sensors

MDPI

Article

Eye-in-Hand Robotic Arm Gripping System Based on Machine Learning and State Delay Optimization †

Chin-Sheng Chen and Nien-Tsu Hu *

Graduate Institute of Automation Technology, National Taipei University of Technology, Taipei 10608, Taiwan
* Correspondence: nthu@ntut.edu.tw

† This paper is an extended version of "The Gripping Posture Prediction of Eye-in-hand Robotic Arm Using Min-Pnet" published in the Proceedings of the 2022 International Conference on Advanced Robotics and Intelligent Systems (ARIS), Taipei, Taiwan, 24–27 August 2022.

Abstract: This research focused on using RGB-D images and modifying an existing machine learning network architecture to generate predictions of the location of successfully grasped objects and to optimize the control system for state delays. A five-finger gripper designed to mimic the human palm was tested to demonstrate that it can perform more delicate missions than many two- or three-finger grippers. Experiments were conducted using the 6-DOF robot arm with the five-finger and two-finger grippers to perform at least 100 actual machine grasps, and compared to the results of other studies. Additionally, we investigated state time delays and proposed a control method for a robot manipulator. Many studies on time-delay systems have been conducted, but most focus on input and output delays. One reason for this emphasis is that input and output delays are the most commonly occurring delays in physical or electronic systems. An additional reason is that state delays increase the complexity of the overall control system. Finally, it was demonstrated that our network can perform as well as a deep network architecture with little training data and omitting steps, such as posture evaluation, and when combined with the hardware advantages of the five-finger gripper, it can produce an automated system with a gripping success rate of over 90%. This paper is an extended study of the conference paper.

Keywords: point cloud; neural network; arm control; object grabbing; state delays

Citation: Chen, C.-S.; Hu, N.-T. Eye-in-hand Robotic Arm Gripping System Based on Machine Learning and State Delay Optimization. *Sensors* **2023**, *23*, 1076. https://doi.org/10.3390/s23031076

Academic Editor: Sašo Blažič

Received: 21 December 2022
Revised: 12 January 2023
Accepted: 13 January 2023
Published: 17 January 2023

1. Introduction

The COVID-19 pandemic means that there is a greater demand for automated production. To avoid a reduction in production capacity due to insufficient manpower, robotic arms with automatic control systems must perform more complex and more varied tasks.

Nowadays, robot arms for pick-and-place and assembly tasks have matured [1–8], and many factory robot arms now use two- or three-finger grippers for tasks, and many research papers have been conducted in this direction or for experiments [9–12]. Some use algorithms for control [13] and others use visual images and machine learning to allow robotic arms to perform tasks [14,15]. Other studies convert images into point clouds that use depth distances for better task execution [16–24]. For example, the study in [25] combines object recognition by Mobile-DasNet and point cloud analysis to generate the coordinates of the arm endpoints for the apple picking task.

However, if the factory is to become more automated, the robot arm needs to be able to perform more complex movements. On this premise, the original two- and three-finger grippers may not be able to complete more delicate or complex movements, so many people began to develop and use other types of grippers to overcome these shortcomings, including the five-finger grippers. Although the five-finger grippers have a larger gripping range, higher flexibility and fault tolerance, because of their complex structure, many control systems that use two- or three-finger grippers, as to derive the gripping attitude may not be compatible with five fingers.

There have been some studies using five-finger graspers, but most of them studied hardware or other aspects [26–29]. The studies used for application have used a virtual space as a result of the study [30,31] or used algorithms for gripping pose prediction [32,33].

This paper is an extended study of the original conference paper [1]. We extend the original conference paper on the effects of time delay for the robot manipulator. We investigate state time delays and propose a control method for a robot manipulator. Many studies on time-delay systems have been conducted, but most focus on input and output delays. One reason for this emphasis is that input and output delays are the most commonly occurring delays in physical or electronic systems. An additional reason is that state delays increase the complexity of the overall control system. In this paper, we use the qbSoftHand five-finger grasper in combination with an RGB-D visual recognition system and modify the proposed mechanical learning network by omitting the pose evaluation part so that the arm can automatically move to the target location and accomplish the task of grasping the target object to the placement area. The state delay of the control system is also optimized to enable the system to operate efficiently. Experiments were conducted with objects that were considered difficult to grasp with a two- or three-finger grasper to demonstrate the superiority of the grasping method and five fingers.

2. Control System Architecture

The hardware for this study is a UR5 six-degree-of-freedom arm, qbSoftHand five-finger gripper, HIWIN XEG-32 gripper and a RealSense D435 camera. The system architecture is shown in Figure 1. Object recognition is achieved using Yolo [34,35]. The point cloud for the object is generated after sampling and adjustment. The data are used as input data for an AI model that predicts the gripping posture for the arm and sends this information to the arm to perform the gripping action.

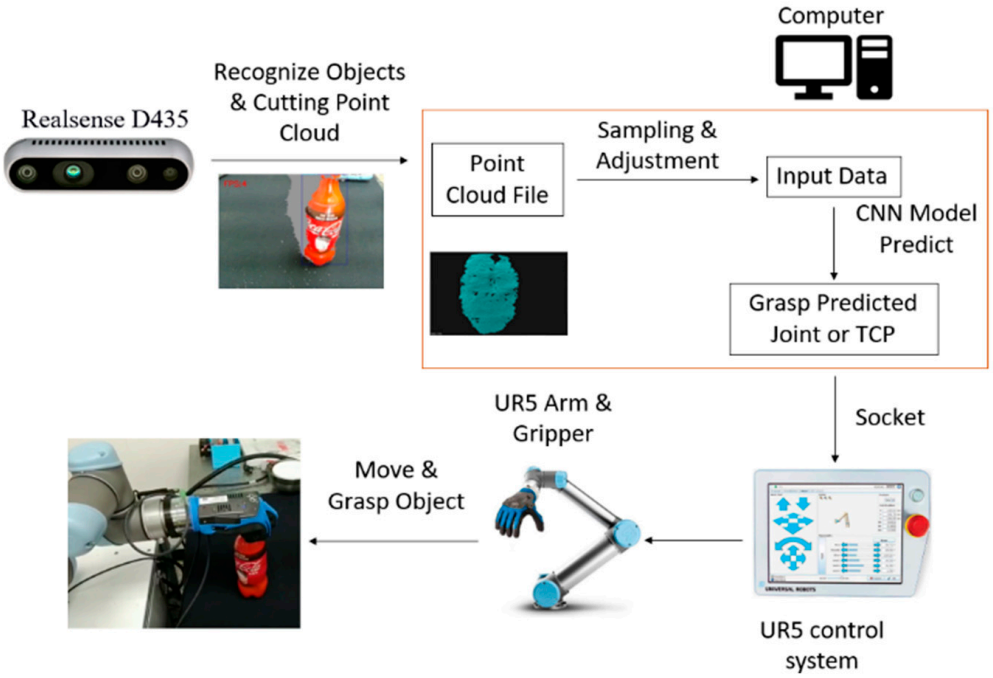

Figure 1. The working principle of the gripping robot.

The pose that is predicted by the network is sent to the UR5 control system via the socket library of the network cable. The current joint angle, the tool center point (TCP) and the analog and digital signals for the arm are also obtained in this way.

A successful grasping action is achieved if the arm moves to the target location, picks up the object and moves it to the target position without dropping the object.

3. Point Cloud Data and Machine Learning Architecture

3.1. Convolutional Neural Network (CNN) Architecture

The architecture of the CNN is shown in Figure 2. This section describes the hidden layer (convolutional layer).

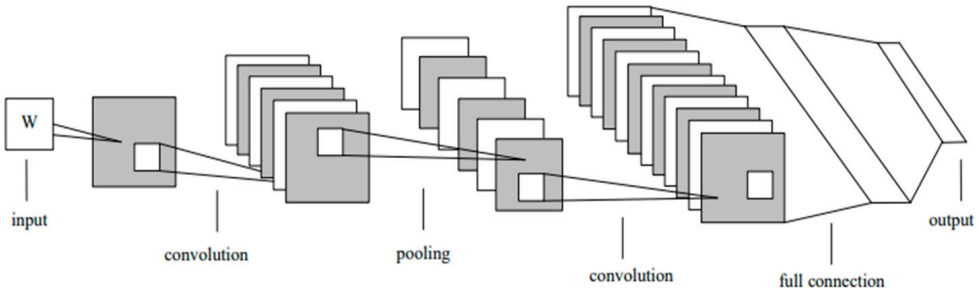

Figure 2. The network architecture of CNN [36].

The convolutional layer is the core of the CNN [36]. When the image is input into the convolution layer, it performs convolutional operations using the convolutional kernel. The formula is:

$$X_j^l = f(\sum_{i \in Pj} X_i^{l-1} * k_{ij}^l + b_j^l) \quad (1)$$

where $f(\sum_{i \in Pj} X_i^{l-1} * k_{ij}^l + b_j^l)$ is a tanh function, P_j is a local receptive field, X_i^{l-1} is the value of the l-1 feature on the i window, (i, j) is the position on the first floor, k_{ij}^l and b_j^l are the respective weights of the convolution kernel and the offset of the feature. More details can be found in [37].

The input data for this study are not images, but numerical values, so the convolution is a one-dimensional convolution. Differences between this and a two-dimensional convolution are described in Section 3.3.

The AI network for this study consists of three one-dimensional convolutional layers, a pooling layer and a multi-layer perceptron [25]. The point cloud data are input and the position at which the object is grasped is predicted. This information is transmitted to the UR5 arm control system. The network architecture using the CNN is shown in Figure 3.

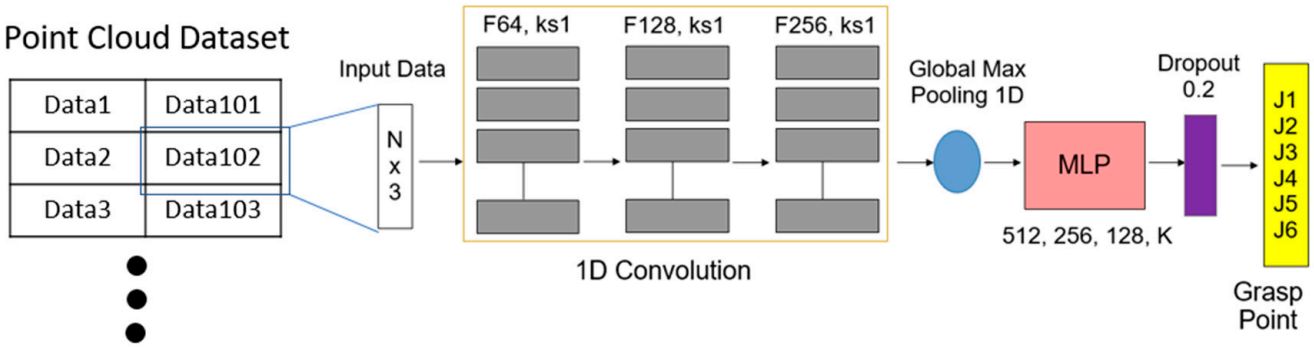

Figure 3. Grasping the estimated CNN network architecture. (N: the number of " points" in input data; F: filters, ks: kernel size; K: output number).

3.2. Min-Pnet Architecture

Min-Pnet is a network designed according to the architecture of PointNet [37] (see Figure 4) In PointNet research, it is mentioned that many studies transform the input data into regular 3D voxel grids or collections of multi-angle images, generating large amounts of unnecessary data and destroying the natural invariance of the original data. This study uses point clouds directly to avoid these problems and to make it easier to learn.

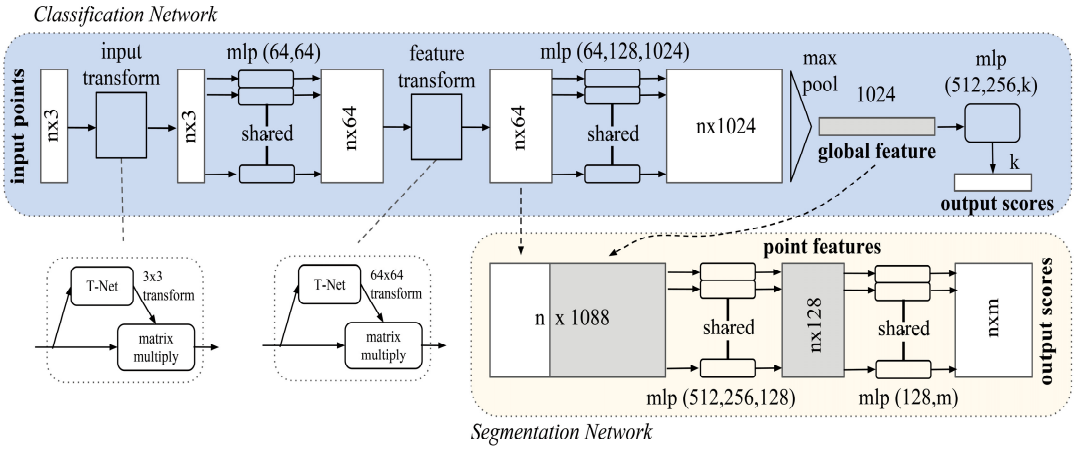

Figure 4. PointNet's network architecture [37].

In order to protect the network itself from point cloud disorder, PointNet uses its "feature extraction layer" to convert disordered point cloud data into 1024-dimensional features to perform subsequent tasks.

The feature extraction layer is to transform the input points into features by assembling the input data into a canonical space and using T-net to predict an affine transformation matrix. The partial and global feature extraction of the point cloud is performed without affecting the correlation and invariance of the points.

In this study, the Min-Pnet (see Figure 5) uses one feature extraction layer instead of two to reduce the training time and to avoid overfitting. The original 2D conv part was changed to 1D conv. This is because our input data size is different from the original setting of Pointnet, which causes problems in the matrix multiply part of the network.

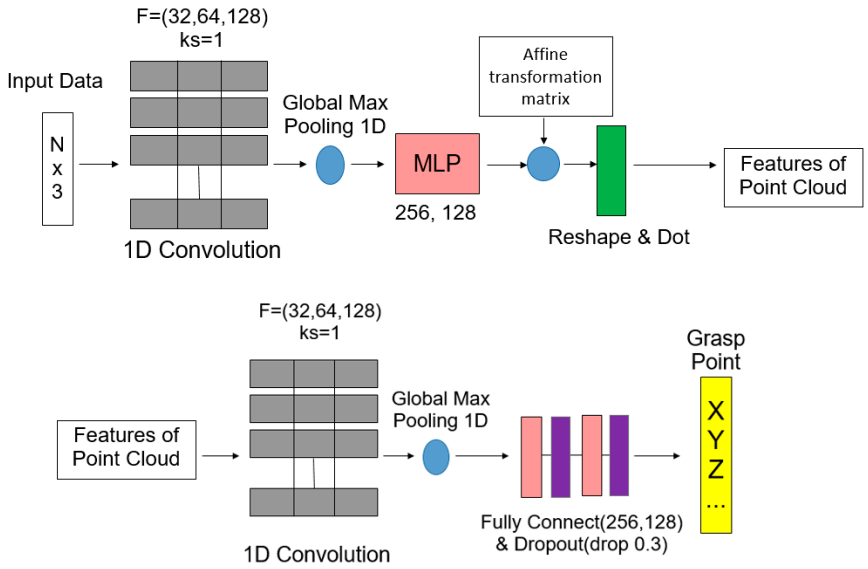

Figure 5. Min-Pnet architecture (N: the number of "points" in input data; F: filters; ks: kernel size).

3.3. Input Point Cloud Data

To ensure that the point clouds are reliable and not easily affected by the external environment, a PCL [38] chopbox and the Yolo Bounding Box are used to remove most of the unwanted point clouds (walls and desktops; see Figure 6).

To increase the amount of training data and to ensure that the model resists manipulation in response to slight errors that are caused by the hardware, Gaussian noise is used (Figure 7.)

To ensure that the input data are the same size, the point clouds for these objects are sampled and processed. The processed input data are a 3000 × 3 (N × 3 in Figure 8) matrix of the XYZ coordinates of the points (Figure 7). These data correspond to grasping point data, including the angle of each joint and a coordinate system based on UR5. These data are the input for the AI network and the prediction.

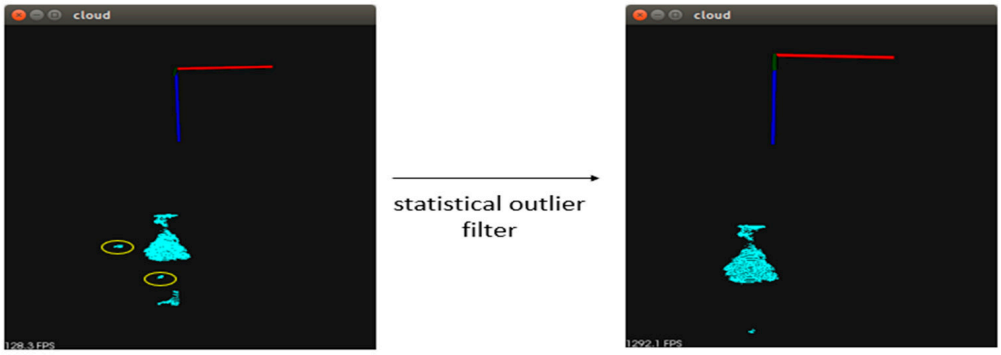

Figure 6. Schematic diagram of the statistical outlier filter. The yellow circle is the noise point removed.

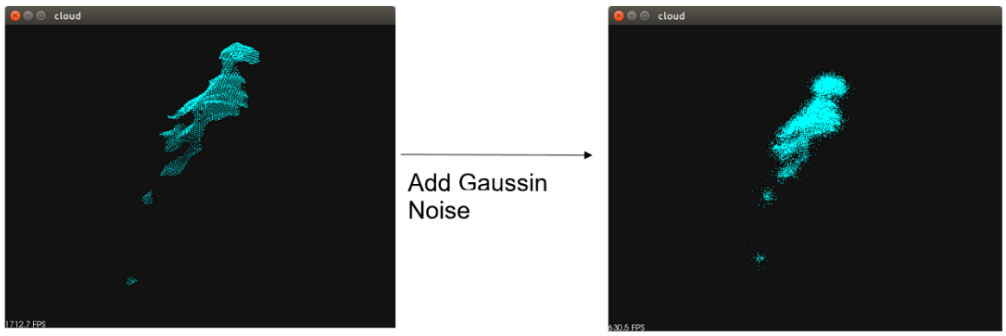

Figure 7. Point cloud sampling process.

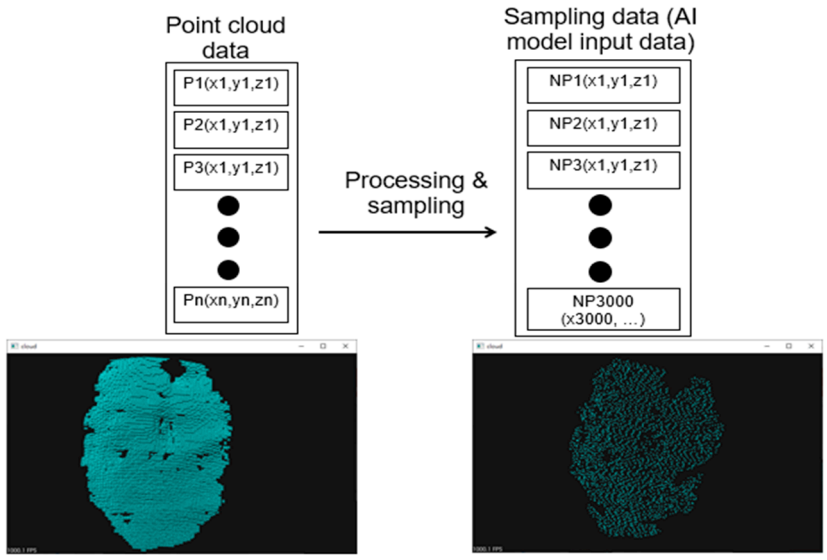

Figure 8. Point cloud sampling process.

A one-dimensional convolution layer (1D conv) is used for convolution and feature extraction from 1D data. Most AI frameworks that are used for object recognition or for grasp prediction, which use images, use a two-dimensional convolution layer (2D conv). The differences between these systems are shown in the Figure 9.

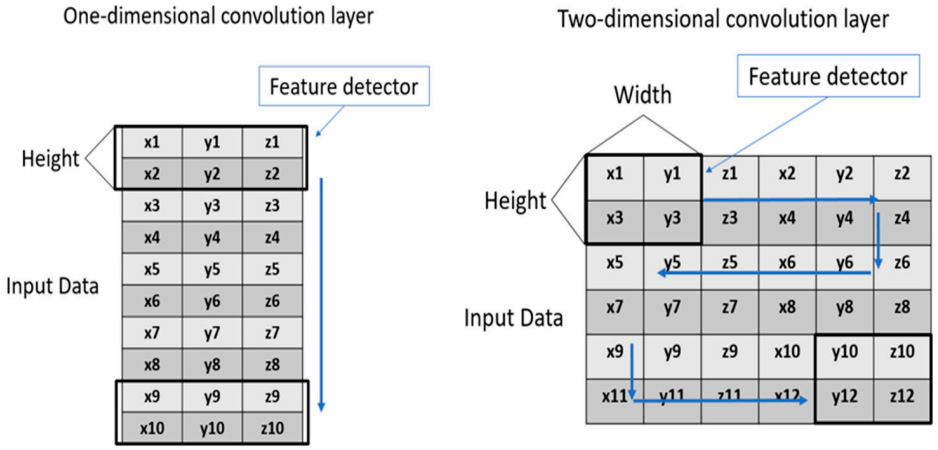

Figure 9. Illustration of 1D and 2D convolution works: the point cloud data for this study are used as demonstration.

In the CNN network, 1D conv directly extracts features from the input point cloud, and after processing with the maximum pooling layer (Max Pooling), it finally inputs the multi-layer perceptron (MLP) composed of several fully connected layers to obtain the output of the final predicted grasp position. This study uses the position of the object in space to predict the grasping position of the gripper so a 1D conv is used to extract the main features. After we sample the point cloud data, it is input to the model as a network, and the model outputs the predicted grasp posture coordinates or joint angles.

A 1D conv is initially used for natural language and data analysis and can also be used to analyze point clouds in data format. It is less computationally intensive and requires a shorter training time than a 2D conv. A 2D conv cannot be used because the point cloud data are not continuous (Figure 9), so the extracted features cannot be applied, which affects the prediction results.

3.4. Pooling Layer

The pooling layer reduces the dimensions of features and filters redundant features to reduce the computational burden and increase the generalization of the network. The pooling layer uses the maximum pooling (Max Pooling; see Figure 10) and mean pooling (Average pooling). The pooling process is expressed as:

$$X_j^i = f(pool(X_i^{l-1}) + b_j^l) \tag{2}$$

where X_i^{l-1} is the value of the ith window in the layer l-1 input feature, b_j^l is the offset for the jth window in layer l and pool represents the sampling function.

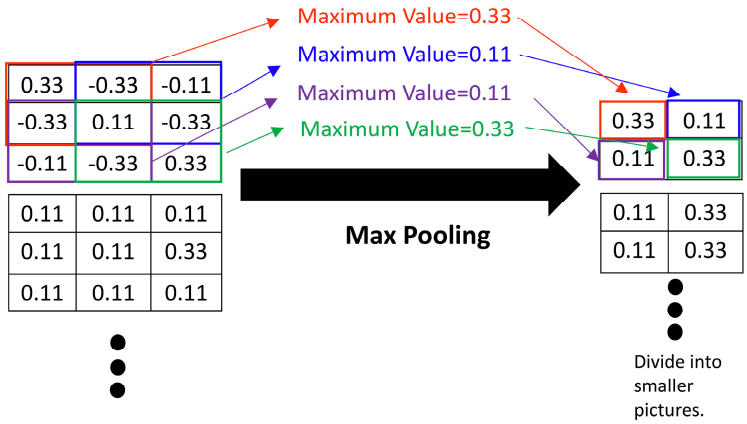

Figure 10. Operation of the Max Pooling layer.

3.5. Multi-Layer Perceptron (MLP)

Multi-layer perceptron uses several fully connected layers. Neurons in the full connection layer are connected to each other in the previous layer. The formula is:

$$X^l = f(u^l).u^l = W^l X^{l-1} + b^l \tag{3}$$

where $f(u^l)$ is the activation function, W^l is the weight of layers l-1 to 1, b^l is the offset for Layer 1, and X^{l-1} is the output feature of Layer l-1.

The last fully connected layer outputs six parameters that are used to predict joint angles or TCP coordinates. To avoid overfitting and prediction errors in the AI model, an exit layer is inserted before the final output layer.

4. State Delays Using Digital Redesign

We present the principles of optimal digital redesign in this section. This system removes the following parts of the original transformation of a time-delay system to a delay-free system [39]:

$$\dot{x}(t) = A\,x(t) + B\,u(t) \tag{4}$$

$$y(t) = C\,x(t)\,,\; x(0) = x_0 \tag{5}$$

The optimal quadratic state feedback control law is used to minimise the following performance cost function:

$$J = \int_0^\infty \left\{ [\,Cx(t) - r(t)\,]^T\, Q_c\, [Cx(t) - r(t)\,] + u^T(t) R_c u(t) \right\} dt \tag{6}$$

where $Q_c \geq 0$, $R_c > 0$. The following formula is obtained by optimising the controller:

$$u(t) = -K_c x(t) + E_c r(t) \tag{7}$$

The entire closed loop system can then be expressed as

$$\dot{x}(t) = (A - BK_c)x(t) + BE_c r(t) \tag{8}$$

For $m = p$, we obtain

$$K_c = {R_c}^{-1} B^T P \tag{9}$$

$$E_c = -{R_c}^{-1} B^T [(A - BK_c)^{-1}]^T C^T Q_c \tag{10}$$

Here, P is the solution to the Riccati equation given below.

$$A^T P + PA - PB{R_c}^{-1} B^T P + C^T Q_c C = 0 \tag{11}$$

The linear quadratic regulator (LQR) (Equation (6)) design characteristics make the resulting closed loop system stable. If the system state is unmeasurable, an observer must be designed to measure the system state. The linear observable continuous system (see Figure 11) that is described by the equation is shown below:

$$\dot{\hat{x}}(t) = A\hat{x}(t) + Bu(t) + L_c[y(t) - C\hat{x}(t)] \tag{12}$$

where $\hat{x}(t)$ is the estimated state, and L_c is the gain of the observer:

$$L_c = P_o C^T {R_o}^{-1} \tag{13}$$

We apply digital redesign to the analogue controller (Equation (7)) to obtain a more practical digital controller. The operation of the discrete time state feedback controller is described by the following equation:

$$u(kT_s) = -K_d x(kT_s) + E_d\, r^*(kT_s) \tag{14}$$

where

$$K_d = (I + K_c H)^{-1} K_c G \tag{15}$$

$$E_d = (I + K_c H)^{-1} E_c \tag{16}$$

$$r^*(kT_s) = r(kT_s + T_s) \tag{17}$$

$$G = e^{AT_s} \tag{18}$$

$$L_d = (G - I)A^{-1} L_c \left(I + C(G - I)A^{-1} L_c\right)^{-1} \tag{19}$$

For the linear model of the sampling system, the digital tracker based on the observer and the observer are shown in Figure 12.

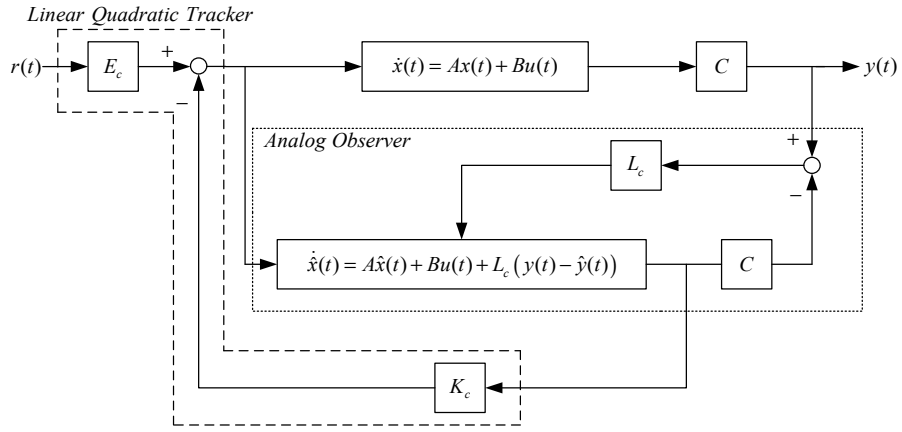

Figure 11. Linear-quadratic analogue tracker based on an observer.

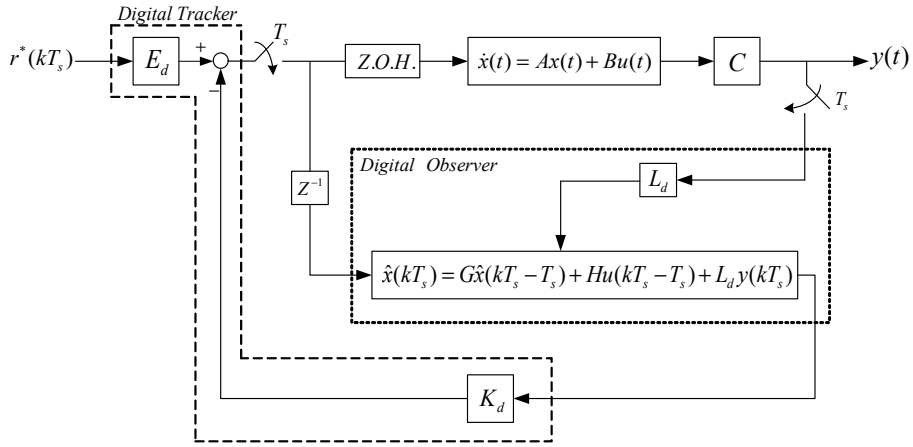

Figure 12. Predicted digital tracker and observer.

5. Results

This study uses bottles, bowls and sports balls for the experiments (see Figure 13). The training parameters for the AI network are a batch size = 32 and an epoch = 10,000 and the loss parameter uses mean squared error (MSE). The selected optimizer is 'adam'.

$$MSE(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{20}$$

$y_i - \hat{y}_i$ is error, n is number of data in Dataset.

This study uses Python 3.7.7 and a Windows10 system environment for model training and a crawling test.

In order to verify that the proposed control method [40] can be applied in subsequent real-world tests and to collect data more easily, a simulator was used to design a model of a nonlinear MIMO robot manipulator, as shown in Figure 14.

Figure 13. Three types of grasping objects for the experiment.

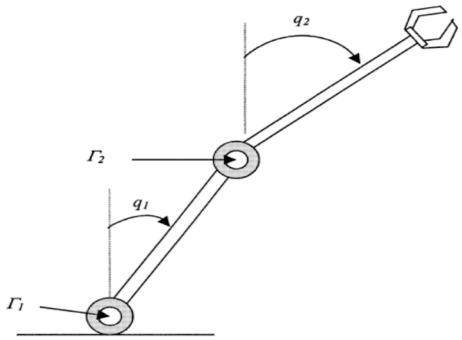

Figure 14. Two-link robot manipulator.

5.1. The Test of the Robot Manipulator

The dynamic equation of the two-link robot system is given below:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \Gamma \tag{21}$$

where

$$M(q) = \begin{bmatrix} (m_1+m_2)l_1^2 & m_2 l_1 l_2(s_1 s_2 + c_1 c_2) \\ m_2 l_1 l_2(s_1 s_2 + c_1 c_2) & m_2 l_2^2 \end{bmatrix}, \quad C(q,\dot{q}) = m_2 l_1 l_2(c_1 s_2 - s_1 c_2)\begin{bmatrix} 0 & -\dot{q}_2 \\ -\dot{q}_1 & 0 \end{bmatrix}$$
$$G(q) = \begin{bmatrix} -(m_1+m_2)l_1 g_r s_1 \\ -m_2 l_2 g_r s_2 \end{bmatrix}$$

and $q = \begin{bmatrix} q_1 & q_2 \end{bmatrix}^T$, where q_1 and q_2 are angular positions, $M(q)$ is the moment of inertia, $C(q,\dot{q})$ includes the Coriolis and centripetal forces, $G(q)$ is the gravitational force, and Γ is the applied torque vector. Here, we use the short-hand notations $s_i = \sin(q_i)$ and $c_i = \cos(q_i)$. The nominal parameters of the system are as follows: the link masses are $m_1 = 5$ kg and $m_2 = 2.5$ kg, the lengths are $l_1 = l_2 = 0.5$ m, and the gravitational acceleration is $g_r = 9.81$ ms^{-2}. Then, (30) can be rewritten in the following form:

$$\ddot{q} = M^{-1}(q)\left(\Gamma - C(q,\dot{q})\dot{q} - G(q)\right) \tag{22}$$

Let x and $f(x)$ represent the state of the system and a nonlinear function of the state x, respectively. The following notation is used:

$$x(t) \equiv [x_1\ x_2\ x_3\ x_4]^T = [q_1\ \dot{q}_1\ q_2\ \dot{q}_2]^T,\ f(x(t)) \equiv [f_1\ f_2\ f_3\ f_4]^T,$$

where

$f_1 = x_2$, $f_3 = x_4$, and $[f_2\ f_4]^T = M^{-1}(-C[x_2\ x_4]^T - G)$. Let $u \equiv \Gamma$, where $\Gamma = \begin{bmatrix}\Gamma_1 & \Gamma_2\end{bmatrix}^T$.

The inverse of the matrix M is calculated as $M^{-1} = \begin{bmatrix}p_{11} & p_{12}\\ p_{21} & p_{22}\end{bmatrix}$, such that $g(x(t)) = \begin{bmatrix}0 & p_{11} & 0 & p_{21}\\ 0 & p_{12} & 0 & p_{22}\end{bmatrix}^T$.

The dynamic equation of the two-link robot system can, thus, be reformulated as follows:

$$\dot{x}(t) = f(x(t)) + g(x(t))u(t - \tau_i) \tag{23}$$

$$y(t) = Cx(t - \tau_o)$$

where $C = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\end{bmatrix}$ and the initial condition is $x(0) = \begin{bmatrix}0 & 0 & 0 & 0\end{bmatrix}^T$.

First, OKID is applied to convert the nonlinear system (Equation (23)) to an equivalent linear system. The system (Equation (23)) is injected with white noise $u(t) = \begin{bmatrix}u_1(t) & u_2(t)\end{bmatrix}$ with a zero mean and covariance $diag\big(\mathrm{cov}(u_{1,2}(t)) = \begin{bmatrix}0.2 & 0.2\end{bmatrix}\big)$ at the sampling time $T = 0.01$ s. Figure 15 shows that the error between the output of the identified equivalent linear system and the original nonlinear system (Equation (23)) can be controlled to within $10^{-6} \sim 10^{-5}$.

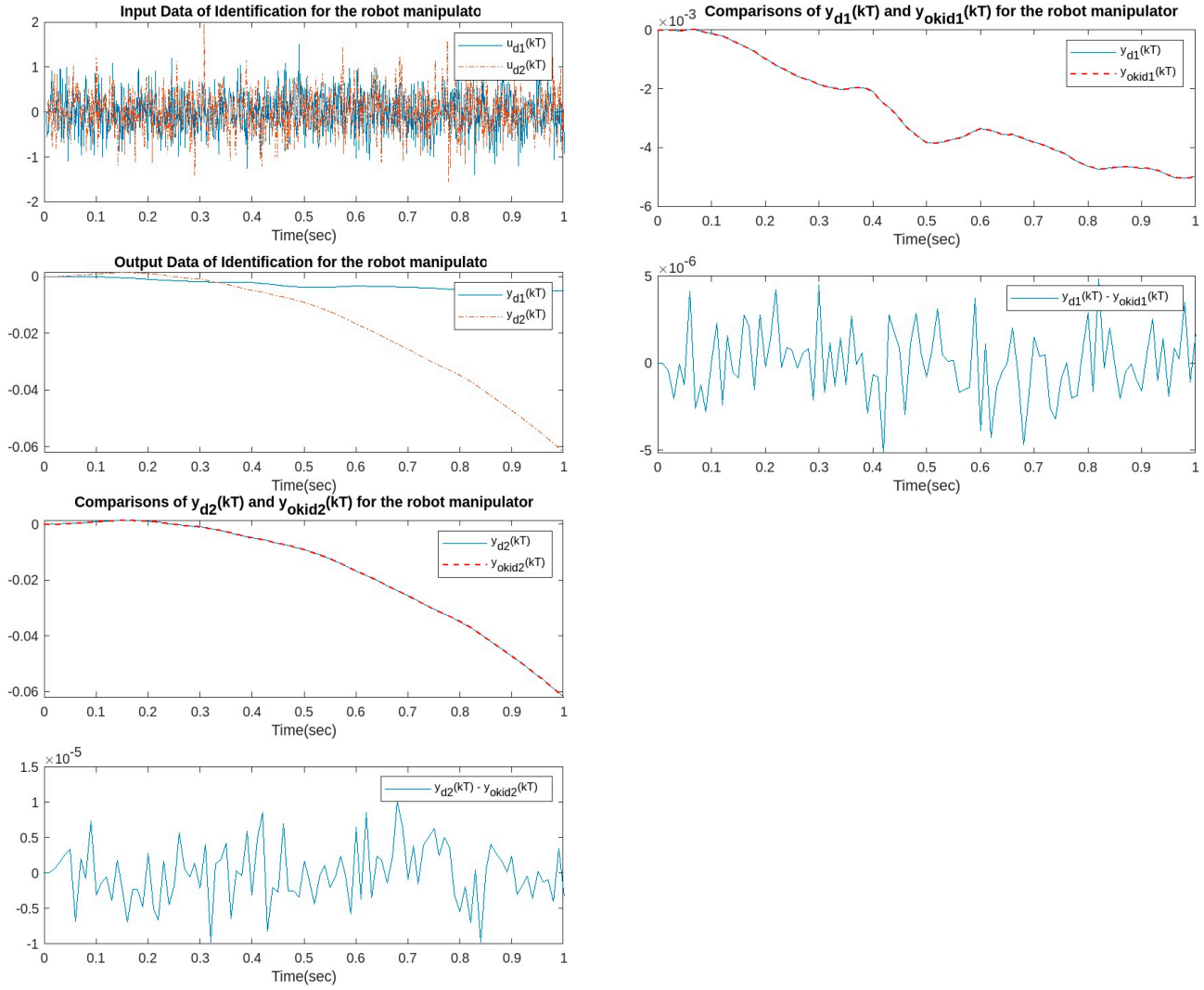

Figure 15. The comparison between the output of the identified equivalent linear system and the original nonlinear system.

We then consider two different reference inputs $r(t) = \begin{bmatrix} r_1 & r_2 \end{bmatrix}^T$ for the identified equivalent linear system at the sampling time $T = 0.01$ s with an input delay $\tau_i = 0.5 \times T$ and an output delay $\tau_o = 0.3 \times T$ for the optimal DR method presented in Section 4. To test whether the designed control can effectively suppress a state delay, we gradually increase the state delay $\tau_s = 0 \times T$ for two different reference inputs, i.e., Types 1 and 2. Figures 16–23 show the ability of the control to suppress the Types 1 and 2 state delays. Table 1 summarises the results in Figures 16–23, showing that the robot manipulator suppresses the state delay for approximately 2.6 s.

Table 1. The ability of the robot manipulator to suppress state delays.

Reference Inputs	Delay Parameters	Max. Output Error (Rad)	Tolerable Delay Time of Manipulator
Type 1	$\tau_i = 0.5 \times T,\ \tau_o = 0.3 \times T,\ \tau_s = 0.13 \times T$	6.48×10^2	<2.6 s
Type 2	$\tau_i = 0.5 \times T,\ \tau_o = 0.3 \times T,\ \tau_s = 0.13 \times T$	9.886	<2.6 s

TYPE 1:
Case 1: $\tau_i = 0.5 \times T$, $\tau_o = 0.3 \times T$, $\tau_s = 0 \times T$

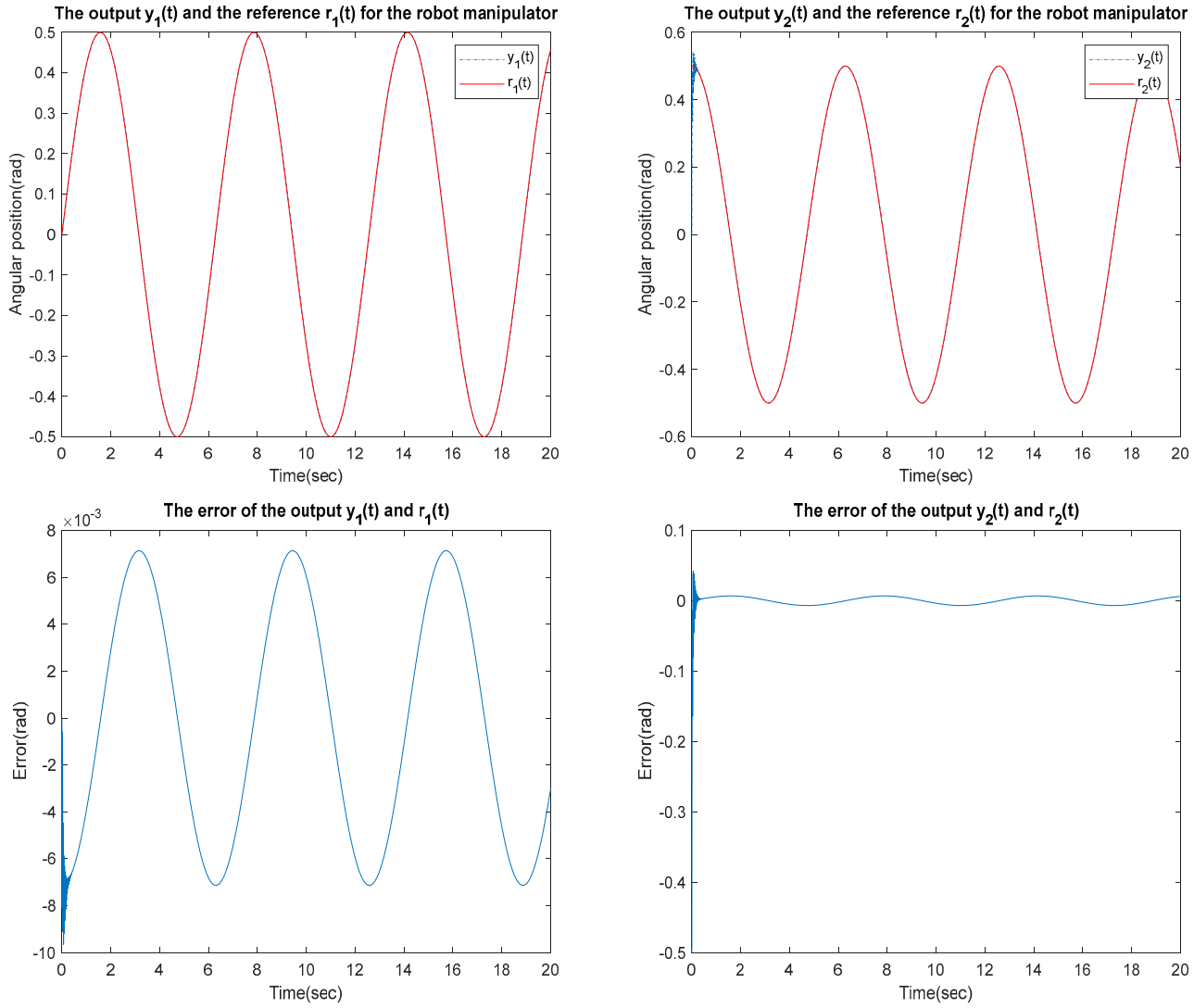

Figure 16. TYPE 1. Case 1: for the ability to suppress state delay.

Case 2: $\tau_i = 0.5 \times T$, $\tau_o = 0.3 \times T$, $\tau_s = 0.1 \times T$

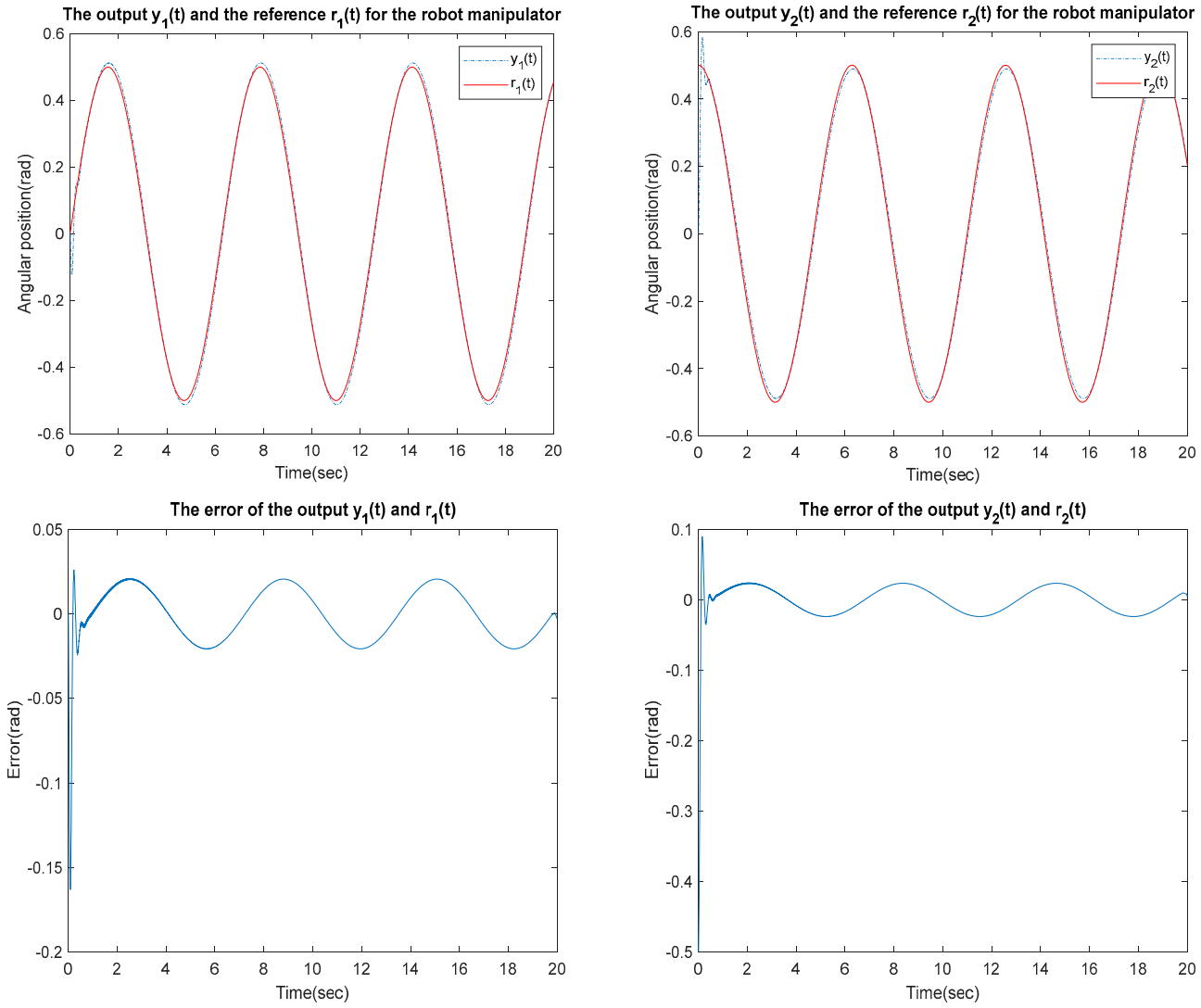

Figure 17. TYPE 1. Case 2: for the ability to suppress state delay.

Case 3: $\tau_i = 0.5 \times T$, $\tau_o = 0.3 \times T$, $\tau_s = 0.12 \times T$

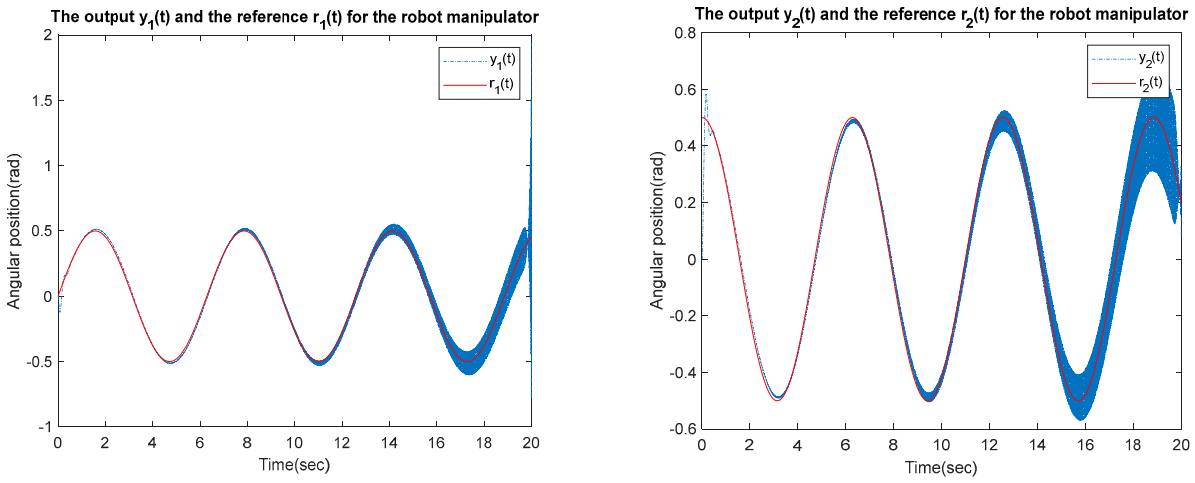

Figure 18. *Cont.*

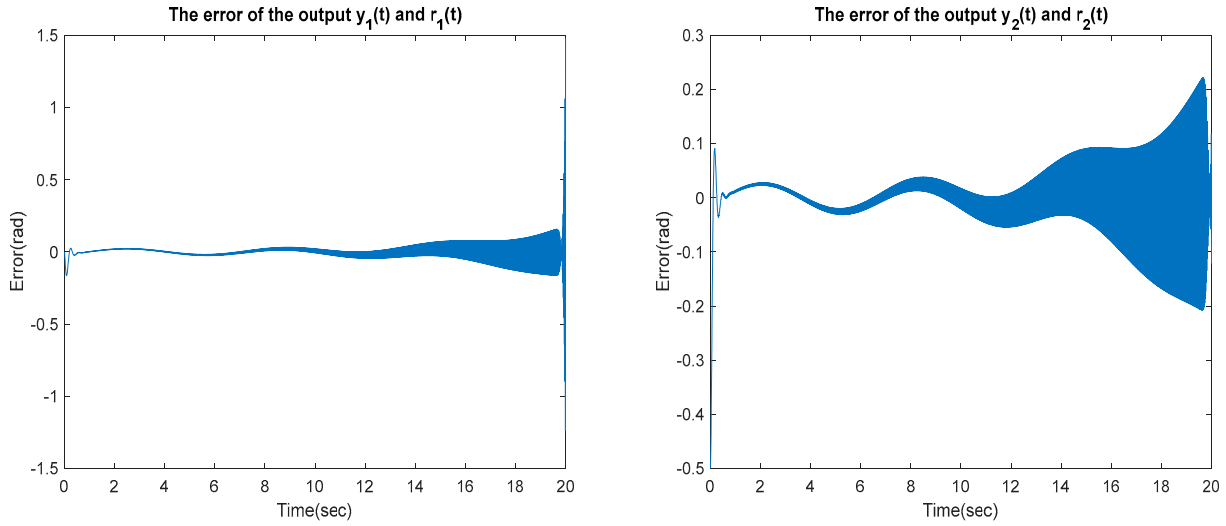

Figure 18. TYPE 1. Case 3: for the ability to suppress state delay.

Case 4: $\tau_i = 0.5 \times T$, $\tau_o = 0.3 \times T$, $\tau_s = 0.13 \times T$

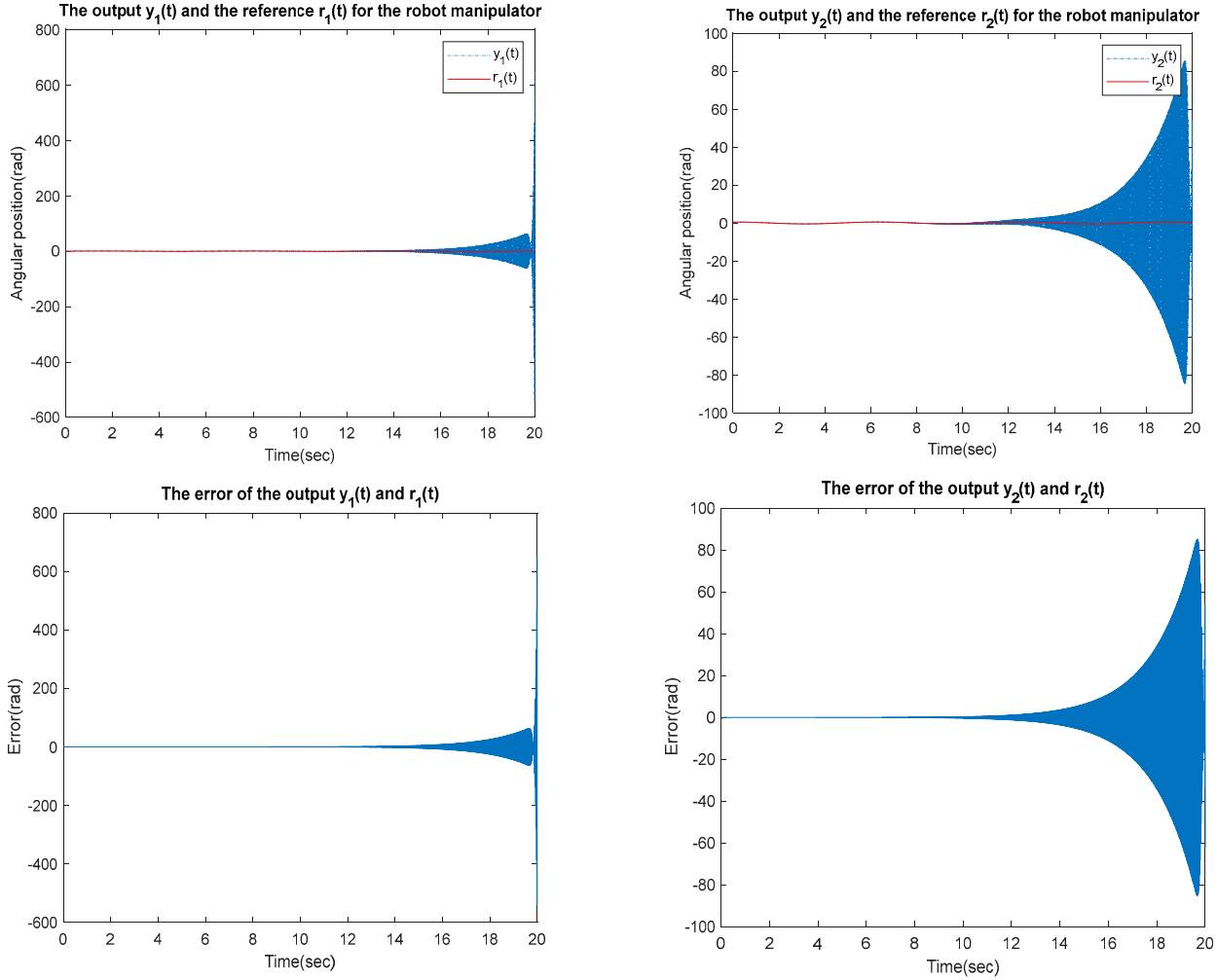

Figure 19. TYPE 1. Case 4: for the ability to suppress state delay.

TYPE 2:
Case 1: $\tau_i = 0.5 \times T$, $\tau_o = 0.3 \times T$, $\tau_s = 0 \times T$

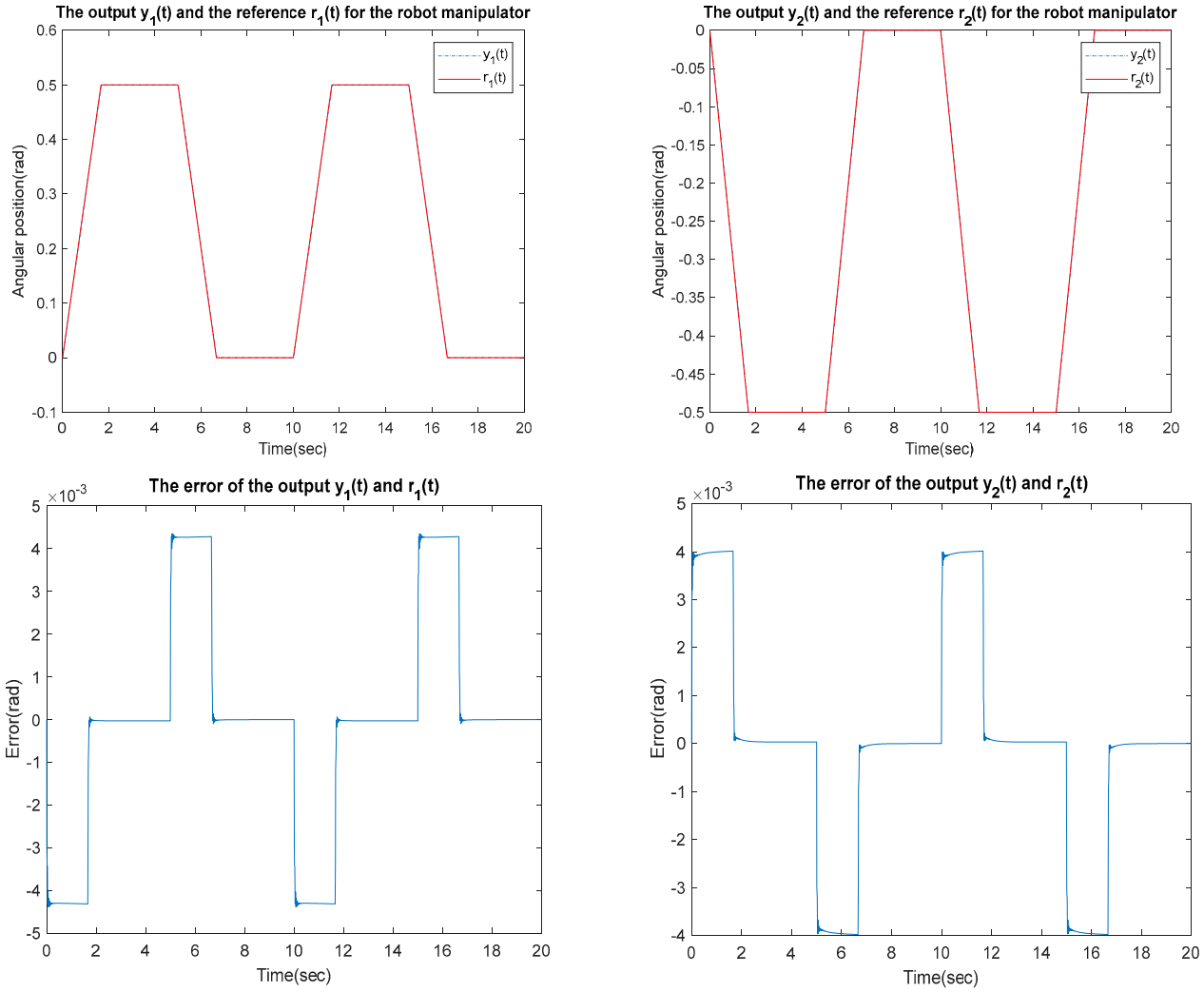

Figure 20. TYPE 2. Case 1: for the ability to suppress state delay.

Case 2: $\tau_i = 0.5 \times T$, $\tau_o = 0.3 \times T$, $\tau_s = 0.1 \times T$

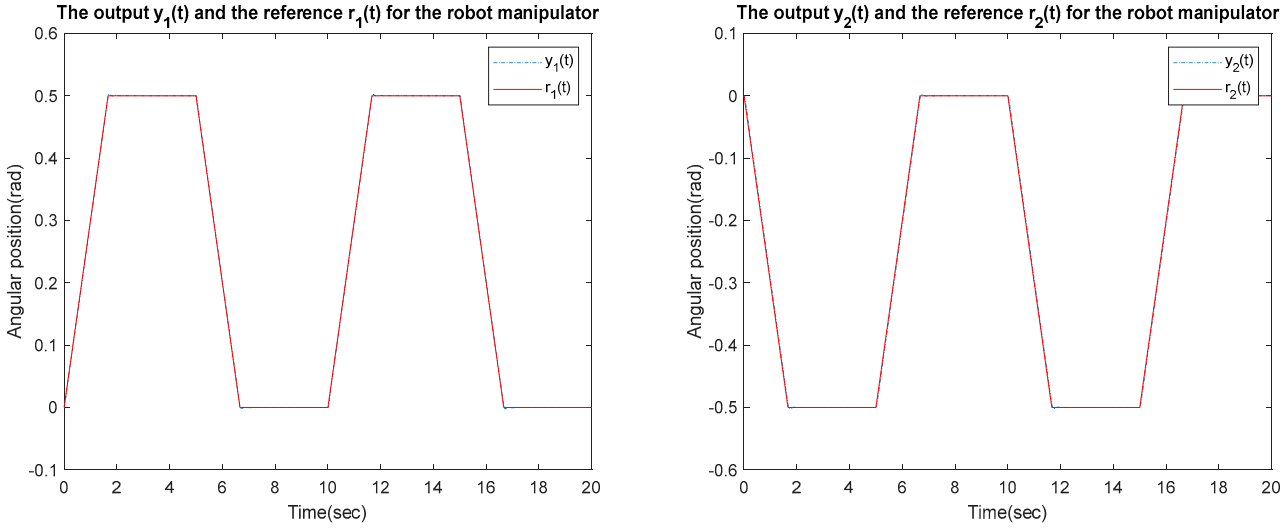

Figure 21. *Cont.*

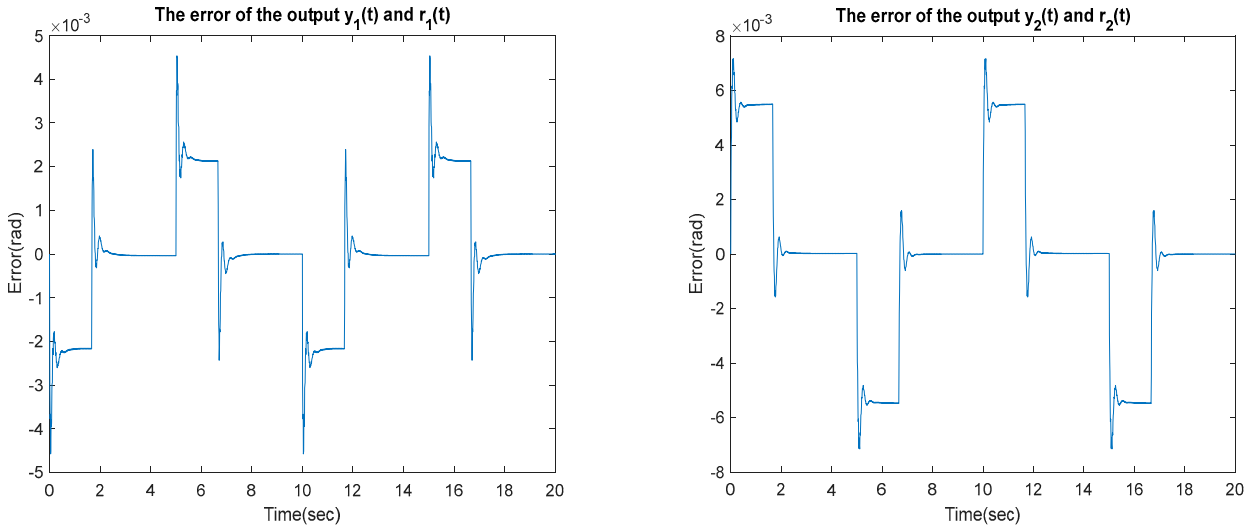

Figure 21. TYPE 2. Case 2: for the ability to suppress state delay.

Case 3: $\tau_i = 0.5 \times T$, $\tau_o = 0.3 \times T$, $\tau_s = 0.12 \times T$

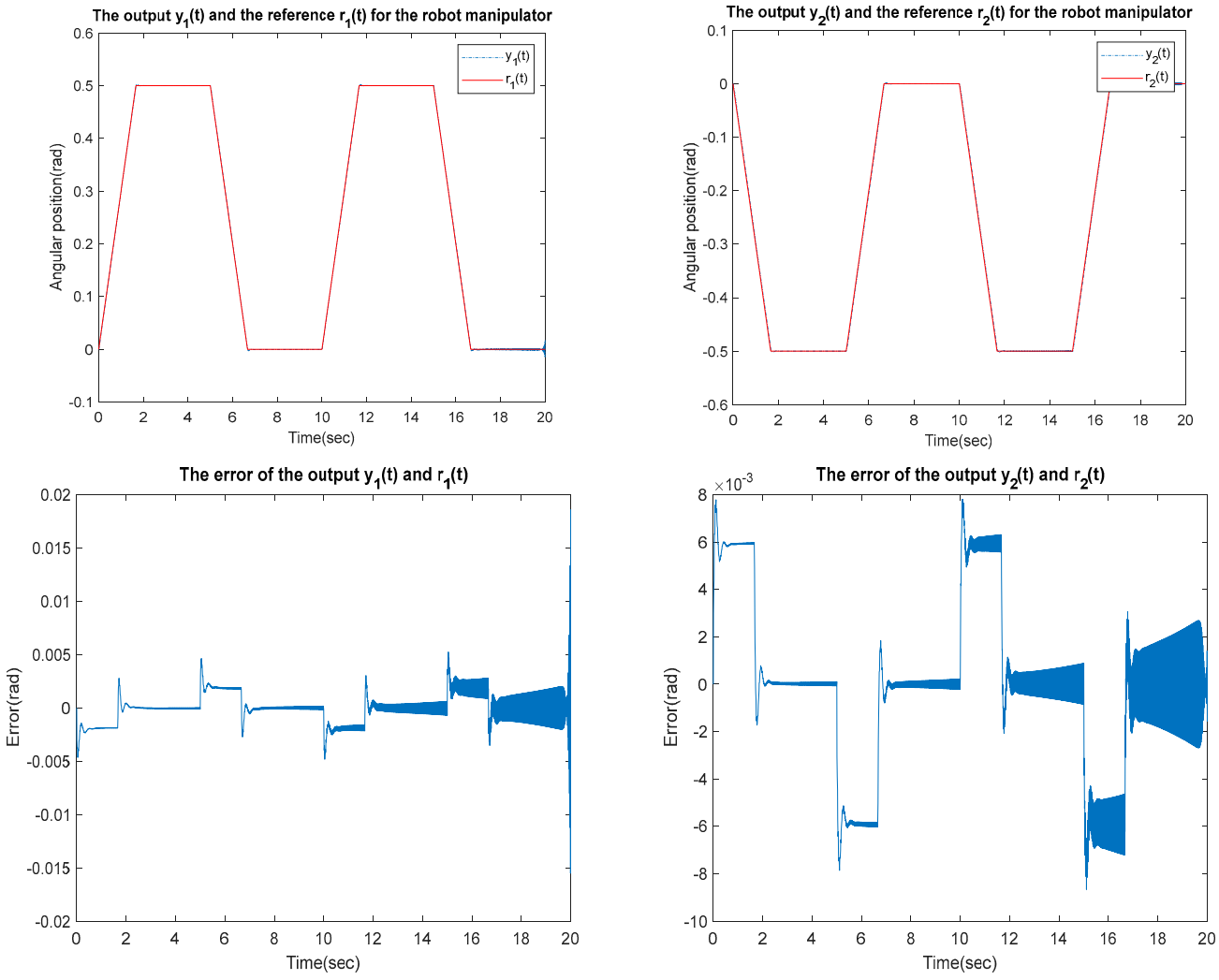

Figure 22. TYPE 2. Case 3: for the ability to suppress state delay.

Case 4: $\tau_i = 0.5 \times T$, $\tau_o = 0.3 \times T$, $\tau_s = 0.13 \times T$

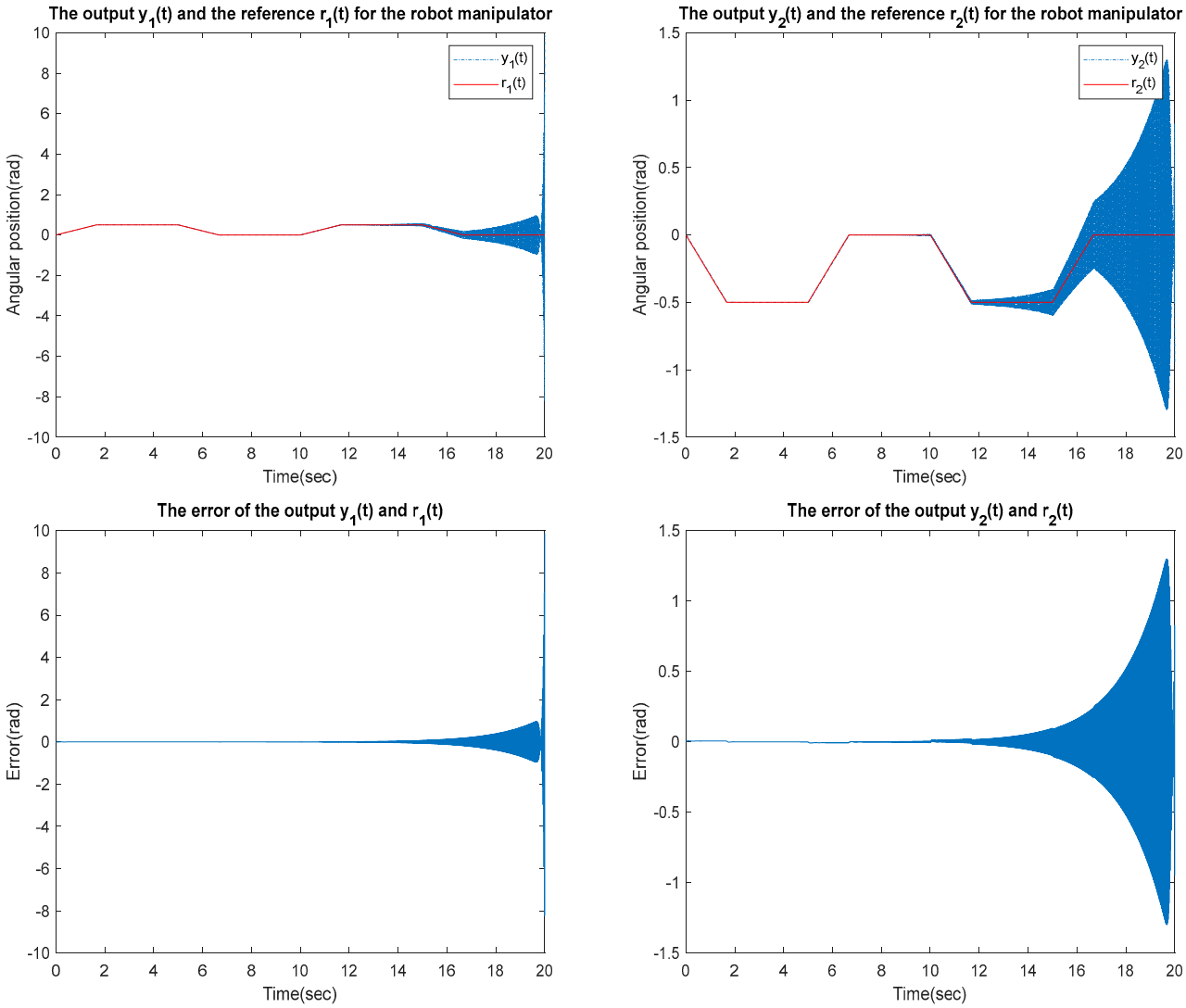

Figure 23. TYPE 2. Case 4: for the ability to suppress state delay.

5.2. AI Training Results

The next section shows the training results of the neural network. There are 325 training data points for the three subjects in the training and (bottle: 180; bowl and ball: 50). Noisy point cloud data are used for training to give a total of 595 training samples. There are 45 data points in a test set (bottle: 25; bowl and ball: 10). During the training period, the predicted output (joint angles or TCP) was standardization. The standardization range (Table 2) is set according to the working range of the arm.

Table 2. Standardization range.

Range	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6
Max (1)	−160°	−80°	140°	−10°	120°	0°
Min (0)	−220°	−120°	80°	−40°	40°	−50°
Range	X	Y	Z	α (Rx)	β (Ry)	γ (Rz)
Max (1)	1.5 m	1.5 m	1.5 m	1.5 m	1.5 m	1.5 m
Min (0)	−1.5 m	−1.5 m	−1.5 m	−1.5 m	−1.5 m	−1.5 m

Three models were trained for these three types of objects. The output for these models in Table 3 above is the UR5 arm joint angle that qbSoftHand uses to grasp the object. These angles are input into the UR5 control system and the arm is moved using the MoveJ command.

Table 3. AI Model training results.

Model	Bowl Loss	Bowl Val-Loss	Ball Loss	Ball Val-Loss	Time Consumed
CNN	2.2469×10^{-5}	0.0011	1.2599×10^{-5}	1.7467×10^{-4}	17 ms/epoch
Min-Pnet	1.4181×10^{-4}	0.0011	5.3420×10^{-5}	3.6437×10^{-5}	215 ms/epoch
OBB	2.8651×10^{-4}	0.0012	1.2412×10^{-4}	2.4541×10^{-4}	25 ms/epoch

5.3. Real Machine Grasping

For the grasping test, the camera observes objects in the work space at an angle of about 45 degrees to the desktop (Figure 24).

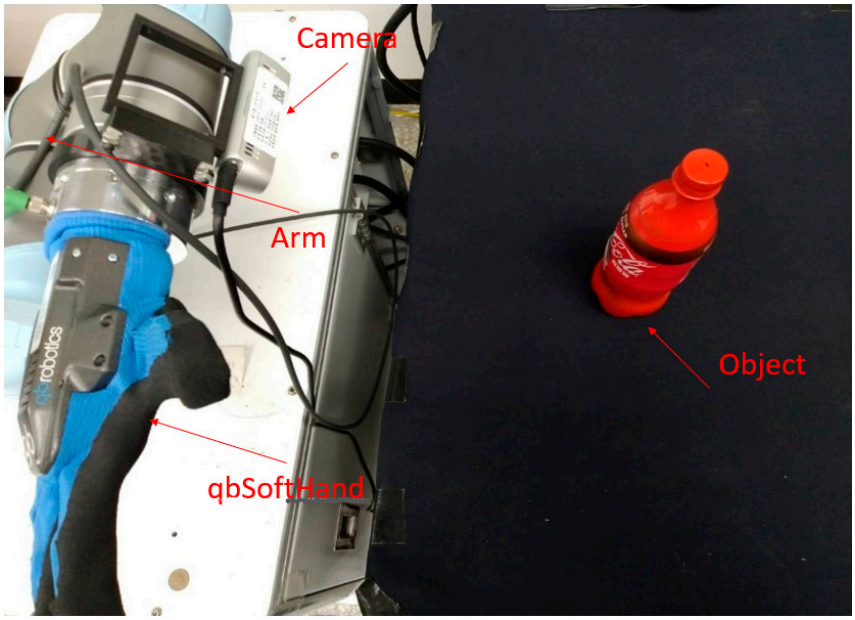

Figure 24. The UR5 arm observes the position of the object.

5.3.1. Five-Finger and Two-Finger Claw Gripping Experiment

The experiment first tested the grip comparison between the two-finger and five-finger grippers (Figure 25). A network was trained using oriented bounding box (OBB) framework data [41] and a point cloud model using a two-finger gripper as a control group. Simultaneously, the results were compared with those of a study similar to [25]. The results are displayed in Table 4.

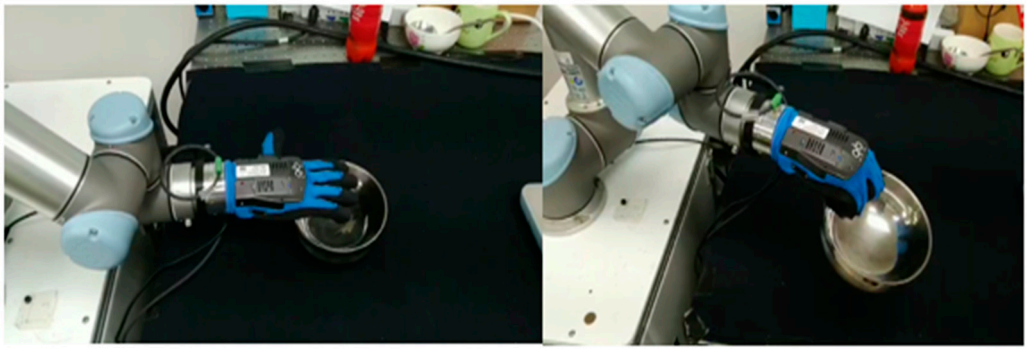

Figure 25. UR5 and qbSofthand grasping the bowl.

Table 4. Bowl and ball gripping test results.

Object and Model	Grasping Success Rate
5-Fin CNN Bowl	90.0%
5-Fin CNN Ball	80.0%
5-finger CNN Bottle	80.0%
2-Fin CNN Bowl	58.82%
2-Fin CNN Ball	52.94%
2-finger CNN Bottle	64.15%
Indoor [25]	85%
Outdoor [25]	80%

According to the above test results, it can be observed that compared to the two fingers, the five-finger gripper achieved a success rate of more than 80% in the grip of various objects. The performance of the two-finger gripper (Figure 26), although it had a success rate of more than half, was much worse than that of the five-finger gripper. In fact, in most of the failed grips with the two-finger, the predicted position was very close to the object, but the object could not be grasped because the gripping area of the gripper was not wide enough and the force applied at the contact point. This demonstrates that even the five-finger grippers, which are only capable of open grip action, show a significant advantage over the two-finger on the hardware level. Reference [25] used a three-finger gripper and achieved at least an 80% success rate in both indoor and outdoor tests as in this study. Considering the larger number of samples they collected (570 in total), the similar success rate achieved in our study may be attributed to the adaptability of the five-finger gripper, in addition to the different experimental environment (clutter vs. single).

Figure 26. Failed attempt using a two-finger gripper.

5.3.2. Comparison of CNN and Min-Pnet Networks

The next section compares CNN networks, Min-Pnet networks designed with reference to the PointNet concept, and networks using OBB. Since the advantages of five fingers in hardware have been demonstrated in the previous section, it is shown that they can effectively compensate for the shortcomings of neural networks. In order to compare the differences in network architectures more clearly, the hardware was compared using a two-finger gripper. The results are shown in Table 5.

Compared to the CNN network, the results of Min-Pnet were significantly better in all aspects. Although the difference in the results on the bottles was not large (5%), the bowl and sports ball performed significantly better (24% and 20%), which may be due to the hardware limitation mentioned in the previous section, while for the OBB, except for the

bowls with a larger successful area, the results were not very good. In the OBB experiment, it was found that objects that were too close or too far away were not successful, probably due to the angle of view and distance, so that the OBB frame point did not fully represent the size and position of the object.

Table 5. Test result of each network.

Object and Model	Grasping Success Rate
2-Fin Min-Pnet Bowl	82.60%
2-Fin Min-Pnet Ball	73.07%
2-finger Min-Pnet Bottle	69.01%
2-Fin CNN Bowl	58.82%
2-Fin CNN Ball	52.94%
2-finger CNN Bottle	64.15%
2-Fin OBB Bowl	60.00%
2-Fin OBB Ball	33.34%
2-Fin OBB Bottle	53.33%

5.3.3. Out-of-Training Set Object Gripping

Then, we tested whether our network could produce a corresponding grip on an unknown object (see Figure 27) with a similar training set. The objects we tested were: an alcohol spray bottle (considered as bottle), a sock wrapped in a ball, and a small doll (considered as sport ball). The network included CNN, Min-Pnet and OBB, and the hardware was the same as in the previous section. The results are shown in Table 6.

Figure 27. Out-of-training set object.

Table 6. Out-of-training set object grabbing test.

Object and Model	Grasping Success Rate
2-Fin Min-Pnet Ballobj	75.00%
2-Fin CNN Ballobj	50.00%
2-Fin OBB Ballobj	25.00%
2-Fin Min-Pnet Bottleobj	73.33%
2-Fin CNN Bottleobj	60.00%
2-Fin OBB Bottleobj	53.33%

Although the number of experiments is small, the success rate of Min-Pnet proves that the feature transformation architecture can indeed analyze point clouds more effectively, demonstrating its adaptability by still being able to predict effective grip positions

when faced with objects that are similar but not identical to the training data. This also demonstrates the drawback of OBB's difficulty in displaying asymmetric object features.

6. Conclusions

Simulation results in Section 5.1 are presented demonstrating effective optimal digital re-design control of a robot manipulator with a nonlinear delay. In particular, the proposed method can effectively control a state delay within the tolerable scope.

In terms of gripper hardware comparison, although the qbSoftHand used in this study cannot grasp objects more flexibly due to hardware limitations, it can still grasp many objects that are difficult to grasp with the two-finger gripper, such as smooth bowls, balls, and bottles. A comparison with the two-finger gripper and the three-finger gripper of [25] demonstrates the advantage of the five-finger gripper on the hard surface.

In the case of objects outside the training set, the Min-Pnet achieved a success rate of more than 70% despite the small number of experiments, and after analysis, it was found that most of the gripping failures were due to hardware limitations. When testing the limits of the network, it is also proved that the features and poses of the objects cannot be fully expressed by using only the OBB framework.

Due to hardware and time constraints, it is not possible to conduct more complex experiments such as identifying and grasping multiple objects in a chaotic environment or testing the grasping of more objects in this paper. However, our research has partially demonstrated the advantages of the five-finger gripper and provided a simple and effective grasping system to automate the task of grasping objects.

There are many future research directions, such as using wireless communication to control the arm. The original data collection part can be accomplished in a virtual environment, or using 5G combined with AR for arm training, etc.

Author Contributions: Conceptualization, C.-S.C.; methodology, C.-S.C.; software, N.-T.H.; validation, N.-T.H.; formal analysis, C.-S.C.; investigation, C.-S.C.; resources, C.-S.C.; data curation, C.-S.C. and N.-T.H.; writing—original draft preparation, N.-T.H.; writing—review and editing, N.-T.H.; visualization, C.-S.C.; supervision, C.-S.C. and N.-T.H.; project administration, C.-S.C. and N.-T.H.; funding acquisition, C.-S.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Ministry of Science and Technology (MOST), grant number: MOST 111-2221-E-239-031 and MOST 109-2221-E-027-044-MY3.

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. Chen, C.S.; Li, T.C.; Hu, N.T. The Gripping Posture Prediction of Eye-in-hand Robotic Arm Using Min-Pnet. In Proceedings of the 2022 International Conference on Advanced Robotics and Intelligent Systems (ARIS), Taipei, Taiwan, 24–27 August 2022; pp. 1–5.
2. Kim, H.-S.; Park, J.; Bae, M.; Park, D.; Park, C.; Do, H.M.; Choi, T.; Kim, D.-H.; Kyung, J. Advanced 2-DOF Counterbalance Mechanism Based on Gear Units and Springs to Minimize Required Torques of Robot Arm. *IEEE Robot. Autom. Lett.* **2022**, *7*, 6320–6326. [CrossRef]
3. Li, Z.; Li, S.; Luo, X. Using Quadratic Interpolated Beetle Antennae Search to Enhance Robot Arm Calibration Accuracy. *IEEE Robot. Autom. Lett.* **2022**, *7*, 12046–12053. [CrossRef]
4. 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]
5. Righi, M.; Magrini, M.; Dolciotti, C.; Moroni, D. A Case Study of Upper Limb Robotic-Assisted Therapy Using the Track-Hold Device. *Sensors* **2022**, *22*, 1009. [CrossRef]
6. Borst, C.; Fischer, M.; Hirzinger, G. Grasp Planning: How to Choose a Suitable Task Wrench Space. In Proceedings of the IEEE International Con-ference on Robotics and Automation (ICRA), New Orleans, LA, USA, 26 April–1 May 2004; pp. 319–325.

7. Tang, T.; Lin, H.C.; Zhao, Y.; Chen, W.; Tomizuka, M. Autonomous alignment of peg and hole by force/torque measurement for robotic assembly. In Proceedings of the 2016 IEEE International Conference on Automation Science and Engineering (CASE), Fort Worth, TX, USA, 21–25 August 2016; pp. 162–167. [CrossRef]
8. Luo, J.; Solowjow, E.; Wen, C.; Ojea, J.A.; Agogino, A.M. Deep Reinforcement Learning for Robotic Assembly of Mixed Deformable and Rigid Objects. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 2062–2069. [CrossRef]
9. Klingbeil, E.; Rao, D.; Carpenter, B.; Ganapathi, V.; Ng, A.Y.; Khatib, O. Grasping with Application to an autonomous checkout robot. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China, 9–13 May 2011; pp. 2837–2844.
10. Chen, N.; Westling, G.; Edin, B.B.; van der Smagt, P. Estimating Fingertip Forces, Torques, and Local Curvatures from Fingernail Images. *Robotica* **2020**, *38*, 1242–1262. [CrossRef]
11. Cohen, Y.; Bar-Shira, O.; Berman, S. Motion Adaptation Based on Learning the Manifold of Task and Dynamic Movement Primitive Parameters. *Robotica* **2021**, *39*, 1299–1315. [CrossRef]
12. Yao, S.; Ceccarelli, M.; Carbone, G.; Dong, Z. Grasp configuration planning for a low-cost and easy-operation underactuated three-fingered robot hand. *Mech. Mach. Theory* **2018**, *129*, 51–69. [CrossRef]
13. Park, H.; Park, J.; Lee, D.-H.; Park, J.-H.; Bae, J.-H. Compliant Peg-in-Hole Assembly Using Partial Spiral Force Trajectory With Tilted Peg Posture. *IEEE Robot. Autom. Lett.* **2020**, *5*, 4447–4454. [CrossRef]
14. Levine, S.; Pastor, P.; Krizhevsky, A.; Ibarz, J.; Quillen, D. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *Int. J. Robot. Res.* **2018**, *37*, 421–436. [CrossRef]
15. Calli, B.; Wisse, M.; Jonker, P. Grasping of unknown objects via curvature maximization using active vision. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), San Francisco, CA, USA, 25–30 September 2011; pp. 995–1001.
16. Yu, S.; Zhai, D.-H.; Wu, H.; Yang, H.; Xia, Y. Object recognition and robot grasping technology based on RGB-D data. In Proceedings of the 2020 39th Chinese Control Conference (CCC), Shenyang, China, 27–29 July 2020. [CrossRef]
17. Bae, J.-H.; Jo, H.; Kim, D.-W.; Song, J.-B. Grasping System for Industrial Application Using Point Cloud-Based Clustering. In Proceedings of the 2020 20th International Conference on Control, Automation and Systems (ICCAS), Busan, Republic of Korea, 13–16 October 2020. [CrossRef]
18. Jeng, K.Y.; Liu, Y.C.; Liu, Z.Y.; Wang, J.W.; Chang, Y.L.; Su, H.T.; Hsu, W.H. GDN: A Coarse-To-Fine (C2F) Representation for End-To-End 6-DoF Grasp Detection. *arXiv* **2020**, arXiv:2010.10695.
19. Pas, A.T.; Platt, R. Using Geometry to Detect Grasp Poses in 3D Point Clouds. In *Robotics Research*; Springer: Cham, Switzerland, 2018; pp. 307–324. [CrossRef]
20. Liang, H.; Ma, X.; Li, S.; Görner, M.; Tang, S.; Fang, B.; Zhang, J. PointNetGPD: Detecting Grasp Configurations from Point Sets. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 3629–3635.
21. Mousavian, A.; Eppner, C.; Fox, D. 6-DOF GraspNet: Variational Grasp Generation for Object Manipulation. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 2901–2910.
22. Varadarajan, K.M.; Zhou, K.; Vincze, M. Holistic Shape Detection and Analysis using RGB-D Range Data for Grasping. Available online: https://www.semanticscholar.org/paper/Holistic-Shape-Detection-and-Analysis-using-RGB-D-Varadarajan-Zhou/bf74c4e2453608042c23ab94a94edc1e68046e19 (accessed on 1 December 2022).
23. Czajewski, W.; Kołomyjec, K. 3D Object Detection and Recognition for Robotic Grasping Based on RGB-D Images and Global Features. *Found. Comput. Decis. Sci.* **2017**, *42*, 219–237. [CrossRef]
24. Kingry, N.; Jung, M.; Derse, E.; Dai, R. Vision-Based Terrain Classification and Solar Irradiance Mapping for Solar-Powered Robotics. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 5834–5840. [CrossRef]
25. Kang, H.; Zhou, H.; Wang, X.; Chen, C. Real-Time Fruit Recognition and Grasping Estimation for Robotic Apple Harvesting. *Sensors* **2020**, *20*, 5670. [CrossRef] [PubMed]
26. Vignesh, T.; Karthikeyan, P.; Sridevi, S. Modeling and trajectory generation of bionic hand for dexterous task. In Proceedings of the 2017 IEEE International Conference on Intelligent Techniques in Control, Optimization and Signal Processing (INCOS), Srivilliputtur, India, 23–25 March 2017; pp. 1–6. [CrossRef]
27. Zhu, Q. Teleoperated Grasping Using an Upgraded Haptic-Enabled Human-Like Robotic Hand and a Cyber Touch Glove. Ph.D. Thesis, University of Ottawa, Ottawa, ON, Canada, 2020.
28. Chen, X.; Li, Z.; Wang, Y.; Liu, J. Effect of fruit and hand characteristics on thumb–index finger power-grasp stability during manual fruit sorting. *Comput. Electron. Agric.* **2019**, *157*, 479–487. [CrossRef]
29. Anzai, Y.; Sagara, Y.; Kato, R.; Mukai, M. Development of a foldable five-finger robotic hand for assisting in laparoscopic surgery. In Proceedings of the 2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN), New Delhi, India, 14–18 October 2019; pp. 1–6. [CrossRef]
30. Chao, Y.; Chen, X.; Xiao, N. Deep learning-based grasp-detection method for a five-fingered industrial robot hand. *IET Comput. Vis.* **2019**, *13*, 61–70. [CrossRef]

31. Wang, C.; Freer, D.; Liu, J.; Yang, G.-Z. Vision-based Automatic Control of a 5-Fingered Assistive Robotic Manipulator for Activities of Daily Living. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019; pp. 627–633. [CrossRef]
32. Ji, S.-Q.; Huang, M.-B.; Huang, H.-P. Robot Intelligent Grasp of Unknown Objects Based on Multi-Sensor Information. *Sensors* **2019**, *19*, 1595. [CrossRef]
33. Xu, Z.; Yang, C.; Wu, W.; Wei, Q. Design of Underwater Humanoid Flexible Manipulator Motion Control System Based on Data Glove. In Proceedings of the 2020 6th International Conference on Mechatronics and Robotics Engineering (ICMRE), Barcelona, Spain, 12–15 February 2020; pp. 120–124. [CrossRef]
34. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* **2018**, arXiv:1804.02767.
35. Bochkovskiy, A.; Wang, C.; Liao, H. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**, arXiv:2004.10934.
36. Zheng, S. Network Intrusion Detection Model Based on Convolutional Neural Network. In Proceedings of the 2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Chongqing, China, 12–14 March 2021; Volume 5, pp. 634–637. [CrossRef]
37. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 24–29 July 2016; pp. 652–660.
38. Rusu, R.B.; Cousins, S. 3d is here: Point cloud library (pcl). In Proceedings of the IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 1–4.
39. Tsai, J.S.-H.; Hu, N.T.; Yang, P.C.; Guo, S.M.; Shieh, L.-S. Modeling of decentralized linear observer and tracker for a class of unknown interconnected large-scale sampled-data nonlinear systems with closed-loop decoupling property. *Comput. Math. Appl.* **2010**, *60*, 541–562. [CrossRef]
40. Hamzaoui, A.; Essounbouli, N.; Benmahammed, K.; Zaytoon, J. State Observer Based Robust Adaptive Fuzzy Controller for Nonlinear Uncertain and Perturbed Systems. *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **2004**, *34*, 942–950. [CrossRef] [PubMed]
41. Shang, S.-Y. Eye-in-hand Robotic Arm Gripping System Based on Two Dimensional Object Recognition Using Machine Learning and Three Dimensional Object Posture Estimation. Available online: https://hdl.handle.net/11296/n9nqt6 (accessed on 1 December 2022).