp. 549–552.

24. Barbalace, A.; Luchetta, A.; Manduchi, G.; Moro, M.; Soppelsa, A.; Taliercio, C. Performance Comparison of VxWorks, Linux, RTAI, and Xenomai in a Hard Real-Time Application. *IEEE Trans. Nucl. Sci.* **2008**, *55*, 435–439. [CrossRef]
25. IFA. *BG/BGIA Risk Assessment Recommendations According to Machinery Directive—Design of Workplaces with Collaborative Robots*; BGIA—Institute for Occupational Safety and Health of the German Social Accident Insurance: Sankt Augustin, Germany, 2011; p. 15.