

# Artificial Intelligence for Smart Manufacturing

Image processing and prediction model of electrode imprinting in resistance spot  
welding



**Politecnico  
di Torino**

## **SUPERVISORS**

Prof. DE MADDIS MANUELA  
Prof. BRUNO GIULIA

## **CANDIDATE**

HOU XIN

## ABSTRACT

### **Image processing and prediction model of electrode imprinting in resistance spot welding**

Resistance spot welding has the characteristics of concentrated energy, small deformation, high productivity and easy automation, which makes it an indispensable connection method in the car body assembly process. In the process of spot welding, electrode wear is inevitable. When the number of welding, electrode wear degree is not large, the impact on the quality of the welded joint is not obvious, but when the wear reaches a certain level, it will seriously affect the quality of the welded joint, and even cause the welded parts are burned through. Electrode wear is mainly manifested as changes in the shape of the electrode end face, therefore, the characteristics of the electrode image can indirectly reflect the life state of the electrode. In this paper, the following research work is carried out to predict the wear degree of electrode head based on image processing and support vector machine, using electrode imprinted image as the information source, with the goal of judging the wear degree of electrode head.

In this paper, electrode imprinted image preprocessing and image edge detection algorithms are investigated, and five classical edge detection operators are introduced, including Roberts operator, Sobel operator, Canny operator, Prewitt operator and LOG algorithm detection methods. And the electrode images combined with these different operators are processed separately using the correlation functions provided by the MATLAB system, etc., and their characteristics of image processing are briefly analyzed. Satisfactory binary images were obtained using the LOG algorithm, and the shape features of the images were extracted based on the electrode surface images, including electrode perimeter, electrode area, electrode roundness and pitted area. An SVM-based indentation depth prediction model for weld joints was developed. In which the extracted image features are used as input and the number of weld joints already experienced by the electrode is used as output. The correlation coefficient between the prediction results and the number of weld joints reached 0.9989. The results of the study show that the established SVM prediction model can achieve the prediction of indentation depth.

**Key words:** MATLAB; resistance spot welding; image processing; edge detection; support vector machine

## TABLE OF CONTENTS

<b>ABSTRACT .....</b>	<b>2</b>
<b>Background of resistance Spot welding and development of image processing.....</b>	<b>6</b>
1.1 Research Background and Significance .....	6
1.2 Research status of wear and service life of electrodes .....	7
1.3 Development and application of digital image processing .....	9
1.3.1 Development of digital image processing.....	9
1.3.2 Application status of digital image processing .....	9
1.4 Development and application of pattern recognition .....	10
<b>INTRODUCTION OF MATLAB .....</b>	<b>13</b>
2.1 GENERAL .....	13
2.2 IMAGES IN MATLAB.....	13
2.3 INTRODUCTION TO IMAGE PROCESSING TOOLBOX.....	14
2.4 VISUALIZATION OF IMAGES IN MATLAB .....	14
2.5 IMAGE PROCESSING TOOLBOX PRODUCT DESCRIPTION.....	15
2.6 PIXEL INDICES.....	16
2.7 IMAGE COORDINATE SYSTEMS .....	17
2.7.1 SPATIAL COORDINATES.....	17
2.7.2 INTRINSIC COORDINATES.....	17
2.7.3 INTRINSIC COORDINATES SYSTEM.....	18
2.7.4 WORLD COORDINATES.....	18
2.7.5 DEFINE WORLD COORDINATES USING XDATA AND YDATA PROPERTIES.....	19
<b>Electrode image processing and feature extraction.....</b>	<b>20</b>
3.1 Wear mechanism of electrode head .....	20
3.1.1 Shape of electrode head .....	20
3.1.2 The role of resistance spot welding electrodes.....	21
3.1.3 Wear of electrode head .....	21
3.2 Electrode imprint image analysis .....	22
3.3 Image pre-processing .....	24
3.3.1 Image normalization.....	24
3.3.2 Image Filter .....	25
3.3.2.2 Average Filter.....	28

3.4 Image Segmentation.....	29
3.4.1 Threshold-based segmentation method .....	30
3.4.1.1 two-mode method .....	30
3.4.1.2 Iterative method.....	31
3.4.1.3 OSTU Method.....	31
3.5 Edge-Based Image Segmentation .....	34
3.5.1 Edge of image.....	34
3.5.2 Principle of edge detection .....	35
3.5.3 Convolution theory.....	36
3.5.3.1 Basic theory.....	36
3.5.3.2 Edge effect.....	37
3.5.4 Roberts operator.....	37
3.5.4.1 Basic theory.....	37
3.5.4.2 Implementation of the Roberts operator for image edge detection .....	38
3.5.5 Sobel Operator .....	39
3.5.5.1 Basic theory.....	39
3.5.5.2 Implementation of Sobel algorithm for image edge detection.....	41
3.5.6 Prewitt Operator .....	42
3.5.6.1 Basic theory.....	42
3.5.6.1 Implementation of the Prewitt operator for image edge detection .....	43
3.5.7 Krisch Operator .....	44
3.5.7.1 Basic theory.....	44
3.5.7.2 Implementation of Krisch algorithm for image edge detection.....	45
3.5.8 LOG Operator.....	47
3.5.8.1 Basic theory.....	47
3.5.8.2 Gauss operator .....	48
3.5.8.3 LOG.....	48
3.5.8.4 Implementation of Gauss-Laplace operator for image edge detection.....	49
3.5.9 Canny Operator.....	50
3.5.9.1 Basic theory .....	50
3.5.9.2 Implementation of the Canny operator for image edge detection.....	54
3.5.10 Comparative Analysis of Image Edge Detection Operators .....	55

3.6 Electrode imprint image feature parameters .....	59
3.6.1 Electrode imprint area.....	59
3.6.2 Electrode imprint perimeter.....	60
3.6.3 Roundness .....	60
<b>SVM prediction model of electrode wear .....</b>	<b>63</b>
4.1 The basic principle of SVM .....	63
4.2 Feature parametric extraction.....	67
4.2.1 SVM regression prediction model.....	67
4.2.2 Data extraction and pre-processing.....	68
4.2.3 Parameter optimization selection .....	69
4.2.4 Model Training.....	70
4.2.5 Regression prediction .....	71
<b>Conclusion.....</b>	<b>73</b>
<b>BIBLIOGRAPHY.....</b>	<b>75</b>

# CHAPTER 1

## BACKGROUND OF RESISTANCE SPOT WELDING AND DEVELOPMENT OF IMAGE PROCESSING

### 1.1 Research Background and Significance

Welding is a physical method or chemical way to produce a combination of molecules or atoms between the contact surfaces of two separated workpieces to form a whole and to meet certain performance requirements. The welding process has a high status in modern industrial production and has an extremely wide range of applications. The rapid development of welding technology has led to tremendous progress in industrial manufacturing, thus affecting the national economy. The earliest welding technology, applied only to blacksmith forging, has evolved over the last hundred years to nearly a hundred methods of welding, from the early history of only arc welding and gas welding to the present stage of development that includes all available energy sources such as sound, light, heat, electricity, force and magnetism together to promote. People through the research and exploration of various energy sources, the study of materials, mechanics and thermodynamics, so that the welding process continues to progress.

Resistance welding is a welding method in which the workpiece is fixed between electrodes, the weld area is heated to a plastic or molten state, and the joint is formed under pressure. Among them, resistance spot welding uses columnar electrodes that are pressurized and energized between the contact surfaces of the lapped workpiece, and the resulting resistance heat melts the workpiece and forms a welded joint after cooling. Resistance welding is an efficient, economical and environmentally friendly welding method and is indispensable in the processing of metals such as steel because of its concentrated energy, low heating deformation and the absence of fillers. Resistance spot welding is suitable for welding of thin plates, usually below 6mm in thickness, and its economy and practicality exceed those of other processing processes. The contribution of resistance spot welding is indispensable in the field of automotive manufacturing, especially in the welding of body-in-white, where resistance spot welding has been successfully used for over 100 years<sup>[1,2]</sup>. Since the automotive industry has the highest quality standards of spot welds combined with their high number, roughly between 3000 spot welds for a small passenger car up to 9500 for a transporter<sup>[3]</sup>.

Thomson Elihu invented resistance spot welding technology in 1877, and since then the resistance spot welding process has been accepted by many industrial production fields for its advantages of easy operation, good welding quality and low production cost. The electrode cap is in direct contact with the workpiece during the welding process, the electrode and the surface of the welded part have a certain pressure, and the temperature is high when energized, after many times welding, the wear of the electrode is inevitable. When the degree of electrode deformation is not significant on the quality of the welded joint, but when the deformation reaches a certain level, it will

cause uneven forces on both sides of the workpiece, resulting in poor welding quality and even workpiece breakage. In a large number of practices found that even if each welding machine uses the same welding process for production, the quality of the welded joint is still unstable due to the wear and tear of the electrode resulting in a variety of electrode cap shapes. Once quality problems are found in production, it often requires a large number of multi-level sampling experiments to find the root cause of the problem, and when it is not possible to improve the quality of welding, the welding strength will be improved by increasing the number of joints, which will cause an increase in costs<sup>[4,5]</sup>.

Therefore, the use of computer data processing, image processing technology and pattern recognition technology to carry out electrode failure inspection and life analysis, so as to further control the spot welding quality has become an urgent global technical problem in the welding field. Therefore, if the wear degree of the electrode can be observed, it has important research significance to ensure the quality of the solder joint. At the same time, it can reduce the number of electrode grinding and the frequency of replacing the electrode head, which greatly improves the production efficiency of the factory and has important economic value for the production of car body.

## 1.2 Research status of wear and service life of electrodes

The service life of spot welding electrode refers to the number of qualified welding spots that can be welded by a pair of electrodes under the condition of continuous welding. The number of qualified solder joints is specified in the relevant standards of some national resistance welding. The LWS-P7903 standard of Japan stipulates that the service life of electrode refers to the number of solder joints corresponding to the strength of solder joints reduced to 70% of the standard strength under the condition of continuous welding. Because the strength of spot welded joint is approximately proportional to the nugget diameter, nugget diameter is generally used to replace the strength of spot welded joint as the evaluation standard of electrode life. Let the thickness of the weld plate be  $\delta$ , then the standard nucleus diameter specified in the

LWS-P7903 standard is  $d = 5\sqrt{\delta}$ , then when the nucleus diameter drops to the critical value  $d' = 3.5\sqrt{\delta}$ , the number of corresponding weld joints is the life of the electrode [2]. Similarly, according to EU standard iso82166:2003, the electrode shall be considered to have reached its life when the welds being produced have a weld diameter, as indicated in a peel test, of less than  $3.5\sqrt{t}$  (where  $t$  is the sheet thickness in mm) for three welds in a test sample of five consecutive welds.

International researchers have focused more on improving the materials of spot welding electrodes to extend the life of electrodes. The current passed through the electrode head during spot welding is thousands to tens of thousands of amperes, and the pressure is 9.81Mpa~49.1Mpa, and the instantaneous temperature rise reaches

60~90°C, so the electrode is required to have good electrical conductivity, thermal conductivity, thermal hardness and high corrosion resistance. Spot welding electrodes are generally made of copper alloy materials, foreign electrode material standards are generally divided into two groups, a group of Cu-based alloys, containing Cd, Ag, Cr, Zr, Be, Co, Ni, Ti and other different alloy components. The other group is powder metallurgy with different proportion of Cu, W, Mo and other components. In order to improve the performance of copper alloy electrodes, strengthening methods for electrode materials have been developed, and there are generally four strengthening methods, namely, cold working strengthening, aging precipitation strengthening, solid-melt strengthening, and dispersion strengthening<sup>[6]</sup>.

Welding workers have done a lot of preliminary work on the service life and burning loss of spot welding electrode, and studied the effects of electrode design (including copper alloy, coating and contour shape), thin plate and electrode surface condition. Michael A<sup>[7]</sup> plated a layer of nickel on the surface of copper electrode to improve the anti burning performance of the electrode. The research results show that different electrode surface roughness and nickel coating thickness will affect its effect on improving electrode life; It is also pointed out that no matter what kind of workpiece cleaning method is adopted, the adhesion between electrode and workpiece is inevitable when welding with electrode without coating. Patrick<sup>[8]</sup> studied the influence of workpiece surface treatment methods on electrode life. The research results show that the burning loss of electrode is closely related to the thickness of oxide film on the surface of aluminum alloy. The thickness of oxide film on the surface of aluminum alloy workpiece is different, and the burning loss mechanism of electrode is also different, which affects the life of electrode. Rinsei Ikeda<sup>[9]</sup> reduced the electrode burning loss by coating a modified coating on the aluminum alloy plate. The results show that the thickness of the modified coating has a significant impact on the electrode burning loss, and the optimal coating thickness is determined through the test; At the same time, they further found that a layer of MgO would be formed on the surface of the electrode head during continuous welding, and the distribution of MgO affected the distribution of current density, thus affecting the location of nugget formation. Mallya<sup>[10]</sup> studied the effects of different aluminum plate cleaning processes on electrode burning loss and electrode life. Ashton et al<sup>[11]</sup>. Proposed the method of arc cleaning to improve the burning loss of electrode. This method can improve the service life of electrode, but the process is complex and is not conducive to the automation of production process. Holliday et al.<sup>[12]</sup> studied the interaction of plastic deformation, alloying and wear on electrode failure. The research shows that these failure mechanisms promote each other and accelerate the failure of electrode. F. Lu and P. Dong established the evaluation model of electrode end face diameter of resistance spot welding. It was found that for cone electrode, with the increase of cone angle, the expansion rate of electrode end face diameter was reduced, but the axial wear of electrode was increased<sup>[13]</sup>. In recent years some research results have been performed to evaluate electrode life and determine electrode wear degree by experimental method and finite element methods<sup>[14-18]</sup>. Li et al<sup>[19]</sup> developed a new two-stage, sliding-level experiment design and an analysis procedure to study the effect of process fault conditions including electrode wear on



weld quality, and suggested an optimal weld lobe to minimize their effects. Based on the above results, Li proposed modelling and on-line estimation of electrode wear<sup>[20]</sup>.

It can be seen that prolonging the electrode life is of great importance for the production of automotive industry. And most of the previous workers focus on changing the material of electrode and the applied processing conditions to improve the electrode life, but there are few observations on the electrode wear degree, therefore, this paper tries to use the use of digital image processing technology to extract the electrode imprint information, study its relationship with the number of spot welding, and establish the prediction model of the electrode wear degree based on the pattern recognition technology, using the surface image of electrode imprint as the information source.

### **1.3 Development and application of digital image processing**

#### **1.3.1 Development of digital image processing**

Digital image processing is a technology and method that uses computers to process images. Image processing first appeared in the 1950s, when computers had developed to a certain level and people began to use computers to process graphics and image information. Digital image processing as a discipline formed in the early 1960s. Jet Propulsion in the United States in 1964The lab (JPL) successfully mapped the surface of the moon using image processing technology on thousands of lunar photographs returned by Space Explorer Wanderer 7, promoting the birth of the discipline of digital image processing. In 1972, Housfield, an engineer from EMI, UK, invented the X-ray computed tomography (CT) device for skull diagnosis. The basic method of CT is to reconstruct the cross-section image by computer processing according to the projection of human head cross-section, which is called image reconstruction. With the further development of image processing technology, since the mid-1970s, with the rapid development of computer technology, artificial intelligence and thinking science, digital image processing has been developed to a higher and deeper level. By the 1980s, image processing technology has entered a period of popularization. The 1990s has entered a practical period of image technology. Image processing technology in the 21st century is mainly to develop in an all-round way in the direction of high resolution, high speed, stereo and intelligence<sup>[21]</sup>.

#### **1.3.2 Application status of digital image processing**

Digital image processing is a processing technology with strong universality, flexible processing methods, high precision and reliable information preservation and transmission because it is easy to realize nonlinear processing and variable processing programs and processing parameters. It is mainly used for image transformation, measurement, pattern recognition, simulation and image generation. At present, with the rapid development of computer architecture and algorithms, image processing technology has been widely used in many scientific and engineering fields, such as

remote sensing, space observation, communication, criminal investigation, image medicine and many industrial fields. Johannes ruisz<sup>[22]</sup> collected the surface image of solder joint online, and judged the splashing of solder joint by calculating the ratio of elliptical area to perimeter as the threshold value. Abdulhadi. A et al.<sup>[23]</sup> applied image processing technology to evaluate the quality of resistance spot welding electrode cap, and pointed out that the state of electrode cap is an important factor determining the welding quality. Liu Tao et al.<sup>[24]</sup> studied the statistical method of wheat seedling density based on image processing technology. Otsu method is used to separate wheat seedling image from background, and the calculation accuracy of seedling density can reach 92%. Bhagavathi S. I et al.<sup>[25]</sup> studied the automatic recognition and counting system of human blood red and white blood cells based on image technology and used for disease diagnosis. Mohan et al.<sup>[26]</sup> invented an object size recognition system based on image processing technology, which can recognize the size of objects in the state of mutual contact or even overlap. Wang Z et al.<sup>[27]</sup> studied the real-time detection system of steel ball surface defects based on machine vision. Lee<sup>[28]</sup> used digital image processing technology to measure the real-time displacement of the bridge, which is an innovative measurement technology with high efficiency, low cost and easy implementation, but still maintains the advantages of high-resolution dynamic tracking. Marco<sup>[29]</sup> proposed a cardiac cavity segmentation method based on neural network, which compares and analyzes the simplified image multilayer perceptron, trains and creates a universal suitable image, and classifies the results by using image processing technology. This method can detect the cardiac cavity at the edge of the image sequence. To sum up, image processing technology has been widely used in all walks of life. The electrode surface image contains rich information, and with the higher and higher requirements for solder joint quality in actual production, this information becomes more and more important.

#### **1.4 Development and application of pattern recognition**

Pattern recognition was born in the 1920s. Sergios and others defined pattern recognition as the processing and analysis of various forms (numerical, literal and logical) information representing things or phenomena. It is a process of describing, identifying, classifying and explaining things or phenomena. It is an important part of artificial intelligence and Information Science<sup>[30]</sup>. With the emergence of computers in the 1940s and the rise of artificial intelligence in the 1950s, pattern recognition developed rapidly into a discipline in the early 1960s. The research content of this discipline is to enable machines to do things that only human beings can do before through learning, and have some of the abilities that human beings have to describe, analyze and judge various things or phenomena. The theories and methods studied by pattern recognition have been widely valued and applied in many fields such as industry, agriculture, scientific research, national defense, meteorology, astronomy and biomedicine, promoted the development of artificial intelligence technology, signal processing, image processing, computer vision and multimedia technology, and expanded the application field of computer. For decades, pattern recognition has been

applied in many fields, and a large number of research results have been obtained. At present, pattern recognition technology has been successfully applied to weather forecasting, satellite aerial image interpretation, industrial product detection, character recognition, fingerprint recognition, speech recognition, license plate recognition, medical image analysis and other fields, and has made considerable progress in the application research of face recognition, automatic text classification, multimedia data mining, fault diagnosis of systems, weapon guidance finding systems, automatic automobile driving systems and other fields. Different theories and methods of pattern recognition can be used for different research objects and purposes. At present, the mainstream technical methods include statistical pattern recognition, structural pattern recognition, fuzzy pattern recognition and neural network pattern recognition. Statistical pattern recognition is also called decision theory recognition. It is a statistical classification method of patterns inspired by decision theory in mathematics, that is, a method of pattern recognition combined with Bayesian decision system of statistical probability theory. It is a classical statistical classification method established in the long-term development of pattern recognition. Its theory is relatively perfect, and now it has a complete system. The principle of this method is to extract a set of statistical features and define them in a feature space. This space contains all feature vectors. The feature space is divided by using the principle of statistical decision. If the distribution of samples in the feature space conforms to a simple topology and the probability distribution of each sample is known, It is possible to recognize the characteristics of the different objects. Structural pattern recognition is also called syntactic pattern recognition. In many cases, for complex objects, only some numerical features can not fully describe them. At this time, syntactic pattern recognition technology can be used. Syntactic pattern recognition technology is to decompose the research object into several basic units, which are called primitives. These primitives and their structural relations are used to describe the object, and these primitives and their structural relations can be represented by a graph or a string, and then syntactic analysis is carried out by using formal language theory, According to whether it conforms to a certain kind of grammar, its specific category is determined. Syntactic pattern recognition is an advanced technology widely used in pattern recognition. It is suitable for the description and analysis of complex images, and is widely used in power system and automation. Fuzzy pattern recognition introduces the concept of fuzzy set in the process of pattern recognition, and uses the theory and method of fuzzy mathematics to solve the problem of pattern recognition. It is suitable for the situation where the classification and recognition object itself or the recognition result is required to be fuzzy. As a measure of the similarity between sample and template, membership function can reflect the overall and main characteristics. Its fuzzy mode has a considerable degree of anti-interference and distortion, which can allow samples to have a certain degree of interference and distortion, but it is very difficult to establish an accurate and reasonable membership function. At present, some scholars have studied introducing fuzzy pattern recognition into neural network method to form fuzzy neural network recognition system. Neural network method is a nonlinear dynamic system composed of a large number of simple basic units neurons. The structure and function of each neuron are

relatively simple, but the system composed of it can be very complex. It has some characteristics of human brain and has strong abilities in self-learning, self-organization, association and fault tolerance. It can be used for association For identification and decision-making, all calculations are processed in parallel<sup>[31]</sup>. In the aspect of pattern recognition, the feature significantly different from the above methods is that the trained neural network can complete the pattern feature extraction and classification recognition together, which overcomes the limitations of traditional statistical pattern recognition. In the long-term development of pattern recognition theories and methods, statistical pattern recognition has always occupied a dominant position and is the most widely used pattern recognition method. However, due to the diversity and complexity of practical problems, the existing theories and methods of statistical pattern recognition are still far from the actual requirements, and some fundamental problems need to be further studied and solved.

In 1995, Cortes and Vapnik proposed a pattern recognition method based on statistical learning theory, namely support vector machine. It is based on VC dimension theory and structural risk minimization principle of statistical learning theory. According to the limited sample information, the complexity of the model (i.e. learning accuracy for specific training samples) and learning ability (that is, the ability to identify any sample without error) to find the best compromise in order to obtain the best generalization ability Compared with neural network, support vector machine can solve machine learning problems in the case of small samples, improve generalization performance, solve high-dimensional and nonlinear problems, avoid neural network structure selection and local minimum points, and can be extended to other machine learning problems such as function fitting<sup>[32,33]</sup>. Support vector machines solve many problems that cannot be solved by neural networks, bringing an emerging force to problems such as pattern recognition and promoting their widespread use in various areas. Therefore, in this paper, support vector machines will be used to build a prediction model to achieve the evaluation of the wear degree of electrode heads.

## CHAPTER 2

### INTRODUCTION OF MATLAB

Digital image processing is a comprehensive science integrating computer science, optics, mathematics, physics and other disciplines. With the development of computer science, digital image processing technology has made great progress and shown strong vitality. It has made a large number of applications in many fields and promoted the development of society.

As the most widely used software in the field of mathematics, matlab integrates the functions and functions of image processing, and has become a leader in dealing with digital image problems. Its outstanding computing power and simple drawing ability can effectively transform and operate digital images. Based on MATLAB, this paper will explore the operation and implementation of some contents and methods of digital image processing, and analyze the electrode imprint image in resistance spot welding combined with the actual production scene, so as to classify and predict the electrode life.

#### 2.1 GENERAL

The basic data structure in matlab is the array, an ordered set of real or complex elements. This object is naturally suited to the representation of images, real-valued, ordered sets of color or intensity data. matlab stores most images as two-dimensional arrays (i.e., matrices), in which every component of the network compares to a solitary pixel in the showed image. (Pixel is gotten from picture component and as a rule, signifies a solitary speck on a PC show.) For instance, an image made out of 100 rows and 200 columns of various hues would be put away in matlab as a 100x200 matrix. A few images, for example, RGB, require a three-dimensional array, where the primary plane in the third dimension represents the red pixel intensities, the second plane represents the green pixel intensities, and the third plane represents the blue pixel intensities.

#### 2.2 IMAGES IN MATLAB

For the most part, clients manage three kinds of image, consequently three distinct grids. High contrast or twofold image matrix comprises of just zero and one, one being the brighter portion and zero being the dark part. For the most part, images are 8bit and comparing the image matrix is 256x256. The grayscale image is likewise a 2-dimensional lattice with every component esteem differing from 0 to 256. Like the grayscale image, the RGB image can be indicated by the network with every pixel esteem shifting from 0 to 256. In the event of the RGB image, three separate networks for every red, green and blue segment cover to frame an RGB image of 256x256x3 dimension. Since we are present all around familiar with the image as a matrix, now

any numerical activities can be performed on an image that should be possible with a matrix .

Some images, like true color images, represent images employing a three-dimensional array. In true color images, the primary plane within the dimension represents the red pixel intensities, the second plane represents the green pixel intensities, and the third plane represents the blue pixel intensities. This convention makes operating with images in matlab almost like operating with the other form of numeric knowledge and makes the complete power of matlab accessible for image processing applications<sup>[34]</sup>.

## **2.3 INTRODUCTION TO IMAGE PROCESSING TOOLBOX**

Image Processing Toolbox is a gathering of capacities that expand the ability of the matlab numeric registering condition. The toolbox supports a wide range of image processing operations, including:

1. Geometric operations
2. Neighborhood and block operations
3. Linear filtering and filter design
4. Transforms
5. Image analysis and enhancement
6. Binary image operations
7. The region of interest operation

A large number of the tool compartment capacities are matlab M-records, the arrangement of matlab articulations that execute particular picture handling calculations. You can see the matlab code for these capacities utilizing the announcement.

You can broaden the capacities of the Image Processing Toolbox by composing your own particular M-files, or by utilizing the toolkit in the blend with different tool compartments, for example, the Signal Processing Toolbox and the Wavelet Toolbox. You can do perform more confused activities on images.

## **2.4 VISUALIZATION OF IMAGES IN MATLAB**

Read and Display an Image:

We can read standard image documents by utilizing the `imread` function. The sort of information returned by `imread` relies up on the kind of image you are reading. For example, read `image2.jpg` by typing:

```
A = imread('image2.jpg');
```

which will store `image2.jpg` in a matrix named `A`. Presently show the image utilizing the `imshow` function. For example, type:

```
imshow(A); [34]
```

**Direct visualization of images in MATLAB**

Color space conversions (e.g. between RGB, HSV, L\*a\*b\*, and so on) Object grouping and data collection, Filtering and fast convolution, Fourier analysis of images, Image arithmetic, Morphological operations, and many others.

Image Processing Toolbox gives a far-reaching set of reference-standard calculations and work process applications for image preparing, examination, perception, and calculation improvement. You can perform image division, picture improvement, commotion decrease, geometric changes, image enlistment, and 3D image handling.

Image Processing Toolbox applications let you computerize regular picture handling work processes. You can intuitively fragment image information, look at image enlistment methods, and bunch process vast datasets. Perception capacities and applications let you investigate images, 3D volumes, and recordings; modify differentiate; make histograms; and control areas of premium (ROIs).

You can quicken your calculations by running them on multicore processors and GPUs. Numerous toolkit capacities bolster C/C++ code age for work area prototyping and implanted vision system deployment.

Certain Image Processing Toolbox functions have been empowered to create C code utilizing matlab Coder. To utilize code generation with image processing functions, take after these steps:

- Write your matlab function or application as you would ordinarily, utilizing functions from the Image Processing Toolbox.

- Add the codegen compiler mandate to your matlab code.

- Open the matlab Coder application, create a project, and add your file to the project. Once in matlab Coder, you can check the preparation of your code for code generation. For a case, your code may contain functions that are not empowered for code generation.

- Make any adjustments required for code generation.

- Generate code by clicking Generate on the Generate Code page of the matlab Coder application. You can create a MEX file, a mutual library, a dynamic library, or an executable.

Regardless of whether you tended to all preparation issues distinguished by matlab Coder, you may still, experience constructs issues. readiness check only looks at function dependencies.

When you endeavor to create code, matlab Coder may find coding designs that are not bolstered for code generation. View the error report and alter your matlab code until the point when you get a successful build. To produce code from matlab code that contains image processing functions, you should have the matlab Coder programming.

When working with generated code, take note of the accompanying: For some Image Processing Toolbox functions, code generation depends on a precompiled, platform-specific shared library<sup>[35]</sup>.

## **2.5 IMAGE PROCESSING TOOLBOX PRODUCT DESCRIPTION**

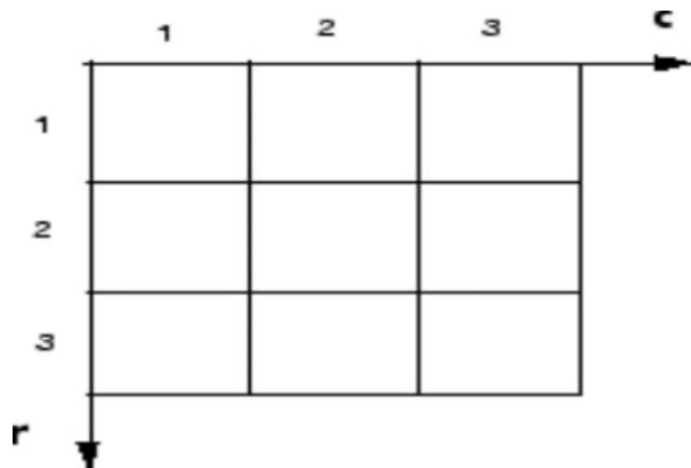
Image Processing Toolbox provides a comprehensive set of reference-standard algorithms and workflow apps for image process, analysis, image, and algorithm

development<sup>[36]</sup>. You'll be able to perform image segmentation, image improvement, noise reduction, geometric transformations, image registration, and 3D image process<sup>[35]</sup>.

Image Processing Toolbox apps allow you to automatize common image process workflows. You'll be able to interactively section image information, compare image registration techniques, and batch-process giant datasets . Image functions and apps allow you to explore pictures, 3Dvolumes, and videos; modify contrast; produce histograms, and manipulate regions of interest (ROIs).You can accelerate your algorithms by running them on multicore processors and GPUs.Many tool box functions support C/C++ code generation for desktop prototyping and embedded Vision system readying.

## 2.6 PIXEL INDICES

Often, the foremost convenient technique for expressing locations in an image is to use pixel indices. The image is treated as a grid of distinct elements, ordered from prime to bottom, and left to right, as illustrated by the following figure<sup>[34]</sup>.



**Figure 2.6.1 Pixel Induces**

For pixel indices, the row will increase downward, whereas the column will increase to the proper. Pixel indices are integer values and vary from one to the length of the row or column.

There is a one-to-one correspondence between pixel indices and subscripts for the first 2 matrix dimensions in matlab. As an example, the data for the pixel in the fifth row, the second is stored in the matrix element (5, 2). You use normal matlab matrix subscription to access values of individual pixels. As an example, the MATLAB code:

```
I (2, 15)
```

Returns the value of the pixel at row 2, column 15 of the image I. Similarly, the matlab code

```
RGB (2, 15)
```



Returns the R, G, B values of the pixel at row 2, column 15 of the image RGB. The correspondence between pixel indices and subscripts for the first 2 matrix dimensions in matlab makes the relationship between an image's data matrix the way the image is displayed easy to understand.

## **2.7 IMAGE COORDINATE SYSTEMS**

Matlab stores most images as two-dimensional arrays (i.e., matrices), during which every component of the matrix corresponds to a single pixel in the displayed image. To access locations in images, the Image Processing Toolbox uses many different image coordinate systems as conventions for representing

Images as arrays.

- “Pixel Indices” as a result of images are arrays; you'll be able to use standard matlab indexing.
- “Spatial Coordinates” you'll be able to contemplate locations in images as positions on a plane using Cartesian coordinates.

### **2.7.1 SPATIAL COORDINATES**

Another technique for expressing locations in an image is to use a system of unceasingly variable coordinates instead of distinct indices. This permits you to think about an image as covering a sq. patch, as an example. In a spatial coordinate system like this, locations in an image are positions on a plane, and that they are delineate in terms of x and y (not row and column as within the picture element categorization system). From this Cartesian perspective, an (x, y) location like (3.2, 5.3) is meaningful and is distinct from pixel (5, 3).

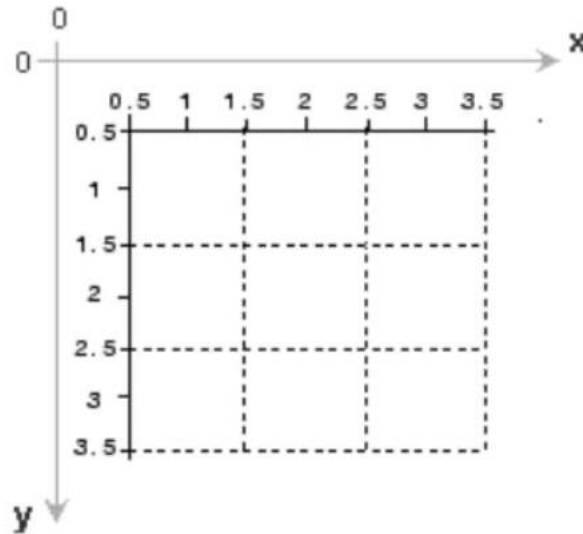
The Image Processing Toolbox defines 2 kinds of spatial coordinate systems:

- Intrinsic Coordinates A spatial coordinate system that corresponds to pixel indices.
  - World coordinates a spatial coordinate system that relates the image to some other coordinate space<sup>[35]</sup>.

### **2.7.2 INTRINSIC COORDINATES**

By default, the toolbox uses a spatial coordinate system for an image that corresponds to the image's pixel indices. It referred to as the intrinsic coordinate system and is illustrated within the following figure.

Notice that y will increase downward as a result of this orientation is in step with the manner within which digital images are sometimes viewed .



**Figure 2.7.1 Intrinsic Coordinates**

### 2.7.3 INTRINSIC COORDINATES SYSTEM

The intrinsic coordinates  $(x,y)$  of the center point of any pixel are identical to the column and row indices for that pixel. As an example, the center point of the pixel in row five, column 3 has spatial coordinates  $x = 3.0$ ,  $y = 5.0$ . This correspondence simplifies many toolbox functions considerably. Be aware, however, that the order of coordinate specification  $(3.0, 5.0)$  is reversed in intrinsic coordinates relative to pixel indices  $(5, 3)$ .

Many functions primarily work with spatial coordinates rather than pixel indices, however as long as you maintain the default spatial coordinate system (intrinsic coordinates), you'll be able to specify locations in terms of their columns ( $x$ ) and rows ( $y$ ).

When observing the intrinsic coordinate system, note that the higher left corner of the image is found at  $(0.5, 0.5)$ , not at  $(0, 0)$ , and the lower right corner of the image is located at  $(\text{numCols} + 0.5, \text{numRows} + 0.5)$ , wherever  $\text{numCols}$  and  $\text{numRows}$  are the numbers of rows and columns within the image. In contrast, the upper left pixel is pixel  $(1,1)$  and the lower right pixel is pixel  $(\text{numRows}, \text{numCols})$ . The middle of the upper left pixel is  $(1.0, 1.0)$  and the center of the lower right pixel is  $(\text{numCols}, \text{numRows})$ . In fact, the middle coordinates of every pixel are integer valued. The middle of the pixel with indices  $(r, c)$  — wherever  $r$  and  $c$  are integers by definition — falls at the point  $x = c$ ,  $y = r$  within the intrinsic coordinate system.

### 2.7.4 WORLD COORDINATES

In some things, you may need to use a world coordinate system (also known as a non default spatial coordinate system). As an example, you may shift the origin by specifying that upper left corner of an image is the point  $(19.0, 7.5)$ , rather than  $(0.5, 0.5)$ . Or, you may need to specify a coordinate system during which each constituent covers

a 5-by-5 meter patch on the ground. There are many ways that to define a world coordinates system.

### **2.7.5 DEFINE WORLD COORDINATES USING XDATA AND YDATA PROPERTIES**

To define a world coordinate system for an image, specify the XData and YData image properties for the image. The XData and YData image properties are two-element vectors that management the vary of coordinates spanned by the image. once you do that, the matlab axis coordinates become identical to the world (nondefault) coordinates. If you are doing not specify XData and YData, the axes coordinates are identical to the intrinsic coordinates of the image. By default, for an image A, XData is [1 size(A,2)], and YData is [1 size(A,1)]. With this default, the world coordinate system and intrinsic coordinate system coincide perfectly. (Another way to define a world coordinate system is to use spatial referencing.

As an example, if A is a 100 row by 200 column images, the default XData is [1 200], and the default YData is [1 100]. The values in these vectors are actually the coordinates for the centre points of the first and last pixels (not the pixel edges), therefore the actual coordinate vary spanned is slightly larger. for example, if XData is [1 200], the interval in X spanned by the image is [0.5 200.5].

It's additionally attainable to set XData or YData such that the x-axis or y-axis is reversed. You'd do that by inserting the larger worth initial. (For example, set the YData to [1000 1].) This is a common technique to use with geospatial data.

Several toolbox functions accept this XData and YData as arguments and return coordinates within the world coordinate system: bwselect, imcrop, impixel, roipoly, and imtransform .

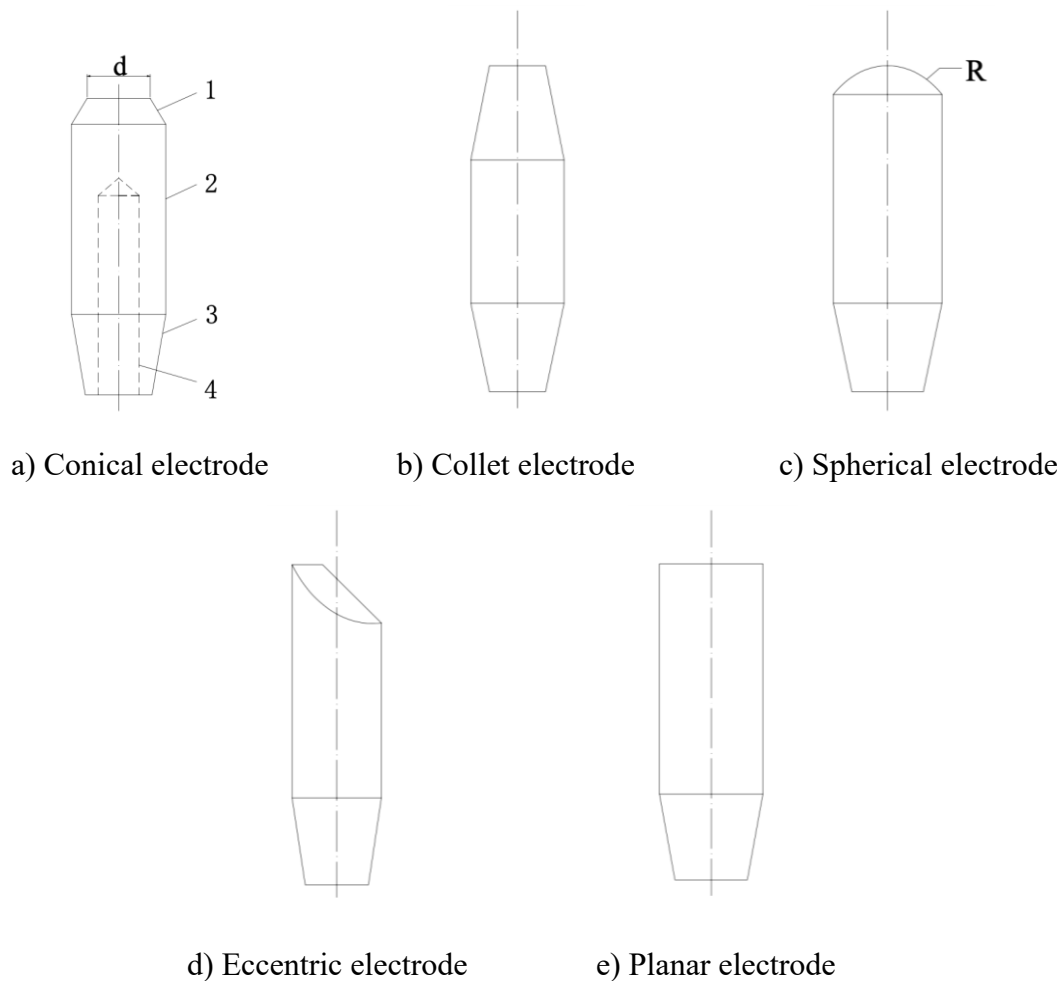
## CHAPTER 3

### ELECTRODE IMAGE PROCESSING AND FEATURE EXTRACTION

#### 3.1 Wear mechanism of electrode head

##### 3.1.1 Shape of electrode head

The spot welding electrode is composed of tip, main body, end and cooling water hole. The standard electrode (i.e. straight electrode) has five forms, as shown in Figure 3.1.1. The tip of the electrode is in direct contact with the surface of the high-temperature weldment. In the welding production, it repeatedly bears high temperature and high pressure. Therefore, the adhesion, alloying and wear of the electrode generally occur at the tip of the electrode. The tip diameter  $d$  of the electrode and the spherical radius  $R$  of the spherical electrode depend on the thickness of the workpiece and the required nugget size. In the actual production, the appropriate electrode shape is generally selected according to the welding materials and application requirements.



**Figure 3.1.1 Standard electrode shape**

### 3.1.2 The role of resistance spot welding electrodes

With the rapid development of thin sheet materials, their rust resistance is getting better and better, but the new materials have poor weldability, which can affect the electrode's spot welding life. The role of electrodes in the spot welding process are:

1. Transmission of current to the welding area.
2. Transferring pressure.
3. Rapid release of heat from the weld surface and the weld area.
4. Controlling the thermal balance of the spot welding process.
5. Positioning the workpiece to the proper location.

Spot welding process electrode will occur upsetting phenomenon, which will lead to the quality of the spot welded joints decline, workers tend to repair the electrode, the main reason for this phenomenon is because the electrode and the surface of the weld occurs adhesion this vicious circle. Therefore the choice of electrode material must be:

1. High temperature hardness, strength and recrystallization temperature.
2. A low tendency to alloy with the weld material.
3. Excellent electrical and thermal conductivity.
4. Excellent processing properties.

### 3.1.3 Wear of electrode head

In the spot welding process, the diameter of the electrode head generally increases with the increase in electrode use time, and the increase in the diameter of the electrode head leads to a decrease in the diameter (or strength) of the formed welded joint due to the decrease in current density during spot welding, thus the criterion for judging the failure of the spot welding electrode is that the electrode fails when the diameter of the welded joint drops to a specified value, which is  $4 \times \sqrt{\delta}$  mm (where  $\delta$  is the thickness of the welded part)<sup>[12]</sup>. The main failure mechanisms are :

1. Plastic deformation

The head of the electrode is plastic deformed by the combined effect of welding pressure and heat, resulting in an increase in the diameter of the electrode head. In fact, when spot welding, if the compressive stress of spot welding is greater than the yield strength at the temperature of the electrode, plastic deformation will occur, because the highest temperature of the electrode head in contact with the weld, so plastic deformation is concentrated in the head of the electrode, this process is imaginatively called "mushrooming".

2. Wear and tear

In the spot welding process, the electrode ends are the most serious area of wear, which generally occurs through electrode adhesion to the workpiece, spattering during spot welding, etc. Electrode wear causes the diameter of the electrode ends to become larger, leading to a reduction in welding current density during spot welding. X.Q. Zhang et al<sup>[37]</sup>. conducted an experimental study of the electrode wear characteristics during welding of galvanized duplex steel DP600 using a servo torch. The results

showed that the early wear stage of the spot welding process produces more alloy products and electrode deformation, which increases the electrode diameter, resulting in a higher electrode wear rate at this stage. Electrode failure is accelerated by pitting growth on the electrode tip surface. As a result of electrode wear, undersized weld cores are formed. Klimov, A.S. et al<sup>[38]</sup>. designed a control system capable of automatically increasing the actual secondary current according to the degree of wear of simple electrodes, based on the principle of current stabilization operation in secondary circuits. When correcting the value of the secondary current according to the degree of wear of the welding electrode, it is possible to control the change in the electrode cross section and the quality of the welded joint. It is determined that the reason for the change in electrode wear during welding may be caused by the change in electrode cross-section resistance.

### 3. Pit corrosion and self-healing

Electrode spot welding under the action of high temperature pressure, the end will form a low melting point alloy layer. When the spatter phenomenon occurs during spot welding, the contact area produces low melting point alloy spatter will leave the electrode surface, leaving an arc pit on the electrode end face, this spatter phenomenon if repeatedly occurs, the arc pit is increasingly formed pit corrosion. Pit erosion will affect the current density of the electrode end surface, resulting in the electrode end surface around the pit erosion easily plastic deformation, so that the electrode end surface diameter becomes larger, affecting the quality of the welded joint. When spot welding galvanized steel, the electrode end pits will be filled with liquid plating metal at high temperature, and the "self-healing" phenomenon. Even if the electrode end face "self-healing" phenomenon, still can not make the end face current density uniform, which will lead to electrode spatter and bonding when spot welding, accelerating the speed of electrode failure.

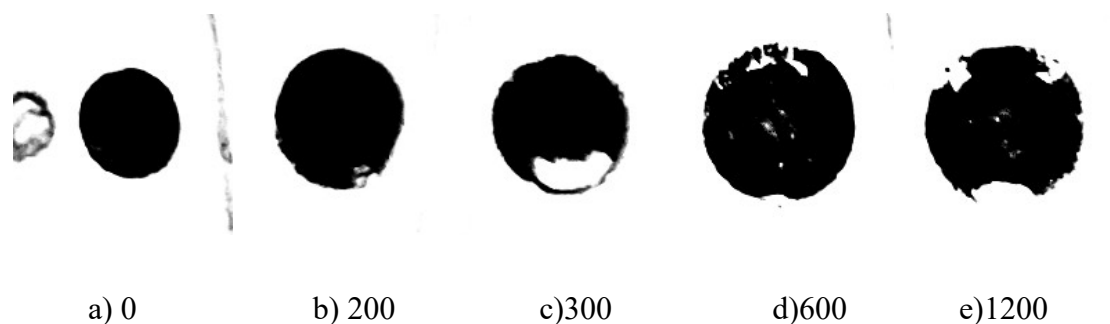
It can be seen that the electrode failure behavior mainly causes the diameter of the electrode head end face in contact with the workpiece to increase, such as the appearance of etching pits and other changes in the end face shape, resulting in the wear of the electrode, affecting the life of the electrode.

## 3.2 Electrode imprint image analysis

With the rapid development of digital image processing technology as well as computer technology, the foundation for a new kind of electrode life judging method has been laid. Combining digital image processing technology and computer technology, we obtain the surface image of electrode imprinting, and extract the features of each region, analyze the correlation between each feature and electrode head life, and explore an image feature recognition method of electrode head wear degree.

Figure 3.2.1 shows the acquired digital images of the electrode surface morphology. Table a, b, c and d show the electrode surface images of the same electrode head in the initial state and at the completion of the 200th, 300th, 600th and 1200th solder joints respectively, which can visually reflect the shape of the electrode. From the figure can be seen, the electrode end surface diameter has a tendency to increase, and in the 300th

point when the electrode embossed part of the region appears blank area, presumably these blank areas for the pit, electrode spot welding at high temperatures and electrical pressure, the end will form a low melting point alloy layer, in the spot welding process to produce spatter phenomenon, the contact area to produce low melting point alloy spatter will leave the electrode surface, leaving arc pits in the electrode end surface. This spatter phenomenon if repeatedly appear, arc pit more and more formed pit corrosion. Pit erosion will affect the current density of the electrode end surface, resulting in the electrode end surface around the pit erosion easily plastic deformation, so that the electrode end surface diameter becomes larger, affecting the quality of the welded joint. Electrode at 600 points, the electrode blank pit corrosion point disappears, representing the electrode surface re-healing phenomenon, but the electrode edge at the indentation blank area, it is presumed that these areas of the electrode ends are subject to greater wear, in the spot welding process, the electrode and workpiece adhesion phenomenon is serious, and spot welding process spatter is also larger, these phenomena will produce wear, so that the electrode failure speed. Until 1200 points when the electrode end surface not only appears pit corrosion again, there are also other blank wear area, the larger the electrode end surface blank area, the more serious the electrode pit corrosion and flaking phenomenon, the faster the electrode failure.



**Figure 3.2.1 Electrode imprint picture**

It can be seen that there is a large amount of information in the surface image of electrode imprinting, which directly reflects the electrode life, and this information can also infer the condition of the solder joint quality. In order to further study accurately, the surface image of electrode imprinting will be processed in order to better analyze the correlation between electrode imprinting and electrode life.

Because of the inevitable interference of various factors during the electrode imprinting surface image acquisition process, the image needs to be pre-processed before the extraction of the electrode imprinting surface image feature parameters. Firstly, the image is pre-processed with grayscale, grayscale histogram equalization and noise removal. Secondly, in order to obtain the information of feature regions in the image, edge detection and segmentation are needed, which is also the basic prerequisite for image recognition.

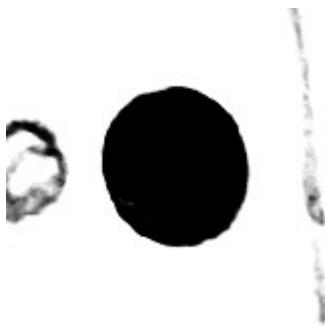
### 3.3 Image pre-processing

#### 3.3.1 Image normalization

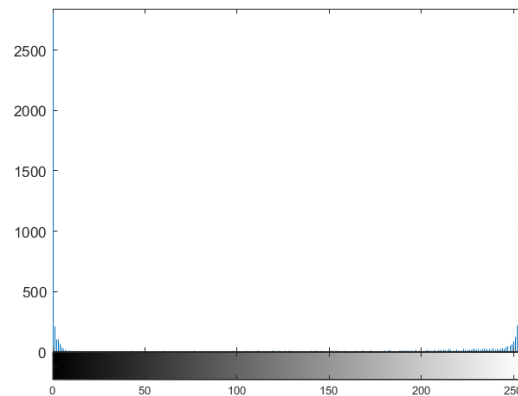
Photoshop was used to cut the original image into an image with electrode embossing located in the center part of the photo and uniform size (160×160 pixel). Since MATLAB has format requirements for image processing, many image processing jobs have specific requirements for image types, such as to filter an index image, it must first be converted to true color image, otherwise the result is meaningless. Grayscale images are higher in clarity than color images in low light conditions compared to color images, so in order to adapt to different light conditions, it is necessary to convert color images into grayscale images, which not only meets the processing needs, but also reduces the amount of information operations. So, first of all, grayscale processing will be performed on the images with the same size cut. The grayscale conversion and histogram results are shown in Figure 3.3.1a), b).

Programming:

```
clear all;  
close all;  
clc;  
ori = imread('0.png');  
gray = rgb2gray(ori);  
figure(1);  
subplot(2,3,1),imshow(gray),title('Gray image');  
subplot(2,3,[2 3]),imhist(gray);title('Gray histogram');
```



a) Grayscale image



b) Grayscale histogram

**Figure 3.3.1 Image gray processing**

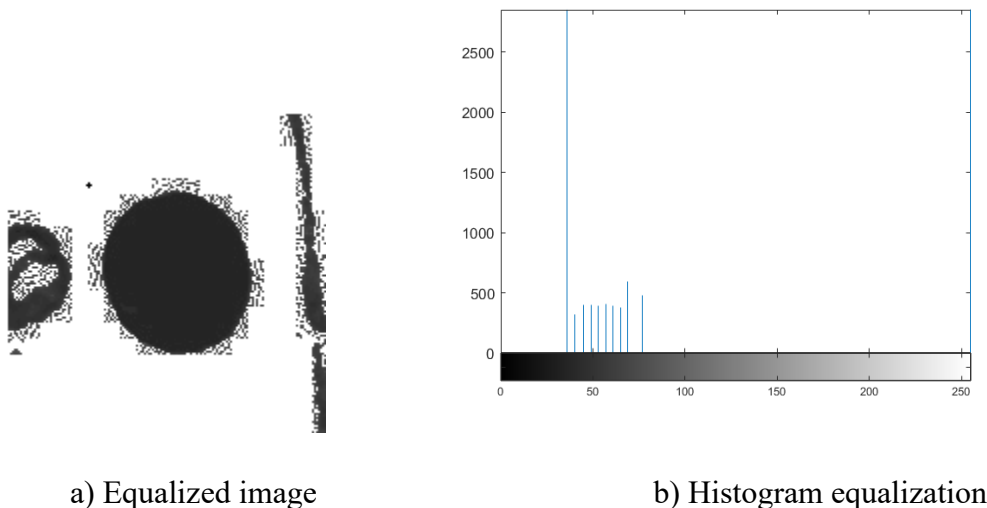
Histogram equalization is a function that transforms the histogram of the original image into a uniform histogram, and then trims the original image according to the uniform histogram. The trimmed image has the same frequency of occurrence for each



gray level, so that the dynamic range of the gray level of the image becomes larger and the overall contrast of the image is enhanced. Each gray level of the trimmed image has the same frequency of occurrence, i.e., the gray levels have a uniform probability distribution. The advantage of this method is that it can automatically enhance the contrast of the whole image, and its disadvantage is that the local enhancement effect is not easy to control. Figure 3.3.2(a) shows the equalized image, and Figure 3.3.2(b) shows the histogram of the equalized image. As seen in the figure, the image contrast increases and the details become clearer. However, the features of its image histogram are weakened, and the frequencies of each gray level appear close to each other, which is not conducive to the segmentation of the image. So the later experiments will not equalize the image.

Programming:

```
clear all;
close all;
clc;
ori = imread('0.png');
gray = rgb2gray(ori);
eqgray=histeq(gray);
imshow(eqgray),title('Gray equalization ');
imhist(eqgray),title('Histogram equalization');
```



**Figure 3.3.2 Image equalization**

### 3.3.2 Image Filter

Noise is all the remaining signals in the image processing except the effective signal, which is not only limited to the distortion and deformation of the image that can be observed by human eyes, some noise may be found only when image processing is performed. Noise can be divided into internal noise and external noise according to the source of its generation; it can be divided into non-smooth noise and smooth noise

according to the statistical point of view; it can be divided into two categories of multiplicative noise and additive noise according to the impact of noise on the signal. In general, the grayscale interval of the electrode imprint image is narrow, and during the image capture process, it will be disturbed and influenced by various noises, such as random noise, quantization noise generated by digitization, etc., thus causing the quality of the captured image to deteriorate. The common noise in general image processing techniques are: quantization noise, additive noise, multiplicative noise. These noises reduce the contrast of image brightness, resulting in blurred image details, which brings inconvenience to image analysis. In order to improve the quality of the image, highlight the features of the target, and filter out interference, denoising must be performed.

Image denoising consists of two aspects: 1) elimination of noise; 2) enhancement of image features. These two goals are contradictory to some extent. Because removing noise means removing the high-frequency part of the image, and the boundary of the image is also the high-frequency part of the image, so while removing noise, it often makes the boundary of the image become blurred. How to solve this contradiction is an important criterion to evaluate how good the image denoising model is. In image processing, the widely used filtering methods are mainly median filtering and mean filtering. We add Gaussian noise and pretzel noise to the existing electrode embossed image, compare the advantages and disadvantages of different filtering methods and select the suitable filtering method.

Programming:

```
clear all;
close all;
clc;
ori = imread('0.png');
gray = rgb2gray(ori);
g_noise = imnoise(gray,'gaussian',0.02); % add Gaussian noise, variance 0.02
figure(1),imshow(g_noise);
s_noise = imnoise(gray,'salt & pepper',0.02); % add salt & pepper noise, density
0.02
figure(2),imshow(s_noise);
imwrite(g_noise,'gaussian noise.png');
imwrite(s_noise,'salt & pepper noise.png');
```



a) Gaussian noise with variance of 0.02      b) Salt and pepper noise with density of 0.02

**Figure 3.3.3 Noise image**

### 3.3.2.1 Median filter

The median filtering method uses a nonlinear smoothing technique to replace all the pixels of an image with the median of the pixels in the neighborhood (i.e., a square interval centered on a pixel). The median filtering has an excellent smoothing effect in dealing with impulse noise, not only to remove the noise, but also to protect the edge signal so that the edge signal is not blurred.

$$g(x, y) = \text{Med}\{f(x - k, y - l), (k, l \in W)\}$$

Where  $f(x, y)$ ,  $g(x, y)$  are the original image and the processed image, respectively,

and  $W$  is the template. The template is generally set to 3\*3 and 5\*5 regions, but of course it can also be set to regions of different shapes, such as rings, bars, crosses, etc.

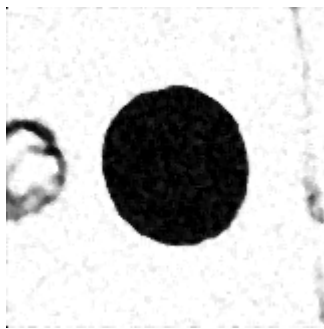
Median filtering can reduce or eliminate the high frequency components in Fourier space, and can also affect the low frequency components. Median filtering relies on two factors, the region of the neighborhood space and the number of pixels associated with the median calculation, to eliminate noise. Usually objects with larger sizes will be preserved unchanged, while objects in regions of bright or dark bands smaller than half of the said filter will be largely filtered out, so the spatial size of the filter must be chosen according to the actual problem.

Programming:

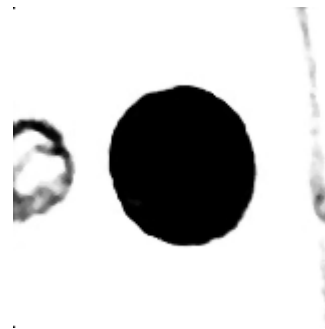
```
clear all;
close all;
clc;
ori = imread('0.png');
gray = rgb2gray(ori);
g_noise = imnoise(gray,'gaussian',0.02); % add Gaussian noise, variance 0.02
figure(1),imshow(g_noise);
s_noise = imnoise(gray,'salt & pepper',0.02); % add salt & pepper noise, density
```

0.02

```
medf_ga = medfilt2(g_noise); % median filtering of Gaussian noise picture
figure,imshow(medf_ga);
medf_sa = medfilt2(s_noise); % median filtering of salt & pepper noise picture
figure,imshow(medf_sa);
imwrite(medf_ga,'medfilter gaussian.png');
imwrite(medf_sa,'medfilter salt & pepper.png');
```



a) Median filter gaussian



b) Median filter salt & pepper

**Figure 3.3.3 Median filter**

### 3.3.2.2 Average Filter

Average filtering is a relatively simple linear filtering method. After average filtering, all pixels of the output image are the average of the input image pixels corresponding to the window, which is actually normalized box filtering to make the pixels smooth and achieve the purpose of noise removal by reducing the influence of prominent pixel points. The essence of average filtering is to remove irrelevant details in the image by blurring the image, ignore the fine details, and get a descriptive image with overall information. Although average filtering has the advantages of fast operation rate, simple method and good suppression of periodic noise, it also has certain drawbacks, i.e., the process of denoising the image may lead to the destruction of the detailed parts of the image, which cannot be preserved intact and blur the original image. The coefficient template of the average filter is shown below:

$$g(x,y) = \sum I(x,y)$$

Where  $g(x,y)$  is the center pixel of the neighborhood, the template is similar to the median filter which can be  $3*3$ ,  $5*5$ ,  $7*7$ , and  $n$  is related to the coefficient template, in the  $3*3$  template,  $n$  is generally set to 9.

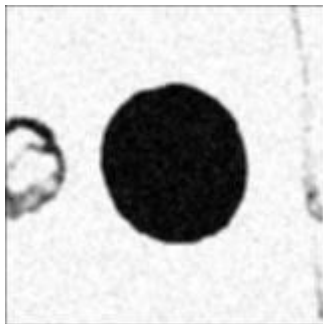
Programming:

```
clear all;
```

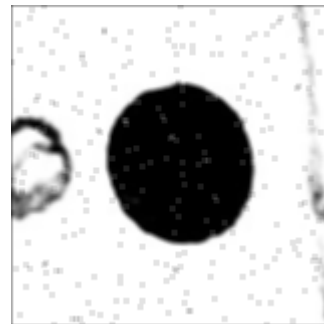
```

close all;
clc;
ori=imread('0.png');
gray=rgb2gray(ori);
g_noise = imnoise(gray,'gaussian',0.02); % add Gaussian noise, variance 0.02
s_noise = imnoise(gray,'salt & pepper',0.02); % add salt & pepper noise, density
0.02
filter=fspecial('average',[3,3]);% definition 3 * 3 filter
avef_s=imfilter(s_noise,filter); % 3*3 average filtering for salt & pepper noise
figure,imshow(avef_s);title('3*3 average filter salt & pepper noise');
avef_g=imfilter(g_noise,filter); % 3*3 average filtering for gaussian noise
figure,imshow(avef_g);title('3*3 Mean filter gaussian noise');
imwrite(avef_g,'avefilter gaussian.png');
imwrite(avef_s,'avefilter salt & pepper.png');

```



a) Average filter gaussian



b) Average filter salt & pepper

**Figure 3.3.4 Average filter**

As shown in the above figure, Gaussian noise and salt and pepper noise with signal-to-noise ratio of 0.02 are added to the original image, and then median filter and average filter are used to denoise the image respectively. The experimental results show that the median filter can eliminate the Gaussian and salt and pepper noise, and retain the information of the original image on the basis of eliminating the noise without blurring the edge of the original image. Although average filtering can make the image smoother, the effect of removing noise is not obvious.

### 3.4 Image Segmentation

Image segmentation is the division of an image into several disjoint regions based on features such as grayscale, color, geometric shape, spatial texture, etc., such that these features show consistency or similarity within the same region, while showing significant differences between regions. In simple terms, it is to separate the target from the background in an image for the next step of processing. Image segmentation is one of the most basic and important areas of image processing and computer vision in the field of low-level vision, it is the basic premise of visual analysis and pattern recognition of images, and the accuracy of segmentation will directly affect the

effectiveness of subsequent work. Currently, the following image segmentation methods are commonly used: threshold-based segmentation, edge-based segmentation, etc.

### 3.4.1 Threshold-based segmentation method

Thresholded image segmentation is one of the most commonly used image segmentation methods, which is particularly suitable for range images where the target and background occupy different gray levels, and it not only greatly compresses the amount of data, but also greatly simplifies the analysis and processing steps. The threshold segmentation method uses a parallel region technique and is commonly applied to image segmentation. The threshold segmentation method is implemented by transforming the image from the input image  $f$  to the output image  $g$  as follows.

$$g(x,y) = \begin{cases} 1, & f(x,y) \geq T \\ 0, & f(x,y) < T \end{cases}$$

The  $g(x,y)$  thus obtained is a binary image. Where  $T$  is the threshold value, for the image element  $g(i,j) = 1$  for objects and  $g(i,j) = 0$  for the image element of the background. The process of implementing this algorithm is as follows:

- 1) converting the color image into a grayscale image.
- 2) calculating a histogram and selecting a threshold based on the histogram.
- 3) Perform image segmentation based on the threshold value.

Based on this principle, we found that if we can determine a suitable threshold value we can segment the image accurately, in short the determination of threshold value is the key of threshold segmentation algorithm, as long as the threshold value is determined, the image can be segmented easily. It has received a lot of attention from many scholars in the past four decades, resulting in hundreds of threshold selection methods<sup>[39-41]</sup>. Although scholars have done a lot of research work on image segmentation, the segmentation algorithms that have been proposed nowadays are generally problem-specific and there is no one general segmentation algorithm that is suitable for all images. That is to say, when dealing with a specific image segmentation problem, a large number of analyses should first be performed on its grayscale image, and a suitable thresholding algorithm should be selected according to the overall characteristics it presents. In this paper, we will try to implement the selection of threshold values using the 2-mode, iterative and OSTU methods.

#### 3.4.1.1 two-mode method

Two-mode method is a simple threshold segmentation method. The two-mode method first converts the original image into a gray image, and then converts the gray image into a gray histogram. The gray histogram is the two-dimensional relationship

between the number of pixels  $n$  of the gray level and the gray level  $I$ . it reflects the statistical characteristics of the gray distribution on an image, which is realized by using the function *imhist* in MATLAB. If the gray histogram shows obvious bimodal shape, the gray level corresponding to the valley bottom between the two-mode is selected as the threshold segmentation.

### 3.4.1.2 Iterative method

The iterative method is based on the idea of approximation, and its steps are as follows:

1) Find the maximum gray value and the minimum gray value of the image, denoted as  $Z_{\max}$  and  $Z_{\min}$ , respectively, so that the initial threshold is

$$T_0 = (Z_{\max} + Z_{\min})/2$$

2) Split the image into foreground and background according to the threshold  $T_0$  and find out the average gray value  $Z_O$  and  $Z_B$  respectively ;

3) Find the new threshold value

$$T_1 = (Z_O + Z_B)/2$$

4) Repeat the above process with the new threshold  $T_1$  instead of  $T_0$ , and iterate until  $T_K$  converges

$$T_K = T_{K+1}$$

### 3.4.1.3 OSTU Method

The maximum interclass variance<sup>[42]</sup> was proposed by the Japanese scholar Otsu in 1979 as an adaptive method for threshold determination.

The optimal threshold value  $t$  for the image should be selected so that the separation between the different classes is best. Firstly, the probability of occurrence of each segmentation characteristic value is obtained from the histogram, and the segmented feature values are divided into two categories by the threshold variable  $t$ . Then, the intra-category variance and inter-category variance of each category are obtained, and

the  $t$  that causes the smallest intra-category variance and the largest inter-category variance is selected as the best threshold value, i.e.

$$\sigma_T^2 = \sigma_B^2 + \sigma_W^2$$

$$\eta(t) = \frac{\sigma_B^2}{\sigma_T^2}$$

$$t = \underset{0 \leq t \leq L-1}{\text{Arg max}} \eta(t)$$

Where  $\sigma_W^2$  is the intra-class variance,  $\sigma_B^2$  is the inter-class variance, and  $\sigma_T^2$  is the overall variance. Since the variance is a measure of the uniformity of the gray distribution, the larger the variance value, the greater the difference between the two parts of the image, when part of the background is misclassified into the target or part of the target is misclassified into the background will lead to a smaller difference between the two parts, so that the segmentation with the largest interclass variance means the smallest probability of misclassification. However, OSTU method is very sensitive to noise and target size, when the size of the target and background ratio is disparate, the criterion function of inter-class variance may show double or multiple peaks, resulting in the global maximum value selected by this method is not necessarily the correct threshold, this method will fail.

Programming:

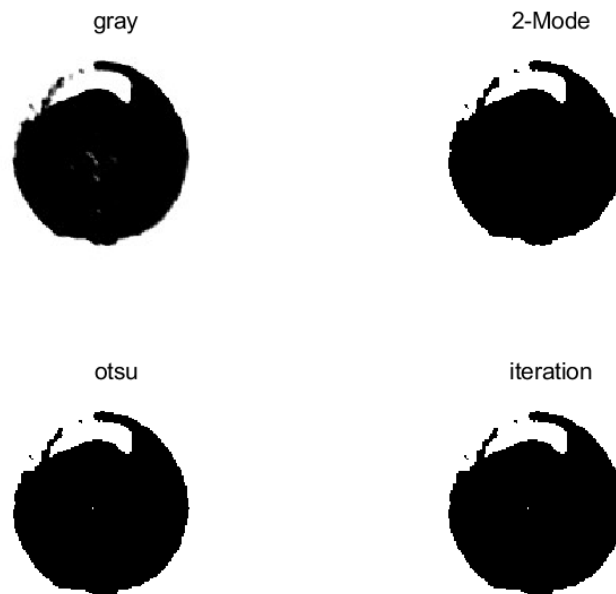
```
clear all;
clc;
close all;
ori=imread('585.png');
gray=rgb2gray(ori);
% 2-mode
figure, imhist(gray);
figure,
subplot(2,2,1),imshow(gray),title('gray');
I=im2bw(gray,150/255);
subplot(2,2,2),imshow(I),title('2-Mode');
% OSTU
level=graythresh(gray);
BW=im2bw(gray,level);
subplot(2,2,3),imshow(BW),title('otsu');
% iterative
gray=double(gray);
```



```

T=(min(gray(:))+max(gray(:)))/2;
done=false;
i=0;
while ~done
r1=find(gray<=T);
r2=find(gray>T);
Tnew=(mean(gray(r1))+mean(gray(r2)))/2;
done=abs(Tnew-T)<1;
T=Tnew;
i=i+1;
end
gray(r1)=0;
gray(r2)=1;
subplot(2,2,4),imshow(gray),title('iteration');
saveas(2,'Image segmentation based on threshold');

```



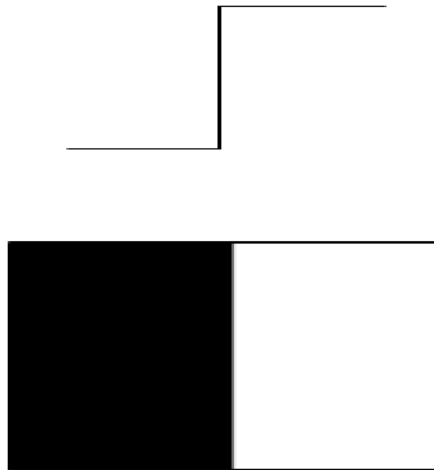
The iterative method and the OTSU method have similar effects and are both automatic threshold selection methods. The design principle is more complicated than the two-mode method, and it is not very good for noisy images, and the iterative does not have a good degree of differentiation for subtle or light-colored lines of the image. Although the two-mode method is simple, but the application range is small, for those images where the peak is not very obvious or the difference in gray value between the pixels on both sides of the target background junction is not very obvious, the effect of processing with the two-mode method is not very obvious. Relatively, among these three methods, the OTSU method is a more general method.

### 3.5 Edge-Based Image Segmentation

#### 3.5.1 Edge of image

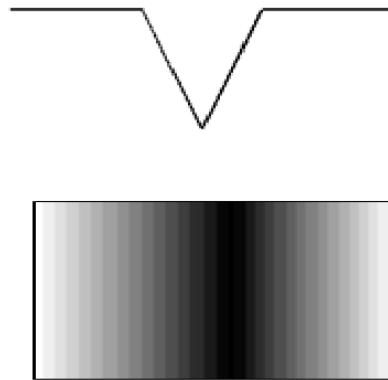
Basic definition of an edge: An edge of an image is actually a collection of pixels, and the location in the image where these pixels are located shows a step change, or roof change, in pixel grayscale. Based on the definition of edge and the shape of the gray change, we can simply group the image edges into two categories, i.e. step-like and roof-like.

1. Stepped edges: Pixel points on this type of edge are located on the vertical steps of the image where the grayscale changes in steps, with good continuity, which is a more ideal digital edge model. As shown in Figure 3.5.1.



**Figure 3.5.1 Stepped edge**

2. Roof edge: This type of image edge is a gently rising and falling edge. As shown in Figure 3.5.2.



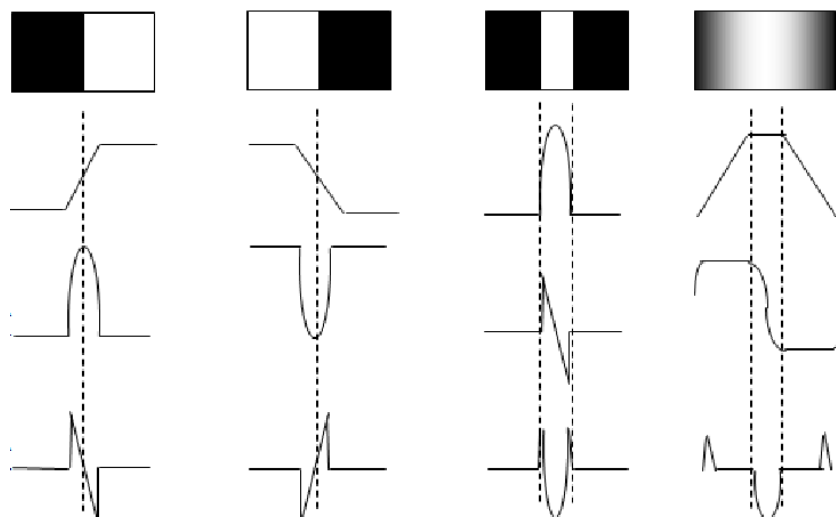
**Figure 3.5.2 Roof edge**

### 3.5.2 Principle of edge detection

The basic idea of image edge detection is to compute the local differential operator of an image.

According to the definition of image edges we know that the edge locations in an image possess a common quality, i.e., the pixel has a large variation in gray level. Then we can determine the location of the edge pixels by examining where their first-order derivatives appear to have a maximum value and where the second-order derivatives appear to have a zero crossing point.

The relationship between image edges and first-order derivatives and second-order derivatives is shown in Figure 3.5.3.



**Figure 3.5.3 Relationship between image edges and first-order derivatives and second-order derivatives**

### 3.5.3 Convolution theory

When processing images, it is common to compute image convolution for the purpose of processing images, that is, traversing all pixels in an image and using small fields of each pixel to generate the final image.

#### 3.5.3.1 Basic theory

In the mathematical definition, we call the method of calculating the infinite integral of two functions, the convolution. We use  $h$  and  $k$  to denote two functions, and convolution is an operator that uses  $h$  and  $k$  to compute a third function by flipping and translating  $h$  and then accumulating the parts that overlap with each other with the  $g$  function. In practical applications and calculations, the convolution algorithm can be used instead of the multiplication operation, because it can be introduced by the convolution theorem that the convolution process in one domain is equivalent to the multiplication operation performed in another domain.

In addition, the convolution operation can also be simplified. It is equivalent to a weighted summation process. Consider the two functions  $h$  and  $k$  that perform the convolution, under the premise that either  $h$  or  $k$  is treated as an indicator function of the interval, and then we can treat the convolution as actually an extended "sliding average".

We usually use a small matrix as the weights of the convolution, the size of which should be the same as the size of the operation area and should be an odd number. We usually call this matrix of weights the convolution kernel, and when performing the convolution process, each element of the kernel is multiplied one by one with the corresponding pixel in the operating region of the image and all the results are summed and reassigned to the central pixel in that region of the image.

For example, suppose  $P$  is a region in the image of size  $3 \times 3$  and the matrix  $K$  acts as a convolution kernel on this region, also of size  $3 \times 3$ . Then the new central pixel value  $P_{22}$  of the region obtained by convolving using  $P$  and  $K$  is expressed as

$$P_{22} = P_{11} \times K_{11} + P_{12} \times K_{12} + P_{13} \times K_{13} + \\ P_{21} \times K_{21} + P_{22} \times K_{22} + P_{23} \times K_{23} + P_{31} \times K_{31} + P_{32} \times K_{32} + P_{33} \times K_{33}$$

Of which:

$$P = \begin{bmatrix} P_{11} & P_{12} & P_{13} \\ P_{21} & P_{22} & P_{23} \\ P_{31} & P_{32} & P_{33} \end{bmatrix} \quad K = \begin{bmatrix} K_{11} & K_{12} & K_{13} \\ K_{21} & K_{22} & K_{23} \\ K_{31} & K_{32} & K_{33} \end{bmatrix}$$

We usually use the convolution coefficients to represent all the constituent elements of the convolution kernel, and the size, order and direction of their values play a crucial role in the image processing process. The convolution kernel we usually use is a 3×3 matrix.

### 3.5.3.2 Edge effect

When using convolution to process images, there are some complex problems that plague us, and the edges of the image are the first of these problems. When we use convolution for image processing, we need to traverse all the regions in the image and let the convolution kernel move around the image in pixel units. However, in the edge region of the image, the convolution kernel does not overlap all the regions of the image, but hangs partly outside the edge of the image. This part of the convolution coefficients hanging outside the image cannot find the pixels in the image that can correspond to them when performing the convolution operation, and then there is a computational problem.

There are two simple methods commonly used to solve this problem:

1. Use constant padding. The positions corresponding to the hanging elements in the convolution kernel are filled with a constant, which is usually 0 by default.

2. Copy edge pixels. The pixels at the edge of the image are copied to the corresponding positions of the hanging elements in the convolution kernel and then processed.

### 3.5.4 Roberts operator

#### 3.5.4.1 Basic theory

The gradient of the function is defined as :

$$\nabla f(x,y) = \frac{\partial f(x,y)}{\partial x} i_x + \frac{\partial f(x,y)}{\partial y} i_y$$

In digital images, we can use the difference to approximate the differentiation. The principle of Roberts' edge detection operator<sup>[43]</sup> is to first calculate the difference in all mutually perpendicular directions in the image, and then use the difference for the calculation of the pixel gradient. The pixels used in the actual calculation are the pixels that are in the diagonal direction of the region and take the difference, i.e. :

$$\nabla_x f = f(x,y) - f(x+1,y+1)$$

$$\nabla_y f = f(x,y+1) - f(x+1,y)$$

The gradient of the image is :

$$|\nabla f(x,y)| = \sqrt{\nabla_x^2 f + \nabla_y^2 f}$$

Simplify the equation to obtain:

$$|\nabla f(x,y)| = |\nabla_x f| + |\nabla_y f| \text{ or } |\nabla f(x,y)| = \max(|\nabla_x f|, |\nabla_y f|)$$

The corresponding convolutional template:

$$\nabla_x f \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \nabla_y f \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

When using Roberts' edge detection operator to process image edges, the first step is to obtain the gradient magnitude  $|\nabla f(x,y)|$  according to the above formula, and then we have to preset an appropriate threshold TH for examining the edge pixels of the image as a criterion for comparison and judgment: if  $|\nabla f(x,y)| > TH$ , then  $(x,y)$  is the edge point of the image, and so on, the set of all edge points is detected, and we have the edge image of the image.

The Roberts operator detects the edges of an image based on the difference of the gray values of four adjacent pixels at diagonal positions in the image localization. It can be seen that Roberts operator is more suitable for processing low noise images with steepness.

#### 3.5.4.2 Implementation of the Roberts operator for image edge detection

1. Get a pointer to the data area of the original image.
2. Create a new buffer for the intermediate process of image processing and cache the intermediate result, which should be the same size as the original image. Initialize all the pixel values in the area and set them to 255.
3. The core algorithm is executed by traversing all the pixel points in the image with the Roberts operator inside our newly created buffer and calculating their grayscale values separately, then according to the above formula.
4. The result of the computed data in the buffer is copied to the original image data area.

Programming:

```
clear all;
close all;
clc;
I = imread('0.png');
```

```

% Read the image file to be processed
subplot(131),imshow(I),title('origina');
% Display the image file to be processed
image = rgb2gray(I);
% Convert images to grayscale
TempImage1 = edge ( image ,'Roberts');
% image is the target image for edge detection, the method used is the Roberts
algorithm, and the threshold is generated by the system adaptively
subplot(132),imshow(TempImage1),title('Roberts');
% Show Roberts Auto Threshold Edge Detection effect image
TempImage 2 = edge ( image ,'Roberts', 0.07);
% image is the target image for edge detection, the method used is the Roberts
algorithm, and the threshold is set to 0.07
subplot(133),imshow(TempImage2),title('Roberts 0.07'); % Processing effect
A comparison of the processing results is shown in Figure 3.5.4.

```



**Figure 3.5.4 Roberts Edge Detection Comparison**

### 3.5.5 Sobel Operator

#### 3.5.5.1 Basic theory

The gradient value at a point in an image is proportional to the difference in grayscale between the pixel points that are close to the point. In other words, the larger the gradient value of a pixel, the more rapidly the grayscale changes in the corresponding area, such as the edge of the image. Conversely, the smaller the gradient value of a pixel is, the slower and flatter the grayscale change in the part of the image region<sup>[44]</sup>. Therefore, we can calculate the first order derivative of the image, which corresponds to the gradient value of the image, and thus we can obtain the edges of the image.

The Sobel edge detection operator is an edge detection operator based on the above theory by means of first-order derivatives. He calculates the weighted difference of the gray value of the neighboring pixels in the four perpendicular directions, up and down and left and right, for each pixel point in the image, and for the consideration of the

weights, the neighboring pixels closer to the center pixel have a larger weight value. Then the differentiation is found and finally the gradient is found.

The size of the convolution kernel used by the Sobel edge detection operator for the convolution operation is a  $3 \times 3$  matrix. The commonly used convolution templates for the Sobel operator include both horizontal and directional templates, as shown in Figure 3.5. The former is most useful when detecting edges in the horizontal direction, while the latter is most useful when detecting edges in the vertical direction.

$$\text{Horizontal Templates} \quad \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

$$\text{Vertical Templates} \quad \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

When detecting image edges using the Sobel operator, the two templates are used to convolve each pixel in the image, respectively, and the central pixel is convolved with the central element of the template, respectively. After the computation is completed, the computation results corresponding to the two convolution kernels are compared, and the larger value is the output bit at that point, and then an appropriate threshold is selected to extract the edges.

For an image template neighborhood, the horizontal convolution operation is :

$$\begin{aligned} \Delta G_x = & [f(x-1, y+1) + 2f(x, y+1) + f(x+1, y+1)] \\ & - [f(x-1, y-1) + 2f(x, y-1) + f(x+1, y-1)] \end{aligned}$$

The convolution operation in the vertical direction is :

$$\begin{aligned} \Delta G_y = & [f(x-1, y-1) + 2f(x-1, y) + f(x-2, y+1)] \\ & - [f(x+1, y-1) + 2f(x+1, y) + f(x+1, y+1)] \end{aligned}$$

Gradient calculation:

$$G[f(x, y)] = \sqrt{\Delta G_x^2 + \Delta G_y^2}$$

Simplify to :

$$G[f(x, y)] = |\Delta G_x| + |\Delta G_y|$$



$$\text{image template neighborhood} \begin{pmatrix} f(x-1, y-1) & f(x, y-1) & f(x+1, y-1) \\ f(x-1, y) & f(x, y) & f(x+1, y) \\ f(x-1, y+1) & f(x, y+1) & f(x+1, y+1) \end{pmatrix}$$

The Sobel algorithm has the advantages of simplicity and speed in image processing, as well as the suppression of noise. In contrast, this algorithm also has some disadvantages. Because this algorithm simply uses two perpendicular convolution kernel templates, they are only effective for edges in horizontal or vertical directions, so they are more suitable for processing images with simple textures, but not for those with complex textures.

However, when detecting edges in certain directions, the Sobel algorithm improves the definition of the templates by providing templates for detecting edges in  $45^\circ$  or  $135^\circ$  directions, as in Eqs. 1.23 and 1.24.

$$\begin{bmatrix} 0 & -1 & -2 \\ 1 & 0 & -1 \\ 2 & 1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 2 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -2 \end{bmatrix}$$

### 3.5.5.2 Implementation of Sobel algorithm for image edge detection

1. Get the pointer to the data area of the original image.
2. Create two new buffers with the same size as the original image to store the original image and its copies for later computational processing. Also initialize these two areas as copies of the original image.
3. Set a Sobel operator template for convolution operation for each of the two buffer regions, and then in the two regions, traverse all pixels in the copy image, one by one, to perform the convolution operation and calculate the result.
4. The results in the two buffer regions obtained in the previous step are compared, and the larger value is reassigned to the pixel points in the image.
5. The selected image in the buffer is copied to the original image data area.

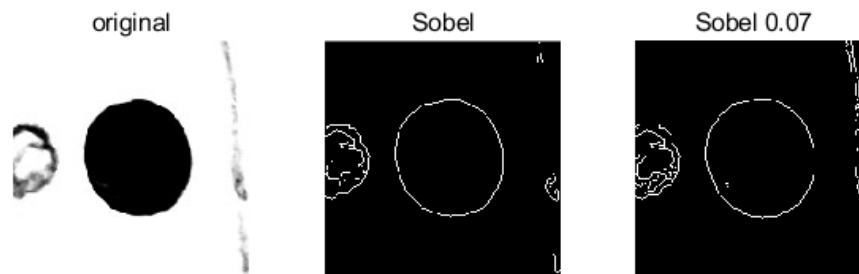
Programming:

```
clear all;
close all;
clc;
I = imread('0.png');
% Read the image file to be processed
subplot(131),imshow(I),title('origina');
```

```

% Display the image file to be processed
image = rgb2gray(I);
% Convert images to grayscale
TempImage1 = edge ( image , ' Sobel' );
% image is the target image for edge detection, the method used is the Sobel
algorithm, and the threshold is generated by the system adaptively
subplot(132),imshow(TempImage1),title(' Sobel');
% Show Sobel Auto Threshold Edge Detection effect image
TempImage 2 = edge ( image , ' Sobel' , 0.07);
% image is the target image for edge detection, the method used is the Sobel
algorithm, and the threshold is set to 0.07
subplot(133),imshow(TempImage2),title(' Sobel 0.07'); % Processing effect
A comparison of the processing results is shown in Figure 3.5.5.

```



**Figure 3.5.4 Sobel Edge Detection Comparison**

### 3.5.6 Prewitt Operator

#### 3.5.6.1 Basic theory

The equations and principles of the Prewitt edge detection operator are basically the same as those described in the previous paper. The former uses an average weighting method, i.e., the average weighting of the influencing pixels around the center pixel, while the latter assigns different weights to the influencing pixels according to their proximity, i.e., the closer the pixel is to the center, the greater the influence, and the greater the weight.

The Prewitt operator template is as follows:

Horizontal Templates

$$\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

Vertical Templates

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

To detect image edges using the Prewitt operator, the same two templates are used to convolve each pixel in the image separately, and the center pixel is convolved with the central element of the template, respectively. After the computation is completed, the computation results corresponding to the two convolution kernels are compared, and the larger value is the output bit at that point, and then an appropriate threshold is selected to extract the edges.

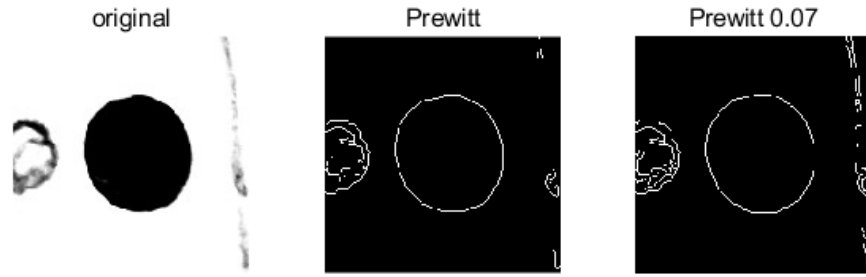
### 3.5.6.1 Implementation of the Prewitt operator for image edge detection

1. Get the data area pointer of the original image.
2. Create two new buffers with the same size as the original image to store the original image and its copies for later computational processing. Also initialize these two areas as copies of the original image.
3. Set up a separate Prewitt template for the convolution operation in each of the two buffer regions, then traverse all the pixels in the copy image in each of the two regions, perform the convolution operation one by one, and calculate the result.
4. Compare the results in the two buffer regions obtained in the previous step and reassign the larger values to the pixel points in the image.
5. Copy the selected image in the buffer to the original image data area.

Programming:

```
clear all;
close all;
clc;
I = imread('0.png');
% Read the image file to be processed
subplot(131),imshow(I),title('origina');
% Display the image file to be processed
image = rgb2gray(I);
% Convert images to grayscale
TempImage1 = edge ( image , ' Prewitt');
% image is the target image for edge detection, the method used is the Prewitt
algorithm, and the threshold is generated by the system adaptively
subplot(132),imshow(TempImage1),title(' Prewitt');
% Show Prewitt Auto Threshold Edge Detection effect image
TempImage 2 = edge ( image , ' Prewitt', 0.07);
% image is the target image for edge detection, the method used is the Prewitt
algorithm, and the threshold is set to 0.07
subplot(133),imshow(TempImage2),title('Prewitt 0.07'); % Processing effect
```

A comparison of the processing results is shown in Figure 3.5.6.



**Figure 3.5.6 Prewitt Edge Detection Comparison**

### 3.5.7 Krisch Operator

#### 3.5.7.1 Basic theory

The Krisch edge detection operator uses a  $3 \times 3$  convolution template to detect the edges of the image. The difference between the weighted sum of the gray levels of three neighboring pixels and the weighted sum of the gray levels of the remaining five pixels is calculated by traversing all the pixel points in the image and examining the gray levels of the pixels in the eight surrounding regions centered on each pixel point one by one. The Krisch algorithm uses eight convolutional templates, representing the eight specific directions of the center pixel. The eight convolution templates are shown below.

Krisch operator convolution template

$$\begin{aligned}
 & \begin{bmatrix} +5 & +5 & +5 \\ -3 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix} \quad \begin{bmatrix} -3 & +5 & +5 \\ -3 & 0 & +5 \\ -3 & -3 & -3 \end{bmatrix} \\
 & \begin{bmatrix} -3 & -3 & -3 \\ -3 & 0 & -3 \\ +5 & +5 & +5 \end{bmatrix} \quad \begin{bmatrix} -3 & -3 & -3 \\ +5 & 0 & -3 \\ +5 & +5 & -3 \end{bmatrix} \\
 & \begin{bmatrix} -3 & -3 & +5 \\ -3 & 0 & +5 \\ -3 & -3 & +5 \end{bmatrix} \quad \begin{bmatrix} +5 & -3 & -3 \\ +5 & 0 & -3 \\ +5 & -3 & -3 \end{bmatrix} \\
 & \begin{bmatrix} -3 & -3 & -3 \\ -3 & 0 & +5 \\ -3 & +5 & +5 \end{bmatrix} \quad \begin{bmatrix} +5 & +5 & -3 \\ +5 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix}
 \end{aligned}$$

Using the above eight volume templates, all pixels in the original image are processed in turn, their edge intensities are calculated, and then detected by thresholding, and the final edge points are extracted to complete Krisch edge detection.

### 3.5.7.2 Implementation of Kirsch algorithm for image edge detection

1. Get the data area pointer of the original image.
2. Create two new buffers with the same size as the original image to store the original image and its copies for later computational processing. Also initialize these two areas as copies of the original image, labeled as Image 1 and Image 2 respectively.
3. For the two cache regions, a separate Kirsch template for convolution is set in each region, and then all the pixels in the copy images are traversed in each of the two regions, and the results are calculated one by one by convolution. Then, the result of the convolution operation is calculated one by one in each of the two regions, and the higher value is temporarily stored in image 1, and then image 1 is copied to cache image 2.
4. Repeat step 3 to set up the remaining six templates and perform the computation, and store the larger grayscale values in Image 1 and Image 2 in Cache Image 1.
5. Copy the processed Image 1 to the original image data area.

Programming:

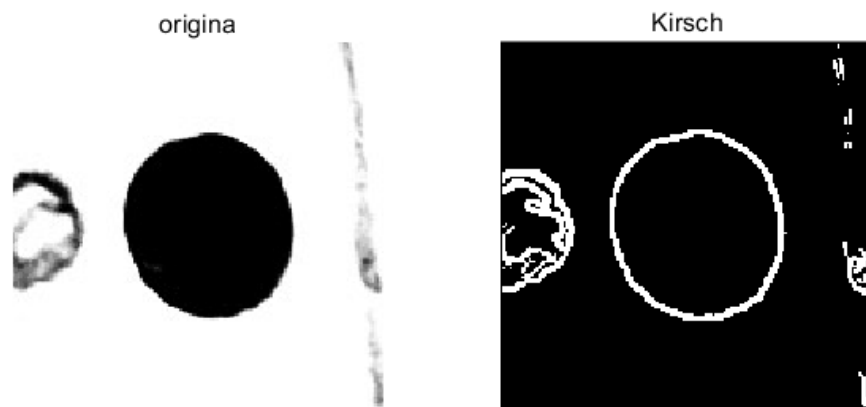
```
clear all;
close all;
clc;
I = imread('0.png');
% Read the image file to be processed
image=rgb2gray(I);
subplot(121),imshow(I),title('origina')
t=1200 ;
% Set default thresholds
colormap(gray(256));
% Set the color palette
BW=double(image);
% Convert the original image to decimal
[m,n]=size(BW);
% Get the length and width of the image
g=zeros(m,n);
% Define a zero matrix of size S
for i=2:m-1
    for j=2:n-1
        % Eight templates for setting up the Kirsch operator
        d1=(5*BW(i-1,j-1)+5*BW(i-1,j)+5*BW(i-1,j+1)-3*BW(i,j-1)-3*BW(i,j+1)-
        3*BW(i+1,j-1)-3*BW(i+1,j)-3*BW(i+1,j+1))^2;
        d2=((-3)*BW(i-1,j-1)+5*BW(i-1,j)+5*BW(i-1,j+1)-3*BW(i,j-1)+5*BW(i,j+1)-
        3*BW(i+1,j-1)-3*BW(i+1,j)-3*BW(i+1,j+1))^2;
        d3=((-3)*BW(i-1,j-1)-3*BW(i-1,j)+5*BW(i-1,j+1)-3*BW(i,j-1)+5*BW(i,j+1)-
        3*BW(i+1,j-1)-3*BW(i+1,j)+5*BW(i+1,j+1))^2;
```

```

d4=((-3)*BW(i-1,j-1)-3*BW(i-1,j)-3*BW(i-1,j+1)-3*BW(i,j-1)+5*BW(i,j+1)-
3*BW(i+1,j-1)+5*BW(i+1,j)+5*BW(i+1,j+1))^2;
d5=((-3)*BW(i-1,j-1)-3*BW(i-1,j)-3*BW(i-1,j+1)-3*BW(i,j-1)-
3*BW(i,j+1)+5*BW(i+1,j-1)+5*BW(i+1,j)+5*BW(i+1,j+1))^2;
d6=((-3)*BW(i-1,j-1)-3*BW(i-1,j)-3*BW(i-1,j+1)+5*BW(i,j-1)-
3*BW(i,j+1)+5*BW(i+1,j-1)+5*BW(i+1,j)-3*BW(i+1,j+1))^2;
d7=(5*BW(i-1,j-1)-3*BW(i-1,j)-3*BW(i-1,j+1)+5*BW(i,j-1)-
3*BW(i,j+1)+5*BW(i+1,j-1)-3*BW(i+1,j)-3*BW(i+1,j+1))^2;
d8=(5*BW(i-1,j-1)+5*BW(i-1,j)-3*BW(i-1,j+1)+5*BW(i,j-1)-3*BW(i,j+1)-
3*BW(i+1,j-1)-3*BW(i+1,j)-3*BW(i+1,j+1))^2;
g(i,j)=round(sqrt(d1+d2+d3+d4+d5+d6+d7+d8));
end
end
for i=1:m
    for j=1:n
        if g(i,j)>t
            BW(i,j)=255;
        else
            BW(i,j)=0;
        end
    end
end
end
% Compare the calculated output gradient value with the threshold value
subplot(122),imshow(BW),title('Kirsch');
%Display the image after edge extraction

```

A comparison of the processing results is shown in Figure 3.5.7.



**Figure 3.5.7 Kirsch Edge Detection Comparison**

### 3.5.8 LOG Operator

#### 3.5.8.1 Basic theory

The Laplace operator is based on the edge characteristics of the image, and we know that the gray scale of the image around him varies widely, so it is known that the first-order derivative here will have a local maximum or minimum, and is also reflected as a zero crossing point of the second-order derivative over the region. The Laplace operator is the second-order derivative of the image function, and determines the edge position of the image by determining where it crosses the zero point.

Assuming that  $f(x,y)$  is a continuous function, the Laplace operator defining him at the point  $(x,y)$  is:

$$\nabla^2 f(x,y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

From the above equation, it can be seen that the Laplace operator is isotropic in nature, i.e., direction-independent, linear and shift-invariant. Based on this property, Laplace operator is fully capable of sharpening and detecting image edges of various orientations in digital images.

The Laplace operator is based on a convolutional template similar to the one described above, and the template used in the Laplace operator is defined by the following rules: the sum of all elements in the template is equal to zero, the central element is positive, and all remaining elements are negative. The following two are commonly used in the discrete case. The following are shown:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$
$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

The Laplace operator is usually expressed by the following equation:

$$\nabla^2 f(x,y) = f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)$$

where  $\nabla^2 f(x,y)$  is the sum of the second-order partial derivative of the X-axis and the second-order partial derivatives with respect to the Y-axis at the point in the

image, which also represents the gray value at that point after the image has been processed using the Laplace operator.

The essence of image edge detection using the Laplace operator is to find the location of the image edge by examining the zero position of the operator. In some cases of image edge detection, we focus on the location of the edge and do not care about the specific value of the grayscale difference, which makes the Laplace operator a better choice based on its direction-independent nature.

By analyzing the processing effect of Laplace operator, we can find some obvious problems and drawbacks. In some parts of the edge image, there are occasionally double pixels, so we can not better and more accurately determine where the real edge is located, and the complex and useless information increases, which undoubtedly multiplies the noise in the resulting image, making the image quality poor and unsatisfactory. Therefore, the Laplace operator is rarely used directly for image edge detection, but is used in combination with a Gaussian smoothing filter.

### 3.5.8.2 Gauss operator

The Gauss filter can attenuate some of the noise in the image, he is a linear smoothing filtering method whose choice of template is based on the shape of the Gauss function, so he can suppress well those noises that obey the positive-terrestrial distribution<sup>[45]</sup>. The Gauss function is used to smooth the noise of the original image, this is because the Gauss filter has the characteristics of spatial smoothness and small spatial position error. The two-dimensional Gaussian function is :

$$h(x,y) = \exp[ - (x^2 + y^2)/2\delta^2 ]$$

where  $\delta$  is the standard deviation and is defined as follows:  $r^2 = x^2 + y^2$ . The Gauss-Laplace operator is obtained by combining the above equation and then calculating the value of the second-order derivative:

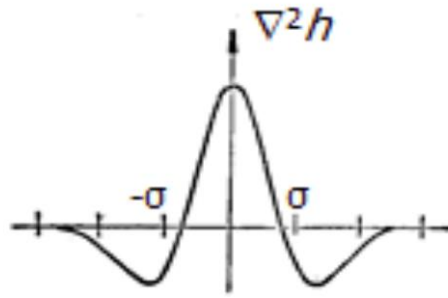
$$\nabla^2 h(x,y) = [(r - \delta^2)/\delta^4] \exp(-r^2/2\delta^2)$$

### 3.5.8.3 LOG

Gauss-Laplace algorithm, in order to get the result of edge detection image effect and quality significantly improved, considering the addition of pre-processing steps to the image, first, the Gauss filter is used to smooth the target image and reduce the noise of the image, as a way to improve the quality of the detection algorithm entrance image itself. Then the real detection is started, using the Laplace operator to process the low-noise image and obtain the edges of the image.

The one-dimensional function of the Gauss-Laplace operator is shown in Figure 3.15.





**Figure 3.5.8 One-dimensional LOG function**

The choice of variance  $\delta$  in the Gauss-Laplace operator formula has a crucial influence on the good or bad results of image detection. When  $\delta$  is small, the position accuracy of the detected edges is high, but the edge details vary a lot, which can be used to detect the image edge details; when  $\delta$  is large, the smoothing effect of the operator is large, which can largely reduce the influence of noise, but at the same time, it also smoothes out part of the meaningful information, causing the loss of details, and the position accuracy of the edges is also low, which can be used to detect the image edge contours. When using the Gauss-Laplace operator to detect image edges, the appropriate  $\delta$  is chosen according to the need for accuracy in edge detection and the noise of the image.

The most commonly used convolution kernel template for the Gauss-Laplace operator is a matrix of size  $5 \times 5$ , as follows:

$$\begin{bmatrix} -2 & -4 & -4 & -4 & -2 \\ -4 & 0 & 8 & 0 & -4 \\ -4 & 8 & 24 & 8 & -4 \\ -4 & 0 & 8 & 0 & -4 \\ -2 & -4 & -4 & -4 & -2 \end{bmatrix}$$

#### 3.5.8.4 Implementation of Gauss-Laplace operator for image edge detection

1. Get the data area pointer of the original image.
2. Apply an image buffer of the same size as the original image, and copy the original image into the buffer.
3. Set a Gauss-Laplace operator template for convolution operation, traverse all pixels in the copy image in the region, perform convolution operation one by one, and calculate the result.
4. Copy the calculation result to the original image data area.

Programming:

```

clear all;
close all;
clc;
I = imread('0.png');
% Read the image file to be processed
subplot(131),imshow(I),title('origina');
% Display the image file to be processed
image = rgb2gray(I);
% Convert images to grayscale
TempImage1 = edge ( image , ' LOG' );
% image is the target image for edge detection, the method used is the LOG
algorithm, and the threshold is generated by the system adaptively
subplot(132),imshow(TempImage1),title(' LOG');
% Show LOG Auto Threshold Edge Detection effect image
TempImage 2 = edge ( image , ' LOG', [], 4);
% LOG operator with standard deviation  $\delta$  set to 4
subplot(133),imshow(TempImage2),title('LOG 4'); % Processing effect
A comparison of the processing results is shown in Figure 3.5.8.

```



**Figure 3.5.8 LOG Edge Detection Comparison**

### 3.5.9 Canny Operator

#### 3.5.9.1 Basic theory

The Canny operator is similar to the LOG method in that it first uses a certain method to smooth the image and reduce the noise of the target image, and after completing the image noise reduction process, and then calculates and examines the derivatives.

The two main problems to be solved by edge detection are resistance to noise interference and precise location of edges. If the edge of the image is detected only by simple differential calculation, it will be difficult to deal with the edge part of the image and the noise part of the same high-frequency component, which will make the noise of the image more serious<sup>[46]</sup>. The Canny edge detection operator uses this kind of processing, and the image is processed by first-order differentiation of the Gauss function, which is the result of its processing of the image, and it is a good balance

between The two main problems are resistance to noise interference and precise location of edges.

Along with the proposed Canny algorithm, three major criteria are given on how to identify various edge detection good and bad<sup>[47]</sup>:

1) Good detection performance. An edge detection algorithm with good performance has both a small miss detection rate and a false detection rate. That is, it is rare to output edge points of an image as non-edge points or non-image edge points as edge points of an image in the resultant image. We use a signal-to-noise ratio (SNR) as a more objective criterion, but the larger the value of the parameter, the better.

$$SNR(f) = \frac{\left| \int_{-W}^W G(-x) f(x) dx \right|}{\delta \sqrt{\int_{-W}^W f^2(x) dx}}$$

$G(-x)$  in the formula is used to express the edge function of the image,  $f(x)$  is used to represent the filter function used in the smoothing operation, and  $\delta$  is used to represent the mean squared error of the noise in the image.

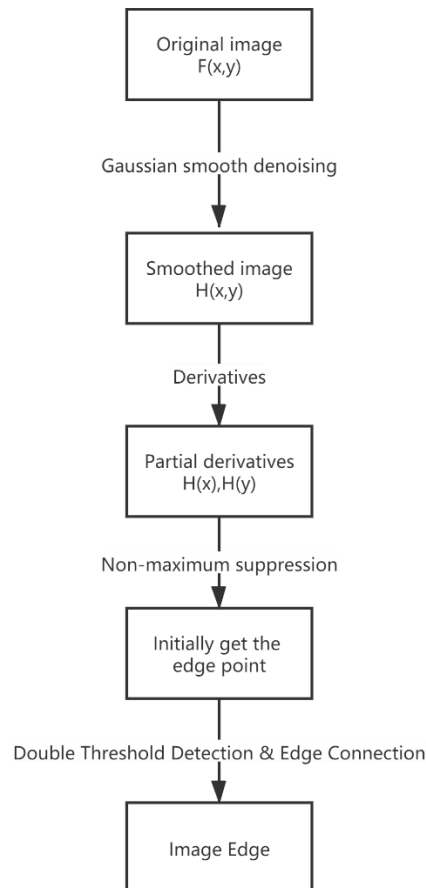
2) High localization accuracy. If the distance between the edge points in the detected image and the corresponding real edge points is smaller, the closer they are, the higher the localization accuracy of the algorithm can be. The larger the evaluation parameter Location, the better.

$$Location = \frac{\left| \int_{-W}^W G(-x) f'(x) dx \right|}{\delta \sqrt{\int_{-W}^W f'^2(x) dx}}$$

3) Minimal number of edge responses. To reduce the probability of multiple pixels responding to a single pixel edge when edge detection is processed, i.e., to try to have a single pixel respond to that edge. If we use the function  $D(f)$  to represent the average of the distances between the zero-crossing points of the impulse response derivatives of the detection operator, then  $D(f)$  should satisfy:

$$D(f') = \pi \left\{ \frac{\int_{-\infty}^{\infty} f'^2(x) dx}{\int_{-\infty}^{\infty} f(x) dx} \right\}^{1/2}$$

To sum up, a simple summary of the criteria for judging the performance of edge detection is that a really good edge detection algorithm is able to improve the sensitivity of the algorithm in detecting edge pixel points and at the same time suppress the noise very well.



**Figure 3.5.9 Canny operator flowchart**

1. The Canny operator first preprocesses the original image with Gauss filter for smoothing and denoising, that is, it uses Gauss function to do convolution operation with the image. The formula is as follows:

$$S(i,j) = G(i,j;\delta) * f(i,j)$$

where  $*$  is the convolution operation,  $G$  is the Gaussian smoothing filter function,  $f(i,j)$  is the gray value of a point in the original image, and  $\delta$  is the variance of the Gaussian function, which mainly affects the degree of image smoothing.

2. Calculate the partial derivatives of the image at the midpoint  $(x,y)$ , respectively, along the two directions, and then calculate the magnitude and direction of the gradient at that point. The formula is as follows:

The gradient amplitude is  $|\nabla f(x,y)| = \sqrt{\nabla_x^2 f + \nabla_y^2 f}$

The gradient direction is  $\Theta(x,y) = \arctan \frac{\nabla_y f}{\nabla_x f}$

3. After obtaining the global gradient of the image by the previous step, we need to filter out the pixel points with the maximum local gradient value to obtain the preliminary edge of the image. extraction.

The non-maximum suppression actually uses the gradient direction to discretize the image gradient into one of the four sectors around the center pixel in the investigated region, and for the convenience of representation, they are numbered from 0 to 3, as shown in Figure 3.5.10. The field centered on pixel B(x,y) is examined, and its gray value is compared with that of the surrounding pixels, and if it is smaller than or equal to the neighboring two at the same time in the gradient direction, then B(x,y) is set. Then set B(x,y)=0.



**Figure 3.5.10 Schematic diagram of edge direction**

4	3	2
1	B	1
2	3	4

**Figure 3.5.11 8 Neighborhood amplitude direction**

4. Double thresholding for edge detection

Since single thresholding is difficult to select a suitable threshold when processing images and usually requires several iterations, double thresholding is used for detection.

After the target image is preprocessed with non-maximum suppression, it is then processed by using two thresholds simultaneously, labeled  $th1$  and  $th2$ , and the magnitude of these two values satisfies the equation  $th1 = 0.4th2$ . First, the image is processed with the threshold  $th1$ , and the gradient value of each point in the image is traversed and compared with  $th1$ . If it is smaller than  $th1$ , the gray value is set to 0. The image output after traversal is used as image 1, and then the image is processed similarly with  $th2$ , and the result image after traversal is used as image 2.

#### 5. Connecting edge

Image 1, obtained with a smaller threshold, retains more edge information, but the effect of noise is still significant, while Image 2, obtained with a larger threshold, removes most of the noise, but loses much useful edge information. In this way, we can use Image 2 as the basis, and use the edge detail information retained in Image 1 to supplement the missing information in Image 2 and connect the image edges.

The operation process of Canny operator to detect image edges is actually solved by finding and examining the extreme value of the signal function of the target image. canny operator uses Gaussian filter to smooth the noise, so it has stronger anti-creation performance. Also, the Canny algorithm uses a double threshold detection algorithm and the use of edge linking to enhance the processing of edge-detected images, so the image edges extracted by this operator have better continuity.

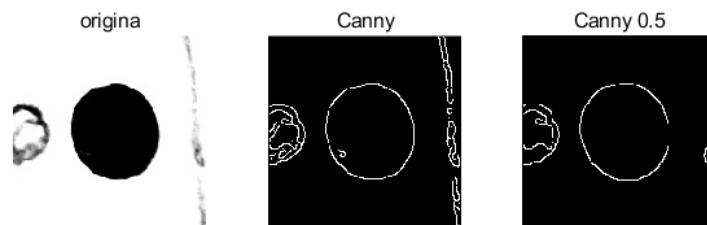
#### 3.5.9.2 Implementation of the Canny operator for image edge detection

1. Obtain the data area pointer of the original image.
2. Smooth the noise of the original image with Gauss filter to reduce the noise effect. That is, the Gauss function is used to do the convolution calculation with the original image.
3. Take the result image of the previous step as the input of this step, first calculate the first-order partial derivative finite difference, then calculate the gradient amplitude and direction of its local area, and perform non-extreme value suppression.
4. Create two new image buffer regions for storing copies of the target image, set their size to be the same as the original image, and initialize the copies of the original image into the two regions respectively.
5. Extract the image edge information in the two buffers with  $th1$  and  $th2$  to get image 1 and image 2 respectively
6. In the operation of connecting edges, we mainly use image 2 as the base, and continuously add the effective detail information to it by comparing and examining in image 1 to get more continuous image edges.

7. Copy the connected image to the original image data area.

Programming:

```
clear all;
close all;
clc;
I = imread('0.png');
% Read the image file to be processed
subplot(131),imshow(I),title('origina');
% Display the image file to be processed
image = rgb2gray(I);
% Convert images to grayscale
TempImage1 = edge ( image , ' Canny');
% image is the target image for edge detection, the method used is the Canny
algorithm, and the threshold is generated by the system adaptively
subplot(132),imshow(TempImage1),title(' Canny');
% Show Canny Auto Threshold Edge Detection effect image
TempImage 2 = edge ( image , ' Canny', 0.5);
% Canny operator with threshold 0.5
subplot(133),imshow(TempImage2),title(' Canny 0.5'); % Processing effect
A comparison of the processing results is shown in Figure 3.5.12.
```



**Figure 3.5.12 Canny Edge Detection Comparison**

### 3.5.10 Comparative Analysis of Image Edge Detection Operators

When we process images and perform edge detection, we want to detect all the image edges sharply without the interference of image noise<sup>[48]</sup>. Combined with the above image processing effect comparison graph analysis, we can get the characteristics, advantages and disadvantages of these edge detection operators in terms of edge detection effect and the range of use.

The Roberts edge detection operator is a  $2 \times 2$  operator that works best for processing low-noise images with steepness, but in the resulting images processed with this operator, the displayed image edges are often coarse and cannot locate the exact edge position well.

Sobel and Prewitt, both of which use the same size convolutional template, are  $3 \times 3$ , and are more accurate in locating image edges than the Roberts operator, which only uses a  $2 \times 2$  template, but both operators are not isotropic, and the edge images they detect have many breakpoints and are not continuous.

In this case, since the Krisch operator takes into account the edge information in eight directions, its edge localization ability is stronger and the processing effect is relatively better. Since the algorithm of Krisch algorithm is based on the calculation of first-order differentiation, it is more suitable for the detection of those step-type edges.

The Gauss-Laplace edge detection algorithm uses a larger  $5 \times 5$  template and is isotropic. This method is more ideal for images with more noise, and at the same time, the grayscale is almost no sudden change, but is slowly changing, and the continuity and sharpness of the extracted edge images are relatively good. However, the Gauss-Laplace algorithm is more difficult to choose the threshold value for zero-crossing when detecting step-type edges, and the detection accuracy is relatively low.

In contrast, the Canny edge detection operator extracts better continuity and clarity of the edge image, suppresses the noise better, and detects the real weak edges in the image more easily, among which the Canny operator is more effective in detecting the step-type edges. But similar to Gauss-Laplace operator, he is easy to smooth out some image edge information in the smoothing operation of the image.

In practical applications, we can use more suitable edge detection operators according to the specific needs of processing images, the efficiency and accuracy requirements of the algorithm, or we can combine the use of several operators to achieve the different needs of image edge information.

The edge detection algorithm is more suitable for simple image segmentation with obvious edge grayscale transition and low noise. For images with complex edges and strong noise, there is a conflict between noise immunity and detection accuracy. If the detection accuracy is improved, the false edges generated by noise will lead to unreasonable contours; if the noise immunity is improved, missing contour detection and position bias will occur.

The boundary demarcation of the electrode imprinted image is relatively clear. In order to better extract the information of the electrode imprinted area in the image, this paper uses LOG operator for image segmentation of electrode imprinting. From the electrode imprinted image 0.png, it can be seen that there are more interference regions within the image, which is not the extraction target. In order to reduce the interference of noise, the connected areas (number of pixels) less than 3000 are considered as noise points and removed (i.e. set to background black), and finally the boundary map of the electrode is derived.

Programming:

```
clear all;  
clc;  
close all;  
I=imread('0.png');
```

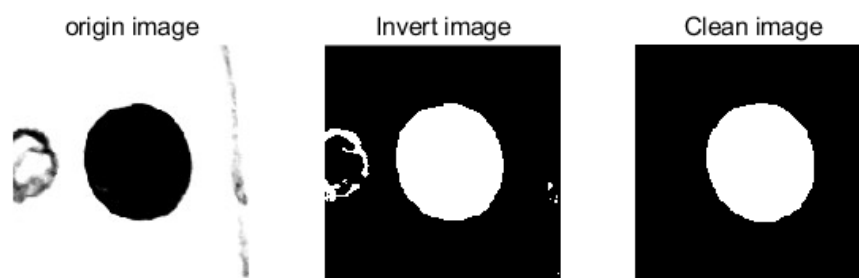


```

BW = im2bw(I);          %Image binarization
BW=~BW  %Image inversion
figure,imshow(BW),title('Invert image');
[L,num]=bwlabel(BW,8); %Marking area
STATS=regionprops(L, 'all');
for i=1:num
area(i)=STATS(i).Area; %Compute a simple estimate of the object's perimeter
end
% Observed the monitoring area larger than 3000
BWout=bwareaopen(BW,3000,8); %Wipe out areas less than 3000
figure,imshow(BWout),title('Clean image');

```

A comparison of the processing results is shown in Figure 3.5.13.



**Figure 3.5.13 Removal of interference targets**

With the increase of solder joints, the pit erosion inside the electrode embossing will cause some interference to the image feature extraction, so the internal area needs to be filled.

Programming:

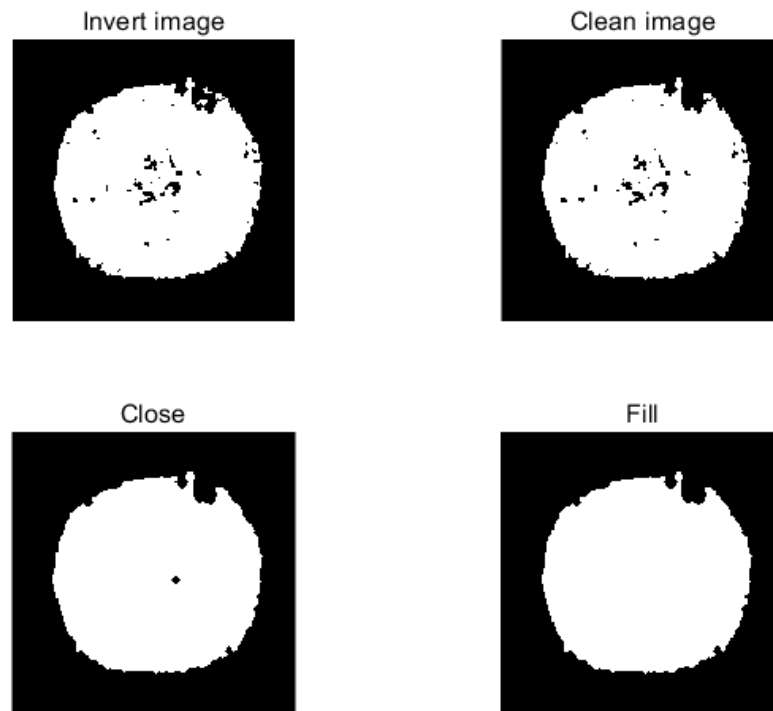
```

clear all;
clc;
close all;
I=imread('1044.png');
BW = im2bw(I);          %Image binarization
BW=~BW  %Image inversion
figure,
subplot(221),imshow(BW),title('Invert image');
[L,num]=bwlabel(BW,8); %Marking area
STATS=regionprops(L, 'all');
for i=1:num
area(i)=STATS(i).Area; %Compute a simple estimate of the object's perimeter
end
BWout=bwareaopen(BW,3000,8); %Use the morphology function to remove pixels
that do not belong to the object of interest

```

```
subplot(222),imshow(BWout),title('Clean image');
BW3=imclose(BWout,strel('disk',2));%Fill gaps
subplot(223),imshow(BW3),title('Close');
FILL=imfill(BW3,'holes');%Fill holes
subplot(224),imshow(FILL),title('Fill');
```

A comparison of the processing results is shown in Figure 3.5.14.



**Figure 3.5.14 Removal of pit corrosion**

In addition to edge detection and segmentation of the complete image of the electrode imprint, the image of the pit erosion needs to be processed to extract its image features to observe the relationship between the pit erosion area and the number of welds.

Programming:

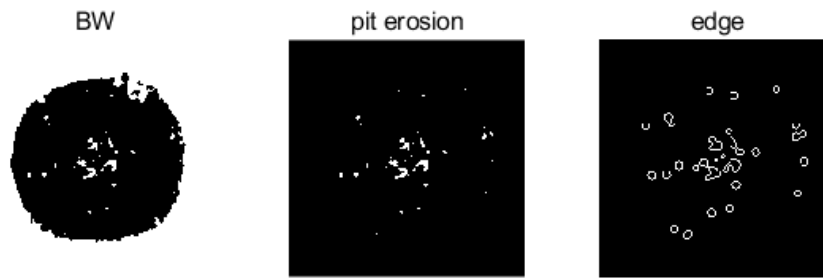
```
clear all;
clc;
% close all;
I=imread('1044.png');
BW = im2bw(I); %Image binarization
figure,
subplot(131),imshow(BW),title('BW');
[L,num]=bwlabel(BW,4); %Marking area
s=zeros(num,1);
for ii=1:num
s(ii)=sum(sum(L==ii)); %Calculate the area of each connected area
```

```

end
[ss IX]=sort(s,'descend'); %Area in descending order
BWout=xor(BW,L==IX(1)); %Wipe out the largest area
subplot(132),imshow(BWout),title('pit erosion');
edge_can=edge(BWout,'LOG');
subplot(133),imshow(edge_can),title('edge');
A=bwarea(edge_can)

```

A comparison of the processing results is shown in Figure 3.5.15.



**Figure 3.5.15 Pitting edge detection**

### 3.6 Electrode imprint image feature parameters

The surface image of electrode imprint was analyzed above to realize the segmentation of feature areas. By extracting the area of each feature region, it was found that the feature regions presented by the electrode imprint image contain a lot of information about the quality of the welding joint. The correlation between the area of each feature region and the wear degree of the electrode head is studied and analyzed, providing a feature reference for monitoring the wear degree of the electrode head.

Let the image size be  $N \times M$ ,  $f(x,y)$  is the point of the image, then the electrode embossing shape features are calculated as follows:

#### 3.6.1 Electrode imprint area

The area is a metric parameter of the total size of the object and thus it can be used as a parameter to evaluate the variation of the indentation shape of the solder joint. In this paper, we will use the pixel-calculated area method, which is based on the principle that for a region  $R_i$  of an image, the area is the sum of the gray levels of the pixel points inside the boundary (including on the boundary) in the statistical  $R_i$ , calculated as follows:

$$S_i = \sum_{x=1}^N \sum_{y=1}^M f(x,y)$$

For a binary image, if 0 denotes the target and 1 denotes the background, its area is the number of statistics  $f(x,y)=0$ . For a frame with  $k$  regions, i.e.,  $i=1,2,3,\dots,k$ , the total area  $S$  is the sum of the areas of each region.

$$S = \sum_{i=1}^k S_i$$

Calculation method of electrode imprint area:

- 1) Calculate the pixel value per unit area: select the unit area on the acquired image, call up the histogram of that unit area and get its pixel value.
- 2) Selecting the feature area to be calculated: Using the image segmentation method, segment the feature area to be calculated, with the feature area as the target value and all other areas treated as background, and calculate the pixel value of the target area.
- 3) Calculate the area of the target region: the ratio of the pixel value of the target region to the pixel value per unit area is the area of the target region to be calculated.

### 3.6.2 Electrode imprint perimeter

The perimeter is particularly useful for distinguishing between objects with simple or complex shapes. The perimeter of the embossed area is calculated in a similar way to the embossed area, simply by counting the number of pixels in a certain gray value of the pre-processed solder joint image, which is represented by the sum of the areas occupied by the borders, each pixel being a small square of area 1.

$$P = \sqrt{2}(N - N_x - N_y) + N_x + N_y$$

where  $N$  is the number of pixels at the edge of the region, and  $N_x$  and  $N_y$  are the number of pixels in the horizontal and vertical directions, respectively.

### 3.6.3 Roundness

Roundness reflects the fullness of the target image relative to its standard circle. It is a parameter of the similarity between the object and the circle and is also called the circle fitting factor.

The calculation formula is as follows:

$$R = \frac{4\pi \times A}{L^2}$$

In the above equation, A represents the image area and L represents the target area perimeter. The value of the circular fitting factor is 0 to 1, and the maximum value of R is 1 for round objects; for rectangular objects, the value of R becomes smaller and smaller; for slender and curved objects, the value of R is the smallest.

Programming:

```

    BW = im2bw(I);
    BW=~BW
    [L,num]=bwlabel(BW,8);
    STATS=regionprops(L, 'all');
    for i=1:num
        area(i)=STATS(i).Area;
    end
    BWout=bwareaopen(BW,3000,8);
    BW3=imclose(BWout,strel('disk',2));
    FILL=imfill(BW3,'holes');
    edge_can=medfilt2(FILL);
    edge_can=edge(edge_can,'LOG');
    [x,y]=size(edge_can);
    BW = bwperim(edge_can);
    P1=0;
    P2=0;
    Ny=0;
    for i=1:x
        for j=1:y
            if (BW(i,j)>0)
                P2=j;
                if ((P2-P1)==1)
                    Ny=Ny+1;
                end
                P1=P2;
            end
        end
    end
    P1=0;
    P2=0;
    Nx=0;
    for j=1:y
        for i=1:x
            if (BW(i,j)>0)
                P2=i;
                if ((P2-P1)==1)
                    Nx=Nx+1;
                end
            end
        end
    end

```

```

        end
        P1=P2;
    end
end
end
SN=sum(sum(BW));
Nd=SN-Nx-Ny;
H=max(sum(edge_can));
W=max(sum(edge_can));
Perimeter=sqrt(2)*Nd+Nx+Ny;

BW3=imclose(edge_can,strel('disk',1));
FILL=imfill(BW3,'holes');
Aera=bwarea(FILL);
Round=4*pi*Aera/Perimeter^2;

BW = im2bw(I)
[L,num]=bwlabel(BW,4);
s=zeros(num,1);
for ii=1:num
s(ii)=sum(sum(L==ii)); %Calculate the area of each connected area
end
[ss IX]=sort(s,'descend'); %Area in descending order
BWout=xor(BW,L==IX(1)); %Wipe out the largest area
ed=edge(BWout,'LOG');
Aera1=bwarea(ed)

```

After extracting the above feature information, the geometric feature vector  $I$  of the electrode image can be expressed as:

$$I = [P, A, R, A1]^T$$

Where  $P$  is the perimeter of the electrode imprint,  $A$  is the area of the electrode imprint,  $R$  the roundness of the electrode imprint, and  $A1$  is the total area of the pit in the image.

## CHAPTER 4

### SVM PREDICTION MODEL OF ELECTRODE WEAR

The regression problem is a typical problem in mathematical modeling, and the solutions to the regression problem are commonly used in traditional multiple regression prediction, as well as artificial neural network prediction<sup>[49]</sup> and gray prediction, which have been developed in recent years. Regression prediction is difficult to deal with high-dimensional and nonlinear patterns; artificial neural networks do not have strong generalization ability and require a large number of training samples; gray prediction has poor long-term prediction ability. Currently, SVM is the best method for small sample statistics and prediction learning. The main difference between it and traditional statistics is that the theoretical basis of inference is different, and the theoretical inference of SVM can be made for small samples, while the inference of traditional statistics is mainly based on the theorem of large numbers. SVM can avoid the problem of structural selection and local minima of neural networks, and can be extended to other machine learning problems such as function fitting<sup>[32,33]</sup>. SVM has intuitive SVMs have an intuitive geometric interpretation, perfect mathematical form, and good generalization performance to solve the problems of model selection, underlearning, overlearning, and nonlinearity, and can overcome the drawbacks of slow convergence and falling into local optima, thus, support vector machines show superior performance in classification and regression. In this chapter, SVM is introduced into the prediction of electrode head wear, and the characteristic parameters of electrode head wear are extracted as model inputs to build the regression prediction model of electrode head wear.

#### 4.1 The basic principle of SVM

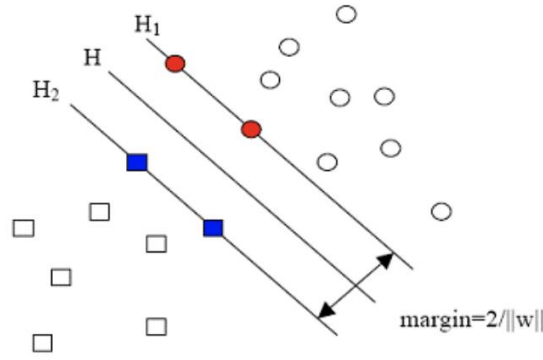
The essence of machine learning is an approximation of a hypothetical model to a real model. To study the gap between the hypothetical model and the real solution of the problem, the concept of structural risk is introduced. Risk is the accumulation of errors between the hypothetical model and the true solution of the problem. The difference between the classification result of the sample data and the true result of the problem is called the empirical risk  $R_{emp}(w)$ . The degree of confidence in the hypothetical model is called confidence risk, which cannot be calculated exactly and can only give a range of estimates. The goal of statistical learning is to find the minimum sum of empirical risk and confidence risk, i.e., the minimum structural risk. Therefore, the concept of generalized error bound is introduced to mean that the true risk consists of two components, one is the empirical risk and the other is the confidence risk. The formula for the generalization error bound is :

$$R(w) \leq R_{emp}(w) + \Phi(n/h)$$

In the formula  $R(w)$  is the true risk,  $R_{emp}(w)$  is the empirical risk, and  $\Phi(n/h)$  is the confidence risk.

SVM is an algorithm that strives to minimize the structural risk. The basic idea of SVM is to map the training data from the nonlinearity of the input space to a higher dimensional space by defining an appropriate inner product function that makes the samples linearly divisible within that space in order to find the optimal linear classification surface.

Figure 4.1.1 shows the two-dimensional linearly divisible case, which can further illustrate the optimal linear classification surface. The hollow and solid points represent two classes of samples, H is the classification surface, H1 and H2 are straight lines parallel to H and nearest to H. The distance between H1 and H, H2 and H is the classification interval. The optimal classification line is the one that not only separates the two types of samples correctly, but also has the largest classification interval.



**Figure 4.1.1 Two-dimensional linear classification schematic**

For linearly divisible sample sets:

$$(x, y), \quad i = 1, 2, \dots, n, \quad x \in R^d, \quad y \in \{+1, -1\}_i$$

Linear discriminant function in d-dimensional space:

$y_i = w \cdot x_i - b$  can be formulated as a constrained minimization problem, i.e.:

$$\min \Phi(w, \zeta) = \frac{1}{2} (w \cdot w) + C \left( \sum_{i=1}^n \zeta_i \right)$$

$$s.t. y_i [(w \cdot x_i + b)] \geq 1 - \zeta_i$$



$w$  is the weight vector;  $\zeta_i \geq 0$  is the relaxation variable;  $C \geq 0$  is the error penalty strength factor. and by introducing a Lagrangian function for determining the solution of the maximum classification interval. The constructed Lagrangian function is :

$$L(w, a, b) = \frac{1}{2} \|w\|^2 - \sum_{i=1}^n a_i y_i (x_i \cdot w + b) + \sum_{i=1}^n a_i$$

$$a_i \geq 0, \quad i = 1, 2, \dots, n,$$

where  $a_i$  is a Lagrangian operator and satisfies the following equation:

$$0 \leq a_i \leq c \quad \text{和} \quad \sum_{i=1}^n a y_i = 0$$

The support vector determined by  $a_i$  can be obtained by maximizing the dual form of Lagrange function under the above two conditions:

$$Max L_D = \sum_{i=1}^n a_i - \frac{1}{2} \sum_{i,j=1}^n a_i a_j y_i y_j x_i x_j$$

The optimized  $L_D$  mainly depends on the number multiplication between the transformed eigenvectors. If a kernel function  $K(x, y)$  is used to replace the number multiplication, the calculated transformation can be avoided. The discriminant function can be written as:

$$\sum a_i y_i K(x_i x) + b$$

For the linearly indistinguishable classification problem, the sample  $X$  can be first mapped onto a high-dimensional feature space  $H$ , and the inner product operation can be realized by applying the function of the original space in this space, so that the nonlinear problem can be converted into a linear problem in another space to obtain the attribution of this sample. According to the theory of general functions, as long as a kernel function can satisfy Mercer's condition, it corresponds to the inner product in a certain space, so as long as the appropriate inner product function is used on the optimal classification plane, this linear indistinguishable classification problem can be realized. In this case, the objective function is:

$$\max w(a) = \sum_{i=1}^n a_i - \frac{1}{2} \sum_{i,j=1}^n a_i a_j y_i y_j K(x_i \cdot x_j)$$

The classification function is:

$$f(x) = \sum_{i=1}^n y_i a_i K(x \cdot x_i) + b^*$$

The conditional requirement of Mercer is to make it a convex quadratic optimization problem, ensuring that a globally optimal solution is obtained and avoiding falling into a local minimum during the training process. In SVM theory, the different kernel functions used will lead to different algorithms for SVMs.

The commonly used kernel functions are:

Linear kernel functions :

$$K(x, x_i) = x \cdot x_i$$

Polynomial kernel function :

$$K(x, x_i) = [\gamma(x \cdot x_i) + d]^p$$

RBF kernel function :

$$K(x, x) = \exp(-\gamma \|x - x\|)$$

Sigmoid kernel function :

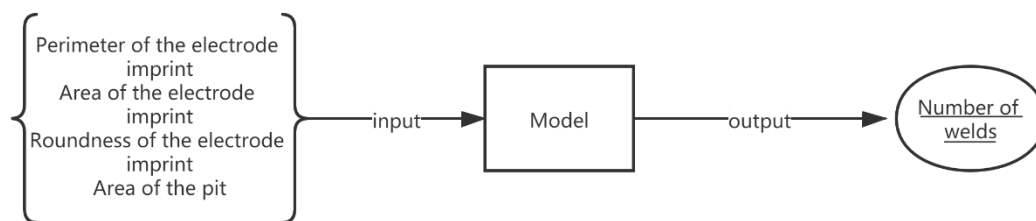
$$K(x, x_i) = \tanh[\gamma(x \cdot x_i) + d]$$

The type of kernel function needs to be chosen by comparing experiments based on the distribution of training samples. the performance of SVM is not so much related to the choice of kernel function type, but is mainly influenced by the kernel parameter  $\gamma$  and the error penalty factor  $C$ . Different kernel parameters  $\gamma$  implicitly characterize different mapping functions and thus affect the complexity of the distribution of sample data in the subspace, which will determine the minimum empirical error that can be achieved for a linear classification surface. The penalty factor  $C$  is used to determine the proportion of confidence range and empirical risk in the subspace of the data, which affects the generalization ability of the learning machine by adjusting the level of

empirical error in the feature space. When  $C$  is large, it means that the penalty for empirical error is large, and the complexity of the learning machine is large but the empirical risk value is small, i.e., "over-learning"; conversely, it is "under-learning". Therefore, only when the kernel parameter  $g$  and the penalty factor  $C$  are matched with each other, the excellent generalization capability can be obtained.

## 4.2 Feature parametric extraction

From the principle of SVM, it is clear that when building a prediction model, the inputs and outputs of the model are first determined. Using the characteristic covariates of the electrode wear level extracted from the electrode imprinted image as input and the number of welds as output, a classification prediction model is used to predict the number of welds of the electrode based on the electrode wear.

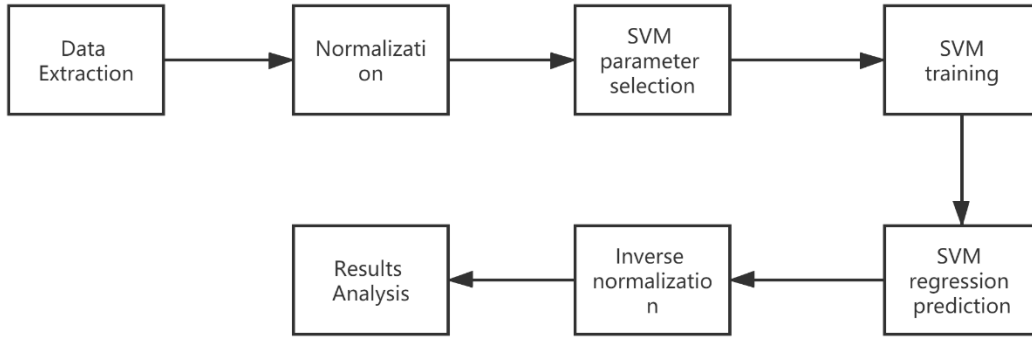


**Figure 4.2.1 SVM model**

### 4.2.1 SVM regression prediction model

A model of electrode lifetime regression prediction was developed based on SVM in MATLAB environment with different feature parameters as model inputs and lifetime interval model outputs for the feature parameters obtained from the previous analysis. The training (svmtrain) and prediction (svmpredict) functions used in the model are derived from the Libsvm software package, which is a simple, easy-to-use and fast and effective software package for SVM pattern recognition and regression developed by Professor Lin Chih-Jen et al. at National Taiwan University<sup>[32,50]</sup>.

The algorithmic flow of the SVM regression program is shown in Figure 4.2.2. It mainly consists of data extraction, data preprocessing, parameter selection, network training, regression or classification prediction, and result analysis.



**Figure 4.2.2 SVM flowchat**

#### 4.2.2 Data extraction and pre-processing

MATLAB writes functions and uses Image Batch processing to batch process the image and extract the electrode imprinting feature parameters.  $TSX[80 \times 4]$  constructs the model input vector from the feature parameters, 80 is the number of groups of feature parameters and 4 is the number of dimensions of feature parameters;  $TS[80 \times 1]$  outputs the vector from the number of electrode welds, 80 is the number of groups of feature parameters. Normalization is an effective way to simplify calculations and reduce the magnitude, i.e., transforming a dimensional expression into a dimensionless expression, so that the absolute value of the physical coefficient becomes some kind of relative value relationship. Normalization is done for the convenience of data processing later, so that the input components are given equal importance at the beginning of the network training, and secondly, to preserve the convergence of the program when it runs faster. There are various methods to perform normalization, the main ones being the minimum maximum, root mean square, mean and standard deviation methods.

$$y = (y_{max} - y_{min}) * (x - x_{min}) / (x_{max} - x_{min}) + y_{min}$$

MATLAB built-in function `mapminmax()` is a normalization function designed based on the minimum maximum algorithm. The minimum maximum algorithm is shown in the above formula, where  $y$  is the result of processing,  $y_{max}$  is the maximum value of the range to be normalized,  $y_{min}$  is the minimum value of the range to be normalized,  $x$  is the data to be processed,  $x_{min}$  is the minimum value of the processed data,  $x_{max}$  is the maximum value of the processed data.

$[y,ps] = \text{mapminmax}(x,min,max)$  is the normalized function call format.

$y$  is the normalized data;  $ps$  is the normalized mapping structure, which is used to denormalize the training results;  $x$  is the data to be normalized;  $min$  is the minimum value of the range to be normalized;  $max$  is the maximum value of the range to be normalized.

Model in the training process, as the model training input and output data if not transformed processing, is bound to make the value of large components of the absolute error, the value of small components of the absolute error is small, the result is a small share of the total error in the relative error of the components of the larger. In the electrode embossed image features, there are thousands and thousands of areas and hundreds of perimeters, while the number of welds may be only single digits. They differ significantly in value and magnitude. Therefore, the input and output vectors are normalized. For normalization, the input and target vectors are combined together as a whole to be normalized by the same algorithm. Normalize the model in the range of  $[0,1]$ . Normalizing the vector space does not change its trend, only the amplitude of the signal is changed.

#### 4.2.3 Parameter optimization selection

The main factors that determine the performance of SVM are the kernel parameter  $g$  and the penalty factor  $C$ . Therefore, the selection of suitable and mutually matching  $g$  and  $C$  is the condition that ultimately determines the regression prediction results of the model. The process of parameter optimization based on the classification accuracy is as follows: first, a range of possible  $g$  and  $C$  is defined, and all the values of  $g$  and  $C$  are selected within the range in turn; the selected  $g$  and  $C$  are applied to the SVM training function and cross validation is performed to obtain the corresponding classification accuracy, finally, the set of  $g$  and  $C$  with the highest classification accuracy is selected. Parameter search function:

$[mse,bestc,bestg] = \text{SVMcgForRegress}(train\_label,train,c\_min,c\_mac,g\_min,g\_max,v,cstep,gstep,msestep)$

The meaning of each parameter in the function:  $train\_label$  is the training set label;  $train$  is the training set;  $cmin$  is the minimum value of the range of variation of the penalty factor parameter  $C$  (after taking the logarithm of the base 2), i.e.,  $c\_min=2^{(cmin)}$ ;  $cmax$  is the maximum value of the range of variation of the penalty factor parameter  $C$  (after taking the logarithm of the base of 2), i.e.,  $c\_max=2^{(cmax)}$ ;  $gmin$  is the minimum of the range of variation of the kernel parameter  $g$  (after taking the logarithm of the base of 2).  $gmin$  is the minimum of the range of variation of the nuclear parameter  $g$  (after

taking the logarithm of the base of 2), i.e.,  $g\_min=2^{(gmin)}$ ;  $gmax$  is the minimum of the range of variation of the nuclear parameter  $g$  (after taking the logarithm of the base of 2).  $gmax$  is the minimum of the range of variation of the kernel parameter  $g$  (after taking the logarithm of the base 2), i.e.,  $g\_min=2^{(gmax)}$ ;  $v$  is the Cross Validation parameter, i.e., the test set is divided into several parts for cross validation;  $cstep$  is the size of the parameter  $C$  step;  $gstep$  is the size of the parameter  $g$  step;  $mstep$  is the step size when the MSE graph is displayed at the end;  $bestacc$  is the highest classification accuracy during cross-validation;  $bestc$  is the best parameter  $C$ ;  $bestg$  is the best parameter  $g$ .

#### 4.2.4 Model Training

Model training function:

*model = svmtrain(training\_label\_vector, training\_instance\_matrix, -s, -t, -g, -c, -v)*

The meaning and optional type of each parameter in the function are as follows.

-s-svm type: SVM setting type (default 0); 0-C-SVC; 1-v-SVC; 2-class SVM; 3 -e -SVR; 4-v-SVR.

-t-kernel function type: type of kernel function setting (default 2).

0 - linear; 1 - polynomial; 2 - RBF function; 3 - sigmoid.

-g- $\gamma$ (gamma): setting of the gamma function in the kernel function

-c-cost: setting the parameters (loss functions) of C-SVC, e -SVR and v-SVR (default 1).

-v-n: n-fold interaction test mode,  $n$  is the number of fold (default 3), randomly split the data into  $n$  parts, and calculate the interaction test accuracy. and calculate the interaction test accuracy and root mean square error.

These parameters can be set in any combination according to the type of SVM and the parameters supported by the kernel function. If a parameter is set that is not present in the function or SVM type, the program will not accept the parameter; if If the parameters are set incorrectly, the default values are used.

The parameters of the training function for the constructed SVM model are set as follows.

*model = svmtrain(TS,TSX,'-s 3 -t 2 -c bestc -g bestg')*

where  $TSX$  is a vector of feature parameters  $[m \times n]$ ,  $m$  is the number of groups of feature parameters,  $n$  is the dimension of feature parameters  $m$  is the number of groups of feature parameters and  $n$  is the dimension of feature parameters;  $TS$  is a vector  $[m \times 1]$  composed of the regression results corresponding to each group of feature parameters;  $model$  is the training model is the model obtained, which is a structure.

#### 4.2.5 Regression prediction

$$[predict, mse] = svmpredict(TS, TSX, model)$$

Where: predict is prediction result; mse is mean square deviation; TS is output vector; TSX is input vector; model is the model obtained by training. The regression prediction result is denormalized to obtain the value with physical dimension.

The inverse normalization function:

$$predict = mapminmax('reverse', predict, TSps)$$

where: reverse is inverse normalization; predict is uninverse normalized regression prediction result; TSps is normalized mapping structure.

Figure 4.2.3 shows the regression prediction data obtained compared with the original data, and the correlation coefficient R reaches 0.9989.

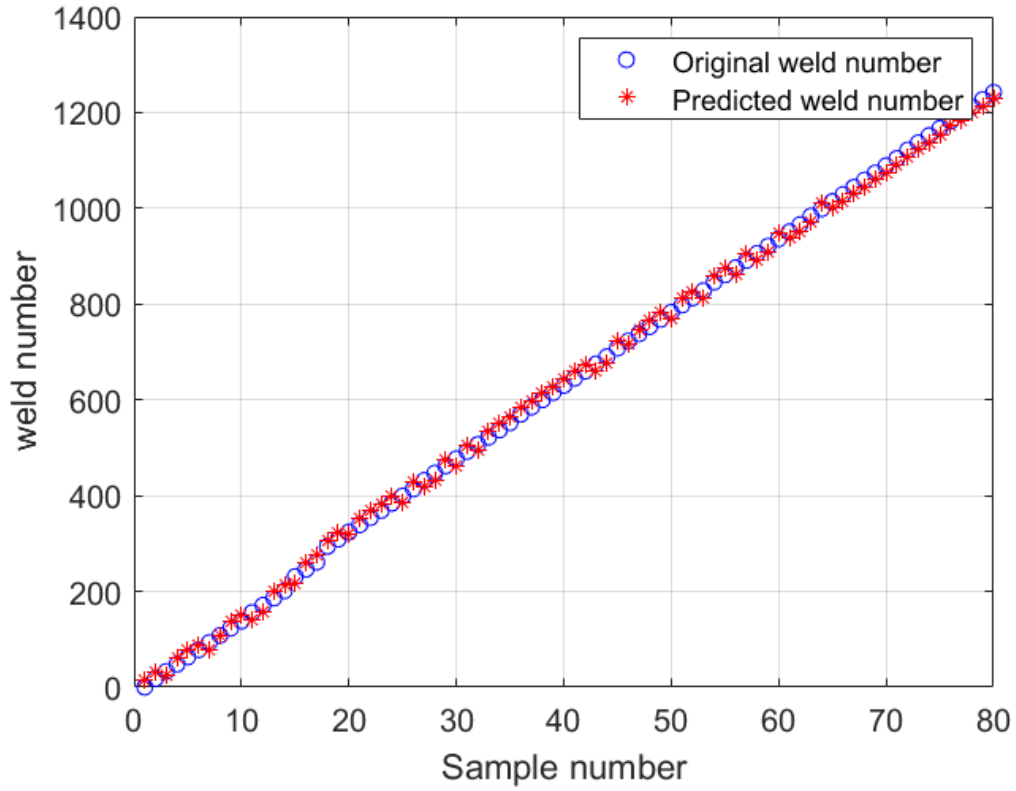
Programming:

```
clear all;
close all;
clc;
data=xlsread('feature.xlsx','Sheet1','A2:E81');
TSX1=data(:,1:4);
TS1=data(:,5);
[temp1,ps1]=mapminmax(TS1',1,10);
TS=temp1';
[temp2,ps2]=mapminmax(TSX1',1,10);
TSX=temp2';

[mse,bestc,bestg] = SVMcgForRegress(TS,TSX,-8,8,-8,8,3,1,1,1.5)
model=svmtrain(TS,TSX,['-s 3 -t 2 -c 500 -g 0.5']);
[predict,mse,dec_values]=svmpredict(TS,TSX,model);
predict1 = mapminmax('reverse',predict',ps1);
predict=predict1'

figure,
plot(TS1,'bo');
hold on;
plot(predict,'r*')
grid on;
xlabel('Sample number');
ylabel('weld number');
legend('Original weld number','Predicted weld number');
```

```
set(gca,'fontsize',12)
```



**Figure 4.2.3 SVM predict result chart**

Below are the formulas for the mean square error and correlation coefficient. The mean square error of the model is 0.0093 and the correlation coefficient is 0.9989.

$$MSE = \frac{1}{l} \sum_{i=1}^l (f(x_i) - y_i)^2$$

$$r^2 = \frac{\left( \bar{l} \sum_{i=1}^l f(x_i) y_i - \sum_{i=1}^l f(x_i) \sum_{i=1}^l y_i \right)^2}{\left( \bar{l} \sum_{i=1}^l f(x_i)^2 - \left( \sum_{i=1}^l f(x_i) \right)^2 \right) \left( \bar{l} \sum_{i=1}^l y_i^2 - \left( \sum_{i=1}^l y_i \right)^2 \right)}$$



## CONCLUSION

The formation process of the car body resistance spot welding joint is a complex process of multiple factors coupled with each other. The random appearance of any fault factor will cause instability in the strength of the spot welding joint, and also threaten the appearance quality of the body welded joints. Electrode wear is closely related to the quality of welded joints and it can reflect the quality of welded joints. The following research work is carried out in the paper with the help of artificial intelligence, with the objective of judging the degree of electrode wear and using the electrode surface image as an information source for the prediction of the number of welds experienced by the electrode.

This paper briefly introduces the MATLAB software. MATLAB has a strong capability for mathematical problem solving, which is convenient with us for dealing with many problems based on mathematical theory, including digital image processing. With the powerful mathematical function library provided by MATLAB, the function library provided for image processing, and the friendly user interface, we can understand the theory of image processing and master the basic methods of image processing more easily and efficiently. Due to my limited knowledge and mastery of image processing, I have only given a preliminary introduction and a rough analysis of some of the more commonly used methods in image edge detection technology in this paper. I hope that through further study and research in image processing in the future, I will be able to master the relevant knowledge and techniques more maturely.

The results show that the median filter and LOG algorithm are suitable for edge extraction of welded joint images, and the binary images are more satisfactory. The electrode surface image feature covariates P/A/A1/R were extracted, and they all have a large correlation with the number of welds and can be used as feature covariates to predict the number of welds experienced by the electrode.

After extracting the feature covariates from the surface images, the minimum maximum algorithm is applied to normalize the model vector space. The trend of the original signal does not change after the normalization process, but the amplitude of the signal is changed. Based on the principle of minimum classification accuracy, the optimal parameters of the model are selected. The SVM regression prediction model was developed using the feature parameters of the surface image (P/A/A1/R) as the input vector and the actual number of welds as the output vector. The correlation coefficient of the regression prediction results of the proposed model reached 0.9989 with a mean square error of 0.0093, indicating that the established SVM prediction model can achieve the evaluation.

Nowadays, how to improve electrode life has been of great interest to companies and researchers, and the degree of electrode wear as one of the main aspects affecting electrode life. Thus, the prediction model of the number of electrodes experiencing weld seams established in this paper can effectively predict the degree of electrode wear by extracting the feature parameters of the image after electrode wear. The method explored is proved to be feasible, but there are still many aspects that need to be improved, such as using more accurate instruments for its measurement, better

extraction of feature parameters in the electrode images, and increasing the number of samples. In addition, the research method proposed in this paper can help to carry out more in-depth research. The degree of electrode head wear should be combined with the improvement of electrode life, and by predicting the degree of electrode head wear, the number of electrode resharpening should be reduced and the electrode life should be improved by changing the process parameters appropriately without affecting the quality of the welded joints, which will improve the productivity of the company.

## BIBLIOGRAPHY

- [1] Jian C, Feng Z. IR-based spot weld NDT in automotive applications[C]. Spie Sensing Technology + Applications, 2015.
- [2] Ltd T. What Is Spot Welding? [J]. TWI Ltd.: Cambridge, 2020.
- [3] Deniz C, Cakir M. In-line stereo-camera assisted robotic spot welding quality control system[J]. Industrial Robot: An International Journal, 2018.
- [4] Hongyan, Zhang, Hu, et al. A Statistical Analysis of Expulsion Limits in Resistance Spot Welding[J]. Journal of Manufacturing Science & Engineering, 2000.
- [5] Zhang H. Expulsion and its influence on weld quality[J]. Welding Journal, 1999, 78(11): 373-s.
- [6] Bhattacharya S, Andrews D R. RESEARCH AND DEVELOPMENT - SIGNIFICANCE OF DYNAMIC RESISTANCE CURVES IN THE THEORY AND PRACTICE OF SPOT WELDING[J], 1974.
- [7] Glagola M A, Roest C A. Nickel Plated Electrodes for Spot Welding Aluminum[C]. Automotive Engineering Congress & Exposition, 1976.
- [8] Patrick E P, Auhl J R, Sun T S. Understanding the Process Mechanisms Is Key to Reliable Resistance Spot Welding Aluminum Auto Body Components[J]. Sae Report, 1984, 840291.
- [9] Ikeda R, Yasuda K, Hashiguchi K. Resistance spot weldability and electrode wear characteristics of aluminum alloy sheets[J], 1998.
- [10] D M U. Effects of contact resistance in resistance welding of aluminum[J]. Welding Journal, 1984.
- [11] Ashton R F. An Arc-Cleaning Approach for Resistance Welding Aluminum[J]. W J, 1976.
- [12] Holliday R J, Parker J D, Williams N T. Prediction of electrode campaign life when spot welding zinc coated steels incorporating electrode tip dressing operations[J]. Ironmaking & Steelmaking, 1996, 23(2): 157-163.
- [13] Lu F, Dong P. Model for estimating electrode face diameter during resistance spot welding[J]. Science and technology of welding and joining, 1999, 4(5): 285-289.
- [14] Chatterjee K L, Waddell W. Electrode wear during spot welding of coated steels[J]. Welding & Metal Fabrication, 1996, 64(3): p.110-114.
- [15] P., Dong, M., et al. Finite element analysis of electrode wear mechanisms: face extrusion and pitting effects[J]. Science & Technology of Welding & Joining, 1998.
- [16] Peng, J., Fukumoto, et al. Image analysis of electrode degradation in resistance spot welding of aluminium[J]. Science & Technology of Welding & Joining, 2004.
- [17] Fukumoto S, Yamamoto A, Tsubakino H, et al. Relation between electrode tip life and electrode degradation in resistance spot welding of aluminum alloy[J]. Welding Journal, 2003, 82(11): 307s-312s.
- [18] Parker J. Relative contribution of electrode tip growth mechanisms in spot welding zinc coated steels[J]. Welding in the World, Le Soudage Dans Le Monde, 1996, 4(37): 186-193.
- [19] Wei L, Cheng S, Hu S J, et al. Statistical Investigation on Resistance Spot Welding Quality Using a Two-State, Sliding-Level Experiment[J]. Journal of

Manufacturing Science & Engineering, 2001, 123(3): 3829-38.

[20] Wei L. Modeling and On-Line Estimation of Electrode Wear in Resistance Spot Welding[J]. Journal of Manufacturing Science & Engineering, 2005, 127(4): 709-717.

[21] Castleman K R. DIGITAL IMAGE PROCESSING Solution Manual - Part Three[M]. DIGITAL IMAGE PROCESSING Solution Manual - Part Three, 1996.

[22] Ruisz J, Biber J, Loipetsberger M. Quality evaluation in resistance spot welding by analysing the weld fingerprint on metal bands by computer vision[J]. The International Journal of Advanced Manufacturing Technology, 2007, 33(9-10): 952-960.

[23] Abdulhadi A, Gdeisat M, Burton D, et al. Assessing the Quality of Spot Welding Electrode Tips Using Image Processing Techniques[J]. Lecture Notes in Engineering & Computer Science, 2011, 2191(1): 1557-1562.

[24] Liu T, Wu W, Chen W, et al. Automated image-processing for counting seedlings in a wheat field[J]. Precision Agriculture, 2016, 17(4): 392-406.

[25] Bhagavathi S L, Niba S T. An automatic system for detecting and counting rbc and wbc using fuzzy logic[J], 2016.

[26] Bolle R M, Connell J H, Haas N, et al. Produce recognition system: US, 1996.

[27] Wang Z, Xing Q, Fu L, et al. Realtime vision-based surface defect inspection of steel balls[J]. Transactions of Tianjin University, 2015.

[28] Lee J J, Shinozuka M. Real-Time Displacement Measurement of a Flexible Bridge Using Digital Image Processing Techniques[J]. Experimental Mechanics, 2006, 46(1): 105-114.

[29] Heart Cavity Segmentation in Ultrasound Images Based on Supervised Neural Networks[J]. Springer-Verlag, 2009.

[30] Sergios T, Dionisis C, Konstantinos K, et al. Introduction to Pattern Recognition: a MATLAB Approach[M]. Introduction to pattern recognition :, 1999.

[31] Hagan M T, Demuth H B, Beale M. Neural network design[M]. PWS Publishing Co., 1997.

[32] Cristianintnello. An introduction to support vector machines and other kernel-based learning methods[J]. Printed in the United Kingdom at the University Press, 2000.

[33] Burges C. A Tutorial on Support Vector Machines for Pattern Recognition[J]. Data Mining and Knowledge Discovery, 1998, 2.

[34] Malin D. Image Processing Toolbox User's Guide - MathWorks[J].

[35] Guerrero J. Tutorial III: Image Processing and Analysis with Matlab[C]. International Conference on Electrical, Communications, and Computers;CONIELECOMP 2009.

[36] <http://www.directindustry.com/prod/mathworks/product-12865-130905.html>[J].

[37] Zhang X Q, Chen G L, Zhang Y S. Characteristics of electrode wear in resistance spot welding dual-phase steels[J]. Materials and Design, 2008, 29(1): 279-283.

[38] Controlling the quality of resistance spot welding by stabilising secondary current during wear of welding electrodes[J]. Welding International, 2008, 22(2): 110-112.

- [39] Oezyuerek D. An effect of weld current and weld atmosphere on the resistance spot weldability of 304L austenitic stainless steel[J]. *Materials & Design*, 2008, 29(3): 597-603.
- [40] Jr R, Min J, Li C J. An intelligent control system for resistance spot welding using a neural network and fuzzy logic[C]. *Industry Applications Conference*, 1995.
- [41] Cullen J D, Athi N, Al-Jader M, et al. Multisensor fusion for on line monitoring of the quality of spot welding in automotive industry[J]. *Measurement*, 2008, 41(4): 412-423.
- [42] Martín ó, López M, Martín F. Artificial neural networks for quality control by ultrasonic testing in resistance spot welding[J]. *Journal of Materials Processing Tech*, 2007, 183(2-3): 226-233.
- [43] Lu G. A multi-scale algorithm to calculate the gradient of image morphology[J]. *Chinese Journal of Graphics*, 2001(3期): 214-218.
- [44] Haralick R M, Shapiro L G. Survey: Image Segmentation Techniques[J]. *Computer Vision Graphics & Image Processing*, 1985, 29(1): 100-132.
- [45] Zhang X. Edge detection of digital images based on MATLAB[J]. *Journal of Jilin Institute of Chemical, Technology*, 2010(02): 59-61.
- [46] Scharcanski, Jacob, Jung, et al. Adaptive Image Denoising Using Scale and Space Consistency[J]. *IEEE Transactions on Image Processing*, 2002.
- [47] Zhu Z, Liu G. Study of first-order edge detection algorithm[J]. *Modern Electronic Technology*, 2009, 32(24): 88-90.
- [48] Min C S, Goldgof D B, Bowyer K W. Comparison of Edge Detector Performance through Use in an Object Recognition Task[J]. *Computer Vision & Image Understanding*, 2001, 84(1): 160-178.
- [49] Yan P, Zhang C. Artificial neural network and simulated evolutionary computation[M]. *Tsinghua University Press*, 2000.
- [50] Chang C C, Lin C J. LIBSVM: A library for support vector machines[J]. *ACM Transactions on Intelligent Systems and Technology*, 2007, 2(3, article 27).