


Article

A Kalman Filter-Based Kernelized Correlation Filter Algorithm for Pose Measurement of a Micro-Robot

Heng Zhang, Hongwu Zhan * , Libin Zhang, Fang Xu and Xinbin Ding

Mechanical Engineering College, Zhejiang University of Technology, Hangzhou 310023, China; hengz067@zjut.edu.cn (H.Z.); lbz@zjut.edu.cn (L.Z.); fangx@zjut.edu.cn (F.X.); 15658092668@163.com (X.D.)

* Correspondence: waltzhan@hotmail.com; Tel.: +86-0431-13758266571

Abstract: This paper proposes a moving-target tracking algorithm that measures the pose of a micro-robot with high precision and high speed using the Kalman filter-based kernelized correlation filter (K2CF) algorithm. The adaptive Kalman filter can predict the state of linearly and nonlinearly fast-moving targets. The kernelized correlation filter algorithm then accurately detects the positions of the moving targets and uses the detection results to modify the moving states of the targets. This paper verifies the performance of the algorithm on a monocular vision measurement platform and using a pose measurement method. The K2CF algorithm was embedded in the micro-robot's attitude measurement system, and the tracking performances of three different trackers were compared under different motion conditions. Our tracker improved the positioning accuracy and maintained real-time operation. In a comparison study of K2CF and many other algorithms on Object Tracking Benchmark-50 and Object Tracking Benchmark-100 video sequences, the K2CF algorithm achieved the highest accuracy. In the 400 mm × 300 mm field of view, when the target radius is about 3 mm and the inter-frame acceleration displacement does not exceed 5.6 mm, the root-mean-square error of position and attitude angle can satisfy the precision requirements of the system.

Keywords: Kalman filter; kernelized correlation filter; target tracking; machine vision; pose measurement



Citation: Zhang, H.; Zhan, H.; Zhang, L.; Xu, F.; Ding, X. A Kalman Filter-Based Kernelized Correlation Filter Algorithm for Pose Measurement of a Micro-Robot. *Micromachines* **2021**, *12*, 774. <https://doi.org/10.3390/mi12070774>

Academic Editors: Jose Luis Sanchez-Rojas and Sukho Park

Received: 12 May 2021
Accepted: 28 June 2021
Published: 30 June 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Technological developments in information, electronics and mechatronics have advanced the use of micro-robots in precision operation fields, such as complex assembly [1,2], advanced machining [3,4], intelligent manufacturing [5], automatic monitoring [6,7], non-destructive testing [8,9] and digital printing [10,11]. To accomplish a wide range of accurate operations, micro-robots such as miniature assembly robots and micro-robots for large-format digital printing must plan their motion trajectories and determine if there is a deviation in the path that requires accurate pose measurements in real-time [12–14]. It is important to measure the pose change of these robots for their trajectory planning and control effect detection. Pose measurement technologies of micro-robots are broadly divisible into relative localization and absolute localization [15–17]. Kelly and Martinoli located the relative position of a robot among a crowd of micro-mobile machines using an infrared transmitter and an infrared receiver. The positioning accuracy of their system was 40 cm within a measurement range of 3 m, and the maximum error in their attitude angle was 17.4° [18,19]. However, estimating the range from the amplitude of an incoming infrared signal is intrinsically limited to low-precision measurements. Qazizada designed an inertial navigation system with a gyroscope and an accelerometer. Although this system determines a robot's current pose, it is unsuitable for long-term high-precision positioning [20]. Absolute localization usually requires beacons and global vision positioning. Diederichs detected, classified, located and tracked micro-mobile robots by a micro-camera, which achieves nanometer-level manipulation but requires the size and speed of the target [21]. Equipped with a target tracking algorithm, vision measurement is widely used for

tracking and locating micro-robots with high precision and high speed under no-contact conditions [22,23].

With respect to the moving target tracking algorithms, they are generally divided into production model methods and discriminative model methods. The production model methods conduct modeling for the target region in the current frame and detect the most similar region in the next frame [24–26]. The classical algorithms mainly contain the particle filter tracking algorithm [27], mean shift algorithm [28] and optical flow algorithm based on feature point [29,30]. Production model tracking algorithms detect the target region by dense or sparse search of each frame image. Nevertheless, they are not suitable for fast detecting in large pixel images due to the high computation load. For the discriminative model methods, the target regions are used as positive samples and the background areas are regarded as negative samples, which are used to train the classifier [31–33]. Kalal proposed a tracking-learning-detection (TLD) algorithm for long time tracking, and performed adaptive processing for various scales, attitudes and illumination changes by offline learning [34]. For the tracking speed, detection accuracy and tracking success rate, the TLD algorithm is lower than the correlation filter tracking algorithm. Due to the high computational efficiency and cyclic dense sampling, the correlation filter tracking algorithm can realize high precision and high-speed tracking. Bolme developed an adaptive correlation filter to achieve visual object tracking, which could reach hundreds of frames per second [35]. Henriques presented a circulant structure of tracking-by-detection with kernels [36]. It adopted a cyclic matrix to achieve a dense sample and transformed computation into the frequency domain by Fourier transform. Then, using the histogram of oriented gradient (HOG) as a sample feature [37], the kernelized correlation filter (KCF) tracking algorithm is proposed [38]. Lim constructed a series of image sequence data with real geographical location and range to analyze the performance of various tracking algorithms and obtained the performance of each algorithm under the same evaluation criteria [39]. According to their results, the KCF tracking algorithm has no requirement for tracking objects. It can update the classifier and recognize the target by self-learning. Within a certain error threshold and overlapping threshold, it can perform fast detection for moving targets.

However, the KCF is very sensitive to occlusion, scale changes and fast motions. To overcome these difficulties, Li replaced the fixed-size template in the KCF tracker with an effective scale-adaptive KCF tracker [40]. In an experimental evaluation, the scale-adaptive tracker showed a promising ability to change the scale of a target. By integrating a Kalman filter with KCF, Huynh developed a new tracking method that overcomes occlusion and human-crossing [41]. The KCF in their method estimates the target position based on the Kalman filter prediction and updates the kernel model accordingly. When the tracker encounters an occlusion, the Kalman filter omits the observed values of the KCF and updates the state based on the previous state. This tracker properly handles occlusion and human-crossing tasks, but its Kalman filter cannot effectively predict the target information of fast-moving objects. When the target moves near or over the boundary, the object information will be filtered out.

Apparently, the boundary effect is the main problem in KCF tracking algorithms, as it risks error fluctuations and tracking loss, especially for fast-moving targets. To correct the boundary effect in the KCF tracking algorithm, this paper proposes an adaptive kernelized correlation filter (K^2CF) algorithm that integrates the adaptive Kalman filter and can effectively predict the position of the target in different moving states. The predicted position is then refined to a precise precision by the KCF algorithm, and the moving state of the target is corrected based on the detection results. The K^2CF algorithm reduces the kernel response value of the KCF algorithm and improves its detection accuracy and tracking stability. The least-square method and morphology processing method are used to extract the feature points of the tracking area, which further improves the tracking accuracy. The image coordinates and motion coordinates are transformed by the camera calibration

principle to complete the pose measurement of the target. The method can measure the pose of the tiny moving target quickly and effectively, and the detection accuracy is high.

2. Pose Measurement Principle for a Micro-Mobile Robot

The pose of a micro-mobile robot in two-dimensional space is measured from the position information and direction angle in a moving plane. To meet the detection accuracy requirements of a mobile robot, we first design a high-precision visual detection platform (see Figure 1). Applying the principle of camera calibration, we then establish the mapping relationship between the moving plane and the image plane. Finally, we extract the feature points of the micro-robot in the field of view and acquire the position and direction angle of the moving target.

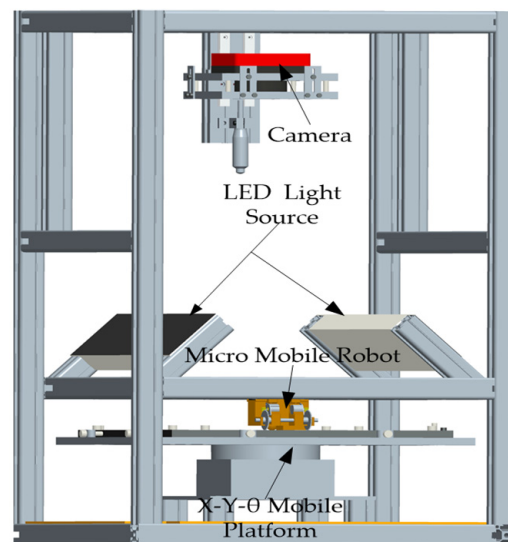


Figure 1. Visual detection platform for micro-mobile robot.

2.1. High-Precision Vision Measurement Platform

When a target moves at speeds below 0.4 m/s in a 400 mm × 300 mm range, the trajectory plan of a micro-mobile robot must resolve the motion to within 0.1 mm. That is, the precision of the vision detection platform must be at least 0.1 mm, and the moving speed of the target cannot exceed 0.4 m/s.

The camera in our system is an IO-Flare series 12M125C with a CMOS global shutter (Figure 1). The pixel size of the images is 4096 × 3072, and the frame rate is 124 fps. The acquisition card (AS-FBD-2XCLD-2PE8) meets the requisite high-rate acquisition through a camera-link serial bus. The uniform light source of the LED strip lamp is flanked on both sides. The required illumination is obtained by an external trigger control that ensures a target displacement not exceeding one pixel while imaging [42–44]. An X–Y–θ triaxial mobile platform, which simulates random movements in the two-dimensional plane, is used for verifying the accuracy of the detection algorithm and adjusting the field of view.

2.2. Visual Platform Calibration

The visual detection system is calibrated using various external and internal parameters. The internal parameters are the intrinsic parameters of the camera, which provide the conversion relationship between the camera and image coordinate systems [45–47]. The external parameters provide the relative position and attitude relationships between the camera and motion coordinate systems. The camera imaging model of the visual platform calibration is shown in Figure 2.

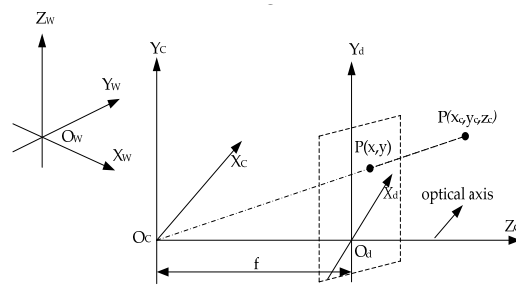


Figure 2. Camera imaging model for visual platform calibration.

The image coordinates are converted to world coordinates as follows:

$$Z_c \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{f}{dx} & 0 & u_0 & 0 \\ 0 & \frac{f}{dy} & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} R_1 & T \\ 0 & 1 \end{pmatrix} \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix} \tag{1}$$

where (u, v) are the image coordinate, (X_w, Y_w, Z_w) are the word coordinate, f is the focal length of the camera, dx and dy are the pixel sizes in the x and y directions, respectively, and (u_0, v_0) are the optical-center camera coordinates, R_1 and T are the rotation and moving matrices, respectively, Z_c is the distance between the camera lens and the target, M_1 and M_2 are matrices of the internal and external parameters, respectively:

$$M_1 = \begin{pmatrix} \frac{f}{dx} & 0 & u_0 & 0 \\ 0 & \frac{f}{dy} & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} M_2 = \begin{pmatrix} R_1 & T \\ 0 & 1 \end{pmatrix} \tag{2}$$

The camera calibration is mainly required for solving the perspective projections of M_1 and M_2 between the imaging and target planes. Therefore, if M_1 and M_2 are obtained, the world coordinates can be determined from the known image coordinates. For a mobile micro robot in a two-dimensional plane, we construct an $X_c O_c Y_c$ coordinate system with origin placed at the starting point of the mobile robot.

As shown in Figure 3, we choose P_{00} and P_{10} as the feature points of mobile robot, and (x, y, θ) indicates the pose of mobile robot and the pose (x, y, θ) of the mobile robot is calculated as follows:

$$\begin{cases} x = \frac{x_{11} + x_{10}}{2} - \frac{x_{01} + x_{00}}{2} \\ y = \frac{y_{11} + y_{10}}{2} - \frac{y_{01} + y_{00}}{2} \\ \theta = \tan^{-1} \frac{y_{11} - y_{10}}{x_{11} - x_{10}} - \tan^{-1} \frac{y_{01} - y_{00}}{x_{01} - x_{00}} \end{cases} \tag{3}$$

where (x_{00}, y_{00}) and (x_{10}, y_{10}) are the initial coordinates of P_{00} and P_{10} , respectively, (x_{01}, y_{01}) and (x_{11}, y_{11}) are the coordinates of the corresponding feature points P_{01} and P_{11} , respectively (see Figure 3).

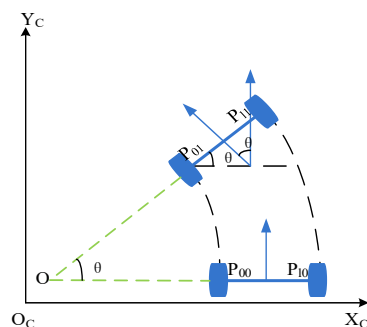


Figure 3. Target pose in a 2-dimension plane.

3. Tracking Algorithm for the Moving Target

To track a fast-moving target with small geometric size, the tracking algorithm requires a fast tracking rate and high detection precision. Because of the circular correlation, the KCF algorithm can expand limited training data by implicitly including all shifted samples of given patches [48,49]. Moreover, the computational effort of training and detection is significantly reduced in the Fourier domain. Owing to these advantages, the KCF is especially suitable for micro-target tracking. However, the periodicity of the circular correlation of KCF produces unwanted boundary effects, which severely limit the target-search region at the detection step. To resolve these inherent problems, we propose the K²CF algorithm for moving target tracking. The K²CF includes a Kalman predictor that efficiently predicts the positions of the detection samples before the KCF detection. The effect of incomplete negative training samples is weakened by a Gaussian model.

3.1. Analysis of the KCF Tracking Algorithm

3.1.1. Training and Detection

The KCF tracking algorithm attempts to train a function $f(\mathbf{z}) = \mathbf{w}^T \mathbf{z}$ that minimizes the squared error over the samples x_i and their regression labels y_i . The optimal solution of the optimization problem can be expressed as:

$$\min \sum_i (f(x_i) - y_i)^2 + \lambda \|\mathbf{w}\|^2 \quad (4)$$

where λ is a regularization parameter that controls overfitting (the state vector machine uses a similar parameter). Equation (4) is a linear least-squares problem. To map the inputs of a linear problem to a nonlinear feature space $\varphi(x)$, define the correlation function as: $k(x, x') = \varphi^T(x)\varphi(x')$, we apply the kernel trick and express the solution of \mathbf{w} as a linear combination of samples: $\mathbf{w} = \sum_{i=1}^n \alpha_i \varphi(x_i)$ [50,51]. The function $f(\mathbf{z}) = \mathbf{w}^T \mathbf{z}$ is then represented as

$$f(\mathbf{z}) = \mathbf{w}^T \mathbf{z} = \sum_{i=1}^n \alpha_i k(\mathbf{z}, x_i) \quad (5)$$

where \mathbf{z} and x denote the candidate detection patches and the training samples, respectively. Consequently, Equation (4) is solved as

$$\boldsymbol{\alpha} = (\mathbf{K} + \lambda \mathbf{I})^{-1} \mathbf{y} \quad (6)$$

where \mathbf{K} is a $n \times n$ kernel matrix with elements $K_{ij} = k(x_i, x_j)$, \mathbf{y} is the regression output of all samples, \mathbf{I} is the identity matrix. $\boldsymbol{\alpha}$ is a parameter matrix needs to be optimized.

By Theorem 1 in [38], the cyclic matrix \mathbf{K} can be transformed into a diagonal matrix by the discrete Fourier transform (DFT) [52,53]. Accordingly, Equation (6) is solved in the Fourier domain as

$$\hat{\boldsymbol{\alpha}} = \frac{\hat{\mathbf{y}}}{\hat{\mathbf{k}}^{XX} + \lambda} \quad (7)$$

The quantities k^{xx} , expressing the kernel correlations of each x with itself, occupy the first row of the kernel matrix \mathbf{K} ; the hat symbol $\hat{\cdot}$ denotes the DFT of a matrix.

The parameter $\boldsymbol{\alpha}$ of the classifier function $f(\mathbf{z})$ is obtained through the training samples x . Using the learned function $f(\mathbf{z})$, we evaluate the classification scores of all cyclic shifts of a candidate detection sample:

$$f(\mathbf{z}) = \hat{\mathbf{k}}^{xz} \odot \hat{\boldsymbol{\alpha}} \quad (8)$$

where \odot is a dot product of the matrix. The quantities k^{xz} , expressing the kernel correlations between \mathbf{z} and x , occupy the first row of the kernel matrix \mathbf{K}^z . The position of the shifted samples that maximizes the classification score is the target location.

After locating the target by Equation (8), we obtain a training sample \mathbf{x}_k around the target position. The parameter α_k of the classifier function $f(\mathbf{z})$ at the current time is computed by Equation (7). The algorithm is updated by the following strategy:

$$\begin{cases} \alpha'_k = \beta\alpha_k + (1 - \beta)\alpha_{k-1} \\ \mathbf{x}_k = \beta\mathbf{x}_k + (1 - \beta)\mathbf{x}_{k-1} \end{cases} \quad (9)$$

where β is the learning factor.

3.1.2. Boundary Effect of KCF

The KCF formulation trains the parameter α of the classifier function $f(\mathbf{z})$ in Equation (7) on the training sample x and its shifted samples. At the detection step (8), the detection samples are also shifted by the base candidate sample \mathbf{z} .

As shown in Figure 4, shifting the base training sample (a) obtains a set of negative training samples (b). However, these negative training samples fail to capture the true image content, because the periodicity of the shift samples introduces a boundary effect that reduces the discriminative power of the learned classifier. To ensure the integrity of the shift samples, we preprocess the target region through a cosine window filter.

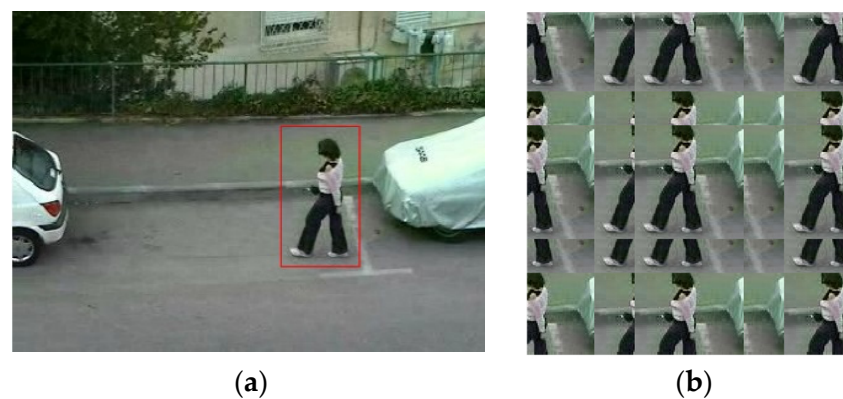


Figure 4. Circular shift: (a) original image; (b) periodicity in shift samples.

Figure 5 is the result of extending the target region and smoothing the edges through the cosine window filter. A circular shift of the base sample then acquires complete negative training samples. However, if the center of the shift target is near the boundary, some incomplete negative samples are also generated. In the detection step, when a fast-moving target is near the boundary of the detection region, the cosine window filter will filter out the target information (see Figure 6).

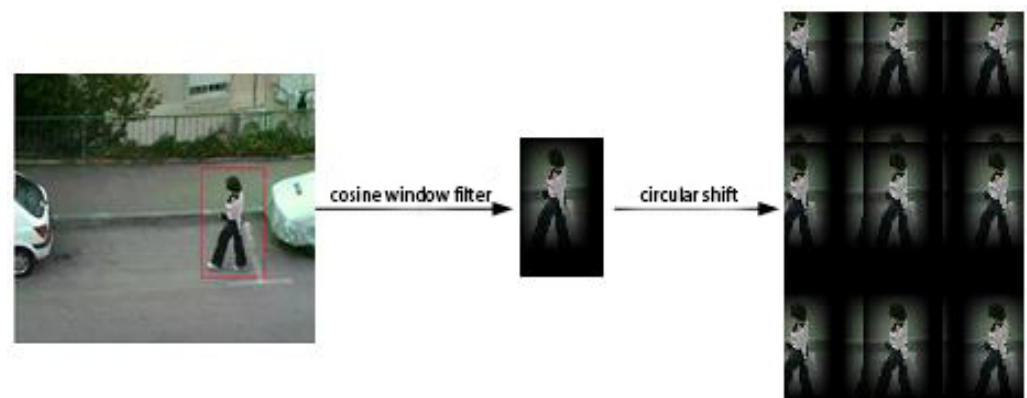


Figure 5. Extraction process of cyclic samples.



Figure 6. Detection samples of fast target.

The above-mentioned limitations of the KCF hamper the tracking performance in several ways. First, KCF-based trackers cannot easily detect fast target motions, as their search regions are limited. Second, any incomplete negative training patches will affect the accuracy of the learned model. Third, the training and detection limitations reduce the re-detection potential of the tracker after interruption by an occlusion.

3.2. Framework of K2CF Tracking Algorithm

To resolve the problems caused by the circular shift and cosine window filter, we define the importance of negative training samples by a Gaussian function, and predict the position of the detection target by a Kalman filter [54–56].

3.2.1. Training and Prediction

To solve Equation (4) in the KCF, we train the classifier $f(\mathbf{z})$ on a set of training samples $\{(x_i, y_i)\}_{i=1}^t$. Each training sample x_i consists of a C-channel HOG feature map. The desired output value y_i is the label for each sample x_i . We define the base sample extracted from the image region as the positive sample and the shift samples as negative samples. To discriminate the incomplete and complete training samples, we calculate the value of the label y_i , which represents the importance of the training samples, by the following Gaussian function:

$$G(a, b) = \frac{1}{2\pi\sigma_1^2} e^{-\frac{(a^2+b^2)}{2\sigma_1^2}} \tag{10}$$

where a and b are the shifted distances of the negative samples along the row and column, respectively. The Gaussian model assigns low importance to shift samples away from the base sample. Incomplete samples, which reduce the discriminant of the classifier model [57–59], also receive lower weights. The unequal weights significantly mitigate the emphasis of these shift samples in the learned classifier.

The kernel correlation $k^{xx'}$ in Equation (7) is computed with Gaussian kernels $k(x, x') = \exp(-\frac{1}{\sigma_2^2} \|x - x'\|^2)$ as follows:

$$k^{xx'} = \exp\left(-\frac{1}{\sigma_2^2} \left(\|x\|^2 + \|x'\|^2 - 2f^{-1}\left(\sum_C \hat{x} * \odot \hat{x}'\right)\right)\right) \tag{11}$$

where $k^{xx'}$ expresses the kernel correlation of x with x' . The Gaussian kernel is less interfered by noise than the linear and polynomial kernels. Using Equations (7), (9), and (10), we can train the parameter α of the classifier function $f(\mathbf{z})$.

The classification scores of all cyclic shifts of test sample \mathbf{z} in Equation (8) are evaluated by a learned classifier. The detection base sample \mathbf{z} is extracted from the image region that is the target location in the previous frame of the KCF formulation. Therefore, the target can move only within a limited region.

To solve the above problem, we estimate the position of the test region by a predictor. The target state is denoted by a vector $S = \{x, y, v_x, v_y\}$, where x and y are the coordinates of the tracking object center, and v_x and v_y are the velocity components along the x and y axes, respectively. Following the basic principle of the Kalman filter, the state equation of the discrete dynamic system is recursively obtained as

$$\begin{cases} s_{k|k-1} = \mathbf{A}s_{k-1|k-1} + \mathbf{B}u_{k-1} \\ p_{k|k-1} = \mathbf{A}p_{k-1|k-1}\mathbf{A}^T + \mathbf{Q}_{k-1} \end{cases} \tag{12}$$

where $s_{k|k-1}$ is the estimated state of the tracking object at time k , $s_{k-1|k-1}$ is the predicted state of the tracking object at time $k - 1$, \mathbf{A} and \mathbf{B} are constant matrices describing the state transition and state control of the object, respectively, u_{k-1} is the control vector, $p_{k|k-1}$ is the estimated covariance matrix at time k , $p_{k-1|k-1}$ is the predicted covariance at time $k - 1$, and \mathbf{Q}_{k-1} is the error matrix in the state transition process.

3.2.2. Detection of KCF

To resolve the boundary effect, we predict the location of the detection region by a Kalman filter, which ensures that the candidate detection sample is within the predicted boundary of the region. The candidate test patch \mathbf{z} is then extracted at the prediction position $s_{k|k-1}$. Exploiting the cyclic property, a set of detection samples is obtained as

$$\{\mathbf{P}^u \mathbf{z} \mid u = 0, 1, \dots, n-1\} \tag{13}$$

where u denotes a shift element. Each sample \mathbf{z} is periodically obtained after n shifts.

In Equation (8), the kernel matrix \mathbf{K}^z denotes the correlation between all training samples \mathbf{X} and all candidate patches \mathbf{Z} . The samples and patches are cyclic shifts of the base sample x and base patch, respectively. The KCF algorithm processes the kernel matrix \mathbf{K}^z as a circulant, defined as

$$\mathbf{K}^z = \mathbf{C}(k^{xz}) \tag{14}$$

where k^{xz} (occupying the first row of the kernel matrix) denotes the kernel correlation between x and \mathbf{z} , and \mathbf{C} denotes that \mathbf{K}^z is generated by shifting k^{xz} .

This cyclic matrix property means that \mathbf{K}^z can be diagonalized by the DFT as

$$\mathbf{K}^z = \mathbf{F} \hat{\text{diag}}(k^{xz}) \mathbf{F}^T \tag{15}$$

where \mathbf{F} is the DFT matrix. Here, the hat symbol $\hat{}$ and T denote the DFT and conjugate transpose of a matrix, respectively [60,61].

From Equation (5), the correlation values of all detection patches are computed as

$$f(\mathbf{z}) = (\mathbf{K}^z)^T \boldsymbol{\alpha} \tag{16}$$

where $f(\mathbf{z})$ contains the classification scores of all cyclic shifts of \mathbf{z} . Equation (16) can be efficiently calculated by diagonalizing Equation (16) in the DFT domain (Equation (15)). Accordingly, the location of maximum in $f(\mathbf{Z})$ is the target position in the current frame.

3.2.3. Adaptive Optimization of the Position Predictor

Equation (11) gives a preliminary estimate of the state. The target state is then optimized by the real observed value Y_{k-1} , which corrects the estimating error. In the discrete dynamic system, the target state is corrected as

$$\begin{cases} \mathbf{K}_k = p_{k|k-1} \mathbf{H}^T (\mathbf{H} p_{k|k-1} \mathbf{H}^T + \mathbf{R})^{-1} \\ s_{k|k} = s_{k|k-1} + \mathbf{K}_k (Y_{k-1} - \mathbf{H} s_{k-1|k-1}) \\ p_{k|k} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}) p_{k|k-1} \end{cases} \tag{17}$$

where K_k is the state gain matrix, H is the observation transfer matrix, $p_{k|k}$ is the covariance matrix after correcting the target state, I is the identity matrix, and R denotes the observation error in the system. Because the imaging system is low-noise, R is set as a constant matrix. Equation (12) corrects the estimated state using the observation error at the previous time. Here, $s_{k|k}$ denotes the predicted position of the detection region in the current frame.

First, the target position is predicted by a Kalman filter. The testing samples around the prediction location are then extracted, and the target in the current frame is located by the KCF tracker. Finally, the predicted target position is refined by the detected value. The authors of [41] selected the position of an occluded target using a threshold value σ . If the detected value of the KCF tracker is less than σ , the Kalman filter is updated using the detected value, and the result is set as the target location. Otherwise, the detected value is ignored and the predicted value of the Kalman filter is set as the target location. However, this tracker is unsuitable for fast-moving targets.

Many fast-motion targets follow nonlinear dynamics. If the target position of a nonlinearly moving object is predicted by a conventional Kalman filter, the predicted state will deviate from the actual state. As the acceleration of the tracked object varies over time, the conversion error matrix Q_{k-1} varies throughout the transition process. The error matrix Q_{k-1} must then be updated by analyzing the target acceleration.

In this system, the mobility of the target is not high, so Q_{k-1} can be updated using the frequency δ and variance σ^2 of the acceleration:

The frequency of acceleration α is computed from three adjacent frames as

$$Q_{k-1} = 2\delta\sigma^2 I \quad (18)$$

$$\delta = \left| \frac{a_{k-1|k-1} + a_{k-3|k-3} - 2a_{k-2|k-2}}{T} \right| \quad (19)$$

where $a_{k-1|k-1}$, $a_{k-2|k-2}$, $a_{k-3|k-3}$ indicate the accelerations at three adjacent moments $k-1$, $k-2$, and $k-3$, respectively. The sample interval T is usually set to 1.0 [62,63]. When the target is moving at uniform speed, the frequency of the acceleration δ is unchanged, which conforms to actual situations.

The maximum value of $f(z)$ given by Equation (16) is the highest classification score of the cyclic shift samples, which is assigned to the target location in the current frame. In the detection step, the correlation filter detects the candidate patch extracted from the image region at the prediction position. If the predicted position is exactly at the target center, the highest detection score is allocated to the 0-element shift. However, if the predicted position deviates from the target center, the location of the highest detection score is an n -element shift. The n -element shift denotes the distance between the predicted position and the actual target position. In fact, the predicted and actual target positions usually deviate when the acceleration is variable, indicating that the prediction error is caused by an acceleration disturbance:

$$n = S_1(k/k-1) - S_1(k) \quad (20)$$

In Equation (20), $S_1(k/k-1)$ denotes the predicted position at time k from time $k-1$, and $S_1(k)$ is the actual detection location at time k . Within the sampling period T , the error between the predicted and detected positions is related to the acceleration variation Δa as

$$S_1(k/k-1) - S_1(k) = \frac{T^2}{2} \Delta a \quad (21)$$

The acceleration covariance is linearly related to the absolute value of the acceleration variation. In turn, the acceleration variation is linearly related to the position error as follows:

$$\sigma^2 = \eta \left| \frac{2(S_1(k/k-1) - S_1(k))}{T^2} \right| \quad (22)$$

where η is the linear factor. Using Equation (22), we can adaptively adjust the acceleration covariance.

Because the acceleration defines a velocity change, we can update the predictor state and modify the predictor velocity with the prediction error n , which is related to the acceleration variation Δa .

$$\begin{cases} s_{k|k}.v_x = s_{k-1|k-1}.v_x + n.x \times T \\ s_{k|k}.v_y = s_{k-1|k-1}.v_y + n.y \times T \end{cases} \quad (23)$$

where n denotes the shift distance between the predicted and actual positions. Therefore, the actual location of the target in the current frame is determined as

$$Y_k = s_{k|k} + n \quad (24)$$

In the K^2CF , the weights of all shift training samples are determined by the Gaussian model. By assigning lower labels to incomplete samples affected by the boundary, we can reduce the influence of the incomplete samples on the learned classifier model. Prior to detection, the test region is identified by the Kalman filter. At the predicted position, the candidate base patch is extracted from the image region, which ensures that the target lies within the boundary. During the detection step, the location of the maximum $f(z)$ determines the prediction error of the predictor. Knowing the detected position, we can promptly correct the state transfer error and update the target state to optimize the predictor.

4. Feature Points for High-Precision Tracking

As mentioned in Section 1, the pose measurement of a micro-robot requires two feature points. However, when the target moves outside the boundary, the extracted target is incomplete. Therefore, rather than taking the center of the tracking object, we select the center of the extracted circular features as the feature points on the micro-robot (see Figure 7).

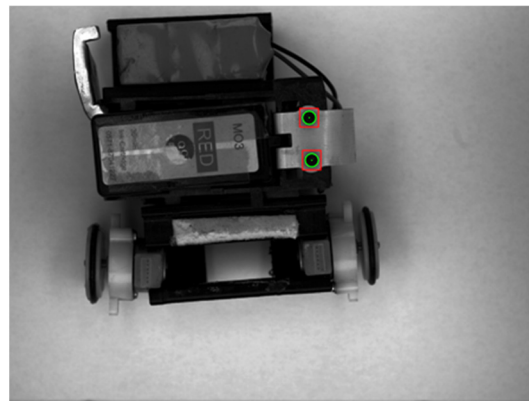


Figure 7. Circular tracking object.

4.1. Contour Extraction of a Tracked Object

To extract the contour of a tracked circular object, we first separate the circular object by the region-growth method. As shown in Figure 8a, one point in the circular object is randomly selected as a seed. The region around the seed grows by judging the similarity of the gray value of the neighborhood points. The growth continues until the circular object is completely segmented. After segmenting the circular object, the contour points are extracted by the Canny operator. The result is shown in Figure 9.

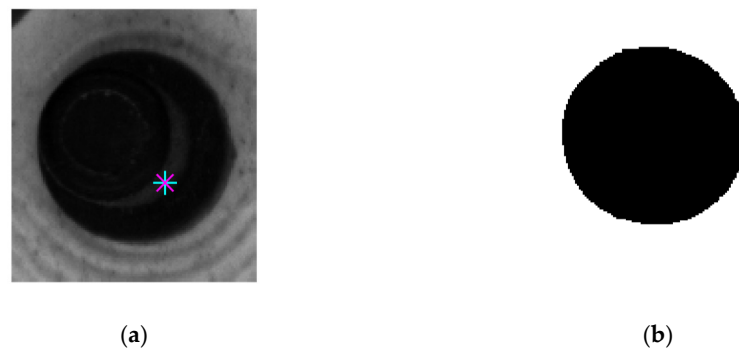


Figure 8. Tracking object segmentation: (a) original tracking target; (b) segmented circular object.

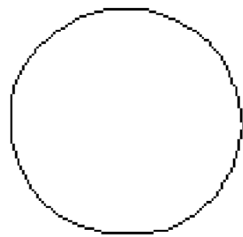


Figure 9. Contour of the tracked circular object in Figure 8.

4.2. Optimization of the Tracked Object Contour

The segmented circular contour is roughened by “burr points” among the extracted contour points. Removing these points will improve the accuracy of the circle fitting. The curvature at the midpoint of three adjacent contour points $Q_1(x_1, y_1)$, $Q_2(x_2, y_2)$, $Q_3(x_3, y_3)$ on the contour is calculated as

$$k = \frac{1}{r} = \frac{1}{\sqrt{(x_0 - x_2)^2 + (y_0 - y_2)^2}}. \quad (25)$$

where (x_0, y_0) is the center of the circle determined by the three adjacent points. When the curvature at a contour point exceeds some given threshold, that point is considered as a “burr point” and is rejected. The optimized contour points are drawn in Figure 10.

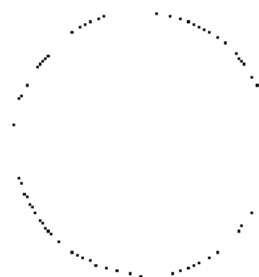


Figure 10. Optimized contour points.

4.3. Roundness Fitting to the Optimization Contour Points

The optimized contour points are fitted to a circle by the least-squares method. The fitting-circle equation is given by

$$(x - x_c)^2 + (y - y_c)^2 = R^2 \quad (26)$$

where (x_c, y_c) is the center of the fitting circle (the required feature point), and R is the radius of the fitting circle. The least-squares method minimizes the quadratic sum of the distance from the contour points to the circle center. This optimization problem is given by

$$f = \sum_{j=0}^n \left(\sqrt{(x_j - x_c)^2 + (y_j - y_c)^2} - R \right)^2 \quad (27)$$

To acquire the extremum of (27), we set the partial derivatives equal to 0.

$$\frac{\partial f}{\partial x_c} = 0, \quad \frac{\partial f}{\partial y_c} = 0 \quad (28)$$

The fitting circle obtained by Equation (28), and its center, are shown in Figure 11.

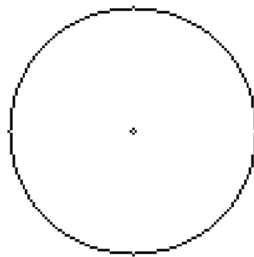


Figure 11. Optimized fitting circle.

The high-precision feature points are extracted by the region-growth method and the Canny operator as the contour points of the tracked object. These contour points are then optimized by calculating the contour point curvatures. Finally, the optimized contour points are fitted to a circle by the least-squares method, which extracts the precise center of the circle as a feature point.

5. Numerical Results and Discussion

5.1. Implementation Details

The proposed K^2CF algorithm was competed against the KCF algorithm and the Kalman filter- based tracking algorithms in several numerical instances. The present trackers were implemented using OpenCV library. In the Kalman filter, the initial motion state was set as $s = \{0, 0, 0, 0\}$ and the transition matrix A was set as

$$A = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (29)$$

The conversion error matrix Q_{k-1} follows a metabolic transition process. Its initial value was set as

$$Q_{k-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (30)$$

The observation transfer matrix H was set as a constant matrix.

The performance of our tracker was compared with those of the KCF and Kalman filter-based tracking. All trackers were implemented in the OpenCV library and C++ on Windows 10 running on a computer with an i7 CPU and 16 GB RAM.

To compare the performance of each method in micro target estimation, we constructed numerical motion models under different circumstances in 3D-Max software. These models obtained the actual target position in each frame by detecting a mark along

the coordinate axis. Next, we varied the motion states and target speeds in the simulation models. To test the adaptive ability of K^2CF , the initial states of predictor should also be varied. For a fair comparison, all trackers were given the same sample size and frame sequence. For an intuitive demonstration of the tracking effect, the tracking error was denoted as $\Delta\varepsilon = \sqrt{(x' - x)^2 + (y' - y)^2}$, where (x, y) and (x', y') are the true and detected coordinates, respectively.

5.2. Tracking Experiment of a Uniformly Moving Target

In the uniform-speed models, we varied the amplitude of the speed in each frame sequence. Because the sampling time between two frames was equal, the moving speed of the target was represented by the displacement of the target between two frames. The detection results are shown in Figures 12 and 13.

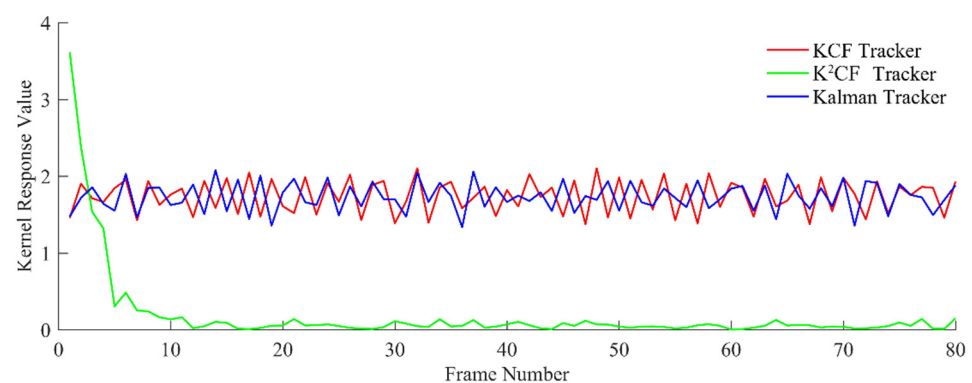


Figure 12. Output detection value of uniform state.

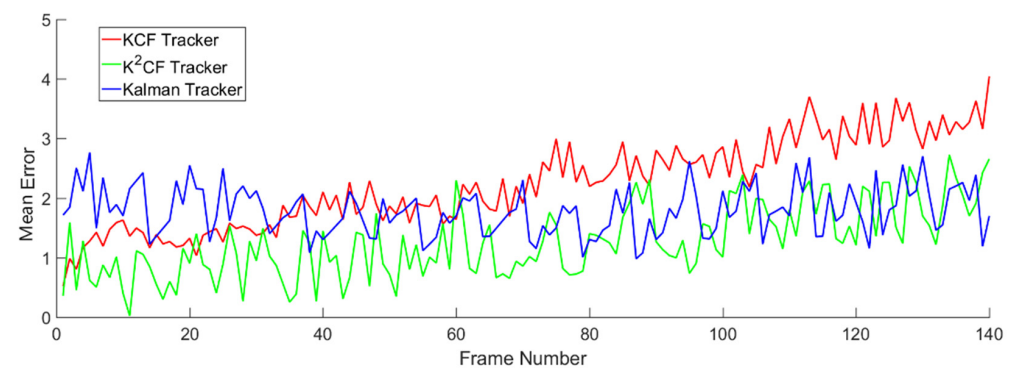


Figure 13. Detection errors of the trackers for a target with uniform velocity.

Figure 12 shows the output value of the kernel correlation detection. The values were almost unchanged across the frame sequence, confirming a constant displacement between pairs of frames. These results reflect the actual motion state. The output value of K^2CF rapidly converged to 0, confirming that the Kalman predictor in K^2CF adjusted the initial prediction speed to the actual speed using the kernel correlation filter. In contrast, the output value of the Kalman tracker converged to a nonzero value, indicating that the Kalman predictor alone cannot adjust the prediction state. Meanwhile, the KCF tracker has a fixed detection range, so its kernel response value was almost constant at some value above 0.

Figure 13 shows the detection error of the three trackers. The mean errors in the KCF, Kalman and K^2CF trackers were $\bar{\Delta}_1 = 0.9597$ pixels, $\bar{\Delta}_2 = 0.9290$ pixels, and $\bar{\Delta}_3 = 0.9074$ pixels, respectively. Apparently, K^2CF achieved a higher detection precision than the other trackers.

Next, the tracking effects of the three trackers were tested on a fast-moving target. For this purpose, a target of radius 3 mm was moved at speeds of $\{v_1, 2v_1, 3v_1, 4v_1, 5v_1, 6v_1, 7v_1 \dots\}$, where $v_1 = \sqrt{v_x^2 + v_y^2}$, $v_x = 1$ mm/s, $v_y = 0.5$ mm/s. The average detection errors of the KCF, Kalman, and K^2CF trackers were obtained and reported at each speed.

When the velocity exceeded $3v_1$, the tracking error of KCF increased sharply and the tracking failed (Figure 14a). The graduations of the velocity axis is clearly seen in Figure 14b. The Kalman tracker exhibited similar behavior. In contrast, the average error of the K^2CF tracker was less than 2 pixels at any target speed, and even the fastest-moving object was stably tracked. This result confirms that the K^2CF tracker perfectly combined the Kalman predictor with the kernel correlation detector, and hence diminished the boundary effect during fast tracking.

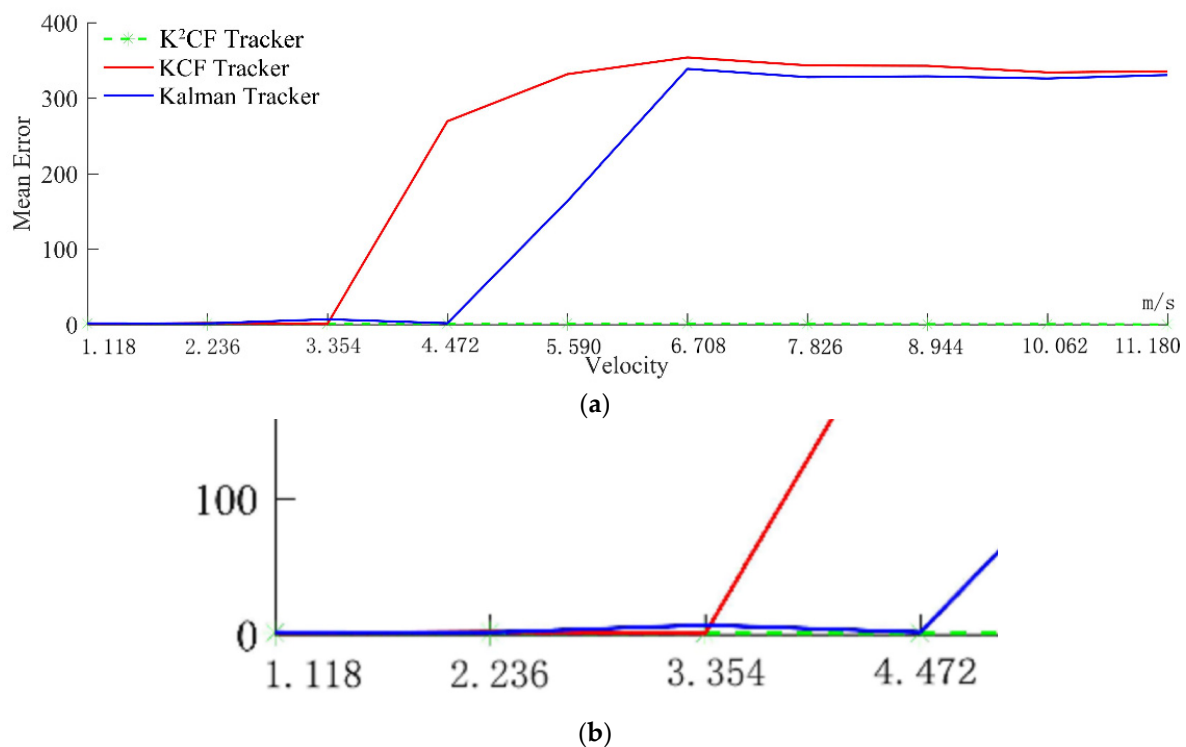


Figure 14. Detectable velocity of uniform state: (a) the mean error of velocity from 1.118 m/s to 11.180 m/s; (b) amplification of velocity error from 1.118 m/s to 4.472 m/s.

5.3. Tracking Experiment with a Uniformly Accelerating Target

In the uniform acceleration model, the accelerated speed was set to $a_x = 0.21$, $a_y = -0.1$, the initial velocity was $v_x = 0.1$, $v_y = -0.4$, and the target radius was 3 mm. The detection results of the three filters are shown in Figures 15 and 16.

The moving distance of the target between two frames increased with velocity. As shown in Figure 15, the output values of the KCF and Kalman trackers initially increased, but (as explained in Section 2), the boundary effect of the KCF causes target loss when the target moves outside the detection range. After frame 42, the KCF tracker lost the target because the object was moving too quickly. The Kalman tracker also lost the target, because its predicted speed cannot adaptively adjust. However, the output value of the K^2CF tracker was always close to 0, demonstrating that the Kalman filter effectively resolved the boundary effect of kernel correlation detection.

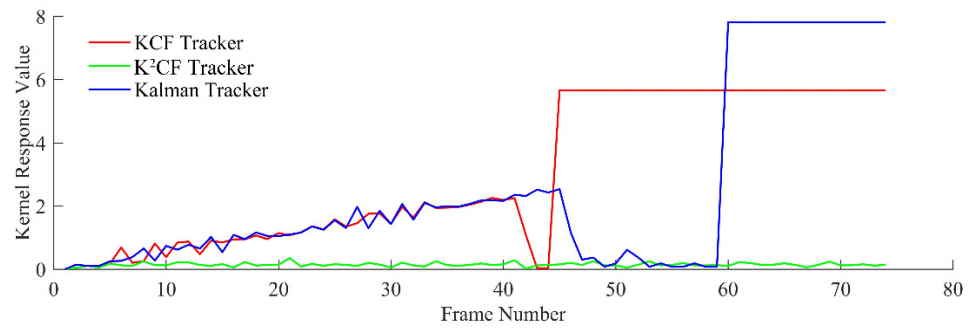


Figure 15. Output detection value of uniform acceleration state.

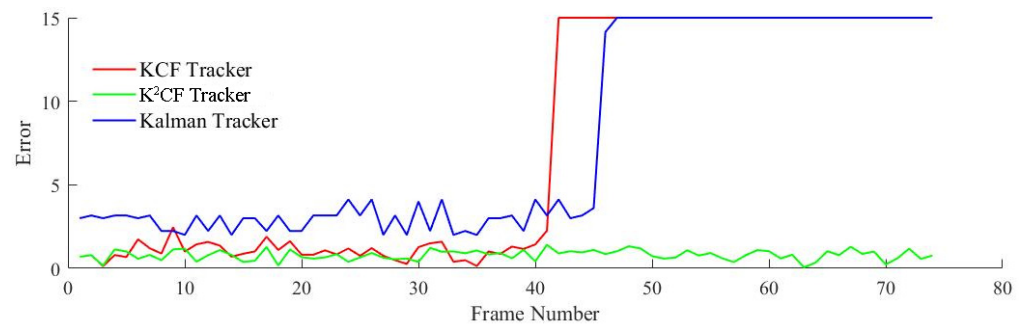


Figure 16. Detection error of uniform acceleration state.

The detection errors of the KCF and Kalman trackers increased steeply at frame 42, and the target was lost shortly afterward. However, the K^2CF tracker maintained continuous tracking with an average error of 0.7956.

To verify the adaptive effect of the K^2CF tracker, we then changed the initial state of the predictor. By observing the predictor speed throughout the tracking process, we can certify the correction effect.

In Figure 17, for different initial velocities, the Kalman predictor in the K^2CF tracker can automatically correct the error between the setting velocity and actual velocity. Nevertheless, the correction range is related to the detection range of the kernel correlation filter according to Equation (23). If the initial error exceeds the target sample size, the Kalman predictor will lose efficacy.

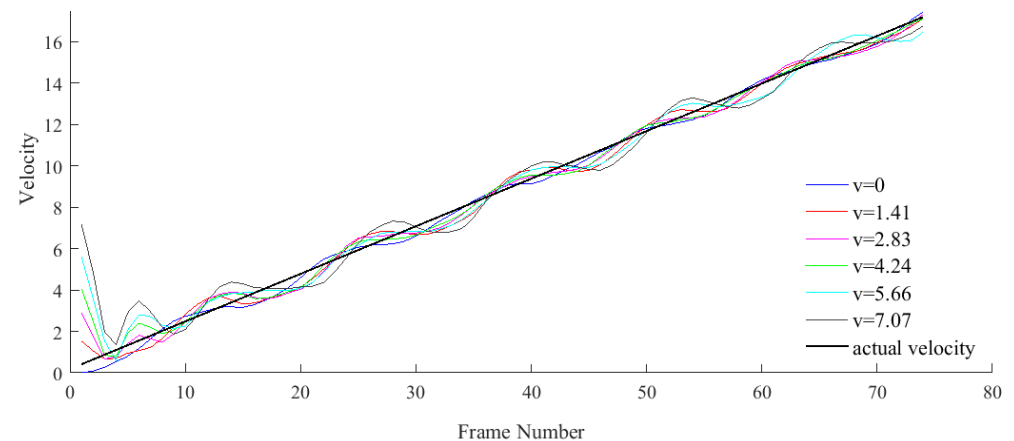


Figure 17. Adaptive velocity adjustment.

As shown in Figure 17, the Kalman predictor in the K^2CF tracker automatically corrected the error between the set and actual velocities, regardless of the velocity magnitude.

Nevertheless, the correction range was related to the detection range of the kernel correlation filter, as dictated by Equation (23). If the initial error exceeds the correction range, the Kalman predictor will lose its efficacy.

5.4. Tracking Experiment of a Nonuniformly Accelerating Target

Actual objects do not move with an ideal uniform speed or acceleration because they are interfered with by system noise. Actual accelerations are nonuniform and affected by different disturbances. Therefore, the tracking performances of KCF and K²CF were tested in a helix motion model with constantly changing accelerations and velocities. (See Figure 18).

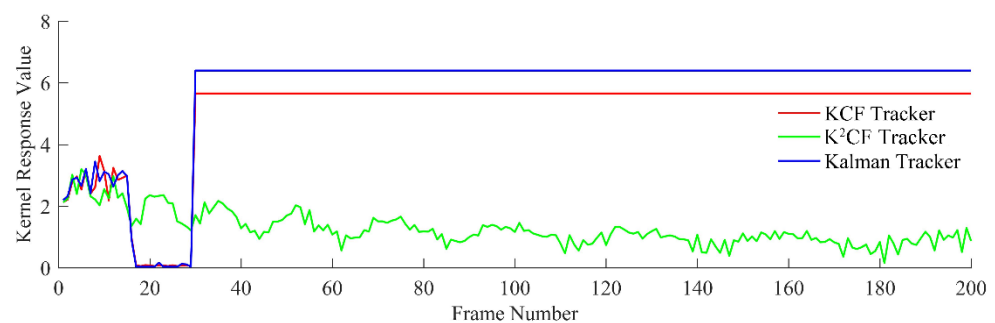


Figure 18. Kernel response value of nonuniform state.

Changing the target speed changed the displacement of the target between two consecutive frames. The output values of these trackers varied accordingly (Figures 18 and 19). When the target moved too quickly, it escaped the detection boundary of the kernel correlation filter, and the KCF tracking failed. Meanwhile, the Kalman tracker could not correct the predictive state in response to the motion variations. Therefore, it easily lost the nonlinearly moving target. Conversely, the Kalman predictor in the K²CF tracker constantly corrected the predicted moving state, and the kernel correlation detection correctly predicted the target position. Therefore, the target remained within the boundary of the detection region and was continuously tracked. The mean tracking accuracy of the K²CF tracker was 1.0704 pixels.

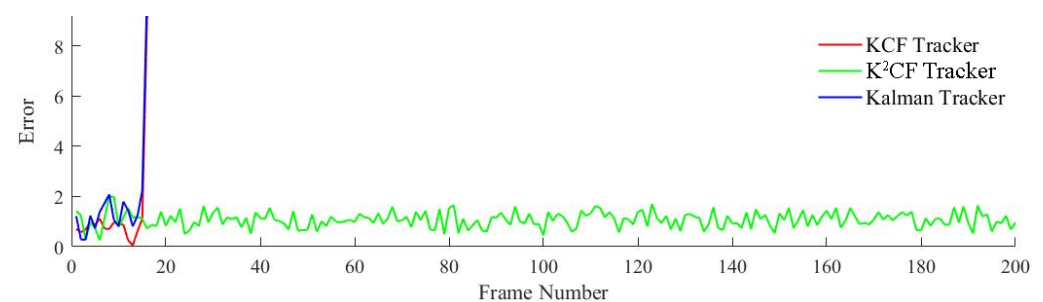


Figure 19. Detection error of nonuniform acceleration state.

5.5. Baseline Comparison

The performances of K²CF and various state-of-the-art methods were compared on OTB-50 and OTB-100 video sequences extracted from the OTB-2013 dataset. These videos present different tracking challenges, such as illumination variation, rotation, scale change, motion blurring, occlusion and fast motion. Figure 20 shows the location precision curves of K²CF and the existing trackers: KCF, visual tracking decomposition (VTD) [64], visual tracking by sampling (VTS) [65], cyclic kernel tracking detection (CSK) [36], structured output tracking with kernels (Struck) [66], the sparsity-based collaborative model (SCM) [67],

local sparse and K-selection tracking (LSK) [68], adaptive structural local sparse appearance (ASLA) [69] and online Ada boosting (OAB) [70].

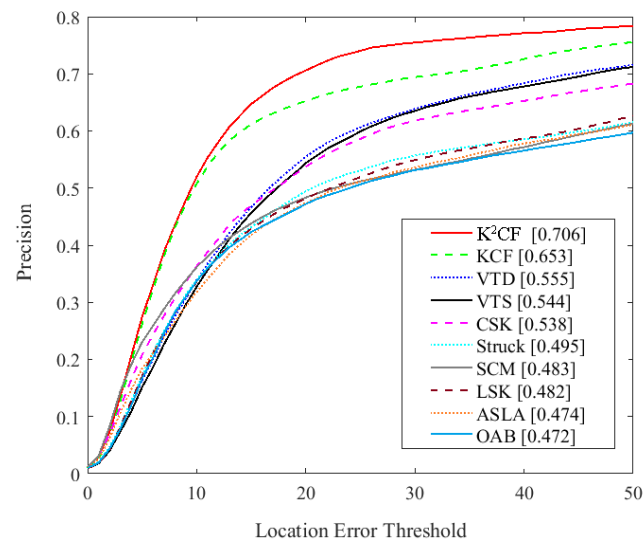


Figure 20. Comparison of state-of-the-art trackers.

As shown in Figure 20, the K²CF achieved the highest mean precision score (0.706) among the tested methods, followed by the KCF. This result confirms the high detection precision of the kernelized correlation filter. Therefore, the KCF-based method can precisely track the moving target.

Because the KCF tracker is sensitive to fast motion, occlusion and scale change, the performance of K²CF was tested in these scenarios. As the test sample, we extracted the “Jogging” and “RedTeam” sequences from the OTB-100 sequences. Our tracker coped well with both occlusion (Figure 21) and scale changes (Figure 22).



Figure 21. Tracking object for occlusion case.



Figure 22. Tracking object for scale change case.

As the K²CF tracker delivers its best performance on fast-moving smaller objects, further analysis is required. Using the visual platform designed in Section 2, we collected pictures of the moving micro-robot at a sampling frame rate of 1.0 frame/s.

The KCF tracker, the minimum output sum of squared error (MOSSE) tracker, and the improved KCF tracker proposed in paper [41] could not track the micro-object correctly, whereas our tracker successfully tracked all frames (Figure 23). The established trackers coped poorly with the fast motions of the micro-object.

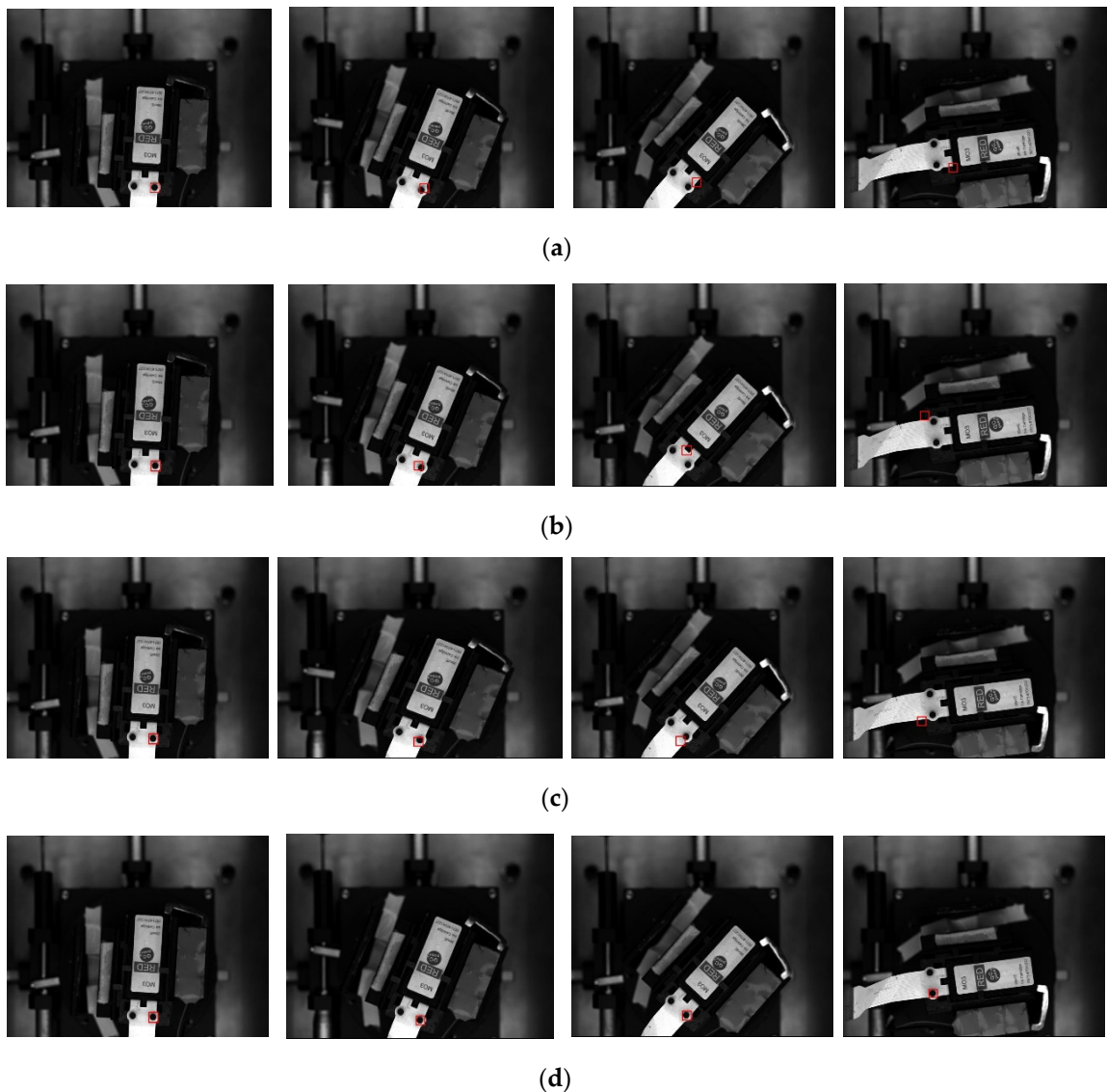


Figure 23. Comparing the trackers' performance for micro-object fast motion case: (a) MOSSE tracker starts to fail at these frames; (b) improved KCF tracker starts to fail at these frames; (c) KCF tracker starts to fail at these frames; (d) K2CF tracker is successful at these frames.

Our tracker markedly outperformed the KCF, MOSSE and the existing improved KCF tracker, particularly when tracking fast-moving micro-objects. We then positioned the micro-robot with the X - Y - θ triaxial mobile platform and obtained the ground truth location of the small object. The tracking speeds (average frame rates) and mean precisions of the four trackers are listed in Table 1.

Table 1. Average precision and frame fps of 4 trackers.

| | Mean of Precision (%) | Standard Deviation of Precision (%) | Mean of fps | Standard Deviation of fps |
|--------------------------------|-----------------------|-------------------------------------|-------------|---------------------------|
| K²CF tracker | 96.45 | 6.7 | 118 | 110.67 |
| Improved KCF | 78.35 | 17.5 | 125 | 111.76 |
| KCF | 80.28 | 33.67 | 135 | 125.64 |
| MOSSE | 65.17 | 36.89 | 254 | 207.4 |

As shown in Table 1, K²CF achieved the highest precision and a mean speed of 110.67 fps. Although KCF operated at 125.64 fps, our tracker is suitable for real-time applications while also improving location precision.

6. High-Precision Pose Measurement Experiments

6.1. Calibration of the Experimental Platform

The effectiveness of the proposed K²CF algorithm was checked on an experimental calibration platform. Following the observation method in [71], we employed a (63 × 63 × 6) mm³ ceramic checkerboard with a precision of 2 μm as the calibration plate. The calibration plate was photographed from different viewpoints relative to the camera (Figure 24).

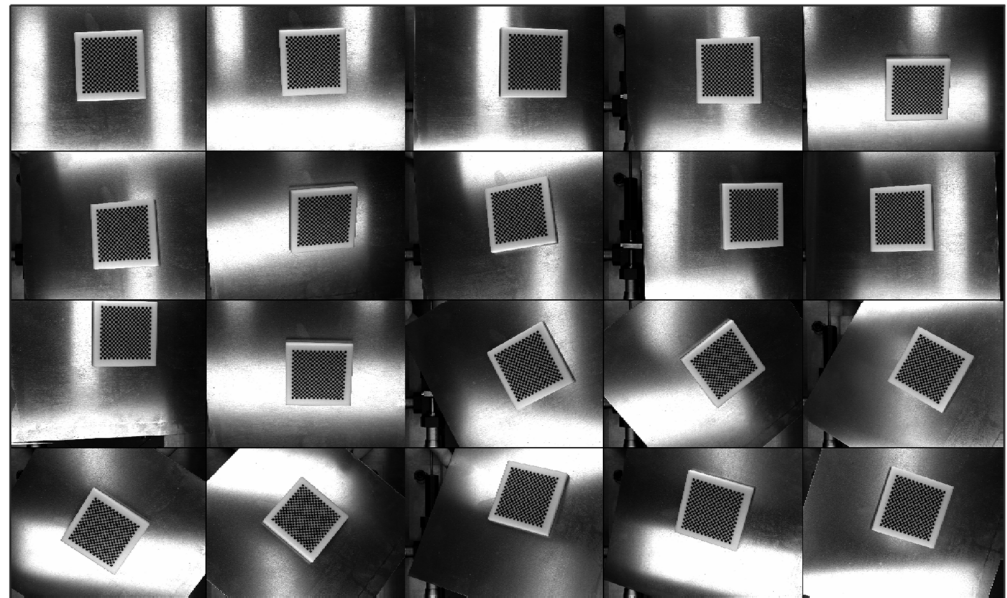


Figure 24. Calibration pattern of camera system.

From the calibration operation, we obtained the inner parameter matrix M_1 , the outer parametric rotation matrix R_1 , and the translation vector T :

$$\left\{ \begin{array}{l} M_1 = \begin{bmatrix} 10313.8 & 0 & 2006.7 \\ 0 & 10313.9 & 1565.8 \\ 0 & 0 & 1 \end{bmatrix} \\ R_1 = \begin{bmatrix} 0.0003 & 0.9997 & -0.0217 \\ 0.9975 & -0.0018 & -0.0694 \\ -0.0694 & -0.0217 & -0.9974 \end{bmatrix} \\ T = [-24.3039 \quad -18.7007 \quad 467.3700] \end{array} \right. \quad (31)$$

Using the rotation matrix R and the translation vector T , we then established the mapping relationship between the image and motion coordinate planes. Because the actual coordinates of the chessboard were known, we extracted the image coordinates of the corner points and calculated them in the moving plane. Comparing the calculated and actual coordinates, we finally obtained the coordinate conversion error. From Figure 25, the average errors in the coordinate transformation were determined as $\text{error}_x = 0.044$ mm and $\text{error}_y = 0.037$ mm.

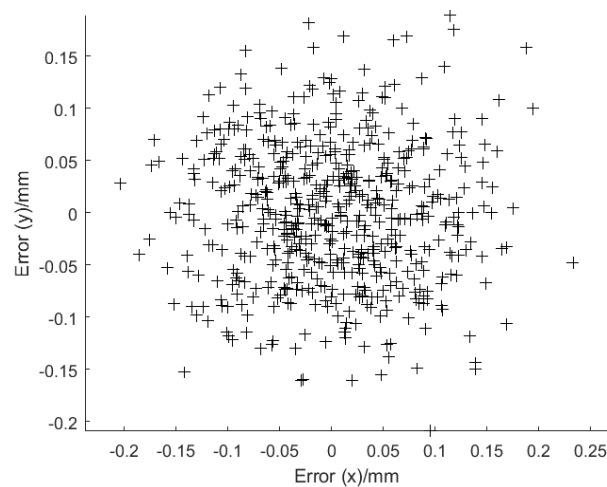


Figure 25. Error distribution of coordinate transformations.

6.2. Pose Measurement Precision Experiment

To test the precision of the pose measurement system, we should also obtain the ground-truth location of the micro-robot. The X - Y - θ triaxial mobile platform locates the micro-robot with a position precision of $10\ \mu\text{m}$ and a rotation precision of 0.1° . Therefore, we can monitor the ground truth pose of the micro-robot as it moves. Comparing the ground-truth pose with the measured pose, we can obtain the precision of the pose measurements.

Because the Kalman predictor corrects the predicted motion error in any state, the kernel correlation filter can adapt to different moving states of the target. The correction ability of the Kalman predictor is related to the range of kernel correlation detection. Acceleration interference increases the displacement between the frames, so the tracking target will be lost when the increment of the acceleration displacement exceeds the detection boundary.

In the pose-measurement experiment, we varied the inter-frame acceleration displacement s between the images. Here, $s = \sqrt{x^2 + y^2}$, where $x = x_i - x_{i-1}$, $y = y_i - y_{i-1}$ are the acceleration displacements in the x and y directions, respectively. The target was tracked by the K^2CF algorithm, and the error e of the system pose measurement was calculated as $e = \sqrt{(\Delta x)^2 + (\Delta y)^2}$, where Δx and Δy are the errors in the x and y directions, respectively. The acceleration displacement between the frames was then calculated.

As shown in Figure 26, the target was lost when the acceleration displacement between two frames exceeded 5.6 mm. However, at such large acceleration displacements, the maximum inter-frame displacement of the target can reach 8 mm. The pose measurement platform satisfies the moving speed requirements of the micro-robot.

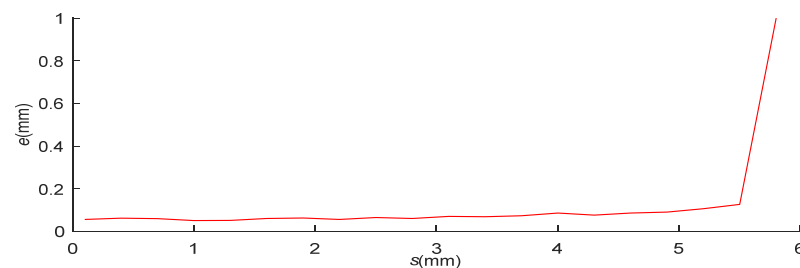


Figure 26. Detection error versus acceleration displacement of the pose measurement system.

Figures 27 and 28 present the position error s and attitude error θ in the actual pose measurements, respectively. The average accuracies of the position and angle measurements were 0.0729 and 0.0824° , respectively, satisfying the precision requirements of the system.

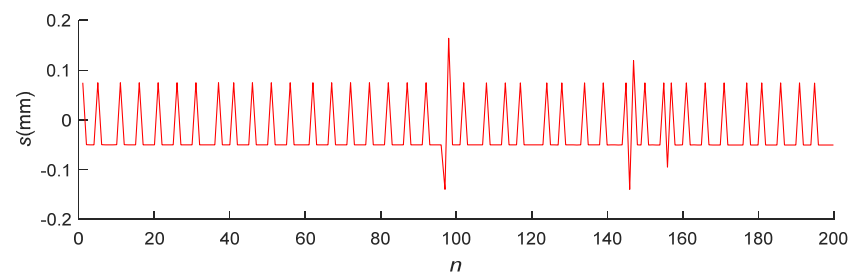


Figure 27. Position detection error of the pose measurement system.

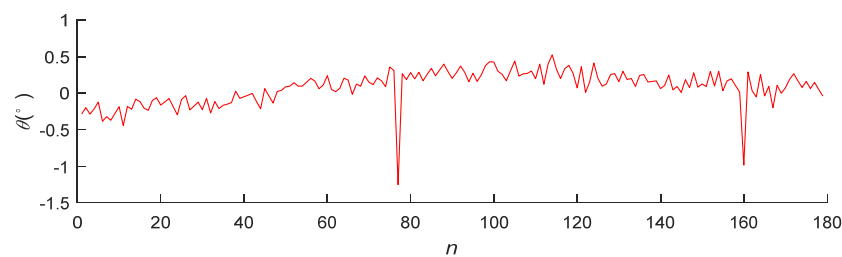


Figure 28. Angle detection error of the pose measurement system.

7. Conclusions

To mitigate the error fluctuations and tracking losses in high-precision pose measurements of a moving target, we proposed the K^2CF algorithm. By removing the boundary effect of the KCF algorithm, K^2CF accurately tracks a fast-moving micro-target. The pose of a micro-robot in a two-dimensional plane was monitored on a visual test platform. Referring to the research goals, the main conclusions are summarized below.

1. A mapping relationship between the moving and image coordinate systems was constructed using the camera calibration principle. A kernel relation filter and a Kalman filter were combined into an adaptive tracking model. The object features were extracted by the region growth method.
2. The feature profile was extracted by the Canny operator, and the feature points were precisely extracted by optimizing the edge points using the least-squares method.
3. The boundary effect of the KCF algorithm was resolved by a Kalman filter that predicts the target position and ensures that the candidate samples lie within the boundary. Comparative simulations confirmed that the proposed K^2CF algorithm adapts to different moving states of the target and corrects the initial velocity error.
4. The precision of the system was experimentally evaluated on a small target of radius 3 mm. For acceleration displacements lower than 5.6 mm, the average position and angle accuracies were 0.0729 mm and 0.0824° , respectively.

In summary, this paper provides an adaptive method for tracking a moving target. The pose detection system lays a foundation for the trajectory planning of micro-mobile robots. The fast target tracking algorithm achieves precise and stable tracking with a certain versatility. However, stable and accurate tracking is difficult to achieve in the long term because it is interfered with by target occlusion and scale change. Solving these problems to maintain long-term stability and high-precision tracking is our future research goal.

Author Contributions: Conceptualization, H.Z. (Heng Zhang) and L.Z.; methodology, H.Z. (Heng Zhang) and F.X.; software, H.Z. (Heng Zhang) and X.D.; validation, H.Z. (Heng Zhang), H.Z. (Hongwu Zhan) and L.Z.; formal analysis, L.Z. and F.X.; investigation, H.Z. (Heng Zhang) and H.Z. (Hongwu Zhan); resources, H.Z. (Hongwu Zhan); data curation, H.Z. (Hongwu Zhan); writing—original draft preparation, H.Z. (Heng Zhang); writing—review and editing, H.Z. (Heng Zhang); visualization, L.Z.; supervision, F.X.; project administration, H.Z. (Hongwu Zhan); funding acquisition, H.Z. (Hongwu Zhan). All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Key R&D Program of China under Grant 2018YFB1309400.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

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

References

1. Jia, Q.; Wang, S.; Chen, G.; Wang, L.; Sun, H. A Novel Optimal Design of Measurement Configurations in Robot Calibration. *Math. Probl. Eng.* **2018**, *2018*, 1–17. [\[CrossRef\]](#)
2. Zhang, L.; Lv, H.; Tan, D.; Xu, F.; Chen, J.; Bao, G.; Cai, S. Adaptive quantum genetic algorithm for task sequence planning of complex assembly systems. *Electron. Lett.* **2018**, *54*, 870–872. [\[CrossRef\]](#)
3. Ji, S.; Xiao, F.; Tan, D. Analytical method for softness abrasive flow field based on discrete phase model. *Sci. China Ser. E Technol. Sci.* **2010**, *53*, 2867–2877. [\[CrossRef\]](#)
4. Li, C.; Ji, S.-M.; Tan, D.-P. Softness abrasive flow method oriented to tiny scale mold structural surface. *Int. J. Adv. Manuf. Technol.* **2012**, *61*, 975–987. [\[CrossRef\]](#)
5. Tan, D.; Zhang, L.; Ai, Q. An embedded self-adapting network service framework for networked manufacturing system. *J. Intell. Manuf.* **2019**, *30*, 539–556. [\[CrossRef\]](#)
6. Zheng, N.; Jin, M.; Hong, H.; Huang, L.; Gu, Z.; Li, H. Real-time and precise insect flight control system based on virtual reality. *Electron. Lett.* **2017**, *53*, 387–389. [\[CrossRef\]](#)
7. Tan, D.-P.; Li, P.-Y.; Ji, Y.-X.; Wen, D.-H.; Li, C. SA-ANN-Based Slag Carry-Over Detection Method and the Embedded WME Platform. *IEEE Trans. Ind. Electron.* **2013**, *60*, 4702–4713. [\[CrossRef\]](#)
8. Zheng, N.; Su, L.; Zhang, D.; Gao, L.; Yao, M.; Wu, Z. A computational model for ratbot locomotion based on cyborg intelligence. *Neurocomputing* **2015**, *170*, 92–97. [\[CrossRef\]](#)
9. Zeng, X.; Ji, S.-M.; Jin, M.; Tan, D.-P.; Ge, J.-Q. Research on dynamic characteristic of softness consolidation abrasives in machining process. *Int. J. Adv. Manuf. Technol.* **2015**, *82*, 1115–1125. [\[CrossRef\]](#)
10. Zhan, H.; Xu, F.; Ni, Z. Fluid dynamic modeling and fuzzy proportional–integral–derivative–based ink-supply method for piezoelectric ink-jet printing. *Adv. Mech. Eng.* **2017**, *9*, 1–17. [\[CrossRef\]](#)
11. Tan, D.-P.; Ni, Y.-S.; Zhang, L.-B. Two-phase sink vortex suction mechanism and penetration dynamic characteristics in ladle teeming process. *J. Iron Steel Res. Int.* **2017**, *24*, 669–677. [\[CrossRef\]](#)
12. Pire, T.; Baravalle, R.; D’Alessandro, A.; Civera, J. Real-time dense map fusion for stereo SLAM. *Robotica* **2018**, *36*, 1510–1526. [\[CrossRef\]](#)
13. Chen, J.; Xu, F.; Tan, D.-P.; Shen, Z.; Zhang, L.; Ai, Q. A control method for agricultural greenhouses heating based on computational fluid dynamics and energy prediction model. *Appl. Energy* **2015**, *141*, 106–118. [\[CrossRef\]](#)
14. Stavrou, D.; Timotheou, S.; Panayiotou, C.G.; Polycarpou, M. Optimizing Container Loading with Autonomous Robots. *IEEE Trans. Autom. Sci. Eng.* **2018**, *15*, 717–731. [\[CrossRef\]](#)
15. Talla, J.; Leu, V.Q.; Smidl, V.; Peroutka, Z. Adaptive Speed Control of Induction Motor Drive with Inaccurate Model. *IEEE Trans. Ind. Electron.* **2018**, *65*, 8532–8542. [\[CrossRef\]](#)
16. Tan, D.-P.; Ji, S.-M.; Jin, M.-S. Intelligent Computer-Aided Instruction Modeling and a Method to Optimize Study Strategies for Parallel Robot Instruction. *IEEE Trans. Educ.* **2013**, *56*, 268–273. [\[CrossRef\]](#)
17. Li, J.; Ji, S.; Tan, D. Improved Soft Abrasive Flow Finishing Method Based on Turbulent Kinetic Energy Enhancing. *Chin. J. Mech. Eng.* **2017**, *30*, 301–309. [\[CrossRef\]](#)
18. Kelly, I.; Martinoli, A. A scalable, on-board localisation and communication system for indoor multi-robot experiments. *Sens. Rev.* **2004**, *24*, 167–180. [\[CrossRef\]](#)
19. Pugh, J.; Martinoli, A. Relative localization and communication module for small-scale multi-robot systems. In Proceedings of the 2006 IEEE International Conference on Robotics and Automation, Orlando, FL, USA, 15–19 May 2006; pp. 188–193.
20. Qazizada, M.E.; Pivarčiová, E. Mobile Robot Controlling Possibilities of Inertial Navigation System. *Procedia Eng.* **2016**, *149*, 404–413. [\[CrossRef\]](#)
21. Chu, H.K.H.; Mills, J.K.; Cleghorn, W.L. Dynamic tracking of moving objects in microassembly through visual servoing. In Proceedings of the 2010 IEEE International Conference on Mechatronics and Automation, Xi’an, China, 4–7 August 2010; pp. 1738–1743.
22. Zhang, L.; Yuan, Z.; Tan, D.; Huang, Y. An Improved Abrasive Flow Processing Method for Complex Geometric Surfaces of Titanium Alloy Artificial Joints. *Appl. Sci.* **2018**, *8*, 1037. [\[CrossRef\]](#)
23. Tan, D.-P.; Li, L.; Zhu, Y.-L.; Zheng, S.; Yin, Z.-C.; Li, D.-F. Critical penetration condition and Ekman suction-extraction mechanism of a sink vortex. *J. Zhejiang Univ. A* **2019**, *20*, 61–72. [\[CrossRef\]](#)
24. Khan, A.; Imran, M.; Rashid, A. Aerial Camera Network for Observing Moving Targets. *IEEE Sens. J.* **2018**, *18*, 6847–6856. [\[CrossRef\]](#)
25. Li, Y.; Tan, D.-P.; Wen, D.; Ji, S.; Cai, D. Parameters optimization of a novel 5-DOF gasbag polishing machine tool. *Chin. J. Mech. Eng.* **2013**, *26*, 680–688. [\[CrossRef\]](#)

26. Tan, D.-P.; Chen, S.-T.; Bao, G.-J.; Zhang, L.-B. An embedded lightweight GUI component library and ergonomics optimization method for industry process monitoring. *Front. Inf. Technol. Electron. Eng.* **2018**, *19*, 604–625. [[CrossRef](#)]
27. Li, J.; Yu, H.; Zhou, L.; Liang, H.; Wang, L. An Adaptive Unscented Particle Filter Tracking Algorithm Based on Color Distribution and Wavelet Moment. In Proceedings of the 2007 2nd IEEE Conference on Industrial Electronics and Applications, Harbin, China, 23–25 May 2007; pp. 2218–2223.
28. Du, K.; Ju, Y.; Jin, Y. Mean shift tracking algorithm with adaptive block color histogram. *J. Wuhan Univ. Technol.* **2012**, *34*, 2692–2695.
29. Liu, M.; Wu, C.; Zhang, Y. Multi-resolution optical flow tracking algorithm based on multi-scale Harris corner points feature. In Proceedings of the 2008 Chinese Control and Decision Conference, Yantai, China, 2–4 July 2008; pp. 5287–5291.
30. Manasa, K.; Channappayya, S.S. An Optical Flow-Based Full Reference Video Quality Assessment Algorithm. *IEEE Trans. Image Process.* **2016**, *25*, 2480–2492. [[CrossRef](#)] [[PubMed](#)]
31. Chen, S.; Tan, D. A SA-ANN-Based Modeling Method for Human Cognition Mechanism and the PSACO Cognition Algorithm. *Complexity* **2018**, *2018*, 6264124. [[CrossRef](#)]
32. Tan, D.-P.; Li, P.-Y.; Pan, X.-H. Application of Improved HMM Algorithm in Slag Detection System. *J. Iron Steel Res. Int.* **2009**, *16*, 1–6. [[CrossRef](#)]
33. Ji, S.; Weng, X.; Tan, D.P. Analytical method of softness abrasive two-phase flow field based on 2D model of LSM. *ACTA Phys. Sin.* **2012**, *61*, 010205.
34. Kalal, Z.; Mikolajczyk, K.; Matas, J. Tracking-Learning-Detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **2012**, *34*, 1409–1422. [[CrossRef](#)] [[PubMed](#)]
35. Bolme, D.; Beveridge, J.R.; Draper, B.A.; Lui, Y.M. Visual object tracking using adaptive correlation filters. In Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Francisco, CA, USA, 13–18 June 2010; pp. 2544–2550.
36. Henriques, J.F.; Caseiro, R.; Martins, P.; Batista, J. Exploiting the Circulant Structure of Tracking-by-Detection with Kernels. In Proceedings of the European Conference on Computer Vision; Springer: Berlin/Heidelberg, Germany, 2012; pp. 702–715.
37. Dalal, N. Histograms of oriented gradients for human detection. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Diego, CA, USA, 20–25 June 2005; pp. 886–893.
38. Henriques, J.F.; Caseiro, R.; Martins, P.; Batista, J. High-Speed Tracking with Kernelized Correlation Filters. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *37*, 583–596. [[CrossRef](#)]
39. Wu, Y.; Lim, J.; Yang, M.-H. Object Tracking Benchmark. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 1834–1848. [[CrossRef](#)] [[PubMed](#)]
40. Yang, L.; Zhu, J. A Scale Adaptive Kernel Correlation Filter Tracker with Feature Integration. In Proceedings of the European Conference on Computer Vision, Zürich, Switzerland, 6–12 September 2014; pp. 254–265.
41. Huynh, X.-P.; Choi, I.-H.; Kim, Y.-G. *Tracking a Human Fast and Reliably Against Occlusion and Human-Crossing*; Springer: Berlin/Heidelberg, Germany, 2016; Volume 53, pp. 461–472.
42. Li, C.; Ji, S.-M.; Tan, D.-P. Multiple-Loop Digital Control Method for a 400-Hz Inverter System Based on Phase Feedback. *IEEE Trans. Power Electron.* **2012**, *28*, 408–417. [[CrossRef](#)]
43. Tan, D.P.; Li, P.Y.; Pan, S.H. Intelligent industry monitoring network architecture UPnP based. *Chin. J. Electron.* **2008**, *17*, 607–610.
44. Tan, D.; Ji, S.; Li, P.; Pan, X. Development of vibration style ladle slag detection methods and the key technologies. *Sci. China Ser. E Technol. Sci.* **2010**, *53*, 2378–2387. [[CrossRef](#)]
45. El Hazzat, S.; Merras, M.; El Akkad, N.; Saaidi, A.; Satori, K. 3D reconstruction system based on incremental structure from motion using a camera with varying parameters. *Vis. Comput.* **2017**, *34*, 1443–1460. [[CrossRef](#)]
46. Zeng, X.; Ji, S.-M.; Jin, M.; Tan, D.-P.; Li, J.-H.; Zeng, W.-T. Investigation on machining characteristic of pneumatic wheel based on softness consolidation abrasives. *Int. J. Precis. Eng. Manuf.* **2014**, *15*, 2031–2039. [[CrossRef](#)]
47. Li, L.; Qi, H.; Yin, Z.; Li, D.; Zhu, Z.; Tangwarodomnukun, V.; Tan, D. Investigation on the multiphase sink vortex Ekman pumping effects by CFD-DEM coupling method. *Powder Technol.* **2020**, *360*, 462–480. [[CrossRef](#)]
48. Shahraki, F.F.; Pour Yazdanpanah, A.; Regentova, E.E.; Muthukumar, V. A Trajectory Based Method of Automatic Counting of Cyclist in Traffic Video Data. *Int. J. Artif. Intell. Tools* **2017**, *26*. [[CrossRef](#)]
49. Pan, Y.; Ji, S.; Tan, D.; Cao, H. Cavitation-based soft abrasive flow processing method. *Int. J. Adv. Manuf. Technol.* **2020**, *109*, 2587–2602. [[CrossRef](#)]
50. Tan, D.-P.; Li, L.; Zhu, Y.-L.; Zheng, S.; Ruan, H.-J.; Jiang, X.-Y. An Embedded Cloud Database Service Method for Distributed Industry Monitoring. *IEEE Trans. Ind. Inform.* **2017**, *14*, 2881–2893. [[CrossRef](#)]
51. Jeong, S.; Kim, G.; Lee, S. Effective Visual Tracking Using Multi-Block and Scale Space Based on Kernelized Correlation Filters. *Sensors* **2017**, *17*, 433. [[CrossRef](#)]
52. Phelan, C.E.; Marazzina, D.; Fusai, G.; Germano, G. Fluctuation identities with continuous monitoring and their application to the pricing of barrier options. *Eur. J. Oper. Res.* **2018**, *271*, 210–223. [[CrossRef](#)]
53. Tan, D.-P.; Zhang, L.-B. A WP-based nonlinear vibration sensing method for invisible liquid steel slag detection. *Sens. Actuators B Chem.* **2014**, *202*, 1257–1269. [[CrossRef](#)]
54. Ali, H.; Nasser; Hassan, G.M. Kalman filter tracking. *Int. J. Comput. Appl.* **2014**, *89*, 15–18.

55. Zhang, L.; Wang, J.-S.; Tan, D.-P.; Yuan, Z.-M. Gas compensation-based abrasive flow processing method for complex titanium alloy surfaces. *Int. J. Adv. Manuf. Technol.* **2017**, *92*, 3385–3397. [[CrossRef](#)]
56. Zeng, X.; Ji, S.; Tan, D.; Jin, M.; Wen, D.; Zhang, L. Development of vibration style ladle slag detection method and the key technologies consolidation abrasives material removal characteristic oriented to laser hardening surface. *Int. J. Adv. Manuf. Technol.* **2010**, *69*, 2323–2332. [[CrossRef](#)]
57. Khan, F.A.; Khelifi, F.; Tahir, M.A.; Bouridane, A. Dissimilarity Gaussian Mixture Models for Efficient Offline Handwritten Text-Independent Identification Using SIFT and RootSIFT Descriptors. *IEEE Trans. Inf. Forensics Secur.* **2019**, *14*, 289–303. [[CrossRef](#)]
58. Mingge, W.; Congda, L.; Dapeng, T.; Tao, H.; Guohai, C.; Donghui, W. Effects of metal buffer layer for amorphous carbon film of 304 stainless steel bipolar plate. *Thin Solid Films* **2016**, *616*, 507–514. [[CrossRef](#)]
59. Da-Peng, T.; Tao, Y.; Jun, Z.; Shi-Ming, J. Free sink vortex Ekman suction-extraction evolution mechanism. *Acta Phys. Sin.* **2016**, *65*, 054701. [[CrossRef](#)]
60. Gao, Z. Reduced order Kalman filter for a continuous-time fractional-order system using fractional-order average derivative. *Appl. Math. Comput.* **2018**, *338*, 72–86. [[CrossRef](#)]
61. Ge, J.-Q.; Ji, S.-M.; Tan, D.-P. A gas-liquid-solid three-phase abrasive flow processing method based on bubble collapsing. *Int. J. Adv. Manuf. Technol.* **2017**, *95*, 1069–1085. [[CrossRef](#)]
62. Weng, S.-K.; Kuo, C.-M.; Tu, S.-K. Video object tracking using adaptive Kalman filter. *J. Vis. Commun. Image Represent.* **2006**, *17*, 1190–1208. [[CrossRef](#)]
63. Tan, D.-P.; Ji, S.-M.; Fu, Y.-Z. An improved soft abrasive flow finishing method based on fluid collision theory. *Int. J. Adv. Manuf. Technol.* **2016**, *85*, 1261–1274. [[CrossRef](#)]
64. Kwon, J.; Lee, K.M. Visual tracking decomposition. In Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Francisco, CA, USA, 13–18 June 2010; pp. 1269–1276.
65. Kwon, J.; Lee, K.M. Tracking by Sampling Trackers. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; pp. 1195–1202.
66. Hare, S.; Golodetz, S.; Saffari, A.; Vineet, V.; Cheng, M.-M.; Hicks, S.L.; Torr, P.H.S. Struck: Structured Output Tracking with Kernels. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *38*, 2096–2109. [[CrossRef](#)] [[PubMed](#)]
67. Zhong, W.; Lu, H.; Yang, M.-H. Robust object tracking via sparsity-based collaborative model. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 1838–1845.
68. Liu, B.; Huang, J.; Yang, L.; Kulikowsk, C. Robust tracking using local sparse appearance model and K-selection. In Proceedings of the CVPR 2011, Colorado Springs, CO, USA, 20–25 June 2011; pp. 1313–1320.
69. Jia, X.; Lu, H.; Yang, M.-H. Visual tracking via adaptive structural local sparse appearance model. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 1822–1829.
70. Grabner, H.; Bischof, H. Real-Time Tracking via On-line Boosting. In Proceedings of the British Machine Vision Conference 2006, Edinburgh, UK, 4–7 September 2006; pp. 47–56.
71. Zhang, Z. Camera calibration with one-dimensional objects. *IEEE Trans. Pattern Anal. Mach. Intell.* **2004**, *26*, 892–899. [[CrossRef](#)]