Multimedia Tools and Applications (2019) 78:27403–27424
https://doi.org/10.1007/s11042-019-07869-7

Real-time dynamic gesture recognition and hand servo tracking using PTZ camera

Songxiao Cao [1] · Xuanyin Wang [2]

Received: 9 October 2018 / Revised: 23 April 2019 / Accepted: 5 June 2019 /
Published online: 18 June 2019

Abstract
A technology of real-time dynamic gesture recognition and hand tracking using a Pan-Tilt-Zoom (PTZ) camera was presented in this study. It was aimed to achieve robust scheme that stably recognized simple hand gestures and tracked the hand by means of a PTZ camera to keep the fingertip remaining in the center of the camera. For this purpose, the hand region was initially segmented in a cluttered environment using skin color segmentation in YCbCr color space to get the silhouette of the hand. Furthermore, the Monte Carlo Sampling method was used to estimate the Cubic Bezier curves best fitted to the sub contour points centralized in each contour point, and the fingertips were detected by combining the local maximums of a cumulative curvature with detection of convex defects. After that, feature triangle analysis was utilized to achieve dynamic recognition of simple gestures including "right click down" and "right click up". Finally, the PTZ camera was driven by the algorithm to achieve servo tracking with the target fingertip when the gesture "right click down" was detected. As is shown by the experimental results, the proposed approach recognized the dynamic gestures and located the fingertips' positions precisely, and realized the follow-up servo tracking using PTZ camera in real-time.

Keywords Gesture recognition · Fingertip detection · PTZ servo tracking · Bezier curve fitting

1 Introduction

In recent years, hand detection, gesture recognition and tracking applied to perceptive user interfaces and virtual reality have gained growing attention [7, 23, 25]. Even though hand posture

✉ Songxiao Cao
caosongxiao@cjlu.edu.cn

Xuanyin Wang
xywang@zju.edu.cn

[1] College of Metrology and Measurement Engineering, China Jiliang University, Hangzhou 310018, China

[2] State Key Laboratory of Fluid Power and Mechatronic System, Zhejiang University, Hangzhou 310027, China

recognition (static hand gesture) has been reported to be very successful in HCI (Human Computer Interaction), dynamic gesture is continually faced with many obstacles for accurate recognition and tracking in single camera due to following reasons [24]. Initially, the modeling of hand gesture is difficult resulting from the deformable, articulated structure of hands and the self-occlusion problem. Secondly, the cluttered environment and varying lighting conditions lead to inaccurate hand segmentation. Thirdly, the hand appearances vary among people of different races, ages, weights, and so on. In order to solve these problems, several hand gesture recognition and tracking methods have been proposed and specifically categorized into the following two classes.

Firstly, hand gestures are modeled and recognized using geometrical characteristics of the segmented hand region [4, 5, 18, 37]. In this method, once the segmented image is obtained, convex hull and convexity defects are calculated so as to find the area of hand and finger position, and whereafter vertices of the convex hull are considered as the fingertips. It causes some mistakes when the vertices represent other corners of hand silhouette rather than fingertips, which leads to mistakes in gesture recognition and tracking. Secondly, the depth image is used for gesture recognition and hand tracking by some researchers [12, 28, 31]. The advantage of 3D cameras lies in higher discrimination between gestures and background while the disadvantage lies in relatively lower resolution. In addition, under the condition of equivalent resolution, the price of 3D cameras is much higher than that of ordinary cameras. Thirdly, deep learning has drawn much attention nowadays in gesture recognition and hand tracking [8, 19, 29]. Methods based on deep learning performs excellently in some particular gestures through long time training, whereas the time consumed in real-time applications is unsatisfactory. Fourthly, other algorithms based on models are used in some literatures. In literature [24], the authors proposed a centroid tracking of hand gestures that captured and retained time sequence information for feature extraction. Zhang [36] used a multi-cue integration hand tracking method by integrating the motion and color cues from a feature point selection view.

In the process of human-computer interaction, there is a problem that has always been existing yet received hardly any attention. That is, the human-computer interaction system using a static camera will stop work after the human gesture leaves the camera's field of view. Actually, a PTZ camera is a desirable option to solve this problem. Nevertheless, none of these literatures focused on the usage of a PTZ camera combined with a dynamic hand tracking algorithm to achieve servo tracking.

Distinguished from the existing approaches, a real-time dynamic gesture recognition and hand tracking using a PTZ camera was proposed in this study, and the aim of which was to achieve a robust scheme that stably recognized simple hand gestures, and tracked the hand using a PTZ camera to keep the fingertip remaining in the center of the camera.

Specifically, the hand region in a cluttered environment was segmented using skin color segmentation in YCbCr color space and by which the silhouette of the hand was got. For every point in the silhouette, a sub contour point set was constructed, and the Monte Carlo Sampling method was used to obtain the best fitted Cubic Bezier curve to the sub contour set. Subsequently, the curvature on the continuous Cubic Bezier curve of the corresponding point was considered as the estimated curvature of the point in the silhouette. Giving the estimated curvatures, the local maximums of a cumulative curvature curve were detected as the candidate fingertips. After that, geometry feature analysis including convexity approach and feature triangle analysis were used to locate the final fingertip and recognize several specific simple gestures. Finally, if a tracking gesture (right click down) was recognized, the system drove the PTZ camera to track the fingertip, and kept the fingertip always in the middle field of view of the camera in real-time. The overview of the proposed method is shown in Fig. 1.

The innovative points of the proposed method included:

- A new fingertip detection method was proposed based on Monte Carlo sampling Cubic Bezier curve fitting.
- A feature triangle analysis was used to realize simple dynamic gesture recognition.
- The gesture recognition was combined with a PTZ camera to achieve servo tracking in real-time.

This paper is organized as follows. Section 2 details the proposed gesture recognition method including hand region detection and extraction, fingertip detection based on Monte Carlo Sampling cubic Bezier curve fitting, and simple gesture modeling using feature triangle analysis. Furthermore, Section 3 introduces the experimental results and analyses. Last but not least, Section 4 outlines the conclusions and future research.

2 The proposed gesture recognition and tracking method

2.1 Hand region detection and extraction

The first step of finger detection was detecting the hand region which exerted an essential role in the whole process since a complete and accurate hand region laid the foundation for further analysis. In the proposed method, this was achieved through skin color segmentation in the YCbCr color space which was perceptually uniform and widely used in video compression standards. Additionally, it performed well in the separation of luminance (Y) and chrominance (Cb, Cr) together with the compactness of the skin cluster. An image with RGB color space was transformed to YCbCr color space using the following equation:

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} + \begin{bmatrix} 65.481 & 128.553 & 24.996 \\ -37.797 & -74.203 & 112 \\ 112 & -93.768 & -18.214 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (1)$$

In this paper, the extraction of hand region from background was accomplished using a threshold technique proposed by Chai and Ngan [6]. Beyond that, the threshold range of Cb

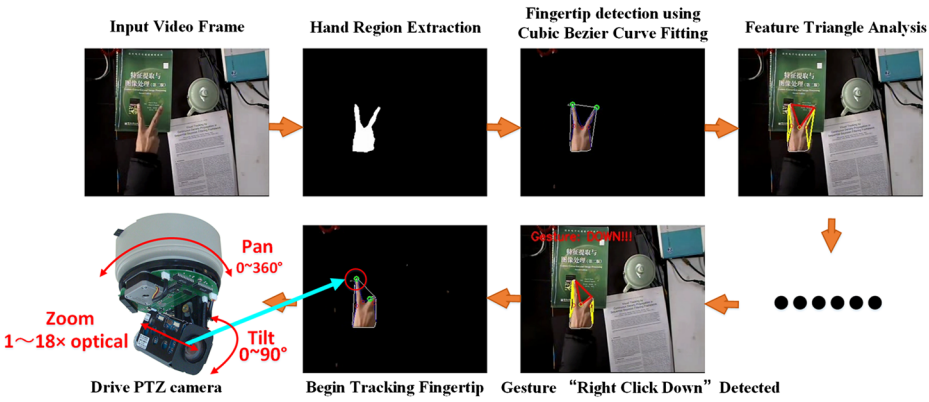

Fig. 1 Overview of the proposed method

and Cr in our method was narrower than that in [6], which was $R_{Cb} = [77, 127]$ and $R_{Cr} = [133, 155]$. It was aimed to minimize the effect of background noise.

After the skin detection and extraction, a morphological filtering process including an erosion operation and a dilation operation was conducted to obtain some more complete candidate regions. Subsequently, the largest region was chosen as the desired hand region (see Fig. 2).

2.2 Fingertip detection based on Monte Carlo sampling Bezier curve fitting

As the first step of gesture recognition, fingertip detection exerted a crucial role in this technology and the previously proposed methods of which were specifically categorized as follows.

Firstly, the fingertip detection was handled by correlation techniques using a circular mask in which the fingertip was modeled as a cylinder with a hemispherical cap [3, 13, 14, 21, 22, 30]. Subsequently, normalized correlation was used with a template of a properly sized circle corresponding to a user's fingertip size. The segmentation of the hand must be very accurate in order to achieve a good detection result under this model and fulfill the requirement of the particular shape of an approximate cylinder. Secondly, the curvature local maximum on the boundary of the silhouette was used as the feature to detect fingertips [2, 15, 20, 26]. The curvature was calculated using the discrete points on the silhouette in all these curvature-based methods, which led to the high sensitivity to noises such as local valleys caused by incompleteness of hand region segmentation. Thirdly, convexity approach was used for finger detection and hand gesture recognition in some latterly literatures, by which convex hull and convexity defects were calculated so as to find the area of hand and finger position as soon as the segmented image was obtained [4, 10]. And then, the vertices of the convex hull were considered as the fingertips, which caused mistakes when the vertices represented other corners of the hand silhouette rather than fingertips. Fourthly, an algorithm based on the distance of the contour points to the hand position was also used in some literatures [11, 17]. In this algorithm, it was assumed that the points with longer distance to the center of the hand were considered as fingertips. However, a hand was deformable and the center of which was probably far away from the expected position if the gesture was not regular, which thus led to an error in fingertip location.

Differed from the existing approaches, an innovative way using Cubic Bezier Curve fitting based on Monte Carlo Sampling was proposed in this study for fingertip detection, by which the robustness to noise and the accuracy of fingertips location were significantly improved.

2.2.1 Bezier curve presenting

After obtaining the contour of the object hand, it was essential to calculate the curvature of every point in the contour as well as locate the local maximums.

Fig. 2 Hand region segmentation Left: input image; Middle: skin color segmentation result; Right: hand contour

The contour points set is $\{CP_m\}\ \ 1 \le m \le M$, where M is the total number of the points of the contour, and a sub contour points set of CP_m including $2L + 1$ points is set as follows:

$$\left\{CP^s_{m+j}\right\}\ \ 1 \le m \le M\ \ -L \le j \le L \tag{2}$$

They are centered in contour point CP_m and consist of L contour points before CP_m as well as L contour points after CP_m (see Fig. 3). A fitted curve to the sub contour points set was discovered and the curvature of the fitted curve at middle point was the estimated curvature of the contour at CP_m. The problem was transformed into seeking for a robust and efficient curve fitting method to the sub contour points set CP^s_{m+j}. In addition, a 'Bezier curve Monte Carlo Sampling and fitting method' with high fitting accuracy as well as computation efficiency was proposed.

Bezier curves were widely used in computer graphics for smooth curve modeling. The curve was completely contained in the convex hull of its control points, thus the points were graphically displayed and used to manipulate the curve intuitively. Affine transformations, such as translation and rotation were applied to the curve by using respective transformation on the control points of the curve.

An nth-degree Bezier curve is represented by the following equation:

$$P(t) = \sum_{i=0}^{n} P_i B_{i,n}(t)\ \ t \in [0, 1] \tag{3}$$

where P_i refers to the control points, n refers to the degree, and t refers to the parameter. The basis function, $B_{i,n}(t)$, is the classical nth-degree Bernstein polynomials defined explicitly by:

$$B_{i,n}(t) = C^i_n t^i (1-t)^{n-i} = \frac{n!}{i!(n-i)!} t^i (1-t)^{n-i} \tag{4}$$

In consideration of the accuracy and calculative efficiency, degree three ($n = 3$), or the Cubic Bezier (see Fig. 4), was adopted in the current study. Namely, four points were required to confirm a Cubic Bezier, in which two were off the curve and called the control points while the other two were on both ends of the curve and called the end points. Giving the sub contour points set CP^s_{m+j}, the curve fitting problem was to find four optimal control points to define

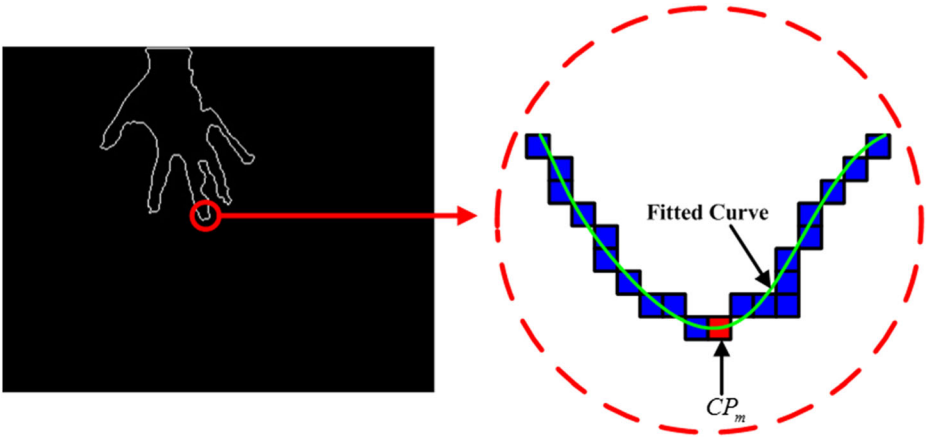

Fig. 3 Sub contour centered at CP_m

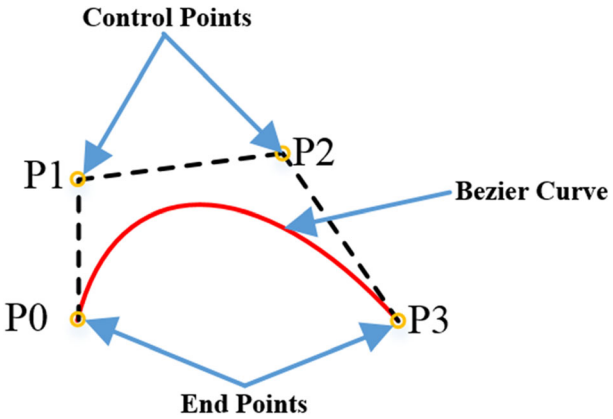

Fig. 4 Cubic Bezier Curve

the fitting curve. An easy way is was to use exhaustive search method through the possible space, yet this brute force method was time-consuming. In this paper, the Monte Carlo method was proposed to find the optimal control points quickly.

2.2.2 Monte Carlo sampling

The Monte Carlo method provides approximate solutions to various mathematical problems by performing statistical sampling experiments on a computer. It applies to problems with absolutely no probabilistic content as well as to those with inherent probabilistic structure. The Cubic Bezier curve fitting problem in this work is formulated [21] as:

$$\min_{x \in X} \{f(x) := E[F(x, \xi)]\} \tag{5}$$

Where $F(x, \xi)$ is a function of two vector variables $x \in \mathbb{R}^n$ and $\xi \in \mathbb{R}^d . X \in \mathbb{R}^n$ is a given set. In this work, it means the sub contour points set CP^s_{m+j}. $\xi \in \xi(\omega)$ is a random vector, which means a candidate sub contour $P(\omega, t)^s_m$, defined by randomly generated control points vector ω. The expectation in Eq. (5) is taken with respect to the probability distribution of ξ which is assumed to be known. The function $F(x, \xi)$ is formulated as:

$$F(x, \xi) = \sum_{j=0}^{2L+1} \left| P\left(\omega, t\right)^s_m - CP^s_{m+j} \right|^2 \tag{6}$$

The objective of the function is to find the best fitted Bezier curve to the given sub contour points set using the least square method. Since ξ has a finite number of possible scenarios with positive probabilities $p_k, k = 1, \ldots K$, the expected value is written as:

$$f(x) = \sum_{k=1}^{K} p_k F(x, \xi_k) \tag{7}$$

However, using exhaustive discretization of the probability distribution lead to an exponential growth of the number of scenarios. Therefore, the Monte Carlo Sampling technique was used to reduce the computational complexity. The proposed approach was accomplished by generating random sequence sub Bezier curve contours ξ^1, ξ^2, ... in the sample space, which were

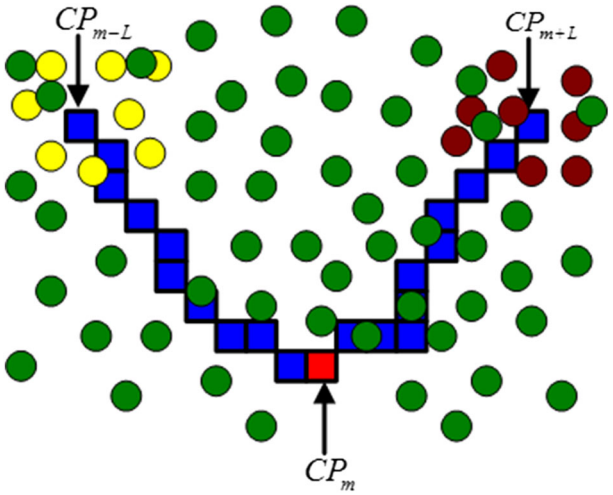

Fig. 5 Distribution of control points Yellow: possible positions of control point P_0; Brown: possible positions of control point P_3; Green: possible positions of control point P_1 and point P_2

independent of each other and uniformly distributed (i.e. independent identically distributed). With the generated N random samples ξ^1, ξ^2, ...ξ^N, the sample's average function is written as:

$$\hat{f}_N(x) \coloneqq \frac{1}{N}\sum_{i=1}^{N} F\left(x, \xi^i\right) \tag{8}$$

Indeed, $\hat{f}_N(x)$ is an unbiased estimator of $f(x)$. Moreover, according to the law of large number, $\hat{f}_N(x)$ is a consistent estimator of $f(x)$ when $N \to \infty$. Subsequently, the Cubic Bezier curve fitting problem in Eq. (5) is approximated:

$$\min_{x \in X}\left\{\hat{f}_N(x) \coloneqq \frac{1}{N}\sum_{i=1}^{N} F\left(x, \xi^i\right)\right\} \tag{9}$$

Specifically, according to the feature of Cubic Bezier curve, the start control point P_0 and the end control point P_3 are on the curve, while P_1 and P_2 are off the curve. Giving the sub contour points set CP^s_{m+j}, the four control points of each candidate sub contour are decided by the

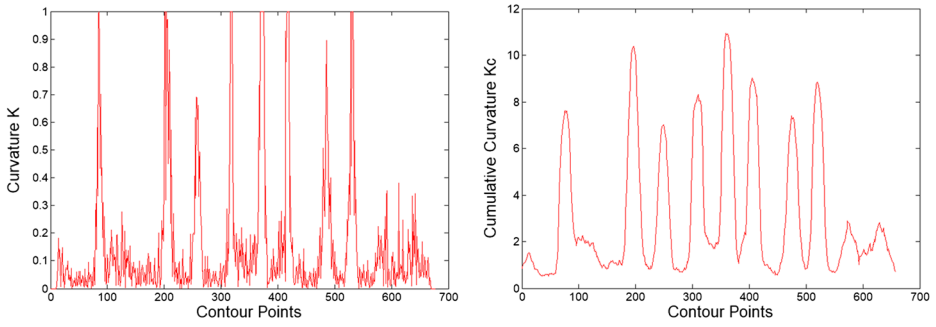

Fig. 6 The curvatures of each contour point of input image shown in Fig. 2. Left: curvature curve; Right: cumulative curvature curve

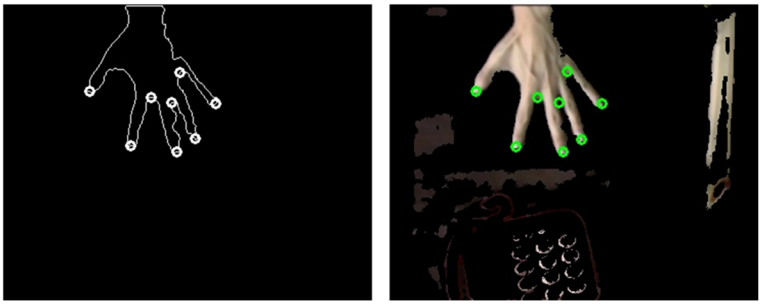

Fig. 7 The detection result of local maximums in cumulative curvature

randomly generated control points vector ω^{ζ}, $\zeta = 1, 2...N$ where N signifies the total random sample number. Consequently, a candidate sub contour using Cubic Bezier curve is expressed as:

$$P_{\zeta}\left(\omega^{\zeta}, t\right)_{m}^{s} = \sum_{i=0}^{3} P_i\left(\omega^{\zeta}\right)_m B_{i,3}(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \omega_{0m}^{\zeta} \\ \omega_{1m}^{\zeta} \\ \omega_{2m}^{\zeta} \\ \omega_{3m}^{\zeta} \end{bmatrix} \tag{10}$$

The control points ω_{0m}^{ζ}, ω_{1m}^{ζ}, ω_{2m}^{ζ} and ω_{3m}^{ζ} are uniformly random variables which can be drawn from the following range independently:

$$\begin{aligned} \omega_{0m}^{\zeta} &\in \left[CP_{m-L}^{s} - T_0, CP_{m-L}^{s} + T_0\right] \\ \omega_{1m}^{\zeta} &\in \left[T_{\min} - T_0, T_{\max} + T_0\right] \\ \omega_{2m}^{\zeta} &\in \left[T_{\min} - T_0, T_{\max} + T_0\right] \\ \omega_{3m}^{\zeta} &\in \left[CP_{m+L}^{s} - T_0, CP_{m+L}^{s} + T_0\right] \end{aligned} \tag{11}$$

where T_0 denotes a consistent threshold which defines a tolerance interval of the corresponding range, and $T_{\min}$, $T_{\max}$ are described as:

$$\begin{aligned} T_{\min} &= \min\left\{CP_{m+j}^{s}\right\} \; -L \le j \le L \\ T_{\max} &= \max\left\{CP_{m+j}^{s}\right\} \; -L \le j \le L \end{aligned} \tag{12}$$

The Eq. (11) means that the start control point and the end control point of a candidate Cubic Bezier curve are supposed to be within an area centered in the corresponding point of the given

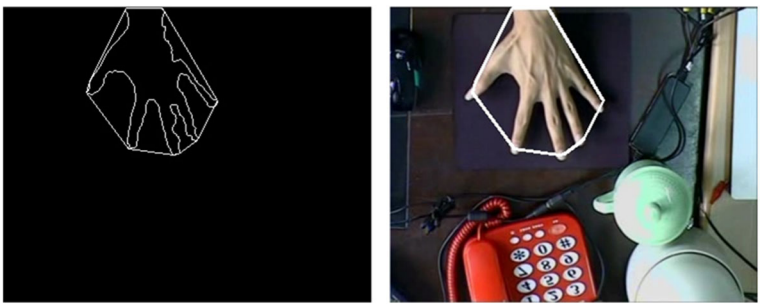

Fig. 8 Detection result of convex hull

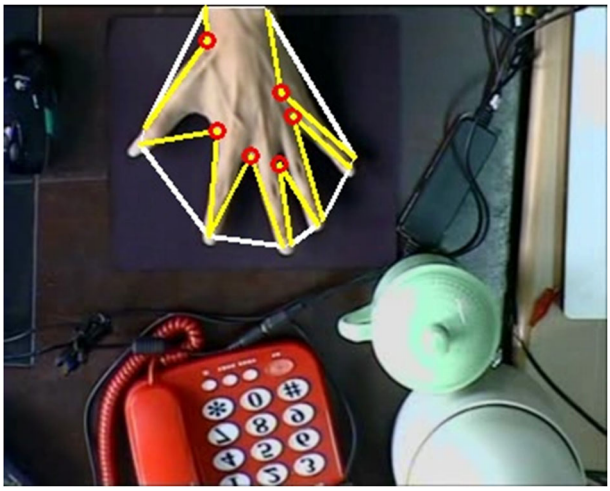

Fig. 9 Detection result of convex defect

sub contour CP^s_{m-j} and CP^s_{m+j}, while the other two control points should possess equal chance to appear in any position of the whole sample space defined by CP^s_{m+j} (see Fig. 5). The start and end control point are not put exactly at the point CP^s_{m-j} and CP^s_{m+j}, since the hand contour extraction is probably so accurate and the position of corresponding contour point possibly contains some uncertainty, which is thus modeled by sampling the points within an area centered in CP^s_{m-j} and CP^s_{m+j}.

2.2.3 Cubic Bezier curvature and local maximum extraction

Using the above-described Monte Carlo Sampling method (i.e. sampling randomly in the sample space N times), a set of control points which fit the given sub contour CP^s_{m+j} best are

Fig. 10 Fingertip detection result

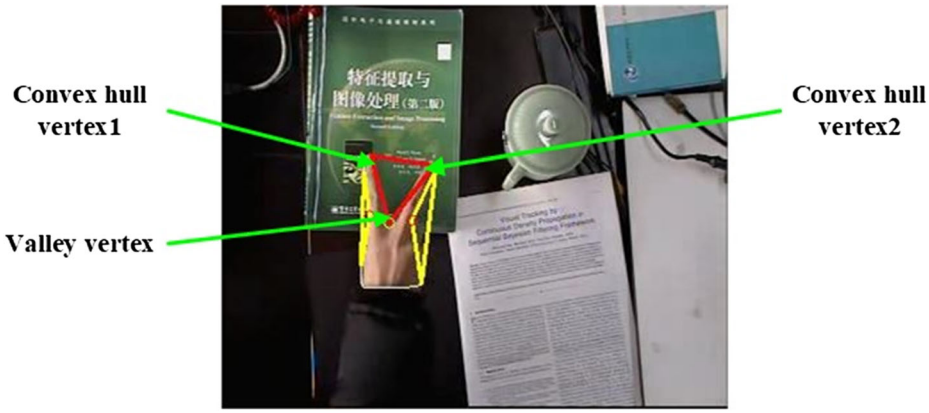

Fig. 11 Feature triangle

discovered as $\{P_i^{best}, i = 0, 1, 2, 3\}$. After that, the curvature at any point in the Bezier curve is derived by the following equation:

$$K(t) = \frac{|P'(t) \times P''(t)|}{|P'(t)|^3} \tag{13}$$

Where the first derivative $P'(t)$ and the second derivative $P''(t)$ of the Bezier curve are represented as follows:

$$P'(t) = \sum_{i=0}^{n} P_i^{best} B_{i,n}'(t) = n \sum_{i=1}^{n} \left(P_i^{best} - P_{i-1}^{best}\right) B_{i-1,n-1}(t) \tag{14}$$

$$P''(t) = n(n-1) \sum_{i=0}^{n-2} \left(P_{i+2}^{best} - 2P_{i+1}^{best} + P_i^{best}\right) B_{i,n-2}(t) \tag{15}$$

Specifically, the first and the second derivative of Cubic Bezier curve are written as:

$$P'(t) = 3\left[\left(P_1^{best} - P_0^{best}\right)(1-t)^2 + \left(P_2^{best} - P_1^{best}\right)t(1-t) + \left(P_3^{best} - P_2^{best}\right)t^2\right] \tag{16}$$

$$P''(t) = 6\left[\left(-P_0^{best} + 3P_1^{best} - 3P_2^{best} + P_3^{best}\right)t + \left(P_0^{best} - 2P_1^{best} + P_2^{best}\right)\right] \tag{17}$$

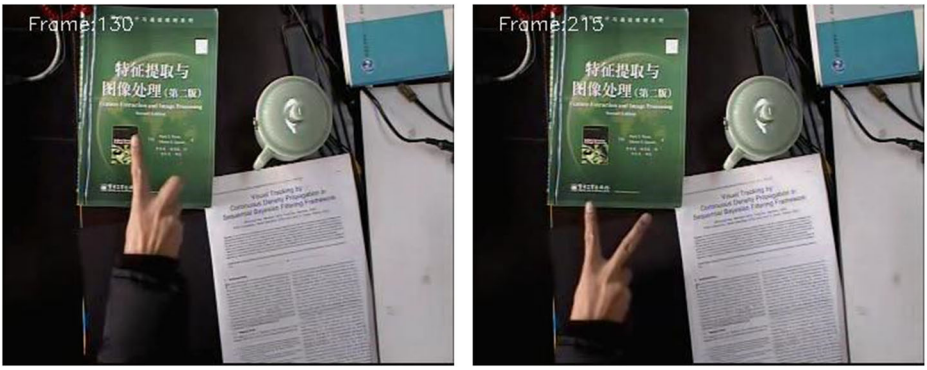

Fig. 12 Gesture "right click down" (left) and Gesture "right click up" (right)

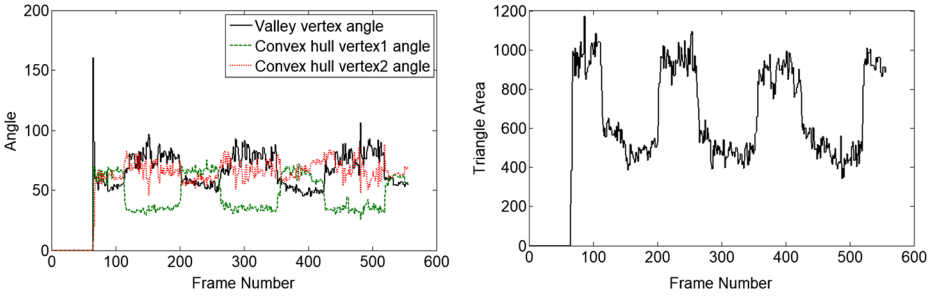

Fig. 13 The three interior angles (left) and the area of the feature triangle (right)

As mentioned above, the given sub contour set is centered in contour point CP_m and consists of L contour points before CP_m as well as L contour points after CP_m. That is to say, the curvature of the point CP_m can be seen as the curvature of the middle point in the best fitted Bezier curve defined by $\left\{P_i^{best}, i = 0, 1, 2, 3\right\}$. Specifically, $K(t = 0.5)$ is assumed to be the curvature that we need.

Since the curvatures of all contour points are calculated, the local maximums with a relatively high probability to be the fingertips are extracted. However, the curvatures obtained by the Monte Carlo Sampling method still contain some noises that make the extraction of local maximums still very difficult (see the left image of Fig. 6). In order to address this problem, a smooth process is introduced to reconstruct a new curvature K^c (also called cumulative curvature), which is described as follows:

$$K_i^c = \sum_{j=i}^{i+2L} K_j \tag{18}$$

The right image of Fig. 6 shows the cumulative curvature of the left one. Obviously, the cumulative curvature curve is much smoother than the original curvature curve, which makes it much easier and more accurate in locating the local maximums. The result of maximums location is shown in Fig. 7.

2.3 Geometry feature analysis

The application of curvature detection alone was not enough to locate the fingertips needed. As shown by Fig. 7, the valley points between two fingers were also local maximums. Consequently, a geometry feature analysis based on convex hull analysis and convex defect detection was proposed in order to overcome this difficult point.

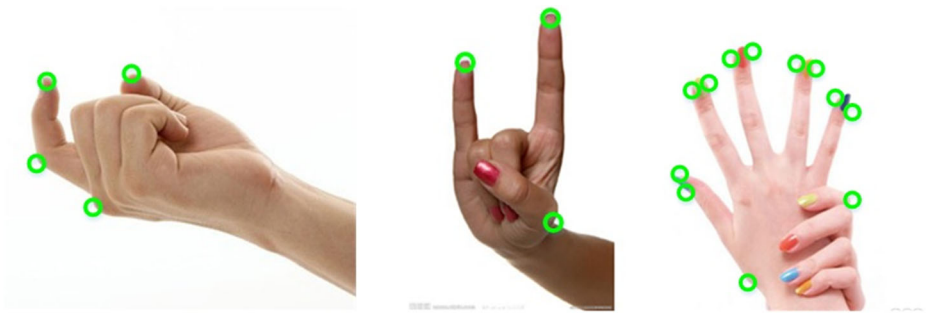

Fig. 14 Fingertip detection result of traditional curvature method

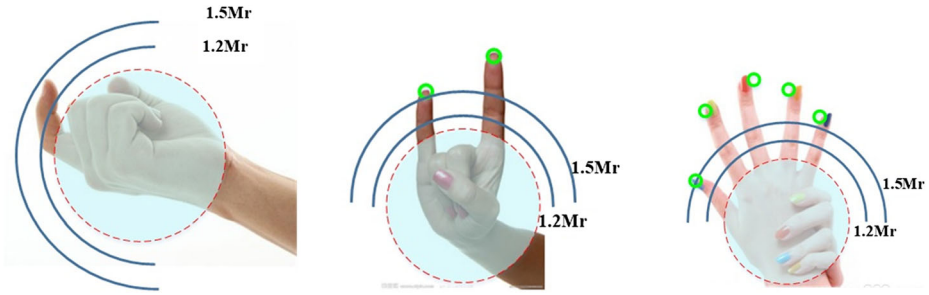

Fig. 15 Fingertip detection result of centroid circle method

Giving a set containing all contour points, the convex hull of the set was detected by Graham scan method [1] (see Fig. 8).

After the detection of convex hull, the valley points between fingers were located by defect detection between two convex hull vertices (see Fig. 9):

The results of curvature detection in Fig. 7 and convex defects detection in Fig. 9 were combined, obviously revealing that the points which possessed high curvature and did not belong to the valleys between fingers were finally considered as the fingertips (see Fig. 10). Beyond that, the locations with green circles alone were fingertips, while those with both green and white circles were the ones that not only possessed high curvature but also belonged to valleys between fingers.

2.4 Feature triangle analysis

Every two fingertip vertices and the valley vertex between them formed a triangle (as seen the red triangle in Fig. 11), called feature triangle. There were several advantages if the feature triangle was used for gesture recognition and hand tracking, including scale-invariance, rotation-invariance, convenience for dynamic analysis, and facility in gesture modeling.

The gesture “right click down” (as seen in the left one of Fig. 12) and “right click up” (as seen in the right one of Fig. 12) were modeled easily using feature triangle analysis. After analyzing a video (556 frames) including several times of gesture “right click down” and “right click up”, the three interior angles and the area of the feature triangle were recorded as seen in Fig. 13.

It is found in Fig. 13 that the internal angles and the area of the feature triangle changed significantly when the gesture changed between “right click up” and “right click up”. By performing appropriate statistical analysis on the area and interior angle of the target feature

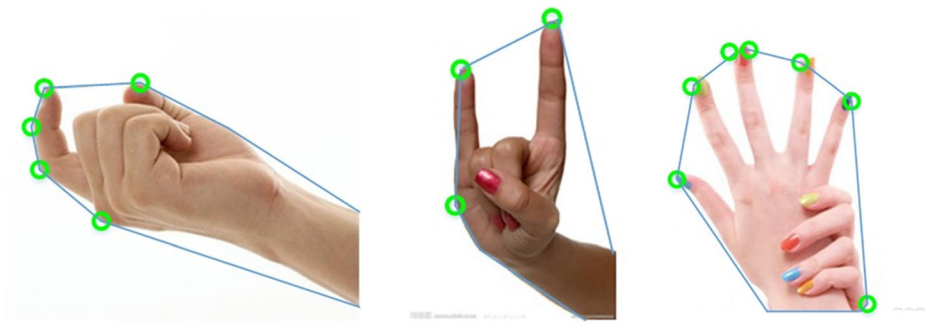

Fig. 16 Fingertip detection result of convex hull and defect analysis method

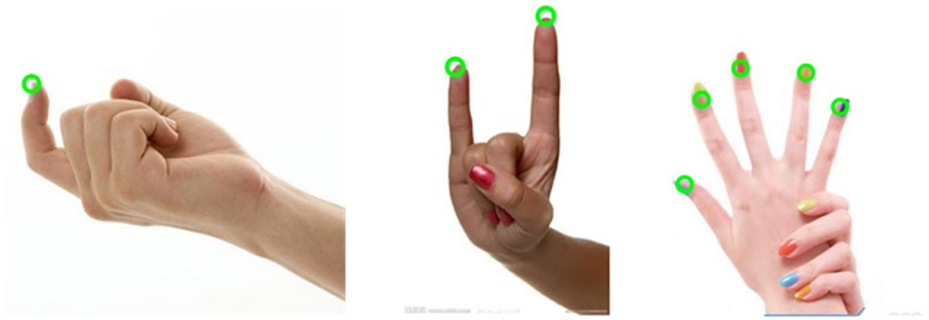

Fig. 17 Fingertip detection result of the proposed method

triangle in n consecutive frames, the switching between "right click up" and "right click down" was judged and recognized in real-time and accurately. Suppose $\{TS_i, i = t - N_h, t - N_h + 1 \cdots, t\}$ represents the triangle area of the frame i from $t - N$ to t, and $\{T\alpha_i\}$, $\{T\beta_i\}$, $\{T\gamma_i\}$ represent the three internal angles. It is shown in Fig. 13 that vertex 1 ($T\alpha_i$) exhibited the most obvious angle change, thus which was exclusively considered in our model. u_t^{s1}, u_t^{s2} represent the mean of first m_h frames and last frames, and u_t^{s1}, u_t^{s2}, $u_t^{\alpha 1}$, $u_t^{\alpha 2}$ represent the mean area and angle of first m_h frames and last frames, while var_t^{s1}, var_t^{s2}, $\mathrm{var}_t^{\alpha 1}$, $\mathrm{var}_t^{\alpha 2}$ represent the variances.

$$u_t^{s1} = \frac{1}{m_h} \sum_{i=t-N_h}^{t-N_h+m_h} Ts_i \quad u_t^{\alpha 1} = \frac{1}{m_h} \sum_{i=t-N_h}^{t-N_h+m_h} T\alpha_i \tag{19}$$

$$u_t^{s2} = \frac{1}{m_h} \sum_{i=t-m_h}^{t} Ts_i \quad u_t^{\alpha 2} = \frac{1}{m_h} \sum_{i=t-m_h}^{t} T\alpha_i \tag{20}$$

$$\mathrm{var}_t^{s1} = \sum_{i=t-N_h}^{t-N_h+m_h} \left(Ts_i - u_t^{s1}\right)^2 \quad \mathrm{var}_t^{\alpha 1} = \sum_{i=t-N_h}^{t-N_h+m_h} \left(T\alpha_i - \alpha_t^{s1}\right)^2 \tag{21}$$

$$\mathrm{var}_t^{s2} = \sum_{i=t-m_h}^{t} \left(Ts_i - u_t^{s2}\right)^2 \quad \mathrm{var}_t^{\alpha 2} = \sum_{i=t-m_h}^{t} \left(T\alpha_i - u_t^{\alpha 2}\right)^2 \tag{22}$$

If both the mean and variance meet the formulas (23)-(26), the occurrence of a switching from gesture "right click up" to "right click down" is considered, while if all the conditions meet the formulas (27)-(30), the occurrence of a switching from gesture "right click down" to "right click up" is considered.

$$u_t^{s1} / u_t^{s2} \geq T_u^s \tag{23}$$

$$u_t^{\alpha 1} / u_t^{\alpha 2} \geq T_u^{\alpha} \tag{24}$$

Table 1 Experimental Result of Fingertip detection using different methods

Method	Average Success Rate	Average Time Cost
Traditional Curvature [27]	60.8%	14.75 ms
Centroid Circle [16]	75.0%	18.12 ms
Convex Hull and Defect Analysis [9]	76.7%	22.43 ms
The Proposed Method	79.1%	28.33 ms

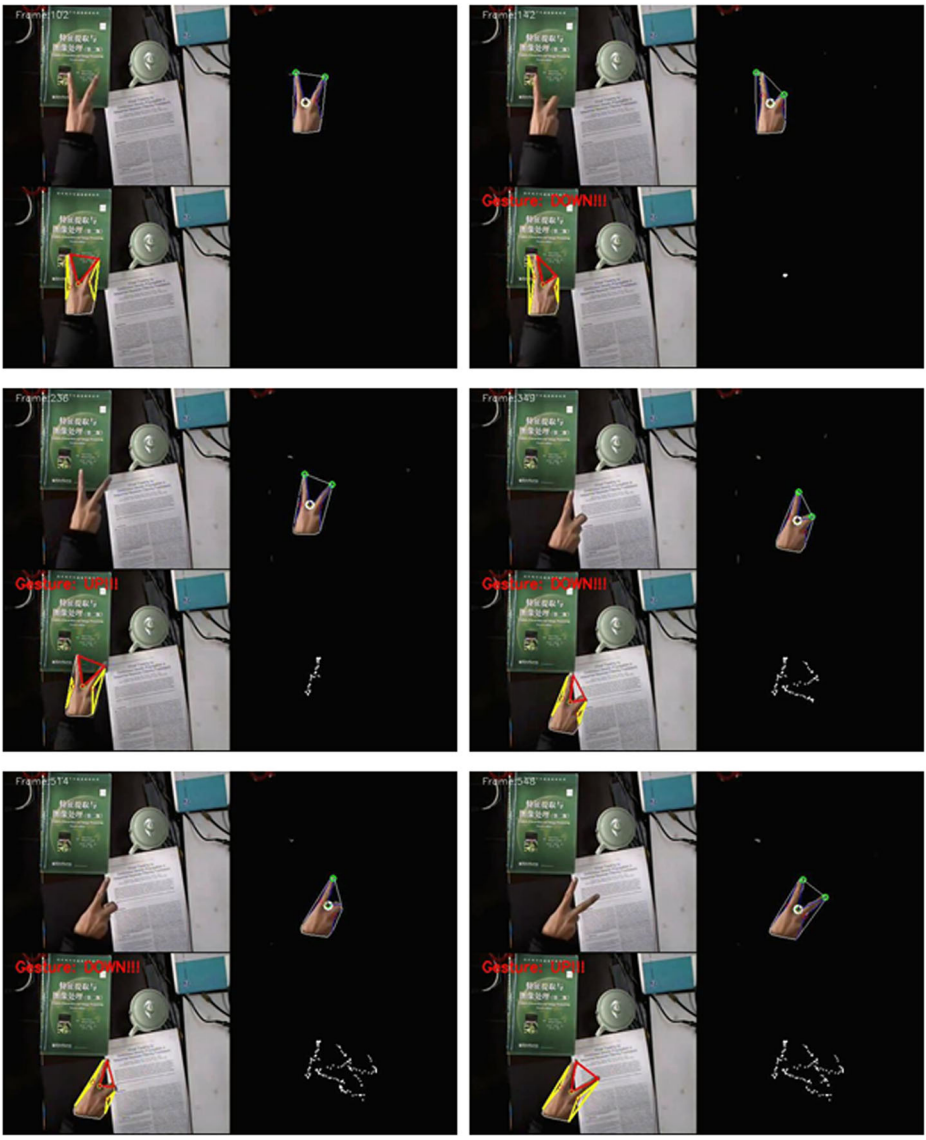

Fig. 18 The tracking results using a still camera

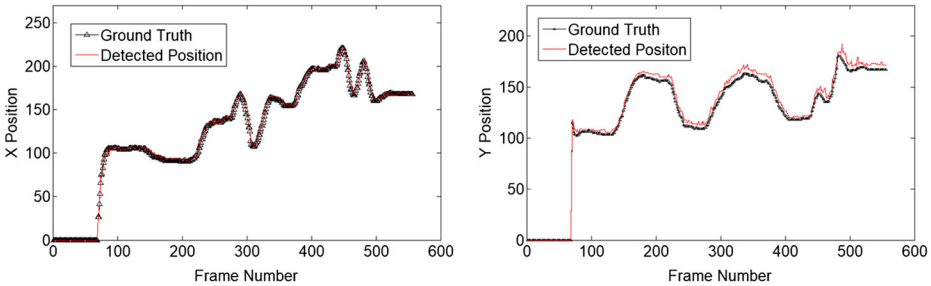

Fig. 19 Fingertip position tracking results, Black: ground truth, Red: detected position

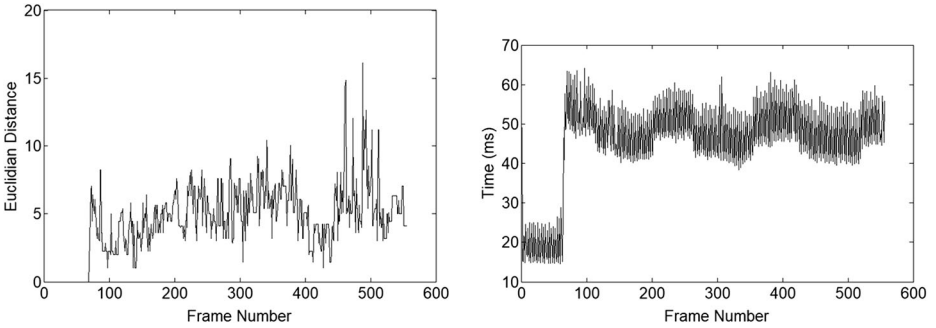

Fig. 20 Fingertip position error and algorithm time

$$T_{\mathrm{var}}^{s1} \le \mathrm{var}_t^{s1} / \mathrm{var}_t^{s2} \le T_{\mathrm{var}}^{s2} \tag{25}$$

$$T_{\mathrm{var}}^{\alpha 1} \le \mathrm{var}_t^{\alpha 1} / \mathrm{var}_t^{\alpha 2} \le T_{\mathrm{var}}^{\alpha 2} \tag{26}$$

$$u_t^{s2} / u_t^{s1} \ge T_u^s \tag{27}$$

$$u_t^{\alpha 2} / u_t^{\alpha 1} \ge T_u^{\alpha} \tag{28}$$

$$T_{\mathrm{var}}^{s1} \le \mathrm{var}_t^{s1} / \mathrm{var}_t^{s2} \le T_{\mathrm{var}}^{s2} \tag{29}$$

$$T_{\mathrm{var}}^{\alpha 1} \le \mathrm{var}_t^{\alpha 1} / \mathrm{var}_t^{\alpha 2} \le T_{\mathrm{var}}^{\alpha 2} \tag{30}$$

3 Experiments and discussions

To demonstrate the effectiveness and robustness of the proposed method, two different kind of experiment conditions were tested. In the first set of experiments, a still camera (SONY CCD named EX-FCB48) was used to test the detecting and tracking accuracy and robustness. In the

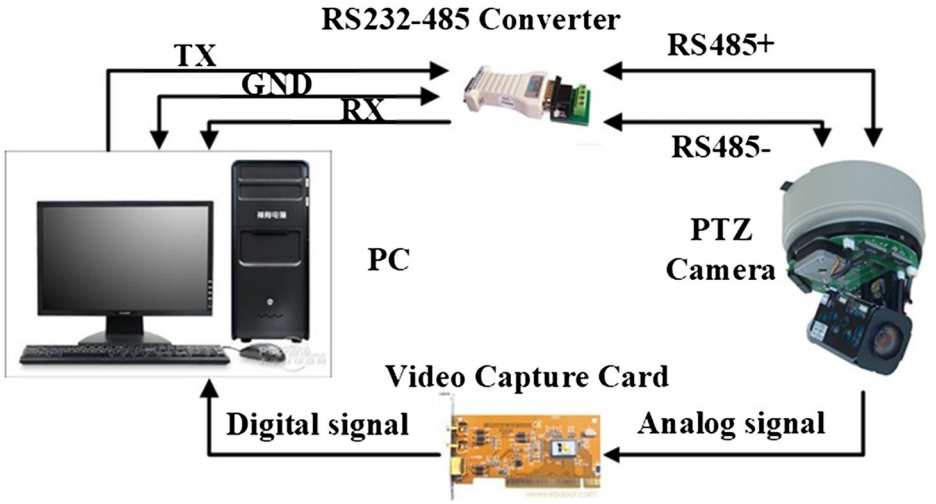

Fig. 21 The hardware of the PTZ camera servo control system

Table 2 Parameters used in the experiments

N	N_h	m	T_u^s	T_u^α	T_{var}^{s1}	T_{var}^{s2}	$T_{\text{var}}^{\alpha 1}$	$T_{\text{var}}^{\alpha 2}$
15	40	10	1.4	1.4	0.5	1.5	0.5	1.5

other set of experiments, a Pan-Tilt-Zoom (PTZ) camera was used for gesture recognition and hand servo tracking in real-time.

The proposed method was implemented in C++ using the OpenCV library and ran on a 1.8 GHz Pentium Dual-Core CPU, 2Gbyte DDR memory. In the experiments, the point number of a sub contour point set was set to be 21, namely $L = 10$, and the sample number $N = 20$.

3.1 Fingertip detection experiments

To demonstrate the effectiveness and robustness of the proposed fingertip detection method, 120 different hand images which included different amounts of visible fingertips, hand gestures, skin colors and races were used in this experiment. The proposed method also was compared with other three commonly used methods, including traditional curvature method [27], the centroid circle method [16], the convex hull and defect analysis method [9].

Qualitative Comparison: As shown in Figs. 14, 15, 16 and 17, traditional curvature method was sensitive to the hand segment noise, and prone to false detection at the local minimum; the centroid circle method was better than traditional curvature method yet prone to miss detection, and finger was bent obviously; the convex hull and defect analysis method was prone to false detection at some convex hull vertex; while the proposed method was better than the above three commonly used fingertip detection methods.

Quantitative Comparison: The percentage listed in Table 1 represented the success rate of fingertip detection, which was defined as the number of correctly detected fingertip images divided by the total number of the fingertip images used in the experiments. It was shown that the proposed method was better than the other three methods. The table also presented the time cost of every method, which showed that the proposed method consumed more time than the other three methods, since the Monte Carlo Sampling was a relatively time-consuming method. Moreover, the time-consuming extent increased with the sampling number, which was a disadvantage of the proposed method yet negligible on use of a better computer.

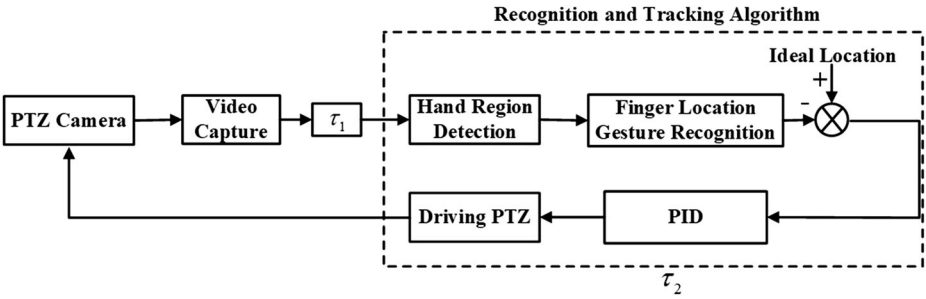

Fig. 22 PTZ camera servo control system

3.2 Detecting and tracking using still camera

The experimental results are shown in Fig. 18. Each frame was divided into four views, the upper left corner was the original input video, the upper right corner was the fingertip detection result, the lower left corner was the feature triangle extraction result, and the lower right corner was the result of simulated handwriting. In experimental result, the red triangle was the target feature triangle, the green dot was the fingertip point, and the green and white dots were the inter-finger points.

From the first frame to the 102nd frame was a process in which the hand extended from the outside of the camera into the middle of the camera, thus all this process was in a non-writing

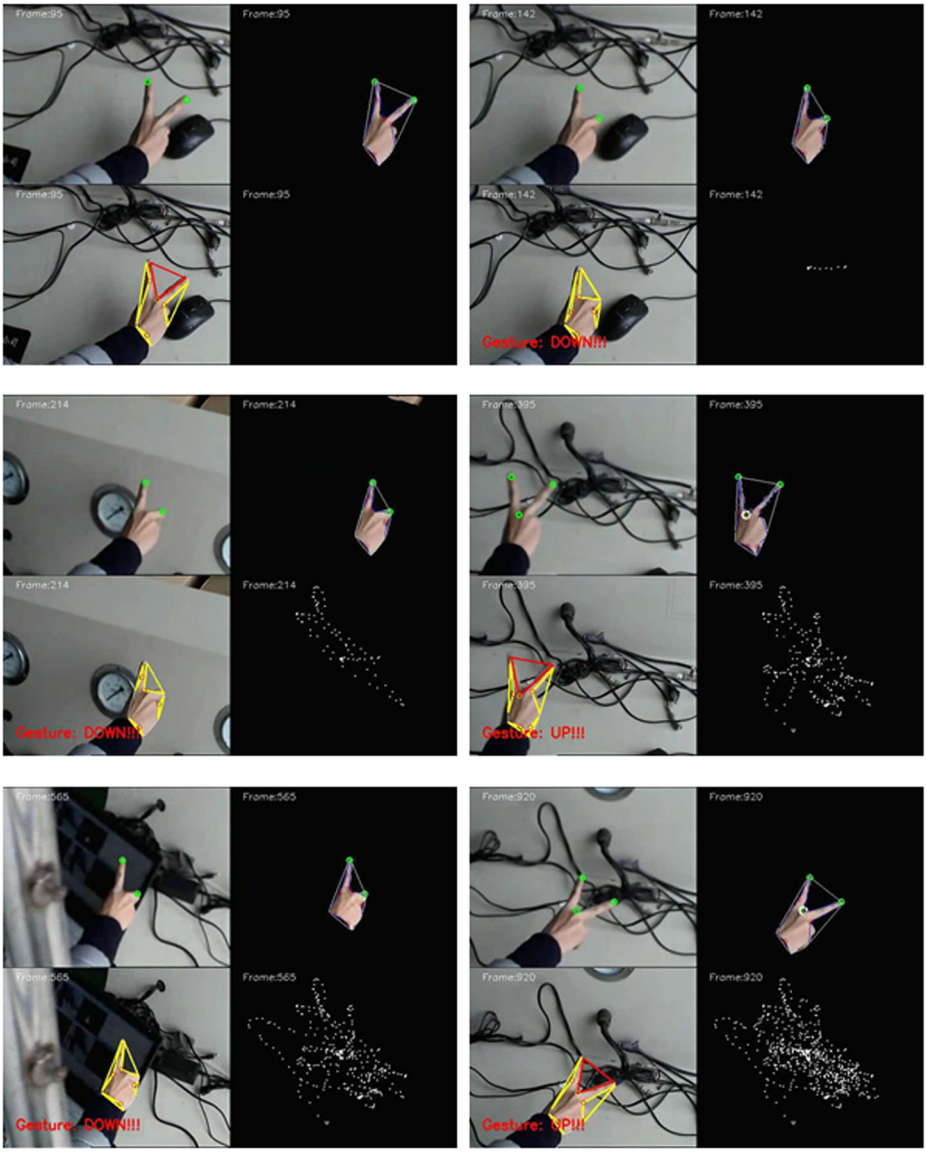

Fig. 23 The tracking results using a PTZ camera

(right click up) state. From the 102nd frame to the 142st frame a blending process of the right finger simulated the action of "right click down", and the state was converted to the handwriting state in the 142th frame when the corresponding lower right corner view drew the current fingertip position. From the 142th frame to the 236th frame, the action of "right click up" was detected. In the handwriting state, the fingertip positions were recorded and plotted in the lower right corner, while in the non-handwriting state, the fingertips positions were ignored. In the process of writing three numbers, a total of six state switching processes occurred, and all of which were detected accurately and timely by the proposed algorithm.

It was revealed by the experimental results that the numbers written down were not very continuous and smooth, since the lower right corner view of the experiment merely recorded the fingertips in each frame, which were just a series of discrete points, and more detailed explanations were shown in the literatures [32–35]. Meanwhile, the distances between points were not identical, since the moving speed of the fingertip was not uniform although the time interval between each frame was constant 25 milliseconds. Therefore the faster the writing speed, the larger the interval between two points while the slower the writing speed, the smaller the interval and sometimes the two points even overlapped if the speed was slow enough. In this case, if the adjacent two points were connected by a straight line, the written words were possibly continuous, and if some smoothing algorithms are further adopted, the hand-writing will be smoother and more continuous.

The tracked fingertip positions were compared with the ground truth as shown in Fig. 19, where the ground truth positions were recorded manually frame by frame. The Euclidian distance between the tracked fingertip and the ground truth is shown in the left one of Fig. 20. The results showed that the tracked fingertip positions were accurate, and the average position error only possessed 5.48 pixels. The average algorithm time was 48.45 ms (as seen in right one of Fig. 20), and accorded with the requirement of real-time applications.

3.3 Detecting and tracking using PTZ camera

The hardware of PTZ camera servo control system was composed of a PTZ camera, a video capture card, a PC, and a RS232-485 converter, as seen in Fig. 21. The PTZ camera used in this system was designed and developed by ourselves. It possessed three degrees of freedom, including Pan (0~360°), Tilt (0~90°), Zoom (1-18x optical).

The servo control model is shown in Fig. 22. The objective of the servo control system was to keep the target in the middle field of camera through driving the PTZ camera in every frame. In this control model, there were two system delay, τ_1 and τ_2. τ_1 was the video system time, and

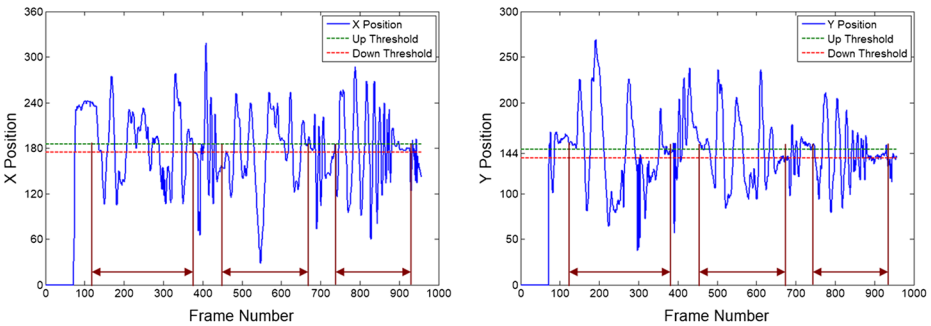

Fig. 24 The horizontal and vertical coordinates of the tracked fingertip

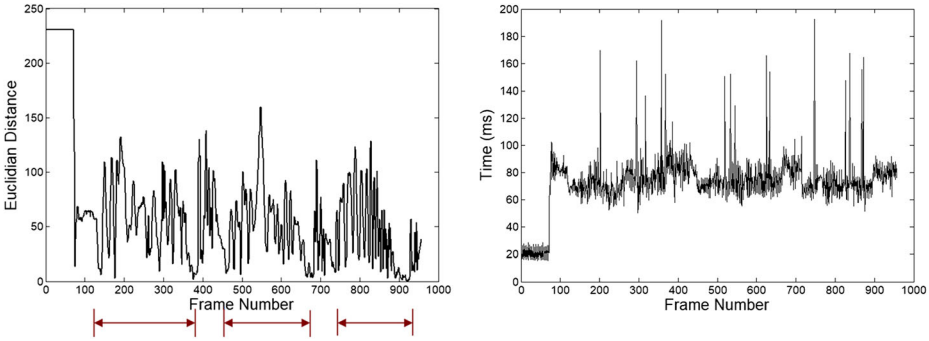

Fig. 25 The Euclidian distance between the tracked fingertip and the image center (left); The algorithm time (right)

in this system τ_1 was 40 ms (25fps) due to the usage of a PAL CCD. τ_2 was the algorithm time and the value of which less than τ_1 made it full real-time. If τ_2 was between 40 and 100 ms, namely reaching more than 10 frames per second, the model was still able to achieve good results in practical applications (Table 2).

Figure 23 is the tracking results using a PTZ camera. From the first frame to the 95th frame was a process in which the hand extended from the outside into the middle of the camera, thus all this process was in a non-tracking state. In the 142th frame, a gesture state switching from "right click up" to "right click down" was detected, and the servo tracking system began to drive the PTZ camera to keep the fingertip in the middle of the camera, as is shown in the 214th frame. While a gesture state switching from "right click down" to "right click up" was detected, as is shown in 395th frame, the servo control system stopped tracking the fingertip and waited for the next "right click down" action to be detected. In the experiment, a total of three state transitions were experienced, and the algorithm made a correct judgment on its state switching, while the hand and fingertips always remained in the middle of the image while tracking.

Figure 24 shows the tracking result of the experiment. The blue line shows the horizontal and vertical coordinates of the tracked fingertips, while the green and red dashed line represent the ideal position range. The camera resolution was 360*288, the ideal horizontal position range was 175~185, and the ideal vertical position range was 139~149. The result shows that during the whole tracking process, the hand and the fingertips always kept in the middle of the camera.

Figure 25 shows the algorithm time. It is revealed that the time in most frames was between 60 ms and 80 ms, while in some frames it was much more than the average time, since the time consuming of fingertip detection using cubic Bezier curve depended on the hand region segments, that is, the more points in the hand contour, the more time consumed.

4 Conclusion

This paper presented a real-time dynamic gesture recognition and hand tracking using a PTZ camera. It was aimed to achieve a robust scheme that stably recognized simple hand gestures and tracked the hand using a PTZ camera to keep the fingertip always in the middle field of the camera. Firstly, a new approach was proposed to detect fingertip by estimating the curvature of hand contour points based on the Monte Carlo Sampling Bezier curve fitting, which was much more robust than existing curvature-based methods against noise. Secondly, a feature triangle analysis method was utilized to

recognize the simple dynamic gesture in real-time. Thirdly, the proposed fingertip detection and feature triangle analysis were applied to a self-made PTZ camera to realize servo tracking of the target fingertip when the gesture "right click down" was detected. As shown by the experimental results, the proposed approach successfully recognized the dynamic gesture and located the fingertips' position precisely, and realized follow-up servo tracking using PTZ camera in real-time.

Moreover, the curvature estimation approach using Cubic Bezier Curve fitting based on Monte Carlo Sampling can be easily extended to other areas (e.g. industrial visual inspection) where the curvature needs to be estimated accurately and quickly.

Authors' contributions Songxiao Cao implemented the core algorithm, designed all the experiments, addressed the resulting data and drafted the manuscript. Xuanyin Wang participated in the design and construction of the system and helped draft the manuscript. All authors have read and approved the final manuscript.

References

1. Anderson KR (1978) A reevaluation of an efficient algorithm for determining the convex hull of a finite planar set. Inf Process Lett 7(1):53–55
2. Argyros AA, Lourakis MIA (2006) Vision-based interpretation of hand gestures for remote control of a computer mouse. Lect Notes Comput Sci 3979:40–51
3. Barrho J, Adam M, Kiencke U (2006) Finger localization and classification in images based on generalized hough transform and probabilistic models. Int Conf Control Autom Robot Vision IEEE 1–6
4. Barros P et al (2017) A dynamic gesture recognition and prediction system using the convexity approach. Comput Vis Image Understanding 155:139–149
5. Bhuyan MK, Neog DR, Kar MK (2011) Hand pose recognition using geometric features. Commun IEEE 1-5
6. Chai D, Ngan KN (1998) Locating facial region of a head-and-shoulders color image. IEEE Int Conf Automatic Face Gesture Recognition 1998. Proc IEEE 124–129
7. Erol A et al (2007) Vision-based hand pose estimation: a review. Comput Vis Image Underst 108(1):52–73
8. Ge SS, Yang Y, Lee TH (2008) Hand gesture recognition and tracking based on distributed locally linear embedding. Image Vis Comput 26.12:1607–1620
9. Gurav RM, Kadbe PK (2015) Real time finger tracking and contour detection for gesture recognition using Open CV. Proc IEEE Int Conf Ind Instrum Control 974–977
10. Hartanto R, Kartikasari A (2017) Android based real-time static Indonesian sign language recognition system prototype. Int Conf Inf Technol Electr Eng IEEE 1–6
11. Jo KH, Kuno Y, Shirai Y (1998) Manipulative hand gesture recognition using task knowledge for human computer interaction. Int Conf Face Gesture Recogn IEEE Comput Soc 468
12. Kim J et al (2017) An adaptive local binary pattern for 3D hand tracking. Pattern Recogn 61:139–152
13. Koike H, Sato Y, Kobayashi Y (2001) Integrating paper and digital information on enhanced desk: a method for realtime finger tracking on an augmented desk system. ACM Trans Comput Hum Interact (TOCHI) 8(4):307–322
14. Letessier J (2004) Visual tracking of bare fingers for interactive surfaces. ACM Symp User Interface Software Technol ACM 119–122
15. Malik S, Laszlo J (2004) Visual touchpad: a two-handed gestural input device. 289–296
16. Malima A, Qzgur E, Cetin M (2006) A fast algorithm for vision-based hand gesture recognition for robot control. Proc 14th IEEE Conf Signal Process Commum Appl 1–4
17. Mo Z, Lewis JP, Neumann U (2005) Smart Canvas:a gesture-driven intelligent drawing desk system. Int Conf Intell User Interf. January 10-13, 2005, San Diego, California, Usa DBLP 239–243
18. Morshidi M, Tjahjadi T (2014) Gravity optimised particle filter for hand tracking. Pattern Recogn 47(1): 194–207
19. Núñez JC, Cabido R, Pantrigo JJ, Montemayor AS, Vélez JF (2018) Convolutional neural networks and long short-term memory for skeleton-based human activity and hand gesture recognition. Pattern Recogn 76:80–94
20. O'Hagan R, Zelinsky A (2000) Visual gesture interfaces for virtual environments. User Interface Conference, 2000. AUIC 2000. First Australasian IEEE 73–80
21. Oka K, Sato Y, Koike H (2002) Real-time fingertip tracking and gesture recognition. Comput Graphics Appl IEEE 22(6):64–71

22. Oka K, Sato Y, Koike H (2002) Real-time tracking of multiple fingertips and gesture recognition for augmented desk interface systems. 情報処理学会論文誌コンピュ タビジョンとイメジメディア(cvim) 44.568:25–32
23. Pavlovic VI, Sharma R, Huang TS (1997) Visual interpretation of hand gestures for human-computer interaction: a review. IEEE Trans Pattern Anal Mach Intell 19(7):677–695
24. Premaratne P, Yang S, Vial P, Ifthikar Z (2017) Centroid tracking based dynamic hand gesture recognition using discrete hidden Markov models. Neurocomputing 228:79–83
25. Sagayam KM, Hemanth DJ (2016) Hand posture and gesture recognition techniques for virtual reality applications: a survey. Virtual Reality 21(2):1–17
26. Segen J, Kumar S (1998) Gesture VR: vision-based 3D hand interace for spatial interaction. ACM International Conference on Multimedia '98, Bristol, England, September DBLP 455–464
27. Segen J, Kumar S (1998) Human-computer interactions using gesture recognition and 3D hand tracking. Proc IEEE Int Conf Image Process. 188–192
28. Suau X et al (2014) Real-time fingertip localization conditioned on hand gesture classification. Image Vis Comput 32.8:522–532
29. Tsironi E et al (2017) An analysis of convolutional long-short term memory recurrent neural networks for gesture recognition. Neurocomputing 268
30. Hardenberg C Von (2001) Bare-hand human-computer interaction. The Workshop on Perceptive User Interfaces ACM 1–8
31. Wu X et al (2015) Depth image-based hand tracking in complex scene. Optik Int J Light Electron Opt 126.20:2757–2763
32. Yan C, Xie H, Chen J, Zha Z, Hao X, Zhang Y, Dai Q (2018) A fast Uyghur text detector for complex background images. IEEE Trans Multimed 20(12):3389–3398
33. Yan C, Li L, Zhang C, Liu B, Zhang Y, Dai Q (2019) Cross-modality bridging and knowledge transferring for image understanding. IEEE Trans Multimed 1–1
34. Yan C, Li Z, Zhang Y, Qin P, Ji X, Dai Q (2019) Depth image denoising using nuclear norm and learning graph model. IEEE Trans Multimed
35. Yan C, Tu Y, Wang X, Zhang Y, Hao X, Zhang Y, Dai Q (2019) STAT: spatial-temporal attention mechanism for video captioning. IEEE Trans Multimed
36. Zhang X et al (2015) Robust hand tracking via novel multi-cue integration. Neurocomputing 157:296–305
37. Zhou Y, Jiang G, Lin Y (2015) A novel finger and hand pose estimation technique for real-time hand gesture recognition. Pattern Recogn 49.C:102–114

Songxiao Cao received the B.S degree in Department of Mechanical Engineering of Zhejiang University, Hangzhou, China, in 2007, the PhD degree in Department of Mechanical Engineering of Zhejiang University, Hangzhou, China, in 2013. Now, he is a lecturer at College of Metrology and Measurement Engineering of China Jiliang University, Hangzhou, China. His research interests including image processing, visual object tracking and multimedia systems.

Xuanyin Wang received the PhD degree in Department of Mechanical Engineering of Harbin Institute of Technology, Harbin, China, in 1995. Now he is a professor at State Key Laboratory of Fluid Power and Mechatronic Systems, Zhejiang University, China. His research interests including image processing, robot control and mechatronic control.