Research on the Improvement of Tool Detection Methods Based on Machine Vision

Chaofan Wang, Junwei Tian

School of Mechanical and Electrical Engineering, Xi'an Technological University, Xi'an, Shaanxi, China

Abstract: To address the dual bottlenecks of accuracy and efficiency in traditional tool detection methods, this paper focuses on the research on the improvement of machine vision-based tool detection methods. It aims to overcome the deficiencies of traditional detection methods in accuracy and efficiency, and establish a highly efficient, precise, and automated system for tool parameter detection. The research centers on the optimization of the image edge rough extraction algorithm and the innovation of the sub-pixel edge detection algorithm, and systematically proposes a dynamic weight adaptive sub-pixel edge detection algorithm. Finally, through the methods of feature point recognition and positioning as well as contour fitting, the on-line measurement of tool parameters is realized. According to the experimental results, the system features a high degree of detection automation and fast operation speed, with the measurement accuracy reaching the micron level, which can be effectively applied to the real-time measurement of the geometric parameters of machining tools in industry.

Keywords: Tool Detection; Rough Edge Extraction; Sub-Pixel Edge Detection; Dynamic Weight Adaptation; Online Measurement.

1. Introduction

With the in-depth advancement of the technology, China's manufacturing industry is accelerating its transformation towards high precision, high efficiency, and intelligence. Numerical control processing technology, as a core support in key fields such as aerospace, high-end equipment, and precision instruments, has achieved large-scale application. As a core functional component in NC processing systems, the geometric parameter accuracy of cutting tools directly determines the machining accuracy of work pieces and production cycles, and also plays a decisive role in controlling production costs and improving the qualification rate of work pieces. In high-precision machining, the requirements for the geometric parameters of cutting tools are extremely strict.

In precision machining scenarios, even a deviation of a few microns in tool parameters can lead to the scrapping of an entire batch of work pieces, resulting in significant economic losses. Therefore, how to achieve efficient, accurate, and intelligent detection of tool conditions, thereby reducing the scrap rate and improving production efficiency, has become a key technical bottleneck that urgently needs to be broken through in the current manufacturing field.

Traditional tool inspection methods mostly rely on manual visual inspection or contact measurement tools such as micrometers and gauges. These methods not only have low inspection efficiency and cannot meet the demands of large-scale production, but also are highly subjective, have poor repeatability, and are easily affected by human factors such as the inspector's experience and fatigue. More importantly, contact measurement may cause minor damage to the cutting edge of the tool, and the offline inspection mode cannot provide real-time feedback on the dynamic parameter status of the tool during the processing, making it difficult to issue timely warnings of potential risks. This severely restricts the continuity and stability of the production line. In contrast, the tool inspection technology based on machine vision, with its significant advantages of non-contact, high speed, high precision, and automation, can achieve real-time monitoring of tool status and batch automated inspection, effectively reducing errors caused by human intervention. It provides key support for the automation and intelligence upgrade of production lines and has become the core technical path to solve the pain points of traditional inspection methods.

For the recognition and positioning technology of cutting tools, scholars at home and abroad have conducted a large number of forward-looking studies: In 2006, Wang [1] proposed a sub-pixel edge extraction algorithm based on moments to address the issue of image noise interference on the edge of the cutting tool, effectively enhancing the robustness and stability of edge extraction; KASSIM et al. [2] tackled the challenge of parameter measurement for complex- structured cutting tools by applying image radial segmentation technology, processing the cutting tool images in different regions, and further improving the accuracy of cutting tool parameter detection. This method was successfully applied to the identification of cutting tool types and the assessment of wear degrees; domestic scholars Guan [3] and Zhang [4] focused on offline measurement scenarios and developed an image-based geometric parameter measurement system for cutting tools based on machine vision. By optimizing the image preprocessing algorithm and edge extraction strategy, they achieved measurement accuracy at the micrometer level, providing important technical references and engineering practical experience for related domestic research.

However, as numerical control machining continues to evolve towards higher speeds, greater precision, and increased flexibility, the existing image recognition and positioning algorithms have gradually revealed significant limitations. To address this issue, this paper proposes a high-precision measurement method for tool geometric parameters based on machine vision: under the collaborative effect of a high-resolution camera and a high-brightness LED coaxial light source, high-quality tool images without shadows and low reflection are captured; camera calibration is completed through the calibration method, and image denoising and detail enhancement are achieved by combining grayscale processing and image guided filtering technology, laying the foundation for subsequent edge extraction; further, a dynamic weight adaptive fusion sub-pixel edge detection algorithm is innovatively proposed. This algorithm adaptively adjusts the fusion weights according to the edge feature differences in different regions of the tool, and combines the edge detection advantages of the Canny operator to accurately extract the edge contour curve of the tool. This method aims to significantly reduce edge detection errors, greatly improve the accuracy of tool target recognition and positioning, and provide a practical technical solution for solving the key problem of tool condition detection in the current manufacturing field, thereby facilitating the intelligent upgrade of high-end manufacturing.

2. Tool Parameter Measurement Methods

For the measurement of tool geometry parameters, after collecting the original tool image, the original image is first converted to grayscale [5]. Then, guided filtering is applied to the grayscale image for noise reduction and optimization to protect the edge details of the tool image and lay the foundation for precise edge extraction. Next, the Canny edge detection method is used to roughly extract the edges of the preprocessed image to obtain the edge feature information of the tool contour. To eliminate redundant interfering contours and focus on the core detection area, the findContours function of OpenCV is used to extract the outermost continuous contour skeleton of the tool image, ensuring the targeted nature of subsequent detection. Subsequently, the dynamic weight adaptive fusion sub-pixel edge detection method proposed in this paper is employed to optimize the accuracy of the roughly extracted contour edges, significantly improving the positioning accuracy of the contour edges and achieving sub-pixel-level precise positioning of the tool edge contour. Finally, the extracted sub-pixel contour point set is subjected to feature point recognition, location, and fitting processing, and the quantitative values of various tool geometry parameters are obtained through precise calculation [6].

2.1 Image Preprocessing

In industrial production scenarios, the process of collecting tool images is susceptible to various interferences, inevitably introducing noise. The collected images often contain noise, which may originate from the inherent sensor noise of the acquisition equipment, circuit interference, or external disturbances. For high-precision measurement of tool geometric dimensions, image noise can seriously interfere with the accurate extraction of edge features, directly restricting the accuracy and reliability of the measurement results. Therefore, before edge detection, it is necessary to filter out the noise through effective image preprocessing methods while retaining the key edge details of the tool to the greatest extent, providing a guarantee for the subsequent precise acquisition of edge information.

As an image processing algorithm with outstanding performance, guided filtering [7] has been widely applied in multiple fields such as image denoising, detail enhancement, image segmentation, and HDR synthesis, thanks to its efficient computational efficiency and stable edge-preserving effect. It is particularly suitable for precision measurement scenarios where edge integrity is strictly required, which is highly consistent with the technical demands of tool geometry parameter measurement.

The core advantage of this algorithm lies in the collaborative realization of denoising and edge preservation. The basic idea is to construct a local linear model, calculate the output value by performing a linear transformation on the local neighborhood of each pixel, and the final output image is constrained by both the input original image and the guidance image. This fundamentally overcomes the inherent defect of traditional methods such as Gaussian filtering and mean filtering, which tend to cause edge blurring during the denoising process, and achieves the dual goals of noise suppression and edge protection.

Its local linear model consists of the guidance image I, the input image p, and the output image q. Within the local region of each pixel, with the guidance image I as the constraint, the structural information of the guidance image is utilized to smooth the input image p by solving the linear transformation of the pixel values within that region. This design effectively suppresses noise in the flat areas of the image during the filtering process, while the edge areas are completely preserved due to the structural constraints of the guidance image. Ultimately, it achieves a collaborative optimization of denoising and edge preservation, laying a solid foundation for the precise extraction of the tool edge contour and the accurate determination of the edge pixel positions in subsequent steps.

The specific steps of the calculation process of the guided filter are as follows:

(1) Local mean calculation: Calculate the mean values of the guidance image I and the input image p within the local region around each pixel. For each pixel i, the local mean values μ_I and μ_{p} can be expressed as:

$$\mu_I(i) = \frac{1}{|\omega_i|}\sum_{j\in\omega_i} I(j) \tag{1}$$

$$\mu_{\mathrm{p}}(i) = \frac{1}{|\omega_i|}\sum_{j\in\omega_i} \mathrm{p}(j) \tag{2}$$

Here, ω_{i} is a specified neighborhood window, which is a window of a fixed size.

(2) Local variance and covariance calculation: Calculate the variance and covariance of the guidance image I and the input image p within the local region:

$$\sigma_I^2(i) = \frac{1}{|\omega_i|}\sum_{j\in\omega_i}\left(I(j)-\mu_I(i)\right)^2 \tag{3}$$

$$\sigma_{I\mathrm{p}}(i) = \frac{1}{|\omega_i|}\sum_{j\in\omega_i}\left(I(j)-\mu_I(i)\right)\left(\mathrm{p}(j)-\mu_p(i)\right) \tag{4}$$

Here, σ_I^2 is the local variance of the guidance image I, and $\sigma_{I\mathrm{p}}$ is the local covariance between the guidance image and the input image.

(3) Calculate the linear coefficients: Compute the linear transformation coefficients within the local region through the local mean, variance and covariance.

The pixel values within the local region can be expressed by a linear model as: a(i) and b(i)

$$\mathrm{q(i)} = a(\mathrm{i})\cdot I(\mathrm{i}) + b(\mathrm{i}) \tag{5}$$

Among them, a(i) and b(i) can be calculated through the following formulas:

$$\mathrm{a(i)} = \frac{\sigma_{Ip}(\mathrm{i})}{\sigma_I^2(\mathrm{i})+\varepsilon} \tag{6}$$

$$\mathrm{b(i)} = \mu_p(\mathrm{i}) - a(\mathrm{i})\cdot\mu_I(\mathrm{i}) \tag{7}$$

Here, a(i) is the coefficient of the local linear transformation, indicating how the input image p is transformed on the guide image I; b(i) is the constant term, which determines the offset. ε is a small constant, usually set to 10^{-6}, to avoid division by zero errors.

After obtaining a(i) and b(i), the output pixel q(i) can be calculated. Since pixel i is involved in the calculation in all windows that contain it, the q(i) values obtained using different windows are different. Therefore, the average values of a(i) and b(i) are taken respectively, resulting in the following formula:

$$\bar{a}(i) = \frac{1}{|\omega_i|}\sum_{j\in\omega_i} a(j) \tag{8}$$

$$\bar{\mathrm{b}}(i) = \frac{1}{|\omega_i|}\sum_{j\in\omega_i} \mathrm{b(j)} \tag{9}$$

(4) Calculate the output image: By using the calculated linear coefficients a(i) and b(i), the output value q(i) for each pixel can be computed. For each pixel i, the output value q(i) is calculated by the following formula:

$$q(i) = \bar{a}(i) \cdot I(i) + \bar{b}(i) \quad (10)$$

This process is applied to each pixel in the image to calculate the output value of each pixel, thereby obtaining the final smoothed image q for the entire image. The smoothing operation is achieved by guiding the structural information of the image and can preserve the edge information of the image during the smoothing process.

2.2 Coarse Extraction of the Image Edge

After the image filtering process is completed, it is necessary to accurately detect the edge contour of the tool and determine the specific positions of all edge pixels [8]. During the edge processing of the tool, common edge detection operators include the Roberts operator [9], Sobel operator, Prewitt operator, Log operator, and Canny operator, etc. Table 1 shows the edge detection situations of each operator.

Table 1. Edge Detection by Various Operators

operator	Application situation
Roberts	It performs well in detecting images with steep gray- scale changes and low noise content, but its edge location accuracy is not ideal. It is more suitable for the initial extraction of horizontal and vertical edges.
Sobel	It has a certain noise smoothing ability and can output relatively accurate edge direction information, but the edge positioning accuracy is limited. It is suitable for scenarios where the detection accuracy requirements are not high.
Prewitt	It has noise suppression characteristics, but the edge positioning accuracy is generally average. In images with gentle gray-scale gradients and high noise content, the edge detection effect is more stable.
Log	The detection results are prone to double-pixel boundaries and are highly sensitive to noise. Therefore, they are less directly used for tool edge extraction and more often for determining the light and dark area attributes of edge pixels.
Canny	It has outstanding anti-noise interference ability and excellent denoising effect, and can generate fine and continuous edge contours, but it has the drawback of over-smoothing some fine edge details.

A comprehensive analysis of the above algorithms reveals that the Canny algorithm [10] detects edges by identifying local maxima of the image gradient. It calculates the gradient magnitude and direction using the first derivative of the Gaussian function and introduces a double-threshold mechanism to distinguish between strong and weak edges. Only when weak edges are directly connected to strong edges are they included in the final edge output. This design not only endows the Canny operator with an extremely strong ability to resist noise interference and effectively suppresses residual weak noise in the image, but also achieves a dynamic balance between noise suppression and edge preservation, enabling precise capture of truly meaningful weak edge information and avoiding the loss of key edges or the introduction of false edges. It can provide high- quality basic data for subsequent sub-pixel-level precise positioning and geometric parameter calculation. Therefore, the Canny operator is selected in this paper to perform the coarse extraction of pixel-level edges. The implementation process includes the following steps.

(1) Gaussian smoothing: Apply a Gaussian filter to the preprocessed tool image to reduce the impact of noise. The standard deviation σ of the Gaussian filter is typically within the range. The value is 0. 5 - 1. 0, and the specific value is adjusted according to the noise level of the image. When the noise is more significant, increase σ appropriately to enhance the denoising effect; when the noise is less significant, reduce σ to preserve edge details. This process is achieved through convolution operations, and the mathematical expression is as follows:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{(-\frac{x^2+y^2}{2\sigma^2})} \quad (11)$$

$$I_S(x,y) = (G * I)(x,y) \quad (12)$$

Here, $I(x,y)$ represents the preprocessed input image, $I_S(x,y)$ is the image after Gaussian smoothing, * denotes the convolution operation, and G (x, y) is the Gaussian kernel function.

(2) Gradient calculation: The Sobel operator is used to calculate the gradients of the image in the x and y directions to capture the intensity and direction of the change in pixel gray values.

The calculation expressions for the x-direction gradient G_x and the y-direction gradient G_y are as follows:

$$G_x = \frac{\partial I_S}{\partial x}, G_y = \frac{\partial I_S}{\partial y} \quad (13)$$

The expressions for the gradient magnitude M(x, y) and the gradient direction θ(x, y) are respectively:

$$M(x,y) = \sqrt{G_x(x,y)^2 + G_y(x,y)^2} \quad (14)$$

$$\theta(x,y) = \arctan 2(G_y(x,y), G_x(x,y)) \quad (15)$$

The gradient magnitude reflects the intensity of the gray-level change at a pixel, while the gradient direction indicates the dominant direction of the gray-level change, providing a core basis for subsequent edge thinning.

(3) Non-maximum suppression: To refine edge contours and eliminate false edge points, after obtaining the gradient magnitude image, non-maximum suppression processing is required. The core objective is to retain the local maximum value of each pixel in its gradient direction and suppress redundant pixels in non-edge regions. A 9 ×9 pixel neighborhood is constructed, with the central pixel as the detection point. The gray-level variation characteristics along the gradient direction of the central pixel are analyzed. If the gray-level values of the adjacent pixels along the gradient direction of the central pixel do not change, the first-order derivatives of the adjacent pixels with significant gray-level changes along the gradient direction and the central pixel are used to replace the partial derivatives in that direction. The gradient magnitude of the current pixel is checked along the gradient direction to determine if it is a local maximum. The gradient direction is quantized into four main directions. Along the quantized gradient direction, the gradient magnitudes of the central pixel and its adjacent pixels are compared. Only pixels with local maximum gradient magnitudes are retained, and key positions with significant gray-level changes are selected, while non-edge pixels with smooth gray-level changes are excluded. This way, the positions of pixels with significant gray-level changes are identified, and those with single gray-level changes are avoided. Finally, to eliminate false edges and retain true continuous edges, a double-threshold processing and hysteresis tracking method is adopted. The hysteresis threshold method is applied to screen all possible edge points. Two thresholds, Max and Min, are defined for the result after non-maximum suppression. If the gradient magnitude is less than Min, the edge point is eliminated. If it is greater than Max, the point is a confirmed edge and retained. If it is between Max and Min, the point is a weak edge. Then, for all weak edge pixels, if at least one strong edge pixel exists in their 8- neighborhood, they are retained as edges; otherwise, they are suppressed as non-edges.

Therefore, the Canny operator will output continuous edge contours of the end mill, effectively eliminating discontinuous edge points and false edge points. These edges are of pixel-level accuracy and are composed of individual pixels connected together. However, the Canny operator can only perform edge detection at the pixel level, which is difficult to meet the high- precision measurement requirements of tool geometry parameters.

Therefore, in order to further improve the accuracy of image contour detection, sub-pixel-level fine positioning processing needs to be carried out on the basis of the coarse positioning by the Canny operator. This can not only continuously resist noise interference but also improve the edge detection accuracy from the pixel level to the sub-pixel level, providing a guarantee for the precise calculation of subsequent geometric parameters.

2.3 Contour Extraction of Images

Based on the pixel-level binary edge map output by the Canny operator, the findContours function in OpenCV is used to quickly extract connected contours. This function retrieves the connected pixel regions in the image and outputs a set of pixel-level contour points composed of integer coordinates. On the one hand, it provides an accurate initial contour reference for subsequent sub-pixel-level precise positioning processing; on the other hand, it can be directly applied to the segmented fitting tasks of key parts such as the tool shank, blade, and cutting edge. The findContours function has stable and reliable contour extraction performance, effectively eliminating pseudo-contours formed by discrete noise points while maintaining the continuity of the core contours.

It has high integrity, and the algorithm has high execution efficiency and strong real-time performance, fully meeting the high-efficiency requirements of tool detection in industrial scenarios.

2.4 Sub-Pixel Edge Detection

In modern manufacturing, the measurement accuracy of tool geometry parameters directly determines the processing quality of precision parts. However, tool images have significant regional heterogeneity characteristics, and the dual strict requirements of industrial inspection for accuracy and real-time performance have made traditional pixel-level detection unable to meet the demands. Against this backdrop, sub-pixel edge detection technology has become the core support for breaking through the accuracy bottleneck. This technology breaks through the physical pixel limitations of hardware imaging systems and uses software algorithms to subdivide pixel units, thereby enhancing the image resolution to the sub-pixel level without changing the hardware configuration, and achieving higher-precision edge positioning. Sub-pixel edge points are typically distributed in the gray-scale gradient regions of the image and their precise positions can be solved through various algorithms such as polynomial fitting and moment calculation. Essentially, it is a high- precision image processing technology that optimizes algorithms to break through hardware limitations. Currently, mainstream sub-pixel edge detection algorithms can be classified into three major categories: moment methods [11], interpolation and fitting methods [12].

Although existing algorithms have made certain progress, significant challenges remain in tool detection in complex industrial scenarios. Traditional sub-pixel algorithms typically employ fixed processes and parameter configurations, lacking robustness and the ability to dynamically adjust detection strategies based on the local features of tool images. In actual tool images, the edge characteristics of different regions vary significantly: the cutting edge area has clear edges and high gray-level contrast, demanding extremely high positioning accuracy; the texture interference area requires suppression of redundant features to avoid false detections; and the reflective area of the cutting edge shows complex nonlinear gray-level gradients, prone to edge breaks or false edges. Fixed detection algorithms cannot adapt to this regional heterogeneity, resulting in a significant decline in detection accuracy in blurred and noise-interfered areas, while also struggling to balance edge positioning accuracy and continuity.

In response to the shortcomings of existing technologies, this paper proposes a two-stage optimization sub-pixel edge detection algorithm based on dynamic weight adaptive fusion. This algorithm is based on a local feature perception mechanism [13] and adopts a dual-core fusion strategy. It innovatively employs a dual-parallel processing branch architecture: the fitting algorithm based on the local gray area effect and the improved region growing algorithm are used as the core detection branches. The feature quality score of the first-stage detection results is extracted through the local feature perception mechanism, and the dynamic weight allocator adjusts the output weight ratio of the two branches in real time according to this score. Specifically, for regions with high feature quality scores, such as clear edge regions, the fitting algorithm based on the local gray area effect is assigned an 80% weight ratio to prioritize sub-pixel- level positioning accuracy; for regions with low feature quality scores, such as reflective and blurred areas, the improved region growing algorithm is assigned an 80% weight ratio to prioritize the continuity and integrity of the edge. This algorithm can dynamically adapt the fusion strategy based on the local features of the tool image, achieving an adaptive balance between high-precision positioning and stable continuity, ultimately achieving a more precise and robust sub-pixel edge detection effect, providing reliable support for the high-precision measurement of tool geometric parameters.

2.4.1 Local feature extraction and feature quality scoring

To accurately quantify the quality level of each Canny edge point and provide a reliable basis for subsequent dynamic weight allocation, this algorithm selects the magnitude of the gray-level gradient of the edge point as the core feature parameter, and directly calculates the feature quality score f based on this parameter, with a value range of [0, 1].

(1) Calculation of the gray gradient magnitude: For each edge point (i, j) obtained by the Canny edge detection, the gray gradient magnitude M(i, j) reflects the intensity and clarity of the edge at that point. The calculation formula is:

$$M(i,j)=\sqrt{(G_x(i,j))^2+(G_y(i,j))^2} \quad (16)$$

Here, G_x and G_y are respectively the gradient components of the edge point (i, j) in the x-direction and y-direction.

(2) Feature quality score mapping: To convert the gray gradient magnitude into an intuitive quality grade, based on the statistical analysis of the tool image data set, the effective distribution range of the gray gradient magnitude at the tool edge is determined to be 0-100. According to this, a trapezoidal mapping function is designed to adaptively convert M(i, j) into the feature quality score f, as expressed in the following equation:

$$f = \begin{cases} 0 & M < 10 \\ \dfrac{M-10}{50-10} & 10 \le M \le 50 \\ 1 & M > 50 \end{cases} \quad (17)$$

When M<10, the edge grayscale change is extremely weak, being noise points or blurred pseudo-edges, and f = 0;

When 10≤M≤50, the edge sharpness is at a medium level, and the intensity of gray-scale change is positively correlated with edge quality. Therefore, a linearly increasing mapping is adopted to ensure the continuity of the score.

When M>50, the edge grayscale contrast is strong, the contour is clear, and it has high positioning reliability, with f = 1.

This mapping relationship not only retains the strong characterization ability of the gray gradient magnitude for edge quality but also simplifies the calculation process through piece wise mapping, avoiding the increase in complexity and redundant calculations caused by multi-feature fusion.

(3) Error Compensation for Feature Quantization: In the medium clarity region (10 ≤ M ≤ 50), the characterization of edge quality by a single gray gradient amplitude has limitations and is prone to score deviations due to local noise or gray-level fluctuations. To address this, the mean gradient of the neighborhood is introduced for error compensation, thereby enhancing the robustness of the score by integrating local neighborhood information. The specific formula is as follows:

$$\overline{M}(i,j) = \frac{1}{N}\sum_{(k,l)\in N} M(i+k, j+l) \quad (18)$$

$$f_{com}(M) = \alpha f(M) + \beta f(\overline{M}(i,j)) \quad (19)$$

Here, N represents the 3 ×3 neighborhood of the edge point (i, j), and M(i, j) is the mean of the gradient magnitudes of all pixels in the neighborhood. The compensation coefficients α and β are determined through grid search optimization: the search range for α is set to [0.5, 0. 9], and for β to [0. 1, 0. 5], with a step size of 0. 1. The combination of coefficients that yields the highest match between the edge quality score and the actual edge clarity is ultimately selected to ensure the effectiveness and scientific nature of the compensation mechanism. Here, α and β are determined to be 0.3 and 0.7, respectively.

2.4.2 Fitting algorithm based on local gray area effect

During the manufacturing process of cutting tools, the edge profile is formed by regular feed motion of the machining process, strictly following the curve generation mechanism. Therefore, for edges with high clarity and stable features, the sub-pixel edge extraction algorithm based on fitting is more capable of accurately capturing the feature edge points consistent with the actual contour of the cutting tool. The sub-pixel edge extraction algorithm based on local gray area effect fitting [14] has the core idea of utilizing the gray distribution law of the edge pixel neighborhood to construct a mathematical model for fitting the edge curve, thereby achieving sub-pixel level edge positioning. The technical principle is shown in Figures 1 and 2.

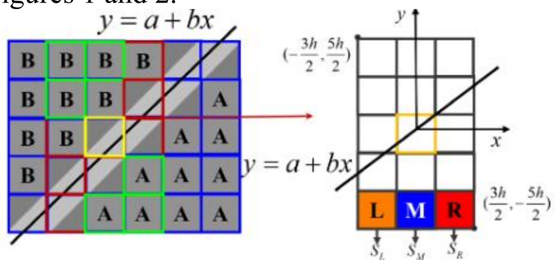
$y = a + bx$
B
A
$y = a + bx$
y
x
$(-\frac{3h}{2}, \frac{5h}{2})$
$(\frac{3h}{2}, -\frac{5h}{2})$
L
M
R
S_L
S_M
S_R

Figure 1. Ideal Straight Line Edge

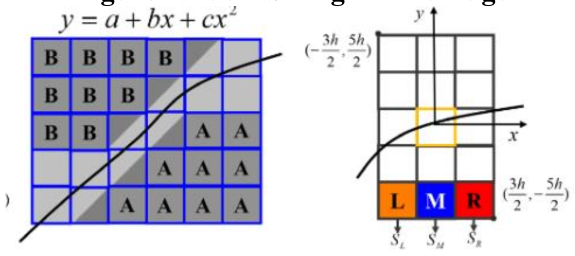
$y = a + bx + cx^2$
B
A
y
x
$(-\frac{3h}{2}, \frac{5h}{2})$
$(\frac{3h}{2}, -\frac{5h}{2})$
L
M
R
S_L
S_M
S_R

Figure 2. Ideal Curve Edge

When the edge passes through a single pixel, the pixel is divided into two sub-regions, one belonging to the tool area and the other to the background. The actual gray level of this pixel is not the gray level at a certain point on the edge, but rather the weighted average of the ideal gray levels of the tool area and the background area, based on their respective areas within the pixel. Based on this characteristic, for a pixel with an edge passing through it, the gray level $F_{i,j}$ can be modeled in the image acquisition process as follows:

$$F_{i,j} = \frac{A \cdot S_A + B \cdot S_B}{h^2} \tag{20}$$

Here, A and B respectively represent the ideal gray values of the regions on both sides of the edge, usually taken as the average gray values of the pixels completely within the corresponding regions near the edge; SA and SB respectively represent the areas of the tool region and the background region divided by the edge within this pixel; h is the side length of the pixel, which is usually normalized to 1, so $S_A + S_B = h^2 = 1$, that is, the sum of the areas of the two sub-regions equals the total area of the pixel.

Based on the established grayscale image acquisition model, ideal equations are respectively constructed for linear and curved edges. Within a 3 ×5 calculation window, the edge perimeter is solved through integral operations.

The distribution of the gray-scale area within the window is used to calculate the fitting edge coefficient by reverse area calculation, ultimately achieving precise determination of the sub-pixel edge position. For each column of pixels within the window, when calculating the sub-pixel edge area based on the linear equation, the edge line divides it into three regions. The areas L, M, and R within each column that are included inside the edge can be obtained by integrating the linear equation, and the formula is:

$$\begin{cases} L = \int_{-3h/2}^{-h/2} (a + bx + 5h/2) dx \\ M = \int_{-h/2}^{h/2} (a + bx + 5h/2) dx \\ R = \int_{h/2}^{3h/2} (a + bx + 5h/2) dx \end{cases} \tag{21}$$

According to the core acquisition model, the sum of the gray values of each column of pixels is linearly related to the area of the corresponding region of that column. Thus, the relationship between the gray area and the area is established, and the equation is:

$$\begin{cases} S_L = \frac{A-B}{h^2} L + 5B \\ S_M = \frac{A-B}{h^2} M + 5B \\ S_R = \frac{A-B}{h^2} R + 5B \end{cases} \tag{22}$$

In actual calculations, A and B are usually estimated by taking the average gray value of the pixels at the corners of the window, as these pixels are most likely to be entirely on one side of the edge and can accurately represent the ideal gray value of the corresponding area. Combining the above linear relationship between gray value and area, a system of equations for the edge line parameters a and b of the equation $y = a + bx$ can be constructed, and the solution is obtained as:

$$a = \frac{2S_M - 5(A+B)}{2(A-B)} \tag{23}$$

$$b = \frac{S_R - S_L}{2(A-B)} \tag{24}$$

Once the parameters a and b of the straight line are determined, the sub- pixel position of the edge can be directly calculated: the intersection coordinates of the straight line with the pixel grid lines are the sub-pixel edge points. If the pixel coordinates of the window center in the entire image are (x_c, y_c), then the coordinates (x_{sub}, y_{sub}) of the sub-pixel edge points are:

$$x_{sub} = x_c + \delta_x, y_{sub} = y_c + \delta_y \tag{25}$$

Among them, δ_x and δ_y are the offsets of the window center relative to the sub-pixel edge point, which can be calculated based on the edge line equation.

Similarly, the area calculation formula and the coefficient solving formula for the curve equation $y = a + bx + cx^2$ are:

$$\begin{cases} L = \int_{-3h/2}^{-h/2} (a + bx + cx^2 + 5h/2) dx \\ M = \int_{-h/2}^{h/2} (a + bx + cx^2 + 5h/2) dx \\ R = \int_{h/2}^{3h/2} (a + bx + cx^2 + 5h/2) dx \end{cases} \tag{26}$$

$$\begin{cases} S_L = \frac{A-B}{h^2} L + 5B \\ S_M = \frac{A-B}{h^2} M + 5B \\ S_R = \frac{A-B}{h^2} R + 5B \end{cases} \tag{27}$$

$$\begin{cases} a = \frac{26S_M - S_L - S_R - 60(A+B)}{24(A-B)} \\ b = \frac{S_R - S_L}{2(A-B)} \\ c = \frac{S_L + S_R - 2S_M}{2(A-B)} \end{cases} \tag{28}$$

This algorithm performs linear or curve fitting within the pixel scale, and the extracted edge features are both smooth and dense. Its positioning accuracy can meet the requirements of high-precision measurement of tool geometry parameters in subsequent steps, effectively compensating for the accuracy deficiency of pixel-level detection.

2.4.3 Improved region growing algorithm

The traditional region-growing algorithm takes the edge points detected by Canny as the initial seeds and merges adjacent pixels based on a single gray- level similarity criterion to form continuous edge regions. However, this method has problems such as weak noise resistance, poor edge continuity, and insufficient threshold adaptability in tool image detection. To adapt to the regional heterogeneity characteristics of tool edges, this paper proposes an improved region-growing algorithm. By optimizing the growth criterion, threshold strategy, and search direction, the robustness and continuity of edge extraction are enhanced. The specific design is as follows:

(1) Multi-constraint growth criterion design: To avoid the incorrect merging of noise points, a dual-constraint criterion based on gray-level difference and gradient similarity is adopted, and the seed point selection mechanism is simultaneously optimized:

a. Gray-level difference criterion: Calculate the gray-level difference $\Delta g = |g_{curr} - g_{seed}|$ between the current pixel to be grown and the seed point. If $\Delta g < T_g$, the merging condition is satisfied. The threshold T_g is dynamically adjusted based on the noise level of the image.

b. Gradient similarity criterion: Constrain the consistency of growth from both the gradient magnitude and direction dimensions to avoid incorrect merging across edge regions. The gradient magnitude difference constraint is $|\nabla I_1 - \nabla I_2| < T_m$, and the gradient direction difference constraint is $|\theta_1 - \theta_2| < T_\theta$, where T_m and T_θ are the gradient magnitude threshold and gradient direction threshold, respectively, ensuring that the growth direction is consistent with the edge extension direction.

Therefore, to eliminate the growth deviation caused by noise points as initial seeds, only strong edge points with a gray gradient magnitude M(i,j) $\geq$30 are selected as seed points, which can effectively filter out weak noise points and blurred pseudo-edges.

(2) Adaptive threshold dynamic adjustment mechanism: The growth threshold is dynamically adjusted based on local region features. For regions with clear edges, smaller gray-level difference thresholds and gradient value thresholds are used to precisely control the growth range and prevent excessive edge expansion. For regions with noise and blurring, larger gray- level difference thresholds and gradient value thresholds are adopted to enhance the algorithm's tolerance to gray-level fluctuations and ensure continuous edge growth. For regions of medium quality, intermediate thresholds are used to achieve a balance between accuracy and continuity.

(3) Multi-directional directional growth strategy: To enhance the continuity and directionality of edge connections, a growth strategy of "8-neighborhood search and gradient direction priority" is adopted. Taking the current seed point as the center, an 8-neighborhood search window is constructed to cover all adjacent pixels, avoiding the omission of potential edge points; combined with gradient direction information to optimize the growth priority, it prioritizes searching adjacent pixels along the gradient vertical direction, and then successively traverses other neighborhood directions.

The improved region-growing algorithm, through the collaborative optimization of multiple constraint criteria, adaptive thresholds, and directional search, can not only maintain edge continuity in noisy areas but also control the growth accuracy in clear areas, providing high-quality edge detection results for subsequent dynamic weight fusion.

2.4.4 Dynamic weight adaptive fusion mechanism

Dynamic weight adaptive allocation is the core innovation of the sub- pixel edge detection algorithm proposed in this paper. Its main objective is to adjust the contribution weights of the local gray area effect fitting algorithm and the improved region growing algorithm in real time based on the local feature quality of edge points, achieving precise complementarity of the advantages of the two algorithms. In clear edge regions, the positioning accuracy of the local gray area effect fitting algorithm is emphasized, while in blurred regions, the continuity guarantee of the improved region growing algorithm is strengthened. Ultimately, sub-pixel edge detection results with both high precision and high robustness are output.

(1) Weight function design

To achieve a smooth transition and precise

adaptation of weights, a piece wise function based on the feature quality score f is adopted to design the weight allocation rule, ensuring that the weights change dynamically with the edge quality and avoiding detection deviations caused by sudden changes. The specific calculation formula is as follows:
For each edge point (i, j), the weight of the local gray-level area effect fitting algorithm:

$$w_1=\begin{cases}0.2 & f<0.5\\ 0.2+0.6\cdot\dfrac{f-0.5}{0.3} & 0.5\le f\le 0.8\\ 0.8 & f>0.8\end{cases}\qquad(29)$$

Weight of the improved region-growing algorithm:

$$w_2=1-w_1\qquad(30)$$

This design achieves smooth switching of weights through piece wise linear mapping, avoiding the problem of insufficient adaptability of a single weight mode.
(2) Weight Optimization Strategy
To further enhance the rationality of weight distribution and the stability of edge detection, a spatial consistency constraint mechanism is introduced to optimize the initial weights, avoiding detection deviations caused by sudden changes in the weights of isolated points: the weight values of adjacent edge points should maintain a certain consistency, and the weights are smoothed through neighborhood averaging.

$$\tilde{w}_1(i,j)=\frac{1}{N}\sum_{(k,l)\in N}w_1(i+k,j+l)\qquad(31)$$

Where N is the 3×3 neighborhood of the edge point (i,j).
$\tilde{w}_1(i,j)$ represents the weight of the optimized local gray-scale area effect fitting algorithm. This constraint can effectively suppress the sudden change in weights caused by isolated noise points and ensure the smoothness of the weights in the local area.
2.4.5 Algorithm implementation process
In the process of sub-pixel edge detection, for the image after contour extraction, the set of pixel points has been obtained. Based on the above-mentioned weight design and optimization strategy, the specific implementation steps of the dynamic weight adaptive fusion sub-pixel edge detection algorithm are as follows:
Step 1: Input data. Read the tool image I after contour extraction and the corresponding pixel-level peripheral contour point set P;
Step 2: Calculate the initial weights. Based on the feature quality score f of each edge point (i, j), calculate the initial weights w1 and w2 for the local gray area effect fitting algorithm and the improved region growing algorithm respectively.
Step 3: Weight optimization. Apply spatial consistency constraints. Optimize the initial weights to obtain the final weights $\tilde{w}_1(i,j)$ and $\tilde{w}_2(i,j)$;
Step 4: Calculate the local fitting results separately: For each edge point (i,j), obtain the sub-pixel edge result E1(i,j) based on the gray gradient magnitude M(i, j) through the local gray area effect fitting algorithm; take the strong edge points M(i, j) output by the Canny edge detection as the initial seeds and execute the improved region growing algorithm to obtain the sub-pixel edge result E2(i,j).
Step 5: Dynamic Weighted Fusion. Based on the final optimized weights, the output results of the two algorithms are weighted and fused to obtain the final sub-pixel edge point E(i,j), with the formula being:

$$E(i,j)=w_1\cdot E_1(i,j)+w_2\cdot E_2(i,j)\qquad(32)$$

Step 6: Output the results. Output the sub-pixel edge image E(i, j) and the corresponding set of sub-pixel edge points Q, providing high-precision data support for the subsequent fitting calculation of tool geometric parameters.

3. Experimental Results and Analysis

3.1 Camera Calibration

In the measurement of tool geometry parameters, to accurately convert the measurement results from pixel units to physical length units, it is necessary to first determine the pixel equivalent, that is, the actual physical size represented by each pixel spacing. Based on the imaging principle, the pixel equivalent can be theoretically calculated through the magnification of the lens and the size of the photosensitive chip. However, due to factors such as manufacturing errors of the lens and assembly deviations, the theoretical calculation results are prone to deviations and cannot meet the high-precision measurement requirements of tool geometry parameters. Therefore, it is necessary to calibrate the camera system through experimental calibration to obtain an accurate pixel equivalent.
This paper uses a standard calibration board to complete the camera calibration [15], as shown

in Figure 3. The specific process is as follows: Fix the calibration board on the actual working plane for tool measurement, control the camera to precisely focus on the calibration board and then take pictures; after preprocessing the collected calibration board images, count the number of pixels q occupied by a single grid in the image; given that the actual physical length of a single grid of the calibration board is l, the calculation formula for the pixel equivalent γ is:

$$\gamma = l / q \tag{33}$$

Through the above calibration process, the actual physical length corresponding to a single pixel was ultimately determined, that is, the pixel equivalent is 47.52 μm. In the subsequent measurement of tool geometric parameters, it is only necessary to detect the image.

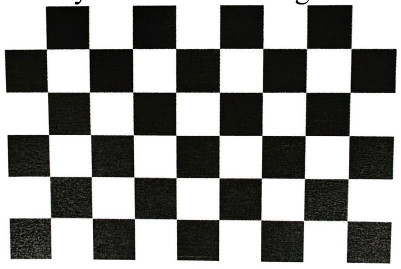
Figure 3. Calibration Board

The actual physical values of the tool geometry parameters can be obtained by multiplying the number of pixels corresponding to the tool geometry parameters measured by the algorithm with the pixel equivalent obtained through calibration.

3.2 Image Preprocessing and Precise Edge Extraction

To verify the effectiveness and measurement accuracy of the tool geometry parameter measurement algorithm proposed in this paper, the algorithm program was developed and implemented on the tool geometry parameter measurement system platform based on VS Code software using Python language [16]. The experiment took straight shank end mills as the measurement objects. Their color images were collected by high-definition industrial cameras. After a series of processing on each image, the core geometry parameters such as the total length of the tool, the diameter of the shank, and the helix angle were measured [17]. Finally, the measurement results of the algorithm were compared with the results of manual precision measurement to verify the performance of the algorithm. The specific steps are as follows:

First, image preprocessing is carried out. To eliminate the interference of color redundancy information in the color image on edge detection and suppress the noise introduced during the acquisition process, the collected color image of the straight shank end mill is first converted to grayscale, compressing the three- dimensional color information into one-dimensional grayscale information. Then, guided filtering technology is used to denoise and optimize the grayscale image. This filtering method can efficiently suppress noise while precisely preserving the edge details of the tool, laying a high-quality image foundation for subsequent edge extraction. The preprocessing effect is shown in Figure 4.

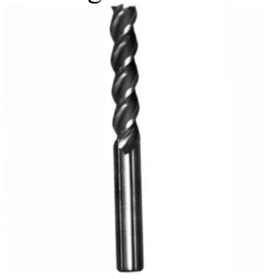
Figure 4. Image Preprocessing

Then, a rough edge extraction is carried out. The Canny edge detection algorithm is used to extract the rough edge contour of the milling cutter, and a single-pixel-level edge image is ultimately obtained, providing a reliable basis for contour extraction. To further screen out effective contours and eliminate redundant interference, the findContours function of OpenCV is called to extract contours from the binary edge image output by the Canny detection, and finally a set of continuous and connected pixel-level contour points is obtained, as shown in Figure 5.

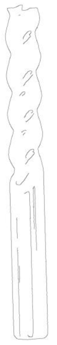
Figure 5. Coarse Edge Extraction

Finally, sub-pixel edge precise positioning is completed. Considering that the pixel-level edge detection accuracy is difficult to meet the high-precision measurement requirements of tool geometry parameters, it is necessary to further improve the positioning accuracy to the sub-pixel level. For this purpose, the dynamic weight adaptive fusion sub-pixel edge detection algorithm proposed in this paper is used to perform fine optimization processing on the above pixel- level contour point set. By dynamically adjusting the contribution weights of the dual algorithms, the precise capture of edge details and the smooth optimization of the contour are achieved. Ultimately, the sub-pixel-level edge contour of the tool is obtained as shown in Figure 6, providing core data support for the subsequent precise fitting and calculation of geometric parameters.

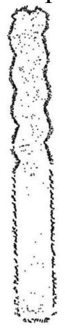

Figure 6. Sub-Pixel Edge Detection

3.3 Tool Geometry Parameter Measurement

After the sub-pixel edge extraction is completed, a method combining feature point recognition, location and contour fitting is adopted to fit the obtained sub- pixel edge coordinate point set. Experiments on the measurement of tool geometric parameters are carried out to verify the performance of the algorithm, and the core geometric parameters are solved through this method. Taking the measurement of the total length of the tool as an example, the specific solution process is as follows: For the tool edge line obtained by least squares fitting [18], a column scanning strategy is used to traverse the effective area of the image, extract the upper and lower boundary y-coordinates corresponding to each column on the edge line, calculate the difference between the upper and lower boundaries coordinates in the same column, and the maximum value is the number of pixels corresponding to the total length of the tool; multiply this pixel number by the pixel equivalent obtained through calibration, and the actual total length of the tool can be converted [19].

The experimental measurement results and measurement errors of the tool geometry parameters are shown in Table 2. Further analysis of the experimental data in Table 2 reveals that the tool geometry parameter measurement method proposed in this paper can output precise measurement results: the total length measurement error can be controlled within 3 μm, and the angle parameter measurement error can be controlled within 0.5, verifying the feasibility and reliability of the algorithm in industrial practical applications.

Table 2. Measurement Results of Tool Geometric Parameters

Geometric parameters	Measurement results					Average measurement results	Actual value	measure error
	1	2	3	4	5			
total length	94.996	94.997	94.998	94.997	94.998	94.997	95.000	0.003
diameter	9.999	9.998	9.997	9.998	9.997	9.998	10.000	0.002
Screw angle	34.84	34.89	34.91	34.88	34.87	34.878	35.00	0.122

4. Experimental Results and Analysis

This paper develops a tool geometric parameter measurement system based on machine vision, with the core focusing on the optimization of coarse image edge extraction and innovation in subpixel edge detection. A dynamic weight adaptive subpixel edge detection algorithm is proposed, providing a reliable solution for high-precision measurement. Experimental verification shows that the system exhibits good accuracy, robustness, and efficient operational performance.

The core technical system of the system follows the "preprocessing [20] - rough extraction - precise positioning" approach: guided filtering is used for preprocessing to achieve noise suppression and edge protection; the Canny algorithm is used for rough edge extraction to obtain a reliable initial edge point set; the local gray area effect fitting and the improved region growing algorithm are innovatively integrated, and the dynamic allocation of weights is used to achieve complementary advantages in accuracy and continuity, significantly improving the edge positioning accuracy of complex tools.

References

[1] Wang W H,Hong G S,Wong Y S. Flank wear measurement by a threshold independent method with sub-pixel ac-curacy. International Journal of Machine Tools and Manufacture, 2006, 46(2): 199-207.

[2] Kassim A A, Mannan M A, Zhu M. Texture analysis methods for tool condition monitoring. Image and Vision Computing, 2007, 25(7): 1080-1090.
[3] Guan B. Research on image measurement technology for cutting tool geometric parameter. North University of China, 2015.
[4] Zhang C C. Study on the fast extraction of the edge of Canny operator cutter. North University of China, 2015.
[5]Chen K K. Research on Key Technologies of Tool Angle Measurement
System Based on Machine Vision. China University of Mining and Technology, 2021.
[6] Zhou J J, Yu J B. Online Measurement of Tool Wear Based on Machine Vision. Journal of Shanghai Jiao Tong University, 2021, 55(06): 741-749.
[7] Wu D H. Software Development for a Tool Measurement System Based on Machine Vision. Xi'an University of Technology, 2023.
[8] Wang J F. Design and Application of an Online Measurement System for End Mills Based on Machine Vision. North China University, 2020.
[9] Xie X, Ge S L, Xie M Y, et al. An improved industrial sub-pixel edge detection algorithm based on coarse and precise location. Journal of Ambient Intelligence and Humanized Computing, 2019, 11(5):2061-2070.
[10]Seo S Y. Subpixel Edge Localization Based on Adaptive Weighting of Gradients. IEEE transactions on image processing: a publication of the IEEE Signal Processing Society, 2018, 27(11):5501-5513.
[11]Bai X, Yang M J, Ajmera Beena. An Advanced Edge-Detection Method for Noncontact Structural Displacement Monitoring. Sensors, 2020, 20(17):4941.
[12]Hou Q L. Research on Tool Detection Technology Based on Machine Vision. Jinan: Shandong University, 2018.
[13]Trujillo-Pino A, Krissian K, Alemán-Flores M, et al. Accurate subpixel edge location based on partial area effect. Image and Vision Computing, 2013, 31(1):72-90.
[14]ZHANG P D. Research on Visual Measurement Methods and Systems for Complex Tool Geometry Parameters. Dalian University of Technology, 2023.
[15]Peng L W. Research on Turning Tool Wear Detection System Based on Machine Vision. Southwest Petroleum University, 2024.
[16]Wang W S, Zhang W, Li Z Y, et al. Improvement of tool geometry parameter measurement algorithm based on OpenCV. Tool Technology, 2021 (01).
[17]Cui T. Research on Contact Measurement Method and Application Technology of Integral End Milling Cutter. Southwest Jiaotong University, 2021.
[18]Zhao C L, Wang K, Wang X X, et al. Research on Tool Thermal Deformation Measurement Based on Image Processing Technology. Machine Tool and Hydraulic, 2021, 49(20): 28-31.
[19]Zhang B, Shen X L, Han Z Y. Machine Vision based Tool Geometric Parameter Measuring Instrument. Technology and Innovation, 2022, (13): 136-139.
[20]Guo RL, Zhang H, Zhi X B, et al. Research on In-machine Detection of Tool Wear Based on Machine Vision. Journal of Lanzhou University of Technology, 2024, 50(06): 33-41.