Hindawi
Scientific Programming
Volume 2021, Article ID 7944300, 10 pages
https://doi.org/10.1155/2021/7944300

Research Article

Automatic Manipulator Tracking Control Based on Moving Target Trajectory Prediction

Haifeng Luo

Hubei University of Education, Wuhan 430205, China

Correspondence should be addressed to Haifeng Luo; haifengluo@hue.edu.cn

Received 12 October 2021; Accepted 17 November 2021; Published 30 November 2021

Academic Editor: Ahmed Farouk

The core issue of automatic manipulator tracking control is how to ensure the given moving target follows the expected trajectory and adapts to various uncertain factors. However, the existing moving target trajectory prediction methods rely on highly complex and accurate models, lacking the ability to generalize different automatic manipulator tracking scenarios. Therefore, this study tries to find a way to realize automatic manipulator tracking control based on moving target trajectory prediction. In particular, a moving target trajectory prediction model was established, and its parameters were optimized. Next, a tracking-training-testing algorithm was proposed for manipulator's automatic moving target tracking, and the operating flows were detailed for training module, target detection module, and target tracking module. The proposed model and algorithm were proved effective through experiments.

1. Introduction

With the rapid development of industrial technology, manipulators have been successfully applied to original manual operations, becoming the most widely used manmade tool for industrial production [1–6]. The application of manipulators makes production more efficient and flexible. The core issue of automatic manipulator tracking control is how to ensure the given moving target follows the expected trajectory and adapts to various uncertain factors [7–11]. It is of great practical significance to derive an automatic tracking control strategy for moving targets under uncertainties and external interference.

Target tracking is an important prerequisite for manipulator-assisted services. Zhu et al. [12] improved the near-field computer vision system for intelligent fire robots. The improved system can predict the falling jet path under the complex light environment and interference during firefighting, identify the jet path based on length and area ratio, and parametrize and extract the features of jet path by superposing radial centroids. Wu et al. [13] adopted a human-following method suitable for a manipulator containing visual sensors with a limited perception range, integrated two physical motion models into an adaptive trajectory prediction algorithm, and improved the prediction accuracy by adaptive adjustment of model parameters. For the trajectory control of Par4 parallel robot, Zhang and Ming [14] designed a type 2 fuzzy predictive compensation proportional-integral-derivative (PID) controller based on the improved dynamic gray wolf optimizer (GWO) based on the mutation operator and the eliminating-reconstructing mechanism (DMR-GWO2). The proposed controller speeds up the response of the parallel robot and improves the adaptability of the entire system.

In actual conditions, two manipulators are often needed to pick up and place moving objects through the planning and execution of collision-free trajectories. Tika et al. [15] put forward a layered control strategy for collaborative picking and placement tasks in a narrow, shared workspace and realized the synchronous execution of task scheduling in top-level design, path planning, and robot tasks. Xia et al. [16] proposed a visual prediction framework based on time granularity. The core of the framework is an integrated moving target prediction module based on multiple long short-term memory (LSTM) neural network. Compared with the latest prediction algorithms, the framework excels

in prediction accuracy, success rate, and robustness. Focusing on the action understanding of mirror neurons, Zhong et al. [17] simulated the walking mode of humanoid robots and predicted the moving direction according to the previous walking trajectory.

Trajectory prediction is the last step in the visual perception of the manipulator. After a series of segmentation, detection, and tracking, the algorithm could determine the type, bounding box, and other information of the object. However, the future movement trend and trajectory of the target must be predicted to realize automatic tracking control. To sum up, the traditional trajectory prediction methods for moving targets mainly rely on features such as color and contour. The recognition effect is very poor, if the target has multiple features. Moreover, the existing moving target trajectory prediction methods rely on highly complex and accurate models, lacking the ability to generalize different automatic manipulator tracking scenarios [18–22]. Therefore, this study develops an approach for automatic manipulator tracking control based on moving target trajectory prediction, aiming to improve the manipulator's trajectory prediction accuracy and automatic tracking control effect. Section 2 establishes a moving target trajectory prediction model and optimizes its parameters. The established model can predict the position and pose of irregular moving objects at the same time and boast a strong generalization ability. Section 3 details the principles of the tracking-training-testing algorithm for manipulator's automatic moving target tracking and explains the operating flows for the training module, target detection module, and target tracking module in the algorithm. The proposed model and algorithm were proved effective through experiments.

This study solves the problems of the manipulator in recognition, positioning, and trajectory prediction of moving objects, models the error in target tracking, and tests the feasibility of the proposed method through tracking experiments. The internal parameters of the proposed trajectory prediction network for moving objects were all trained on datasets. The training ensures the degree of modularity and generalization ability of the network. However, the prediction precision of our network could be further improved by changing network structure and modifying network parameters, when the network is applied to predict the position and pose of complex and irregular moving targets.

2. Moving Target Trajectory Prediction Model

The precision of moving target trajectory prediction hinges on the accuracy of motion model. This study establishes a moving target trajectory prediction model based on LSTM, which is known for its good accuracy and generalizability, and further enables the manipulator to recognize, and automatically track and control the moving target.

2.1. Model Construction. To accurately predict moving target trajectory, this study imports the three-dimensional (3D) spatial position of a moving target from time h to time $h+K$ into the trajectory prediction model, which outputs the 3D spatial position of the moving target at time $h+K+1$.

Figure 1 shows the overall structure of our moving target trajectory prediction model. The model consists of an input layer, a hidden layer, an output layer, and a training module. In the input layer, a complete sequence of moving target trajectories $G = \{g_1, g_2, \ldots, g_m\}$ is subjected to Z-score normalization:

$$G^* = \{g_1^*, g_2^*, \ldots, g_m^*\},$$
$$g_h^{*'} = \frac{\left(g_h - \sum_{h=1}^{m} g_h/m\right)}{\sqrt{\sum_{h=1}^{n} \left(g_h - \sum_{h=1}^{m} g_h/m\right)^2/m}}. \tag{1}$$

To satisfy the input requirements of the hidden layer, the input data were segmented. Let K be the prediction step length of the model. Then, the tensor of the input data after the segmentation can be expressed as follows:

$$A = \{A_1, A_2, \ldots, A_K\}. \tag{2}$$

Batch processing is applied on the input data to fully utilize computer resources and improve the training efficiency of the neural network. That is, A is treated as a tensor composed of a batch of 3D spatial coordinates $[r, K, 3]$, where r is the number of batch processing samples. The training accuracy of the model must ensure that each batch of data is a complete trajectory of the moving target; i.e., the batch size should be defined as $(m-K)$. Then, we have the following equation:

$$A_o = \{g_o^*, g_{o+1}^*, \ldots \infty, g_{m-K+o-1}^*\} 1 \le o \le K; o, K \in m. \tag{3}$$

The theoretical output of the input layer can be expressed as follows:

$$B = \{B_1, B_2, \ldots, B_K\},$$
$$B_o = \{g_{o+1}^*, g_{o+2}^*, \ldots, g_{m-K+o}^*\}. \tag{4}$$

The hidden layer in the trajectory prediction model contains K LSTM nodes, which are connected in chronological order. Each node has F LSTM units. The output of the hidden layer can be expressed as follows:

$$O = \{O_1, O_2, \ldots, O_K\}. \tag{5}$$

The dimensionality $[r, K, F]$ of O should be consistent with that of model output. Let ω_{ti} be the weight of a fully connected layer, and t be the output of the output layer. Before outputting the predicted position of the moving target, the data must be handled by a fully connected layer:

$$t_h = \sum_{i=1}^{F} \omega_{ti} O_h. \tag{6}$$

To test the prediction accuracy, the number r of batch processing samples is set to 1. The first K 3D spatial coordinates of a complete trajectory in the test set are imported:

$$A_g = \{g_1^*, g_2^*, \ldots, g_K^*\}. \tag{7}$$

Based on the input A_g, the model outputs the predicted trajectory:

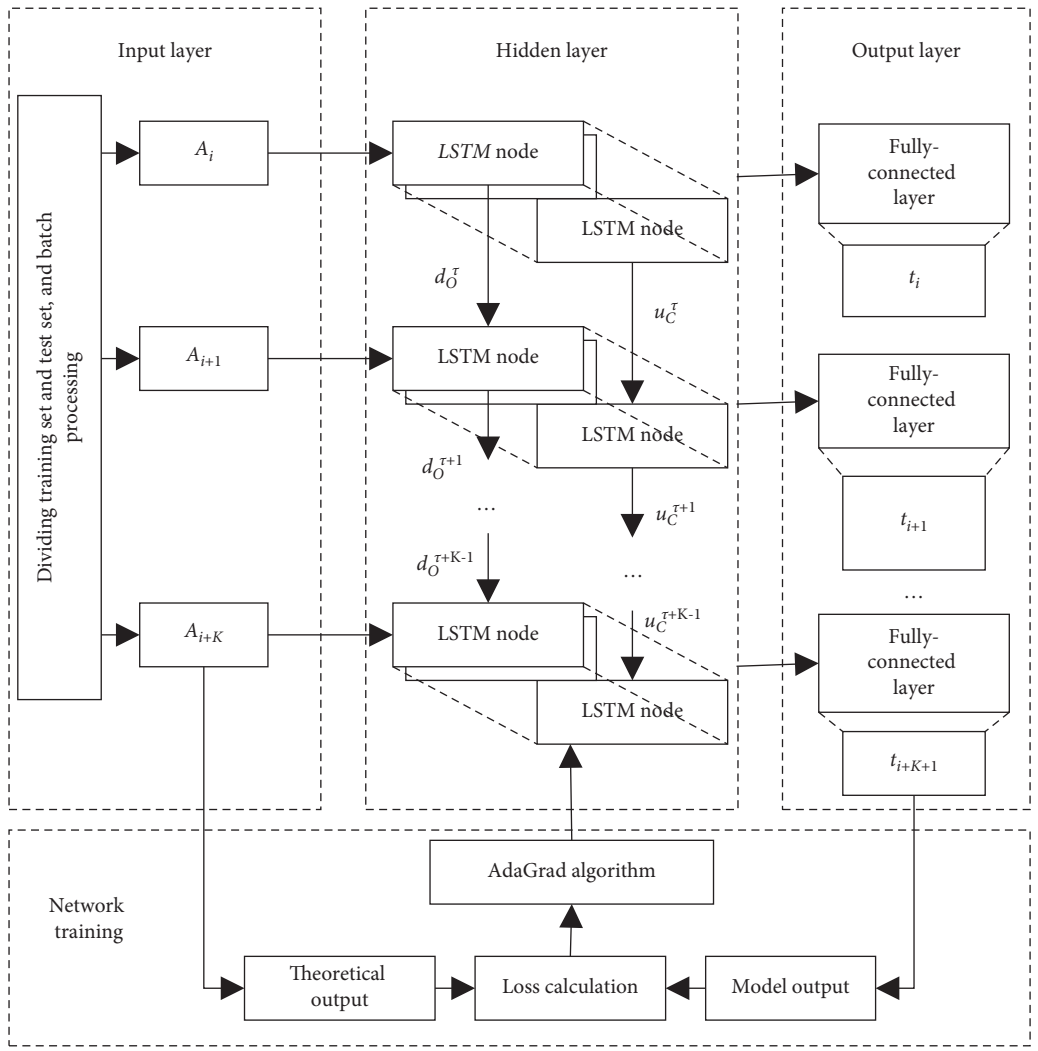

Figure 1: Overall structure of moving target trajectory prediction model.

$$t_g = \{t_2, t_3, \ldots, t_{K+1}\}. \tag{8}$$

Let t_{K+1} be the 3D spatial position predicted for the moving target at time $K+1$. This position is merged with the last $K-1$ 3D spatial positions in A_g to obtain the new input for the trajectory prediction model:

$$A_{g+1} = \{g_2^*, g_3^*, \ldots, g_K^*, t_{K+1}\}. \tag{9}$$

Then, A_{g+1} is imported to the trajectory prediction model. The model will output the predicted 3D spatial position t_{K+2} of the moving object at time $K+2$. The above steps are iteratively executed, and the final prediction of the 3D spatial position of the moving object can be obtained as follows:

$$t = \{t_{K+1}, t_{K+2}, \ldots, t_m\}. \tag{10}$$

The fitting and prediction accuracy of the model can be quantified by the error between input A and output t.

Both the predicted value and theoretical output of the trajectory prediction model are 3D spatial coordinates. The loss of the model is calculated by the Euclidean loss function. Let b be the theoretical output of the model. The error between predicted value and theoretical output can be calculated by the following equation:

$$K(t, b) = \frac{1}{2M} \sum_{m-1}^{M} \|t - b\|_2^2. \tag{11}$$

The model training aims to gradually reduce the value of the loss function. Based on the AdaGrad algorithm, the learning rate δ of our model is updated automatically. Let ξ be the small constant to prevent denominator from being zero; ω be the weight parameter of the model. Then, the model can be updated by the following equation:

$$\delta_m = \frac{\delta}{\xi + \sqrt{\sum_{i=1}^{m-1} p_i \cdot p_i}},$$
$$p = \frac{1}{r} \sum_{i=1}^{r} \frac{\partial K_i(t, b)}{\partial \omega}. \tag{12}$$

2.2. Model Parameter Optimization. There are many parameters in our trajectory prediction model. The most critical ones include prediction step length K, the number of hidden nodes F, and the learning rate δ. To weaken their influence on the prediction of moving target trajectory, this study firstly evaluates the prediction accuracy on all test samples and then chooses the optimal combination of K, F, and δ, which leads to the highest prediction accuracy. The objective function can be expressed as follows:

$$\min \sigma(t, A),$$
$$s.t. \begin{cases} 10 \le K \le K_{\max} \le m, \text{step}_K | K \\ 10 \le F \le F_{\max}, \text{step}_F | F \\ 0.005 \le \delta \le \delta_{\max}, \text{step}_\delta | \delta \\ K, F, \delta, \text{step}_K, \text{step}_F, \text{step}_\delta \in M \end{cases}. \tag{13}$$

The multilayer grid search algorithm is adopted to process K, F, and δ to determine the best values of these crucial parameters. The grid search is carried out from inside to outside in three steps:

Step 1. Set the number of batch processing samples r and number of training steps T_{steps}, which are two key parameters, and preset the value ranges of K, F, and δ based on formula (13).

Step 2. Traverse K, F, and δ layer by layer, and implement model training and prediction in the innermost layer. After the training, maintain the three parameters to obtain the fitting and prediction accuracies of the model.

Step 3. Sort the parameter search results in descending order by the prediction accuracy, and select the K, F, and δ in the top-ranking combination for the optimal model.

3. Automatic Tracking Control Algorithm

3.1. Algorithm Principles. Based on machine vision, manipulator moving target tracking might involve multiple moving targets at a time and needs to consider multiple motion states of each target. The moving targets face changes in moving direction, speed, color, and brightness, and could be occluded by obstacles. Therefore, the tracking technology should be able to detect the 3D spatial position of each moving target in real time and judge whether the target is missing or occluded. This study proposes a tracking-training-testing algorithm for manipulator's automatic moving target tracking and combines the algorithm with moving target trajectory prediction to enable manipulators to grasp, as well as automatically track and control targets.

The automatic tracking algorithm can select the moving target from each frame image of the video stream. The architecture of the algorithm is shown in Figure 2. The training module processes the detection result of the target detection module and the tracking result of the target tracking module. The processing and feedback results from the training module are used to update the target detection module and the target tracking module. This cyclic optimization process can handle complex situations, such as the appearance changes in the moving target over time and the temporary disappearance of the moving target from the shooting range, thereby ensuring the target identification and tracking effects of the algorithm.

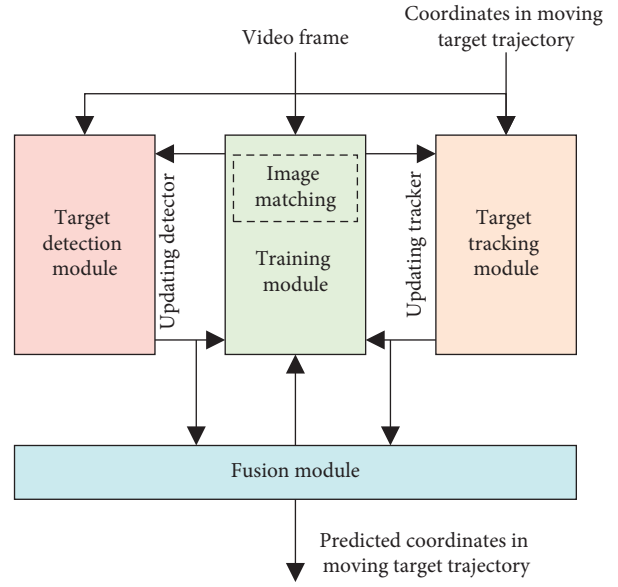

Figure 2: Architecture of manipulator's automatic moving target tracking algorithm.

Let GYH be the normalized cross-correlation coefficient. To select the moving target from the video frame, the similarity between two adjacent frames w_i and w_j must be defined before analyzing the main modules:

$$\text{RE}(w_i, w_j) = 0.5 * (\text{GYH}(w_i, w_j) + 1). \tag{14}$$

The matching image set N containing both positive samples w_i^+ and negative samples w_i^- of moving targets can be expressed as follows:

$$N = \{w_1^+, w_2^+, \ldots, w_n^+, w_1^-, w_2^- \ldots w_n^-\}. \tag{15}$$

Then, n positive sample w_i^+ and n negative samples w_i^- are sorted in the order of $i = 1, 2, 3, \ldots, n$ and then added to the matching image set.

The similarity between a matching image N_G and each frame w can be divided into the similarity with the nearest neighbor of w_i^+ and the similarity with the nearest neighbor of w_i^-:

$$\begin{aligned} \text{RE}^+(w, N_G) &= \max_{w_i^+ \in N_G} \text{RE}(w, w_i^+), \\ \text{RE}^-(w, N_G) &= \max_{w_i^- \in N_G} \text{RE}(w, w_i^-). \end{aligned} \tag{16}$$

The similarity between the frame w and the labeled first half of the positive samples can be calculated by the following equation:

$$\text{RE}_{1/2}^+ = (w, N_G) = \max_{w_i^{+FL < N_G/2}} \text{RE}(w, w_i^+). \tag{17}$$

The cross-correlation of w can be calculated by the following equation:

$$\mathrm{RE}^s = \frac{\mathrm{RE}^+}{\mathrm{RE}^+ + \mathrm{RE}^-}. \quad (18)$$

Formula (18) shows that the value of RE^s falls in [0, 1]. The greater the RE^s, the more credible that the frame contains a moving target. The conservative similarity of w can be calculated by the following equation:

$$\mathrm{RE}^d = \frac{\mathrm{RE}^+_{1/2}}{\mathrm{RE}^+_{1/2} + \mathrm{RE}^-}. \quad (19)$$

The cross-correlation obtained by formula (18) is the threshold for the nearest neighbor classifier that determines the similarity (RE^s, RE^d) between frame w and matching image N_G. If $\mathrm{RE}^s(w, N) > \rho_{\mathrm{MM}}$, w is a positive sample; if $\mathrm{RE}^s(w, N) < \rho_{\mathrm{MM}}$, w is a negative sample. Here, $\mathrm{RE}^s(w, N)$-ρ_{MM} is the classification threshold ensuring the convergence of the classifier.

3.2. Target Detection Module. The variance classifier is the first link of the cascade classifier in the target detection module. Let $Q(w)$ be the expectation of w solved by the integral image method. Then, the variance of any frame w can be calculated by the following equation:

$$Q\left(w^2\right) - Q^2(w). \quad (20)$$

If the total variance of gray values for all pixels in the frame within the window is smaller than half of the total variance of gray values for all pixels in the moving target box, then the window is invalid and needs to be removed. In this way, half of the image contents, including ground and shadows, can be eliminated.

The ensemble classifier is the second link of the cascade classifier in the target detection module. The frame outputted by the variance classifier is imported to the ensemble classifier composed of m basic classifiers. Here, each basic classifier is a decision tree (DT). The output of classifier i is a posterior probability vector composed of code a:

$$\mathrm{GV}_1(b|a), b \in (0, 1). \quad (21)$$

The m classifiers output m posterior probability vectors. The mean of all vectors can be calculated by the following equation:

$$\mathrm{GV}^* = \frac{\left(\sum_{i=0}^{m} \mathrm{GV}_i(b|a)\right)}{m}, b \in (0, 1). \quad (22)$$

If $\mathrm{GV}^* > 1/2$, the window is retained; if $\mathrm{GV}^* > 1/2$, the window is eliminated.

As the eigenvalue of the frame, the combined code vector is distributed to all the basic classifiers of the ensemble classifier. Each basic classifier corresponds to a posterior probability. The ith posterior probability is denoted as $\mathrm{GV}_i(b|a)$. If the posterior probability of each basic classifier is described by binary code a, then

$$\mathrm{GV}_i(b|a) = \frac{\Delta w}{\Delta w + \Delta m}, \quad (23)$$

where

$$\begin{cases} \Delta w = M\left(w^+\right) \\ \Delta m = M\left(w^-\right) \end{cases}. \quad (24)$$

During initialization, $w_i(b|a) = 0$, and the posterior probability corresponding to each basic classifier characterizes a negative sample. During later training, the ensemble classifier classifies the labeled frames and updates $w_i(b|a)$ ($b|a$) (as shown in Figure 3).

Most unqualified contents are eliminated from the input frame through the filtering by both variance filter and ensemble filter. The filtered results are further processed by the nearest neighbor classifier. If $\mathrm{RE}^s(w, N) > \omega_{\mathrm{MM}}$, the frame content in the scanning window is a positive sample.

3.3. Target Tracking Module. The target tracking module combines the Lucas–Kanade (LK) optical flow method with the forward and backward error tracking theory. The forward and backward directions refer to the positive and negative directions of the sequence of video frames, respectively. If there is a large error between the target tracking results in the two directions, then the predicted trajectory of the moving target must be incorrect and unreliable. The forward-backward error helps to judge whether the moving target is tracked successfully, but cannot identify unobvious errors in trajectory prediction. Therefore, this study designs an image frame difference comparison method for slow-moving target tracking points. The frame sequence of slow-moving target can be expressed as follows:

$$\mathrm{FD} = \left(J_\tau, J_{\tau+1}, \ldots, J_{\tau+v}\right). \quad (25)$$

Let A_τ be the coordinates of the moving target at time τ; v be the times of forward tracking of point A_τ. Then, the forward trajectory tracking sequence of the moving target can be given by the following equation:

$$\psi_x^v = \left(A_\tau, A_{\tau+1}, \ldots, A_{\tau+v}\right). \quad (26)$$

The forward tracking and backward tracking are denoted by subscripts x and y, respectively. Then, the pixel coordinates $A_{\tau+v}$ are backward tracked to the previous frame. Then, the backward trajectory tracking sequence can be given by the following equation:

$$\psi_y^v = \left(\dot{A}_\tau, \dot{A}_{\tau+1}, \ldots, \dot{A}_{\tau+v}\right). \quad (27)$$

Combining formulae (26) and (27), the tracking error of the moving object can be obtained by the following equation:

$$\mathrm{FB}\left(\psi_x^v|\mathrm{FD}\right) = \mathrm{distance}\left(\psi_x^v, \psi_y^v\right). \quad (28)$$

To sum up, the forward and backward tracking errors can be obtained by formula (28), as long as a suitable threshold is determined for different image sequences. Then, it is possible to judge the success or failure of target tracking. Figure 4 illustrates the flow of tracking error calculation.

3.4. Training Module. The training module contains the classifier to be trained, labeled training set, positive/negative training sample generator, etc. The classifier is trained on the

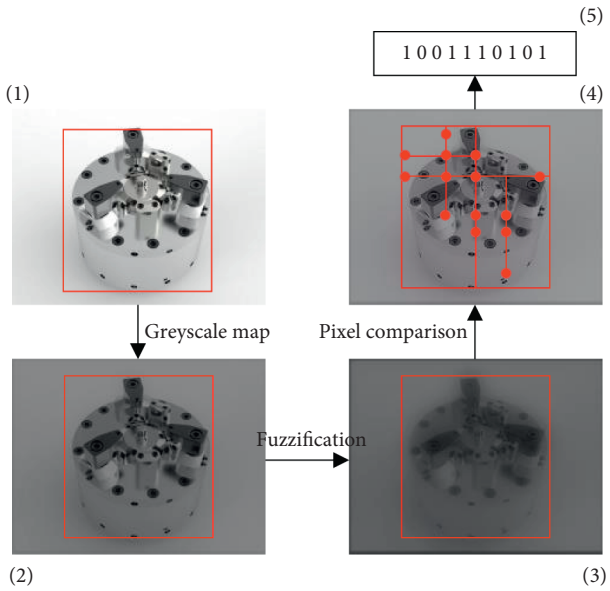

Figure 3: Generation of binary code. Note: subgraph (1) is the input image + window; subgraph (2) is the grayscale map of the input image + window; subgraph (3) is the fuzzified image- + window; subgraph (4) is the pixel comparison image + window; and subgraph (5) is the final binary code.

training set to achieve comprehensive integrated learning. Figure 5 explains the flow of the training module. During classifier training, the training quality is closely associated with the absolute number of labeled positive and negative samples. Hence, the training module should be able to quantify the relationship between the classifier performance and the absolute number of samples. The classifier performance can be characterized by the reliability of positive sample labels, the incorrect detection probability of negative samples, the accuracy of negative sample labels, and the incorrect detection probability of positive samples.

The reliability of positive sample labels can be characterized by the ratio of the number of correctly detected positive samples m_C^+ to the sum of the number of correctly detected positive samples and the number of incorrectly detected positive samples $m_C^+ + m_E^+$:

$$\mathrm{GV}^+ = \frac{m_C^+}{\left(m_C^+ + m_E^+\right)}. \tag{29}$$

The incorrect detection probability of negative samples can be characterized by the ratio of the number of correctly detected positive samples m_C^+ to the number of incorrectly detected negative samples Φ:

$$S^+ = \frac{m_C^+}{\Phi}. \tag{30}$$

The reliability of negative sample labels can be characterized by the ratio of the number of correctly detected negative samples m_C^- to the sum of the number of correctly detected negative samples and the number of incorrectly detected negative samples $m_C^- + m_E^-$:

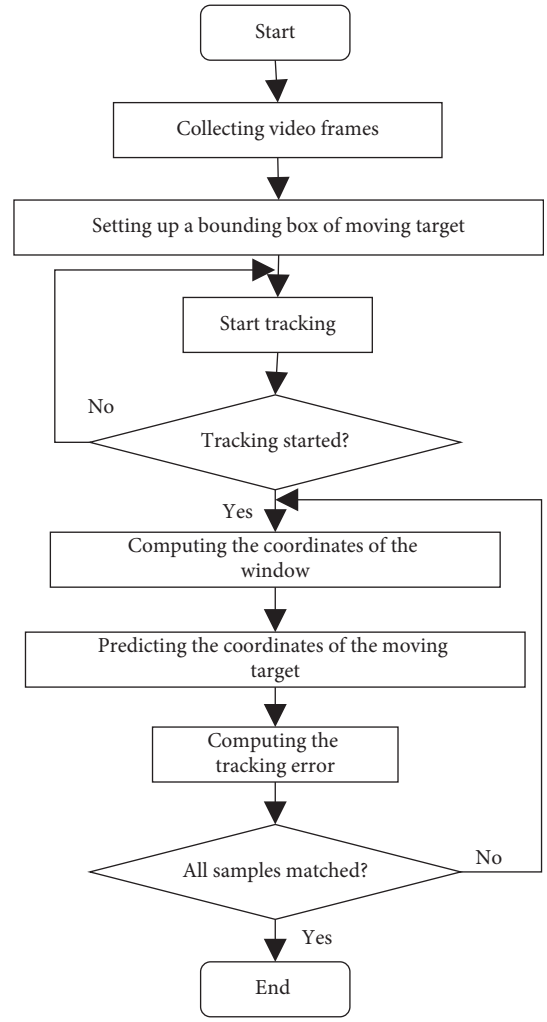

Figure 4: Flow of tracking error calculation.

$$\mathrm{GV}^- = \frac{m_C^-}{\left(m_C^- + m_E^-\right)}. \tag{31}$$

The incorrect detection probability of positive samples can be characterized by the ratio of the number of correctly detected negative samples m_C^- to the number of incorrectly detected positive samples Ω:

$$S^- = \frac{m_C^-}{\Omega}. \tag{32}$$

The classifier performance evaluation equations (29)–(32) must satisfy the following equation:

$$\begin{aligned} m_C^-(v) &= S^+\Phi(v), m_E^+(v) = \frac{\left(1-\mathrm{GV}^+\right)}{\mathrm{GV}^+}S^+\Phi(v), \\ m_C^-(v) &= S^-\Omega(v), m_E^-(v) = \frac{\left(1-\mathrm{GV}^-\right)}{\mathrm{GV}^-}S^-\Omega(v). \end{aligned} \tag{33}$$

The number of incorrectly detected negative samples Φ and the number of incorrectly detected positive samples Ω can be, respectively, updated by the following equation:

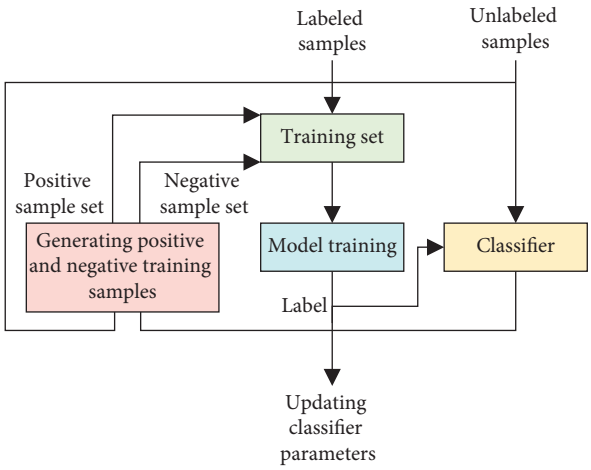

Figure 5: Flow of training module.

$$\Omega(v+1) = (1-S^{-})\Omega(v) + \frac{(1-\mathrm{GV}^{+})}{\mathrm{GV}^{+}}S^{+}\Phi(v),$$
$$\Phi(v+1) = \frac{(1-\mathrm{GV}^{-})}{\mathrm{GV}^{-}}S^{-}\Omega(v) + (1-S^{+})\Phi(v). \tag{34}$$

Assume $\dot{O}(v) = [\Phi(v)\cdot\Omega(v)]^{T}$. Then, a 2×2 matrix Q can be defined as follows:

$$Q = \begin{bmatrix} 1-S^{-} & \dfrac{(1-\mathrm{GV}^{+})}{\mathrm{GV}^{+}}S^{+} \\ \dfrac{(1-\mathrm{GV}^{-})}{\mathrm{GV}^{-}}S^{-} & (1-S^{+}) \end{bmatrix}. \tag{35}$$

After rewriting formulae (25) and (26) as matrices, the recursive formula of $\dot{O}(v)$ can be established as follows:

$$\dot{O}(v+1) = Q\cdot\dot{O}(v). \tag{36}$$

The above formula shows that the recursive system of the manipulator's moving target tracking is both discrete and dynamic. Thus, the ultimate control goal of our algorithm is to gradually reduce the system error increment to zero, with the growing number of iterations.

4. Experiments and Result Analysis

The multilayer search algorithm was introduced to optimize the three parameters K, F, and δ of the proposed moving target trajectory model. Firstly, the number of the training steps was set to 120, and the value ranges of the three parameters were preset as follows: $K\in\{15, 20, 25, 30\}$, $F\in\{60, 120, 180, 240\}$, and $\delta\in\{0.01, 0.02, \ldots, 0.1\}$. The objective function is to maximize the prediction accuracy of moving target trajectory, i.e., minimize the prediction error. The possible parameter combinations were sorted in descending order of error. Table 1 lists the top five parameter combinations and their errors. It can be seen that the optimization of the three parameters greatly enhanced the accuracy of our moving target trajectory model.

Table 1: Top five parameter combinations and their errors.

Ranking	Model parameters			Training error	Test error
	K	H	δ		
1	30	60	0.08	$5.0441e-06$	0.001753
2	30	180	0.08	$4.2582e-06$	0.002096
3	30	60	0.01	$4.7857e-06$	0.002162
4	30	60	0.02	$5.6412e-06$	0.001781
5	30	120	0.06	$4.4325e-06$	0.002318

Table 2: Prediction results of different models.

Models	Model parameters			Training error	Test error
	K	H	δ		
RNN	30	60	0.08	$9.4781e-06$	0.089156
GRNN	30	60	0.08	$6.5428e-06$	0.036843
Our model	30	60	0.08	$5.2657e-06$	0.011891

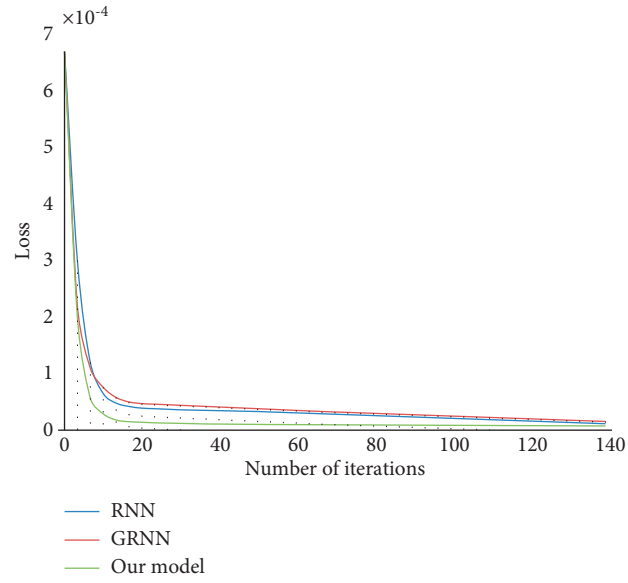

Figure 6: Loss variations of different prediction models.

The three key parameters of the moving target trajectory prediction model were optimized as $K=30$, $H=60$, and $\delta=0.08$. Next, the hidden units in the hidden layer nodes were configured as recurrent neural network (RNN) and gated RNN (GRNN). The prediction results of these two models were compared with those of our model (Table 2). Our model achieved better training accuracy and test accuracy than RNN and GRNN.

Figure 6 records the loss variations of different prediction models during the training. Overfitting occurs to the RNN when the training lasts too long; i.e., the number of iterations is too large. As shown in Figure 6, the loss of the RNN dropped the fastest, but the loss of our model gradually moved below that of RNN and GRNN, with the growing number of iterations.

Table 3: Prediction errors of different models.

Models	Model parameters			Trajectory coordinate error
	K	H	δ	
RNN	30	60	0.08	0.9874
GRNN	30	60	0.08	0.5592
Our model	30	60	0.08	0.4275

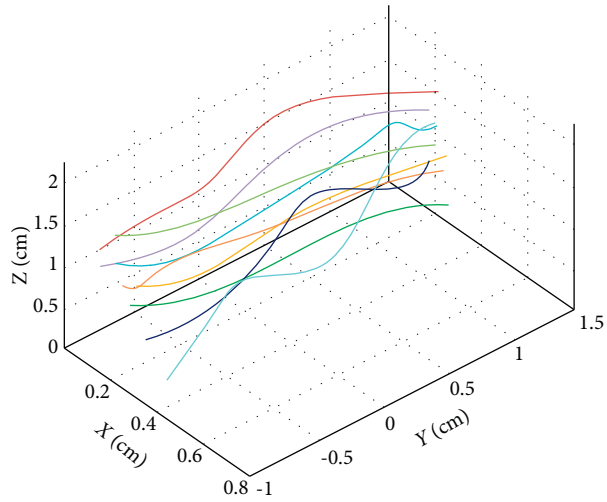

Figure 7: Trajectory of moving targets.

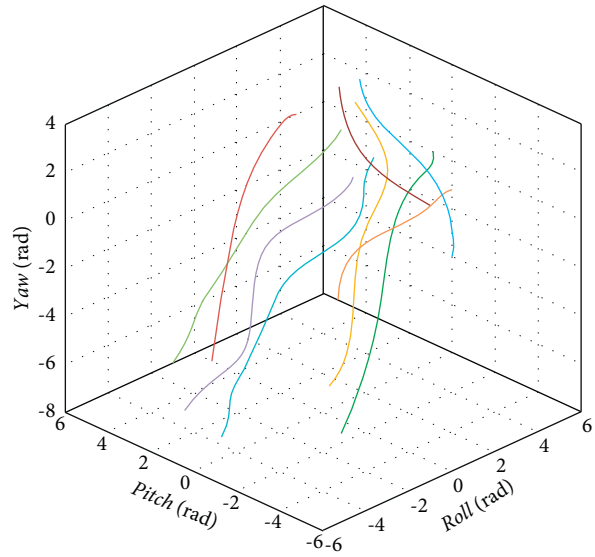

Figure 8: Grasping position trajectory of the manipulator.

The prediction error was defined as the distance from the spatial coordinates on the predicted trajectory of the moving target to those on the actual trajectory. Table 3 compares the prediction errors of our model with RNN and GRNN. When too many trajectory points needed to be predicted, RNN had a lower prediction accuracy than GRNN and our model, because it cannot effectively process the historical positions on distant trajectories. Our model surpassed the GRNN by 56.7% in the prediction accuracy of the spatial coordinates on the trajectory of moving targets.

Figures 7 and 8 show the predicted trajectory of moving targets and the predicted grasping position trajectory of the manipulator. Figure 9 presents the prediction error of moving target trajectory. Table 4 lists the prediction error of moving target trajectory. Most errors were within 0.2 cm, which verify the generalizability of the proposed tracking control algorithm.

To verify the learning effect of our training module, the probability density of classification error was calculated. The classification error of the classifier fell in (−0.9142, 0.8747), which basically obeys normal distribution (as shown in Figure 10).

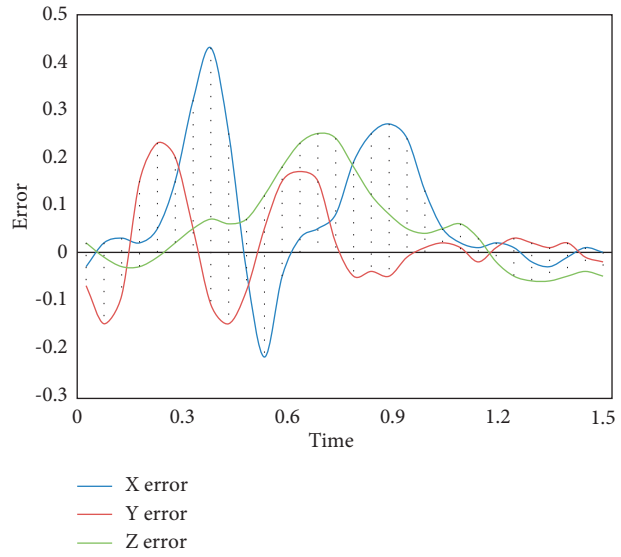

Figure 9: Prediction error of moving target trajectory.

Table 4: Prediction error of moving target trajectory.

Serial number	ΔPitch	ΔRoll	ΔYaw	ΔX	ΔY	ΔZ
1	0.068915	0.031406	−0.07264	0.196543	−0.056134	0.179623
2	0.035234	0.107545	0.0135	0.034126	−0.026247	0.106412
3	0.108972	0.135621	−0.019232	0.049862	−0.00572	0.281565
4	0.16152	−0.008942	−0.003757	0.005741	0.017964	0.297534
Mean	0.09366025	0.0664075	−0.02053225	0.071568	−0.01753425	0.2162835

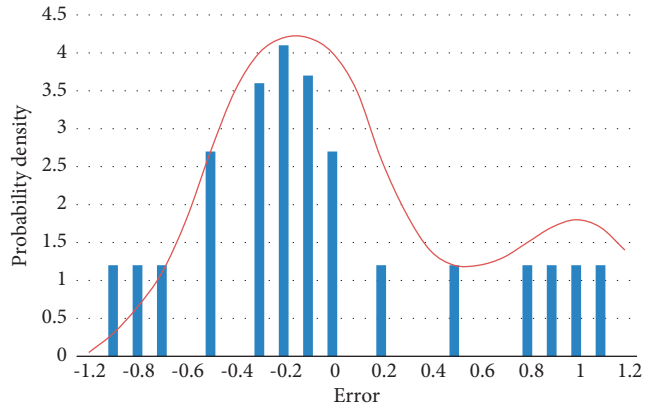

Figure 10: Probability density distribution of classification error.

5. Conclusions

This study explores how to realize automatic manipulator tracking control based on moving target trajectory prediction. Firstly, a moving target trajectory prediction model was established, and its parameters were optimized. Next, a tracking-training-testing algorithm was proposed for manipulator's automatic moving target tracking, and the operating flows were detailed for training module, target detection module, and target tracking module. The experimental results show the effectiveness of the proposed model and algorithm. During the experiments, the parameter combination was optimized, the corresponding errors were obtained, and the values of three key parameters K, F, and δ were optimized. The prediction results and losses of different models were compared, revealing that our model is more accurate in prediction than other models. Finally, the moving object trajectory and the manipulator's grasping position trajectory were predicted, and the prediction error of moving target trajectory was used to confirm the generalizability of the proposed tracking control algorithm.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (Grant No. 51902094).

References

[1] F. Ore, J. L. Jiménez Sánchez, M. Wiktorsson, and L. Hanson, "Design method of human-industrial robot collaborative workstation with industrial application," *International Journal of Computer Integrated Manufacturing*, vol. 33, no. 9, pp. 911–924, 2020.

[2] V. Nguyen, T. Cvitanic, and S. Melkote, "Data-driven modeling of the modal properties of a six-degrees-of-freedom industrial robot and its application to robotic milling," *Journal of Manufacturing Science and Engineering*, vol. 141, no. 12, Article ID 121006, 2019.

[3] F. Gao and P. Wu, "A trajectory planning algorithm for medical manipulators based on adaptive particle swarm optimization and fuzzy neural network," *Journal Européen des Systèmes Automatisés*, vol. 53, no. 3, pp. 429–435, 2020.

[4] G. Cao and G. Cao, "Application of industrial robot in automatic transformation of compressor pump production line," *Journal of Computational Methods in Science and Engineering*, vol. 19, no. S1, pp. 293–298, 2019.

[5] C. Garriz and R. Domingo, "Development of trajectories through the Kalman algorithm and application to an industrial robot in the automotive industry," *IEEE Access*, vol. 7, pp. 23570–23578, 2019.

[6] V. Verma, P. Chauhan, and M. Gupta, "Distributed association rule mining with minimum communication overhead," *Journal Européen des Systèmes Automatisés*, vol. 52, no. 4, pp. 355–362, 2019.

[7] D. Schwung, F. Csaplar, A. Schwung, and S. X. Ding, "An application of reinforcement learning algorithms to industrial multi-robot stations for cooperative handling operation," in *Proceedings of the 2017 IEEE 15th International Conference on Industrial Informatics (INDIN)*, pp. 194–199, Emden, Germany, July 2017.

[8] A. Rout, D. Bbvl, B. B. Biswal, and G. B. Mahanta, "Optimal trajectory planning of industrial robot for improving positional accuracy," *Industrial Robot: The International Journal of Robotics Research and Application*, vol. 48, no. 1, pp. 71–83, 2019.

[9] Y.-H. Lee, S.-C. Hsu, T.-Y. Chi, Y.-Y. Du, J.-S. Hu, and T.-C. Tsao, "Industrial robot accurate trajectory generation by nested loop iterative learning control," *Mechatronics*, vol. 74, Article ID 102487, 2021.

[10] L. Li, Y. Huang, and X. Guo, "Kinematics modelling and experimental analysis of a six-joint manipulator," *Journal Européen des Systèmes Automatisés*, vol. 52, no. 5, pp. 527–533, 2019.

[11] X. Yin and L. Pan, "Enhancing trajectory tracking accuracy for industrial robot with robust adaptive control," *Robotics and Computer-Integrated Manufacturing*, vol. 51, pp. 97–102, 2018.

[12] J. Zhu, L. Pan, and G. Zhao, "An improved near-field computer vision for jet trajectory falling position prediction of intelligent fire robot," *Sensors*, vol. 20, no. 24, p. 7029, 2020.

[13] H. Wu, W. Xu, B. Yao, Y. Hu, and H. Feng, "Interacting multiple model-based adaptive trajectory prediction for anticipative human following of mobile industrial robot," *Procedia Computer Science*, vol. 176, pp. 3692–3701, 2020.

[14] X. Zhang and Z. Ming, "Par4 parallel robot trajectory tracking control based on DMR-GWO2 and fuzzy predictive," *International Journal of Computational Intelligence Systems*, vol. 14, no. 1, pp. 1597–1606, 2021.

[15] A. Tika, N. Gafur, V. Yfantis, and N. Bajcinca, "Optimal scheduling and model predictive control for trajectory planning of cooperative robot manipulators," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 9080–9086, 2020.

[16] C. Xia, C.-Y. Weng, Y. Zhang, and I.-M. Chen, "Vision-based measurement and prediction of object trajectory for robotic manipulation in dynamic and uncertain scenarios," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 11, pp. 8939–8952, 2020.

[17] J. Zhong, C. Weber, and S. Wermter, "Robot trajectory prediction and recognition based on a computational mirror neurons model," in *Proceedings of the International Conference on Artificial Neural Networks*, pp. 333–340, Espoo, Finland, June 2011.

[18] C. Deng, Y. Mao, X. Gan, and J. Tian, "Application of independent component analysis in target trajectory prediction based on moving platform," in *Proceedings of the AOPC 2015: Optical Design and Manufacturing Technologies*, vol. 9676, Article ID 96760T, Beijing, China, 2015.

[19] Y. Du, Y. Wang, X. Zhang, and Z. Nie, "Automatic separation management between multiple unmanned aircraft vehicles in uncertain dynamic airspace based on trajectory prediction," *Revue d'Intelligence Artificielle*, vol. 33, no. 3, pp. 171–180, 2019.

[20] J. Hu and Q. B. Zhu, "A multi-robot hunting algorithm based on dynamic prediction for trajectory of the moving target and hunting points," *Dianzi Xuebao (Acta Electronica Sinica)*, vol. 39, no. 11, pp. 2480–2485, 2011.

[21] G. F. Sun, B. Zhang, and H. Z. Lu, "The detection algorithm based on predicting-matching-window for moving point target trajectory in image sequences," *Journal of National University of Defense Technology*, vol. 26, no. 2, pp. 25–29, 2004.

[22] S. Liu, Y. Guo, H. Jia, S. Lin, and D. Zhao, "Mechanism principle and obstacle-crossing analysis of robot with automatic-strained track and variable main arm configuration," *Journal of Central South University*, vol. 44, no. 6, pp. 2289–2297, 2013.