电 子 科 技 大 学

UNIVERSITY OF ELECTRONIC SCIENCE AND TECHNOLOGY OF CHINA

学士学位论文

BACHELOR THESIS

论文题目　基于机器视觉的轴孔零件测量选配系统设计

专　　业　机械设计制造及其自动化

学　　号　2014080106018

作者姓名　谢明友

指导教师　李迅波　教授

摘　要

机器视觉检测技术是当前精密测量技术领域的研究热点，它以非接触测量、自动化程度高、测量精确稳定等优点，在现代化工业生产领域占据着愈发重要的位置。本论文正是基于机器视觉检测技术，以活门的主活塞的轴和导向盘的孔作为主要检测对象，开发出了一套能够精密快速地对轴孔零件进行尺寸测量并实现自动选配的系统软件。

本文首先在机器视觉检测系统的通用模式基础上分析了轴孔零件尺寸测量系统的实际需求和设计难点，确定了系统的总体设计方案，并对机器视觉检测平台的工业相机、镜头和光源进行了选型分析。

轴孔零件图像的处理与测量是本文的重点，通过对各种图像处理方法的算法理论分析和实践效果对比，本文提出采用加权平均值法进行图像灰度变换，中值滤波法对图像滤波去噪，最大类间方差法确定图像二值化的阈值，Canny 算子进行像素级边缘检测，Zernike 矩法实现亚像素边缘定位，然后通过最小二乘法拟合分别求出轴和孔的像素尺寸。

为进一步提高系统尺寸测量的精度，本文对系统进行了标定，采用张正友标定法获得了成像校正所需的相机内部参数。通过实验标定得到轴和孔的标定系数后，将它们的像素尺寸转化为实际尺寸，并进行了多次重复测量，结果表明本文尺寸测量系统的检测精度达到 0.03mm，测量效率在 1s/件以内，达到了系统预期设计目标。此外，本文还对系统的测量误差进行了分析。

针对轴孔零件选配过程中人工选配和一对一试配法的弊端，本文以最大数量配对为准则，设计了轴孔零件的自动选配算法，对已知尺寸的多组轴孔零件进行选配，结果表明该算法配对成功率高、效率高，满足了实际应用需求。

最后，本文根据机器视觉轴孔零件测量选配系统的设计目标分析了系统软件所需实现的性能指标和开发要求，总结了系统的工作流程，并对系统软件的功能和界面进行了设计。

关键词：机器视觉，边缘检测，图像处理与测量，系统标定，最大数量选配，系统软件设计

ABSTRACT

Detection technology based on machine vision is a research hotspot in the field of precise measurement technologies. It takes advantage of non-contact measurement, high degree of automation, accurate and stable measurement. And it occupies an increasingly important position in modern industrial production. This paper is exactly based on detection technology of machine vision. With the shaft of the main piston and the hole of the guide plate as the main detection objects, a set of system software capable of accurately and quickly measuring the size of the shaft and hole and automatically matching is developed.

This paper firstly analyzes the actual requirements and design difficulties of the size measurement system of shaft and hole parts based on the general model of detection system of machine vision, then determines the overall design scheme of the system and selects the proper industrial camera, lens and light source for the detection platform of machine vision.

The processing and measurement of the image of the shaft and hole parts is the focus of this paper. After analysing the algorithm theories of various image processing methods and comparing their practical effects, this paper proposes the use of the weighted average method for gray transformation, the median filter method for filter denoising, the OSTU method to determine the threshold of image binarization, Canny operator for pixel-level edge detection and Zernike moment method to achieve sub-pixel edge location. And then we obtain the pixel sizes of the shaft and hole by least-squares fitting.

In order to further improve the accuracy of size measurement, the system need to be calibrated. And the camera internal parameters required for imaging correction can be obtained by using Zhang Zhengyou calibration method. After the calibration coefficients of the shaft and hold are determined through calibration experiment, their pixel sizes can be converted into actual size. The results of repeated measurements show that the accuracy of the size measurement system in this paper reaches 0.03mm, and the efficiency of measurement is within 1s/piece, which have achieved the design expectations of the system. In addition, this article also analyzes the measurement error of the system.

In the light of the disadvantages of manual matching and one-to-one matching method, an automatic matching algorithm for the assembling of shaft and hole parts is proposed based on the principle of maximum-number matching. The matching results

show that the algorithm has high success rate and high efficiency, which meets the practical application requirements.

Finally, according to the design goal of measurement and matching system for shaft and hole parts based on machine vision, this paper analyzes the performance quotas and development requirements of the system software, summarizes the work flow and designs the function and interface of the system software.

Keywords: machine vision, edge detection, image processing and measurement, system calibration, maximum-number matching, system software design

目 录

第一章 绪 论

1.1 课题研究的背景与意义

在现代的机械加工行业，随着高精度数控机床的广泛使用，特别是近几年兴起的 3D 打印技术的逐渐应用，机械零部件的加工工艺有了较显著的提高。作为生产流程中的重要一环，机械零部件的加工精度直接影响到最终产品的质量和性能[1]。

随着科学技术的不断发展，传统的机械零部件加工工艺除了向高精度、高效率和高等材料方向发展外，机械加工自动化也是重要发展方向之一[2]，伴随着整个机械加工行业的技术升级，对机械制成品的检验检测有了更高的要求。

本课题来源于航天某厂活门智能装配生产线中的智能检测选配单元。活门是航天运载火箭增压运输系统的重要元件，用于实现气体传输、截止、调节、排放、超压保护和推进剂加泄、溢出等功能，作为运载火箭进行运动的精密机械产品，活门性能直接关系着航天设备的可靠性。而主活塞和导向盘是活门的重要组成零件，它们的尺寸精度对活门的质量稳定性和性能起着至关重要的作用。

目前对主活塞的轴径和导向盘的孔径尺寸测量主要采用人工检测，即通过直尺、千分尺、游标卡尺、万能工具显微镜等传统测量工具进行测量。这种传统的测量方式主要存在以下不足：一是自动化程度低，检验员工作强度大，工作效率偏低；二是测量结果的准确性受制于检验员的检验水平以及工作疲劳程度，如长时间快速的测量，会导致精力不集中，读数误差增大；三是无法实现对大量测量结果的快速保存，动态分析企业生产状况。这些问题大大制约了零件尺寸精度和生产效率的提高，因此需要引入更先进的检验技术。

现代精密测量依靠的技术手段主要有超声波检测、机器视觉检测、光电技术检测和多传感综合检测等类型。其中机器视觉检测技术凭借着非接触测量、自动化程度高、测量精确稳定的优点，市场应用规模越来越大，是现代检测技术中最热门的分支之一[3]。

将机器视觉检测技术用于活门零件尺寸测量，具有以下几个优点：一是易于信息集成和管理，实现智能检测，可一次性实现多个尺寸测量；二是可用于长时间不间断的工作，检测精度和检测效率不受工作时间影响，降低企业生产成本；三是实现非接触测量，避免了磨损或者划伤待检测零件，扩大了检测对象的范围；四是能够保证产品检测的完整性和一致性，避免因人工检测疲劳而产生的误检、漏检尺寸的情况。

此外，活门装配是活门制造的关键环节，活门的主活塞的轴与导向盘的孔存在

配合关系，但在目前它们的选配都由人工在装配过程中完成，难以达到最佳的匹配效果，也降低了装配效率。因此，有必要通过设计零件的自动选配算法解决人工选配带来的问题，使轴和孔的配合间隙满足装配精度要求，同时提高活门装配生产线的自动化程度。

1.2 机器视觉检测技术概述

机器视觉检测技术作为精密测量技术领域的新技术之一，综合使用光电测量、自动控制、计算机、数字图像处理等多个领域的技术把机器视觉与工业检测相结合，具有非接触测量、检测效率高、数据处理灵活、响应速度快、适应性强等优点，在现代化工业生产领域占据着愈发重要的位置[4]。机器视觉检测可以高效获取丰富的数据信息，便于提高生产效率和自动化程度[5]与工业设计信息和加工控制信息结合度高，易于集成管理，具有很强的发展潜力和广阔的应用前景[6]。

机器视觉检测指的是通过机器视觉方法获取被测目标的图像信息，将之与预先得到的标准进行比对，进而确定被测目标的质量情况的过程，实质上是基于给定的标准数据对比确定偏差的过程。在检测中主要关注指定目标的特征信息如配件完整性、表面完好程度以及几何尺寸的测量等。机器视觉检测系统采用相机获取目标图像，并将之转换成数字图像信号，通过先进的计算机软、硬件技术对数字图像信息进行处理，从而获得检测需要的目标特征值图像信息，以此为基础实现灰度分布图、坐标计算、模式识别等多种功能[7]。

机器视觉检测系统的测量精度是检测系统能否得到应用的关键，但由于视觉测量系统涉及光、机、电等多个方面，故影响测量系统精度的因素很多，主要包括硬件系统的误差，标定误差和软件误差三种，其中硬件系统的误差主要来源于光学镜头的畸变误差，标定误差是指在进行系统标定时引入的误差，软件误差是指在进行图像分割、目标特征提取和亚像素定位等图像处理时由于软件算法而带来的误差。对于同一个测量对象，提高测量系统测量精度可以从以下几个方面着手进行，包括提高相机的分辨率，采用最适合的标定技术和采用高精度的亚像素定位技术等。

1.3 国内外研究现状和发展趋势

上世纪八十年代初 ，美国和欧洲率先开始了对机器视觉检测技术的普遍研究，并把机器视觉检测技术运用于物体尺寸的测量，如今已大放异彩。瑞典海克斯康公司推出的三坐标测量机，使用计算机进行自动控制和非接触图像测量，实现了高速处理和自动化测量。日本精密仪器厂商三丰公司研制的数控机床图像三坐标测量

机，能够利用自带的多功能检测系统来测量复杂形状的工业配件，并可对工件实现动对焦，测量精度达到$(1.7+4L/1000)\mu m$。美国 OGP 公司生产的 ZIP 系列图像测量系统既具有强大的影像测量能力，还额外配置了多元传感测量装置，其检测结果能够达到$(2.5+L/1000)\mu m$的单轴精度。意大利公司 COORD3 公司生产的数控机床三坐标测量机，可以实现大尺寸工件的精密图像测量，测量精度能够达到$(3.5+3L/1000)\mu m$。瑞士的 Hauser 公司生产的光电测量投影仪 H602，在一般投影仪的技术基础上，使用高精度的光栅定位仪器配合计算机处理器，以 CCD 相机作为光电瞄准设备，实现了动态瞄准和采样，兼顾了测量的高精确度和高效率。

我国从上世纪九十年代才开始重视机器视觉检测技术的研究，行业产品尚不成熟。各行业的大多数工业企业尚未完全实现生产自动化、信息化的问题，无暇顾及机器视觉检测技术，因此国外的机器视觉检测研究和应用水平超前国内很多。

目前，国内多所大学也开展了机器视觉尺寸测量技术的研究，取得了喜人的成果，但是由于起步较晚，目前技术还不成熟和完善，很多场合仍属于概念导入阶段。张平生等设计的机器视觉系统能够对管孔零件尺寸自动测量，使测量精度维持在 0.01mm 左右[8]。王俊元等利用机器视觉方式，采用 IMAQ Vision 视觉软件，实现对圆筒形零件的直角梯形槽的槽宽检测，所开发的系统重复精度较高，精度维持在 0.002mm 左右[9]。卞晓东等设计的车辆几何尺寸参数测量系统，对车辆几何尺寸进行了测量，采用 Canny 算子、Hough 变换等，利用立体视觉的照相机阵列，使测量极限误差不超过 2mm[10]。吴江等对 O 型密封圈尺寸开展了视觉测量的研究，并利用开发的系统针对外径 36mm、线径 1.7mm 的 O 型密封圈进行测量，误差分别维持在 0.007-0.01mm 和 0.002-0.003mm 左右[11]。

通过检索多方面的文献资料，可以发现我国机器视觉检测技术的发展仍然存在以下阻滞[6]：

（1）机器视觉检测核心设备依靠进口，国产化低，价格昂贵，经济性差。

（2）机器视觉检测设备检测功能较单一，难以同时检测多种对象。

（3）机器视觉检测软件以及核心算法被国外大型视觉厂商所垄断，国产化大面积使用的机器视觉软件寥寥无几。

（4）由于机器视觉检测技术起点高、投入大、产出慢的现状，部分工业企业对视觉检测研究的投入与产出不成正比，从而打击中小型企业研究和使用机器视觉检测技术的兴趣。

机器视觉检测产品要想真正在多种领域被广泛使用，未来发展趋势将会是：

（1）集成功能逐渐增多，测量对象多样化。

（2）价格持续下降，产品小型化和智能化。

（3）测量精度和可靠性进一步提高。

（4）测量效率进一步提升，最终实现实时在线检测。

1.4 论文的主要研究内容

本论文以活门的主活塞的轴和导向盘的孔作为主要检测对象，针对这两种类型的零件特点，结合实际需求，选用合适的相机、镜头以及光源等机器视觉检测平台硬件，设计图像处理与测量算法和轴孔零件自动选配算法，并开发出一套能够精密快速地对轴孔零件进行尺寸测量并实现自动选配的系统软件。

本论文的章节分为七章，各章的内容安排如下：

第一章，阐述本课题设计的研究背景和意义，总体介绍机器视觉检测技术的概念、优点和关键技术，分析机器视觉检测技术的国内外研究现状和发展趋势，并介绍本论文的主要研究内容。

第二章，介绍介绍机器视觉检测系统的通用模式，在此基础上对本文机器视觉尺寸测量系统进行需求分析，设计出合理的软硬件系统，并确定对图像成像有重要影响的工业相机、光学镜头、照明光源的选型。

第三章，介绍图像处理与测量流程中各步骤的关键技术和方法，图像处理步骤包括图像的灰度变换、图像平滑、图像二值化、像素级边缘检测、亚像素级边缘定位、像素尺寸测量等。运用各方法对本文零件图像进行处理，比较它们的处理效果，尤其对边缘检测和定位算法的原理进行分析，讨论各经典算法的优缺点和适用范围，选定各图像处理步骤中能使本文轴孔零件图像处理效果达到最佳的算法，并测量出精确的轴孔零件像素尺寸。

第四章，建立图像成像的几何模型，分析比较各相机标定方法的优缺点，采用张正友标定法对相机镜头进行标定以获取成像畸变校正所需的相机内部参数。在确定系统标定系数后，将轴孔零件的像素尺寸转化为实际尺寸，通过实验验证本尺寸测量系统的测量精度和效率能达到预期设计目标，并对测量误差进行分析。

第五章，针轴孔零件选配过程中人工选配和一对一试配法的弊端，以最大数量配对为准则，设计本文轴孔零件的自动选配算法，对已知尺寸的多组轴孔零件进行选配，，验证该算法的配对成功率和效率能达到实际运用需求。

第六章，根据基于机器视觉的轴孔零件测量选配系统的的设计目标分析系统软件需实现的性能指标和开发要求。总结系统的工作流程，对系统软件的功能和界面进行设计。

第七章，回顾并概括全文，对论文所做的工作进行总结，并对后续的研究工作提出一些建议和展望。

第二章 机器视觉尺寸测量系统的总体方案

2.1 机器视觉检测系统的通用模式

典型的机器视觉检测系统穿插融合了多学科、多领域技术，能实现对多对象、多任务、多目标进行检测，应用范围广泛，结构形式变化多样。但是通用的机器视觉检测系统的检测机理、处理过程、检测方法、工程步骤以及核心部件大致是相同的，只是各个模块的具体构成存在差异，包括硬件的选型、图像处理的算法和检测的类别等。典型机器视觉检测系统的通用模式中的模块构成和数据流方向如图 2-1 所示。

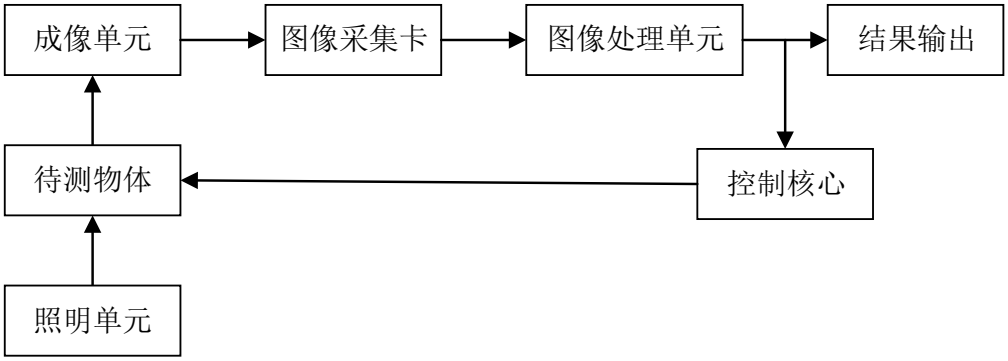


图 2-1 机器视觉检测系统的通用模式

（1）照明单元：属于辅助成像器件，为图像采集提供前置照明条件。照明单元的目的是为待测物体提供广义照明，照亮待测物体，突出待测物体的待测特征与周边环境的对比度，其中包括可见光和非可见光。相机拍摄到清晰图像的前提是正确选择光源和照明方式，因此照明单元对于机器视觉系统来说非常重要[12]。

（2）成像单元：包括镜头和相机。镜头将待测物体反射或遮挡的光线聚焦于相机的图像传感器靶面，实现光束变换，从而获得清晰、特征鲜明的待测物体的数字化图像。镜头是实现系统放大倍率和缩小倍率的基础，相机是将聚焦光线转化为数字图像的前段采集设备，它们共同组合而成的成像单元是机器视觉系统中的重要组成部分，不可缺少[13]。

（3）图像采集卡：提供机器视觉系统中图像采集设备（相机）与图像处理设备（计算机）的接口，是对相机所输出的视频模拟信号实时采集和转换，并通过与计算机的高速接口，将数字化后的图像数据通过总线高速传输至图像处理单元[14]。图像采集卡正逐渐被集合了 IEEE1394 或 USB 接口的工业数字相机和智能相机所取代，发挥的作用正逐渐减弱[15]。

（4）图像处理单元：以图像处理软件和计算机或嵌入式系统为基础，针对成像单元采集的数字图像，完成图像的分析计算，如图像增强、图像滤波、灰度直方图的计算、定位、搜索、角点和几何边缘的提取等[16]。

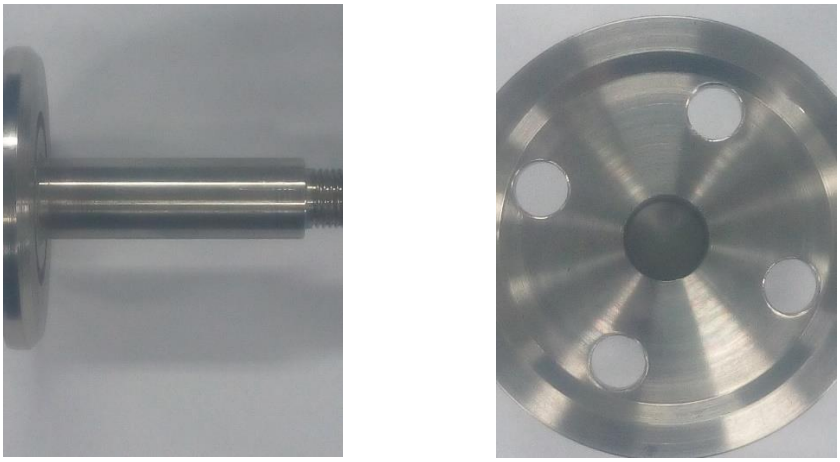
（5）结果输出：将图像处理后，将得到的感兴趣的数据或特征，在显示屏或其他图形界面进行输出和显示，如尺寸大小、目标边缘、缺陷图例等。

（6）控制核心：根据图像处理后得到的结果，控制设备的执行机构，执行相适应的操作，如自动分级系统中的产品分流，质量检测系统中的缺陷报警等[17]。

上述六个部分构成了机器视觉检测系统的通用模式，具有一定的适用性。但是在实际工程应用中会针对不同的需求和功能，选用不同的形式和部件，尤其是集合了图像的采集、简单处理与通信功能的智能相机的出现，为实现具有多功能、模块化、高可靠性嵌入式的微小型机器视觉系统提供了多种可能。

2.2 尺寸测量系统的需求和难点分析

本文的的两个检测对象是活门中的关键零件主活塞和导向盘，对主活塞测量其轴径尺寸，对导向盘测量其孔径尺寸，它们的尺寸均为 16mm 左右。测量的精度和效率是评价该系统的重要指标。图 2-2(a)和图 2-2(b)展示了待检测的零件样本。



(a)　　(b)

图 2-2 待检测的零件。(a)主活塞的轴；(b)导向盘的孔（中间的孔）

本课题的合作企业，其活门的主活塞和导向盘的生产批量大，但产品的结构基本保持不变，为满足轴孔零件尺寸的精密快速测量，结合企业的生产现状，本文基于机器视觉检测技术所设计开发的轴孔零件尺寸测量系统的性能要求如下：检测精度为 0.05mm，检测效率为 1 件/s。

通过对主活塞和导向盘的设计图纸，结合实物分析，归纳出以下几项检测难点：

（1）构建良好的图像采集环境。采集到满足要求的图像是高精度测量的前提条件，在实际环境中由于受到光线、温度、电磁干扰等因素的影响，采集到的图像可能会出现虚化、特征不明显等缺陷。

（2）机器视觉检测平台的搭建。该测量系统对机器视觉检测平台的安装要求较高，必须合理安排镜头视场和角度。比如检测平面必须与相机和镜头的光轴垂直，在检测导向盘的孔径时，其圆心必须在视野的正中心，否则会产生透视畸变，影响检测的精度。

（3）相机镜头的标定。由于透镜等装置在生产过程中不可避免地存在误差，图像采集设备无法按照理想的情况成型，物体在成像平面所成的像点由于光学畸变存在偏差，需要通过标定对镜头畸变进行校正以减小测量误差。

（4）边缘检测算法的研究。不同的检测算法具有不同的适用性，各自的检测精度和具体表现也有所不同。通过分析现有的边缘检测算子，提出一种符合本文系统要求的边缘检测算法，兼顾图像尺寸测量的精度和处理效率，能快速有效地定位目标边缘。由于像素级边缘检测算法的测量精度依赖于图像采集设备的分辨率，需要使用亚像素边缘定位算法进一步提高测量的精度。

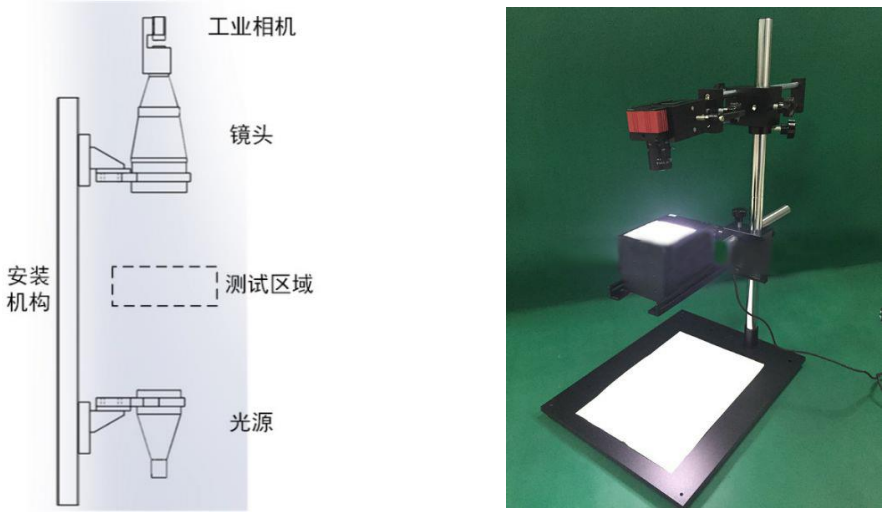
（5）此外，本文的测量对象轴和孔是两类不同的零件，在对它们进行尺寸测量时，提取的边缘形状特征完全不同，轴的边缘是两条直线，而孔的边缘是一个圆。因此，为得到较精确的尺寸测量结果，必须分别设计两套相对独立的图像处理与测量算法，这导致工作量倍增。

2.3 尺寸测量系统的设计方案

本系统的组成分为两部分：硬件系统和软件系统。硬件系统主要完成主活塞的轴和导向盘的孔图像的采集，软件系统主要完成图像的处理和零件的尺寸检测。

2.3.1 硬件系统设计

本硬件系统由计算机、工业相机、光学镜头、图像采集卡、照明光源和工作台等部分组成。其机器视觉检测平台如图 2-3 所示。由于主活塞和导向盘属于非透明体，为使采集到的图像对比度高，能够突出轴和孔的轮廓，有利于后续图像的处理，故采用背向照明的方式。背向照明是指被测物体放在光源和相机之间，光源发出光波，经环形光管照射于被测零件，使零件尽量在均匀照明的可控背景中，经光学系统成像于光敏面上，图像采集卡接受从相机中输入的模拟电信号，由 A/D 转换为离散的数字信号，与计算机进行通讯，计算机通过对采集的图像数据进行处理，从而精密快速地计算出零件的尺寸，并将结果输出显示。



(a) (b)

图 2-3 机器视觉检测平台。(a)模型图；(b)实物图

2.3.2 软件系统设计

本软件系统的基本过程如图 2-4 所示，包括了图像采集、图像预处理、特征提取、边缘检测、尺寸测量等步骤。其中，图像采集主要通过硬件实现，图像预处理可细分为灰度变换、二值化处理、滤波去噪、畸变校正等。边缘检测是对图像进行逐个像素的检测，并通过选取适当的阈值筛选出更加理想的检测结果。对处理算法得出的图像数据进行进一步计算，得到目标物体在图像中的准确坐标，继而通过标定系数计算出世界坐标系中该物体的实际尺寸。

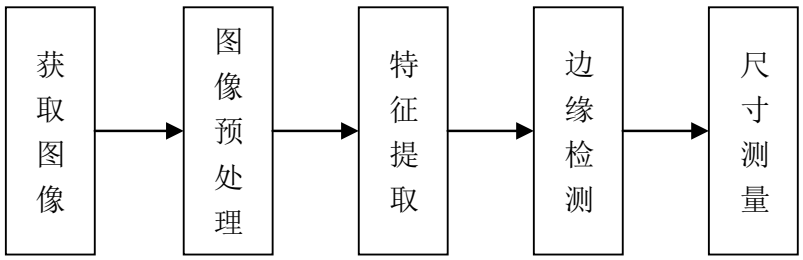


图 2-4 图像测量的基本过程

2.4 尺寸测量系统的硬件选型

系统硬件的选型是后续图像处理的前提与保障，通过正确选型的硬件进行图像采集，将获得高质量的图像，这将降低图像预处理的难度和有利于图像特征的提取，从而能够降低机器视觉测量的误差，因而在选用设备时需要综合考虑设备的性能和配套程度。下面简单阐述用于感光成像的工业相机、用于改变光路的镜头以及

提供光照的光源这三种硬件设备的选择原则。

2.4.1 工业相机选型

工业相机能够得到较稳定的图像，能够在抵御外界干扰的同时快速地传输图像信号。目前工业相机的感光传感器包括 CCD 或 CMOS 传感器。CCD 相机进行图像采集时，首先进行光电互换，存储得到的电荷，通过电荷移动实现电信号的传递，最后对该信号进行读取。CCD 传感器是典型的以电荷作为信号的成像传感器。CCD 成像过程中不容易被烧坏，响应速度快，没有延迟，且工作功率低。

随着大型集成硅制造工艺的发展，CMOS 传感器也发展迅速，CMOS 相机成像过程中，将采集到的微弱图像信号进行调理和转换，将获得的数字信号在处理器上进行分析处理，最终输出图像信号。CMOS 传感器能够将感光器与处理电路集成在小芯片上，工作功率小，信号传输时速度快，越来越成为用户的首选。

与 CCD 传感器相比，CMOS 传感器集成度高、体积小、价格低、功耗小，在高分辨率像数的使用要求中，CMOS 传感器具有更大的优势。因此综合考虑性能参数、相机价格，本文选择使用带 CMOS 传感器的相机。

2.4.2 镜头选型

镜头对相机的作用相当于晶状体对眼睛的作用，使物体能够清晰地在图像传感器上成像。镜头成像时可能存在的像差会导致图像质量受到影响，如导致光束不能交汇主轴同一位置的球差、导致不能在理想平面上形成清晰点的慧差以及像散等。因此需要根据镜头的使用环境和对象来选择合适参数的镜头，减小像差对输出图像的影响。

在选取工业镜头时一般需要根据分辨率、最大相对孔径以及景深等几个实用参数来判断，其中分辨率体现了镜头辨别细节的程度，景深体现了在成像平面成清晰像的空间深度范围的程度，而最大相对孔径定位反映了镜头采光的程度。

根据目标尺寸，可以确定镜头的视场角，图像传感器的尺寸。目标尺寸需要全部处于镜头视野之中才能形成完整的目标像，因此镜头视野大小必须大于与之配套的图像传感器的靶面。且由于系统的最高分辨率受制于传感器的像元分辨率，系统的分辨率并不会由于镜头分辨率的增大而无限制的增大，因此选择镜头的分辨率只要略高于像元分辨率即可。

由于景深影响、待测物体三维空间位置的变化、加工制造安装误差等因素，要把待测物体的光学像精确调焦到与传感器靶面完全重合，技术实现难度较大，进而导致成像的放大倍率不固定，存在变化，影响了测量精度。相对于普通镜头，远心

镜头具有高分辨率，超宽景深，超低畸变以及独有的平行光设计，使物距和工作距离之差小于景深，光学放大倍数不变，从而实现恒定的测量精度。因此本文选择采用远心镜头。

2.4.3 光源选型

照明单元的首要目标是降低采集图像的冗余信息，分离目标信息和背景信息，突出图像的目标和特征，降低图像处理的难度，从而提高边缘定位、尺寸测量的精度，使系统的综合性能将相应提高。不合理的光源，不仅增加图像处理的复杂程度，也会使测量误差增大，甚至不能满足系统的需求。因此，选择合适的照明光源是系统硬件选型的重要步骤。

针对本文以灰度进行量化处理的机器视觉测量系统而言，图像亮度和对比度是较为重要的参数，而决定这一重要参数的因素便是光路系统的质量。一般来说机器视觉系统为了避免环境自然光线或灯光对其图像采集工作的影响，照明设计均采用自足光源，且要求光源亮度大、亮度可调、均匀性好、稳定性高，以抑制外界光照对图像质量的影响，光源与照明方案的配合应尽可能地产生明显的区别，增强对比度。除此以外，光源设计还应保证足够的整体亮度，使成像质量不受物体位置变化的影响。设计时应考虑光源的光谱特性、光度特性、发光面、发光效率、光源的热噪声影响。

在光源类型的选择上，常用的照明光源有冷阴极荧光灯、卤素灯、LED 灯等几种。其性能特点可见表 2-1。

表 2-1 常用照明光源的性能对比

光源类型	亮度	稳定性	使用寿命	温度影响	成本	设计难度	品种数量
荧光灯	低	低	一般	一般	低	低	一般
卤素灯	高	一般	短	低	高	一般	少
LED 灯	一般	高	长	低	一般	高	多

对比三者的性能可以知道，荧光灯具有亮度低、光路稳定性差的缺点。而卤素灯虽然光照表现好，但是使用寿命短，成本过高。LED 灯虽然在设计时难度大，但其光照表现出色、使用寿命长且成本较低，特别是随着近些年半导体照明技术的发展，LED 灯的产品数量激增，可供选择面广。综合对比，在本文的轴孔零件尺寸测量系统中选择 LED 灯作为照明光源是比较合适的。

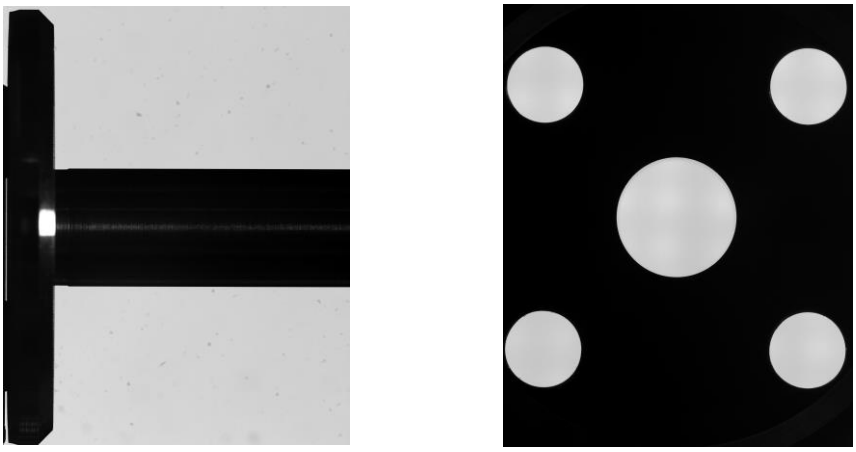
2.5 本章小结

本章首先介绍了机器视觉检测系统的通用模式，研究了各个硬件组成模块的功能及其工作机理，然后对本文轴孔零件尺寸测量系统的需求和难点进行了详细分析，提出了合理的硬件和软件系统设计方案。在总体方案确定的基础上，针对系统的主要硬件组成部分进行了选型分析，选定采用带 COMS 传感器的工业相机作为摄像机，采用远心镜头作为光学镜头，采用 LED 光源作为照明光源。

第三章 图像处理与测量

对于一个稳定快速的机器视觉测量系统，合理选择图像处理算法非常重要，测量软件所使用的图像处理算法直接决定了测量效果的好坏以及测量过程花费的时间[18]。本章将对图像处理的各子算法进行理论和实践分析，以选出对本文所测零件有最佳处理效果的算法。

经过机器视觉检测平台的精心搭建，本文分别采集了主活塞的轴和导向盘的孔的图像，如图 3-1 所示。其中，灰度明亮的部分代表光源背景，灰度黑暗的部分代表目标零件。



(a) (b)

图 3-1 机器视觉图像。(a)主活塞的轴；(b)导向盘的孔

为了减少图片的重复处理，本文将主要采用图 3-1 中更具代表性和处理难度的图(b)进行算法的对比分析。同时，由于(b)图中的内圆才是本文主要的测量对象，为方便图像的处理，首先要对图像中的干扰成分进行蒙蔽，并提取出感兴趣的区域，经过预处理后的图像如图 3-2 所示。下文将以该图作为基础进行深入的图像处理算法分析和比较。

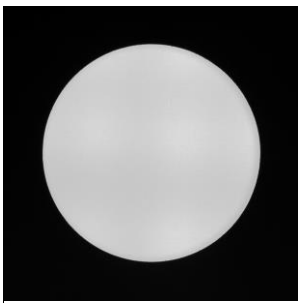


图 3-2 预处理后的图像

3.1 图像灰度变换

从三通道的彩色图像到单通道的灰色图像的转换，又称灰度化。图像在进行特征点检测和匹配之前第一步便是需要将图像灰度化，使图像不包含任何色彩信息而只具有亮度信息。一方面是这种只具有亮度信息的黑白图像数据量小，与彩色图像相比，更容易实现实时算法，降低处理的复杂度。另一方面，这种未经特殊滤光处理的图片有更多我们所需要的信息，还能减少不必要的干扰。为了表示灰度图，需要把亮度值进行量化。通常划分为 0~255 共 256 个级别，0 表示全黑，255 表示全白。在三通道彩色图像中如果 $R=G=B$，此时图像为灰度图，且灰度值为此时的 R、G、B 值，目前常用的灰度变换的处理方法主要有四种：分量法、最大值法、平均值法、加权平均值法。

3.1.1 分量法

根据上述可知，图像的三个分量的每一个分量的亮度值都可以作为灰度图像的灰度值，这便形成了三种灰度图像。因此我们可以根据测量过程中实际所需要的数据值来选取其中一种灰度图像，其灰度化公式为：

$$f_{Gray}(x,y)=\begin{cases}R(x,y)\\G(x,y)\\B(x,y)\end{cases} \tag{3-1}$$

其中，$f_{Gray}(x,y)$ 为图像变换为灰度图后在坐标位置 (x,y) 的灰度值，$R(x,y)$、$G(x,y)$、$B(x,y)$ 表示三个分量的值。

3.1.2 最大值法

最大值法是将三通道彩色图像中 R、G、B 三个分量中数值最大的分量作为图像灰度变换后的灰度值，其灰度化公式为：

$$f_{Gray}(x,y)=Max[R(x,y),G(x,y),B(x,y)] \tag{3-2}$$

3.1.3 平均值法

平均值法是对三通道彩色图像中的 R、G、B 三个分量求平均值，所得的结果则为灰度变换后的灰度值，其灰度化公式为：

$$f_{Gray}(x,y)=\frac{R(x,y)+G(x,y)+B(x,y)}{3} \tag{3-3}$$

3.1.4 加权平均值法

分量法、最值法和平均值法计算量简单，都是对单一分量进行的等量处理，对 R、G、B 三个通道的权重值都划分为相同，这在一般情况下是可行的，但在一些特殊情况下，尤其是如今图像处理应用的更多领域，图像采集环境会有很大的不同，因此前三种方法显得过于片面。而加权平均值法是根据采集的需要划分 R、G、B 三个分量的重要性，对其分配不同的权重，然后再根据权重计算出加权结果，所得的结果则为灰度变换后的灰度值。由于加权平均值法在各领域都有良好的适应性，因此成为目前最常用的方法，其常用经验公式为：

$$f_{Gray}(x,y)=0.2989R(x,y)+0.5870G(x,y)+0.1140B(x,y) \tag{3-4}$$

该公式中的权重是依据人体生理学中所提出的人对绿色敏感度最高，对蓝色敏感度最低的角度所给定的值。结合经验和实际应用，本文将采用加权平均值法对预处理后的图像进行灰度变换，得到如下的图 3-3。

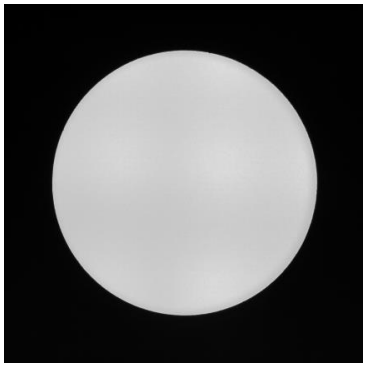


图 3-3 加权平均值法灰度变换后的图像

3.2 图像平滑

图像平滑是一种使用频率很高的图像处理方法，是指用于突出图像的宽大区域、低频成分、主干部分或抑制图像噪声和干扰高频成分，使图像亮度平缓渐变，减小突变梯度，改善图像质量的图像处理方法。

从工业相机采集到的原始图像由于一系列的原因会产生噪声，比如传感器本身存在的噪声，传输过程中出现的噪声等。为了避免噪声对图像分析造成干扰，需要对采集到的图像进行滤波操作，这就需要运用到图像平滑方法。

图像平滑的方法主要有直接计算的空间域法和间接处理的频率域法[19]，空间域法是使用空域模板对图像进行处理，在原始图像中将模板一点一点移动，结合滤波器参数与对应像素值的关系来计算响应。频率域法不是一种直接滤波处理的算法，它通过傅里叶变换将空域中表达的图像转换为频域表达，再与滤波器算子相乘

来改变原图像的频谱成分以达到滤波的目的。

由于空间域法具有简单、计算快速等优点，本文主要讨论空间域滤波法，其可分为线性滤波和非线性滤波。线性滤波是指两信号和的响应等于其响应之和，反之为非线性滤波。其中线性滤波主要包括均值滤波法和高斯滤波法，非线性滤波主要包括中值滤波法和双边滤波法。

3.2.1 均值滤波法

均值滤波法也称为邻域平均法，是一种图像处理的平滑空间域线性滤波方法。其基本原理是通过选择一个一定大小的模板，将模板与被测零件图像进行逐一覆盖，将模板覆盖的图像像素求其平均值，再把这个均值赋给模板中心所对应的原灰度图。

假设有一幅图像 $f(x,y)$，它是 $N\times N$ 的矩阵，处理后的图像假设为 $g(x,y)$，那么图像 $g(x,y)$ 的每个像素的灰度即可由模板所覆盖的邻域内的几个像素的灰度值得平均值所决定，均值滤波法的运算公式为：

$$g(x,y)=\frac{\sum_{i,j\in S} f(i,j)}{M} \tag{3-5}$$

其中，$f(i,j)$ 表示均值滤波前点 (x,y) 的邻域各点的灰度值，$g(x,y)$ 表示均值滤波点 (x,y) 的灰度值，S 是点 (x,y) 的邻域坐标集，M 为该坐标集内的元素的总数，邻域 S 的大小影响着滤波平滑的程度。

均值滤波法的具体思想是通过平滑图像的灰度级，来去除图像由于外界干扰所产生的噪声影响，具体的做法是对当前像素及其相邻的模板内对应的像素点都统一进行均值滤波处理。均值滤波虽然在一定程度上造成图像中的一些细节的模糊，但它具有算法比较简单，处理速度比较快的优点。对于平滑空间域滤波器的图像滤波算法，一个主要的问题就是滤波后所带来的图像模糊程度。也就是说，模板的半径越大，滤波后的图像就更模糊。为了能降低噪声影响的同时又不使图像的边缘变得模糊，本文采用如式 3-6 所示的 3×3 均值滤波模板与图像进行卷积。

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix} \tag{3-6}$$

3.2.2 高斯滤波法

高斯滤波法顾名思义，主要用于消除服从正态分布的噪声，它是通过将输入图

像与高斯函数进行卷积运算来实现图像的滤波。高斯滤波法的公式如下：

$$h(x,y)=\frac{f(i,j)*g(i,j)}{\sum_i\sum_j g(i,j)} \tag{3-7}$$

其中，$f(i,j)$表示高斯滤波前点(x,y)的邻域各点的灰度值，$h(x,y)$表示高斯滤波后点(x,y)的灰度值，*表示卷积运算，$g(i,j)$表示高斯滤波函数，其表达式为：

$$g(i,j)=e^{-\frac{i^2+j^2}{2\sigma^2}} \tag{3-8}$$

其中，σ表示标准差，它决定了高斯滤波器的滤波强度。通常，为了减少计算量，高斯滤波也可以像均值滤波那样，使用高斯滤波模板与原图进行卷积。本文将取σ=1，采用如式 3-9 所示的3×3高斯滤波模板进行滤波。

$$\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix} \tag{3-9}$$

3.2.3 中值滤波法

中值滤波是指对某像素点邻域中元素按灰度值大小排序，将中间值用来替换原像素的灰度值，使之接近真实值，进而消除孤立的噪声点。滤波的效果由邻域空间的范围、滤波模板的形状以及边缘点像素灰度值处理的方式三个方面决定，其中滤波模板大小起主要作用。中值滤波法的公式如下：

$$g(x,y)=Median\{f(i,j),i,j\in A\} \tag{3-10}$$

其中，$f(i,j)$表示中值滤波前点(x,y)的邻域各点的灰度值，$g(x,y)$为中值滤波后点(x,y)的灰度值，A 为选定窗口大小，本文将采用3×3的模板窗口，$Median\{\}$函数用于获取像素集合的中间值。

中值滤波针对的噪声特点不依赖于邻域内那些与典型值差别很大的值，同时可以避免一些线性滤波带来的图像细节模糊，保护边缘信息。

3.2.4 双边滤波法

双边滤波是一种结合图像像素相似度和空间邻近度的一种折中处理，目的是抑制与中心像素的值差别太大的像素。在数学形式上，双边滤波比高斯滤波多了一个高斯方差σ_d，是基于空间分布的高斯滤波函数， 所以在边缘附近，离的较远的像素不会太多影响到边缘上的像素值，这样就保证了边缘附近像素值的保存。

在双边滤波法中，输出的值$g(x,y)$依赖邻域像素值 $f(k,l)$（k，l 表示点(x,y)

的邻域像素位置）的加权组合，其公式如下：

$$g(x,y)=\frac{\sum_{k,l} f(k,l)w(x,y,k,l)}{\sum_{k,l} w(x,y,k,l)} \tag{3-11}$$

权重系数 $w(x,y,k,l)$ 取决于定义域核 d 与值域核 r 的乘积：

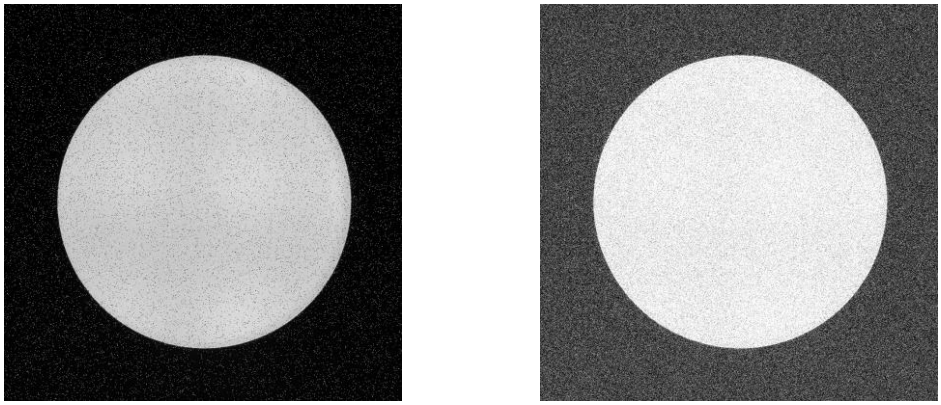
$$d(x,y,k,l)=e^{-\frac{(x-k)^2+(y-l)^2}{2\sigma_d^2}} \tag{3-12}$$

$$r(x,y,k,l)=e^{-\frac{\|f(x,y)-f(k,l)\|^2}{2\sigma_r^2}} \tag{3-13}$$

$$w(x,y,k,l)=e^{-\frac{(x-k)^2+(y-l)^2}{2\sigma_d^2}-\frac{\|f(x,y)-f(k,l)\|^2}{2\sigma_r^2}} \tag{3-14}$$

3.2.5 四种滤波法比较

大量的实验研究发现，由摄像机拍摄得到的图像受离散的椒盐噪声和零均值的高斯噪声的影响较严重。为了体现各种平滑算法的处理效果以及各算法间的差异，下面将分别人为加入了椒盐噪声（如图 3-4(a)）和高斯噪声（如图 3-4(b)）的图像作为原始图像，利用上述四种滤波方法对图像进行滤波去噪。如图 3-5、图 3-6、图 3-7、图 3-8 分别为采用均值滤波法、高斯滤波法、中值滤波法和双边滤波法进行滤波去噪后的效果图。

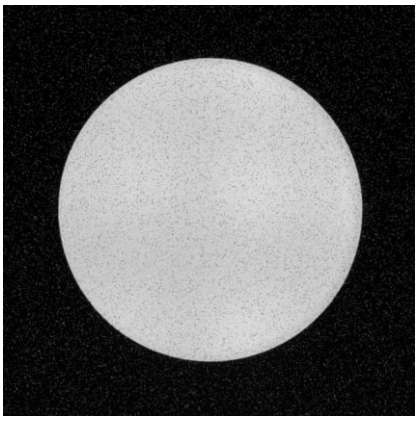


(a) (b)

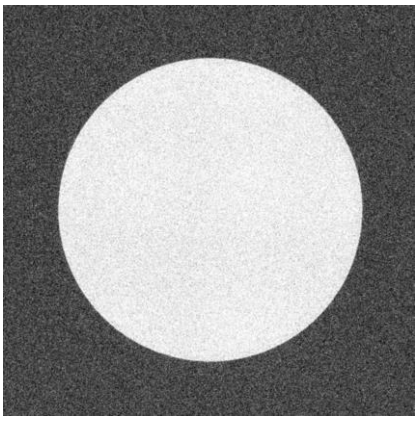
图 3-4 添加噪声后的图像。(a)添加椒盐噪声图像；(b)添加高斯噪声图像

对比以下各图的滤波效果可以看出，对于椒盐噪声，中值滤波的效果明显好于其他三者，双边滤波的效果次之。而对于高斯噪声，四种滤波效果之间差别并不明显。虽然双边滤波的效果较佳，但由于双边滤波结合了图像的空间邻近度和像素值相似度，在设定输入参数时，有可能因取不到合适的参数值而使像素邻域

内的颜色混合到一起，导致边缘出现模糊虚化的现象，这在图 3-8 中已有所体现。

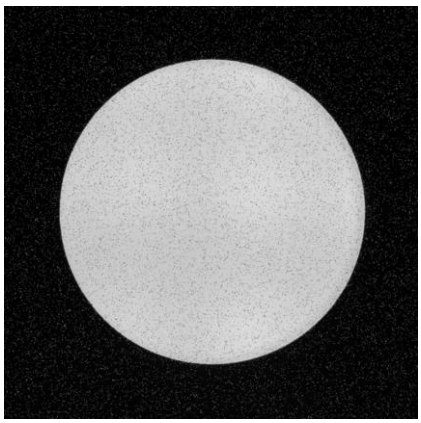


(a)

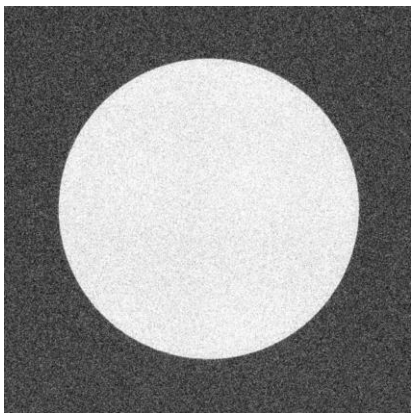


(b)

图 3-5 均值滤波后图像。(a)椒盐噪声图像均值滤波效果；(b)高斯噪声图像均值滤波效果

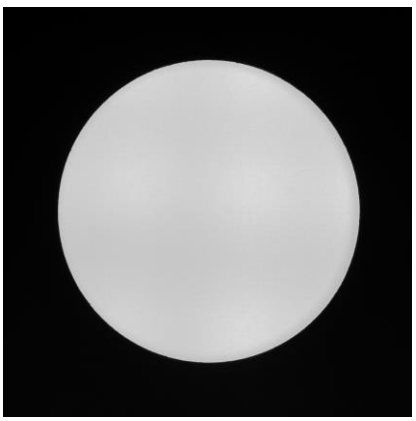


(a)

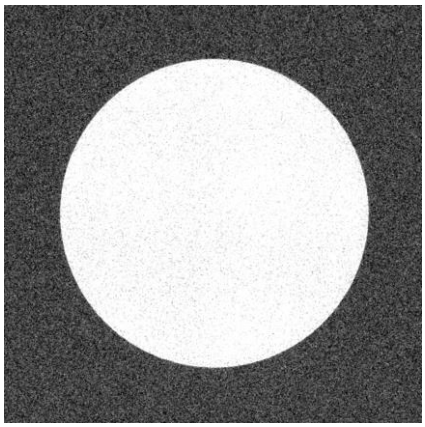


(b)

图 3-6 高斯滤波后图像。(a)椒盐噪声图像高斯滤波效果；(b)高斯噪声图像高斯滤波效果

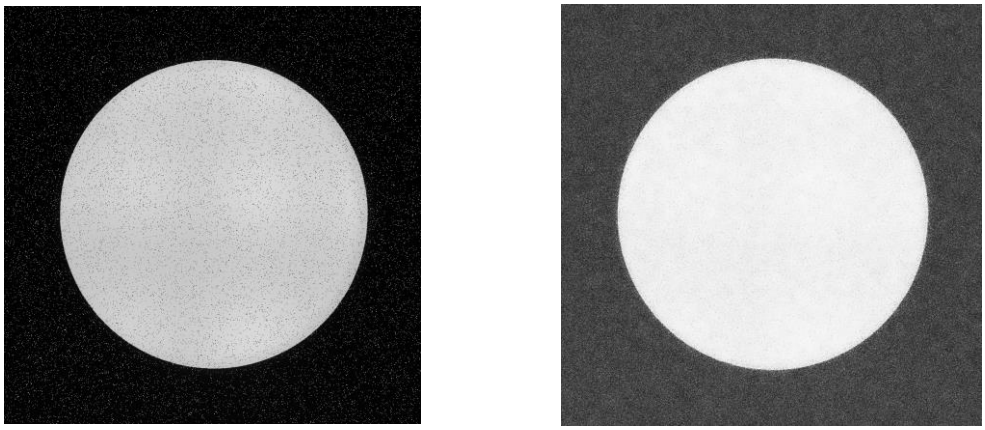


(a)



(b)

图 3-7 中值滤波后图像。(a)椒盐噪声图像中值滤波效果；(b)高斯噪声图像中值滤波效果



(a) (b)

图 3-8 双边滤波后图像。(a)椒盐噪声图像双边滤波效果；(b)高斯噪声图像双边滤波效果

为了获得各种滤波法的的定量化评价，本文分别求取经过图像滤波处理后的图像信噪比，信噪比 *SNR* 越大，表明滤波效果越好，其表达式如下：

$$SNR = 10\log_{10}\frac{\sum_{x}^{M}\sum_{y}^{N} g(x,y)^2}{\sum_{x}^{M}\sum_{y}^{N}[g(x,y)-f(x,y)]^2} \tag{3-15}$$

其中，*M*、*N* 分别为图像长度和宽度上的像素数量，$g(x,y)$ 与 $f(x,y)$ 分别为处理后的目标图像和处理前的原始图像。经计算，上述四种滤波法处理后图像的信噪比如表 3-1 所示。

表 3-1 四种滤波法处理后图像的信噪比 （单位：dB）

滤波法	均值滤波法	高斯滤波法	中值滤波法	双边滤波法
椒盐噪声 *SNR*	7.0806	7.1565	15.8979	15.0241
高斯噪声 *SNR*	4.3607	4.4505	4.5552	5.0392

表 3-1 中的信噪比验证了上述图像滤波效果给人的直观感受，即中值滤波法对椒盐噪声和高斯噪声都有很好的滤波效果，尤其在对椒盐噪声滤波滤波时更为明显。虽然双边滤波法对高斯噪声也有较好的滤波效果，但容易受输入参数的影响，在去除噪声的同时使图像的边缘特征信息丢失。故在本文中，将采用中值滤波法对图像进行滤波去噪。

3.3 图像二值化

在机器视觉测量系统中，我们对于被测目标的形状和边缘最感兴趣，但图像中存在大量的冗余信息，这些信息不仅增加了图像数据处理的计算量，而且对测量结果会产生一定程度的影响。为了后续的边缘提取和尺寸测量，需要将目标形状从图片中分离出来，这就需要通过图像二值化处理。

灰度图像的像素值一般在 0~255 的范围内，二值化图像是指灰度图像的所有像素值只能在两个离散值中选取其一，多数情况下像素值取 0 或 255，呈现为黑白图像。通过阈值分割的方法对灰度图像进行转换，设原灰度图像为 $f(x,y)$，阈值为 T，经过二值化处理后的图像为 $g(x,y)$，则可以得到如下表达式：

$$g(x,y)=\begin{cases}255, f(x,y)\geq T\\0, f(x,y)<T\end{cases} \tag{3-16}$$

其中，阈值 T 对图像的分割起着决定性的作用。因此，选择合适的阈值 T 对图像的二值化来说至关重要。下面将主要分析讨论直方图双峰法、自适应迭代法、最大类间方差法这三种常用的阈值选取方法。

3.3.1 直方图双峰法

图像中的目标区域和背景区域亮度往往存在着明显不同，这种亮度差异反映在直方图上表现为双波峰形状，如图 3-9 所示。其中一个波峰代表目标区域的中心灰度，另一个波峰代表背景区域的中心灰度，而两个波峰之间的波谷则代表边界区域，所以可将波谷处的灰度值 T 作为阈值将目标区域提取出来。当图像信息比较复杂，分为较多灰度区域时，直方图上会出现多个波峰和波谷，此时直方图双峰法可能无法准确分割图像，需要一些辅助手段获取最佳阈值。

图 3-9 图像灰度直方图

3.3.2 自适应迭代法

直方图双峰法是一种人为选取阈值方法，而自适应迭代法则是一种自动阈值化方法。自动阈值化算法通常使用灰度直方图来分析图像中灰度值的分布，并使用特定应用域知识来选取最合适的阈值。自适应迭代法式阈值选取方法为：首先选择

一个近似阈值作为估计值的初始值，然后连续不断地改进这一估计值。比如，使用初始阈值生成子图像，并根据子图像的特性来选取新的阈值，再用新阈值分割图像，这样做的效果将好于用初始阈值分割图像的效果。阈值的改进策略是这一方法的关键。该算法的具体步骤如下[20]：

（1）选择一个初始阈值的估计值 T，比如，图像灰度的均值就是一个较好的初始值；

（2）利用阈值 T 把图像分割成两组，R_1 和 R_2；

（3）计算区域 R_1 和 R_2 的均值 μ_1、μ_2；

（4）选择新的阈值 T，其表达式为：

$$T=\frac{\mu_1+\mu_2}{2} \tag{3-17}$$

（5）重复步骤（2）至（4），直到 μ_1 和 μ_2 的均值不再变化。

3.3.3 最大类间方差法

最大类间方差法又被称为大津法，该方法以最小二乘法为基础，根据图像的灰度直方图进行计算，以某一灰度值为阈值将图像分为两组并计算两组方差，取两组之间方差最大时对应的灰度值作为阈值[21]。其具体计算过程如下：

设图像总像素数为 M，其中存在 n 级灰度，灰度值为 i 的像素数量为 m_i，可得各灰度级的概率为：

$$P_i=\frac{m_i}{M} \tag{3-18}$$

将图像以灰度值 t 分成 C_0 和 C_1 两组，分别代表目标和背景，其中 $C_0=(0,1,2,...,t)$，$C_1=(t+1,t+2,...,n-1)$，则目标 w_0 和背景 w_1 的概率分别为：

$$w_0=\sum_{i=0}^{t}P_i \tag{3-19}$$

$$w_1=1\text{-}w_0=\sum_{i=t+1}^{n-1}P_i \tag{3-20}$$

目标平均灰度 μ_0 和背景平均灰度 μ_1 分别为：

$$\mu_0=\frac{\sum_{i=0}^{t}iP_i}{w_0}=\frac{\mu_t}{w_0}、 \tag{3-21}$$

$$\mu_1 = \frac{\sum_{i=t+1}^{n-1} iP_i}{w_1} = \frac{\mu - \mu_t}{1 - w_0} \tag{3-22}$$

其中 $\mu_t = \sum_{i=0}^{t} iP_i$ ， $\mu = \sum_{i=0}^{n-1} iP_i$ ，由此可得整幅图像的平均灰度值为：

$$\mu = w_0\mu_0 + w_1\mu_1 \tag{3-23}$$

则目标和背景的方差为：

$$\sigma^2(t) = w_0(\mu_0 - \mu)^2 + w_1(\mu_1 - \mu)^2 \tag{3-24}$$

当上式中 $\sigma^2(t)$ 得到最大值，则此时的 t 就是分割目标和背景区域的最佳阈值。

最大类间方差法因能够自动选取图像分割的最佳阈值，实时性好且计算过程简便，在图像处理领域得到了广泛使用。

3.3.4 三种阈值选取方法比较

采用直方图双峰法进行二值化，首先需要了解原始图像的灰度直方图，经过中值滤波后图像的灰度直方图如图 3-10 所示。从图中可以看出，图像呈双峰分布，并且没有任何重叠，其中一个波峰的灰度值在 0 附近，另一个波峰的灰度值在 210 附近，可以选择阈值为 100 对灰度图像进行二值化。

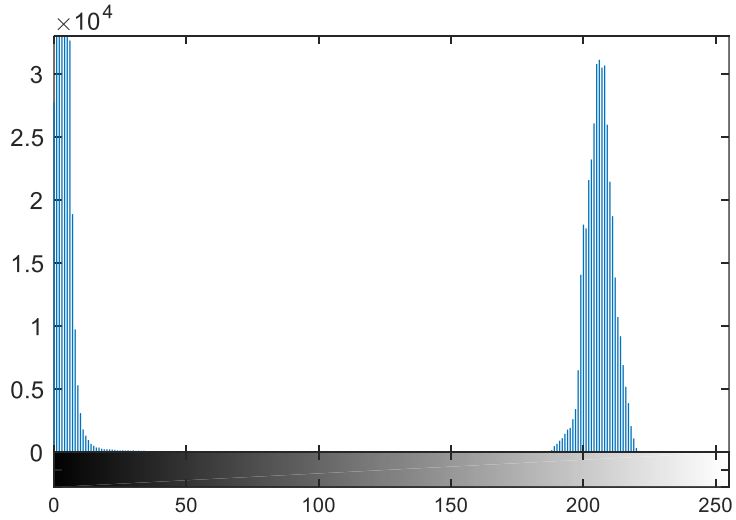


图 3-10 中值滤波后图像的的灰度直方图

由于本文所采用的图像中的目标背景很容易区分，利用自适应迭代法和最大类间方差法确定其阈值时，计算结果均为 104，故这两种方法对本文图像的二值化效果是一样的，均如图 3-11 所示。

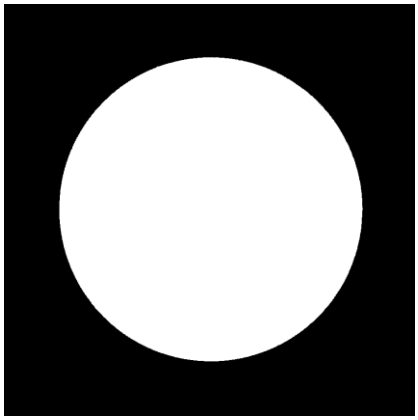


图 3-11 自适应迭代法和最大类间方差法二值化后图像

从三种图像二值化方法的优缺点来看，应用直方图双峰法来分割图像，需要一定的先验知识，因为同一个直方图可以对应若干个不同的图像，直方图表明图像中各个灰度级上有多少个像素，并不描述这些像素的任何位置，所以没有利用到图像灰度的空间信息。该方法不适合直方图中双峰差别较大或双峰间的谷比较宽广而平坦的图像，以及单峰直方图的情况。

相对于直方图双峰法，自适应迭代法不需要人为设定阈值，其分割的效果好，但稳定性比最大类间方差法差。当图像受到很小的噪声干扰、灰度区域分布不均或图像信息较为复杂时，分割效果就会有较大的反差。因此，本文将采用最大类间方差法对图像进行二值化，将该方法运用在下节边缘检测算子处理后的梯度幅值来确定其阈值，可以很好地提取出粗边缘。

3.4 像素级边缘检测

物体的边缘是图像的基本特征，是指图像局部强度变化最显著的部分，也是指灰度图像中灰度有阶跃或尖顶状变化的那些像素的集合。边缘广泛存在于物体与物体、物体与背景之间，是图像分割、纹理特征提取和形状特征提取等图像处理所依赖的重要基础[22]。由于边缘检测十分重要，因此该技术成为了机器视觉研究领域最活跃的课题之一。显然，边缘提取是尺寸测量的关键步骤，为尺寸测量提供所需的轮廓曲线。

图像边缘包含像素邻域内灰度值的阶跃突变或屋脊突变的像素，当需要形状或纹理特征进行图像分析时，图像边缘是首先提取的特征。图 3-12 分别显示了两类边缘[23]，对于阶跃变化的边缘，其邻域一阶导数幅值最大，而二阶导数幅值为零；对于峰形变化的屋脊型边缘，其邻域的一阶导数幅值为零，二阶导数的幅值最大[24]。

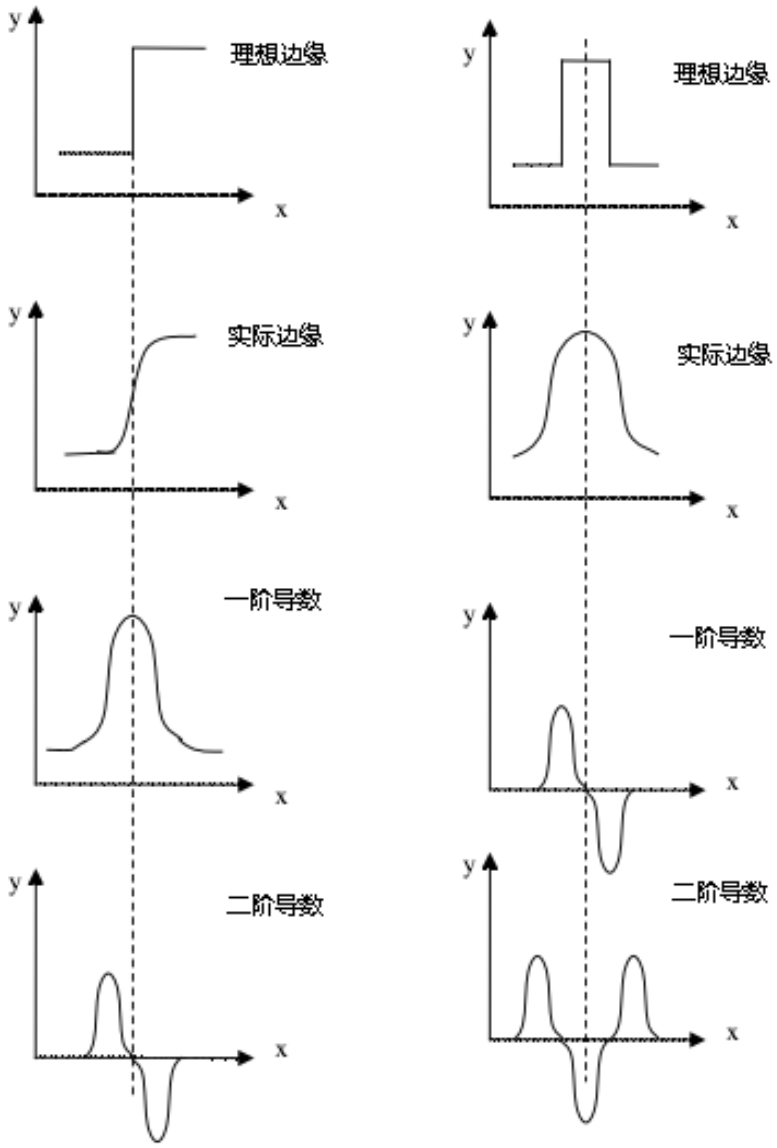


图 3-12 阶跃型边缘和屋脊型边缘及其导数

边缘信息在图像中显出强度和方向两个特性，随边缘方向分布的像素特征表现平缓，与边缘方向垂直的像素特征表现剧烈。图像灰度值的显著变化可用梯度的离散逼近函数来检测。

我们定义一个梯度算子，通过计算一幅图像 f 在 (x,y) 处的梯度，可以得到该像素点的边缘强度和方向，其中梯度的幅值为边缘的强度，梯度方向与边缘方向垂直。函数 $f(x,y)$ 在 (x,y) 处的梯度为一个矢量，其表达式如下：

$$\nabla f(x,y)=\begin{bmatrix}G_x\\G_y\end{bmatrix}=\begin{bmatrix}\dfrac{\partial f}{\partial x}\\\dfrac{\partial f}{\partial y}\end{bmatrix} \tag{3-25}$$

设梯度的方向角度为θ，梯度方向是指在位置(x,y)处图像灰度变化最快的方向，其以x为基准度量，计算公式为：

$$\theta(x,y)=\arctan\left(\frac{G_y}{G_x}\right) \tag{3-26}$$

梯度$G(x,y)$的幅值是指在梯度方向上每增加单位距离后$f(x,y)$值增大的最大变化率，计算公式为：

$$|\nabla f(x,y)|=\sqrt{G_x^2+G_y^2}=\sqrt{\left(\frac{\partial f}{\partial x}\right)^2+\left(\frac{\partial f}{\partial y}\right)^2} \tag{3-27}$$

在实际计算时，由于公式(3-27)中存在平方和开方，计算量较大，因此通常用绝对值来近似梯度幅值：

$$|\nabla f(x,y)|=|G_x|+|G_y|=\left|\frac{\partial f}{\partial x}\right|+\left|\frac{\partial f}{\partial y}\right| \tag{3-28}$$

由于实际图像是经过空间采样、灰度离散化的图像，故在求取梯度时，用差分代替微分进行计算。

一般情况下，我们可以采用边缘检测算子对图像的边缘进行粗提取。其中，通过一阶导数进行边缘检测的算子包括 Roberts 算子、Prewitt 算子和 Sobel 算子等，通过二阶导数进行边缘检测的算子有 Laplacian 算子和 LoG 算子等，它们采用不同的卷积模板依次与图像中的每个像素点做卷积和运算，然后采用上节中介绍的最大类间方差法来确定阈值，如果像素点的梯度幅值大于阈值，则该像素点为边缘像素，否则不是边缘像素。而 Canny 算子则是另一种边缘检测算子，这种算子不需要使用微分方法检测边缘，而是通过一些约束条件进行限制进而计算出结果的边缘检测最优化算子[25]。

以上六种算子都是像素级的边缘检测算子，即采用它们进行边缘检测的最小精度是一个像素。下面将对上述六种边缘检测算子进行分析讨论，比较它们对图像边缘的提取效果。

3.4.1 Roberts 算子

Roberts 边缘检测算子也叫交叉差分算子，在求取梯度时，通过2×2的局部领域并列计算，算出图像数据的空间一次差分，其计算公式为：

$$\nabla f(x,y)=\left|f(x,y)-f(x+1,y+1)\right|+\left|f(x,y+1)-f(x+1,y)\right| \tag{3-29}$$

其水平方向和垂直方向的卷积模板为：

$$f_x=\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}\qquad f_y=\begin{bmatrix}0 & 1\\ -1 & 0\end{bmatrix}\tag{3-30}$$

利用 Robers 算子对中值滤波后的图像进行边缘检测的效果如图 3-13 所示。

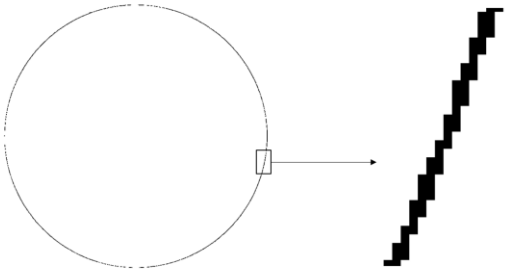


图 3-13 Roberts 算子边缘检测效果

3.4.2 Prewitt 算子

由于 Roberts 算子采用2×2大小的模板，该模板没有中心，故不易使用，Prewitts 算子采用3×3大小的模板，其计算公式为：

$$\begin{aligned}\nabla f(x,y)=&|f(x+1,y+1)+f(x,y+1)+f(x-1,y+1)\\&-f(x+1,y-1)-f(x,y-1)-f(x-1,y-1)|\\&+|f(x-1,y-1)+f(x-1,y)+f(x-1,y+1)\\&-f(x+1,y-1)-f(x+1,y)-f(x+1,y+1)|\end{aligned}\tag{3-31}$$

其水平方向和垂直方向的卷积模板为：

$$f_x=\begin{bmatrix}-1 & 0 & 1\\ -1 & 0 & 1\\ -1 & 0 & 1\end{bmatrix}\qquad f_y=\begin{bmatrix}-1 & -1 & -1\\ 0 & 0 & 0\\ 1 & 1 & 1\end{bmatrix}\tag{3-32}$$

利用 Prewitt 算子对中值滤波后的图像进行边缘检测的效果如图 3-14 所示。

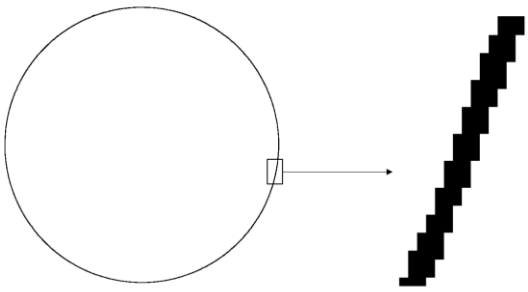


图 3-14 Prewitt 算子边缘检测效果

3.4.3 Sobel 算子

Sobel 算子和 Prewitt 算子的梯度计算方法相似，不同的是，Sobel 算子在上、下、左、右四个位置上的权值为 Prewitt 算子的 2 倍，该边缘检测算子通过增加3×3区域中每行和每列的中心点重要程度来计算 $f(x,y)$ 的梯度，其计算公式为：

$$\begin{aligned}\nabla f(x,y)=&|f(x+1,y+1)+2f(x,y+1)+f(x-1,y+1)\\&-f(x+1,y-1)-2f(x,y-1)-f(x-1,y-1)|\\&+|f(x-1,y-1)+2f(x-1,y)+f(x-1,y+1)\\&-f(x+1,y-1)-2f(x+1,y)-f(x+1,y+1)|\end{aligned}\tag{3-33}$$

其水平方向和垂直方向的卷积模板为：

$$f_x=\begin{bmatrix}-1&0&1\\-2&0&2\\-1&0&1\end{bmatrix}\qquad f_y=\begin{bmatrix}-1&-2&-1\\0&0&0\\1&2&1\end{bmatrix}\tag{3-34}$$

利用 Sobel 算子对中值滤波后的图像进行边缘检测的效果如图 3-15 所示。

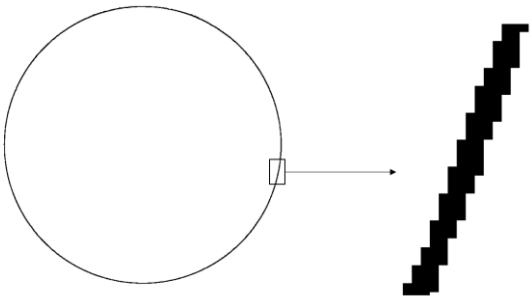


图 3-15 Sobel 算子边缘检测效果

3.4.4 Laplacian 算子

不同于一阶微分算子，二阶微分算子是通过对同向中的每个像素计算关于 x 轴和 y 轴的二阶偏导数之和$\nabla^2 f(x,y)$来进行边缘的检测。Laplacian 算子是各向同性的微分算子，对于二维的图像 $f(x,y)$，Laplacian 算子定义为：

$$\nabla^2 f(x,y)=\frac{\partial^2 f(x,y)}{\partial x^2}+\frac{\partial^2 f(x,y)}{\partial y^2}\tag{3-35}$$

$$\begin{cases} \dfrac{\partial^2 f(x,y)}{\partial x^2} = \dfrac{\partial f_x(x,y)}{\partial x} - \dfrac{\partial f_x(x+1,y)}{\partial x} \\ \qquad\qquad = [f(x,y) - f(x-1,y)] - [f(x+1,y) - f(x,y)] \\ \dfrac{\partial^2 f(x,y)}{\partial y^2} = \dfrac{\partial f_y(x,y)}{\partial y} - \dfrac{\partial f_y(x,y+1)}{\partial y} \\ \qquad\qquad = [f(x,y) - f(x,y-1)] - [f(x,y+1) - f(x,y)] \end{cases} \tag{3-36}$$

将式(3-36)代入式(3-35)中可得：

$$\nabla^2 f(x,y) = 4f(x,y) - f(x-1,y) - f(x+1,y) - f(x,y-1) - f(x,y+1) \tag{3-37}$$

故其相对应的计算梯度卷积模板为：

$$\nabla^2 f = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \tag{3-38}$$

利用 Laplacian 算子对中值滤波后的图像进行边缘检测的效果如图 3-16 所示。

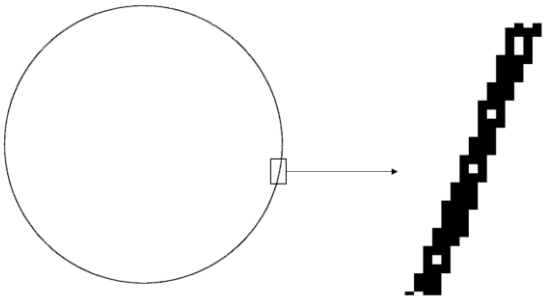


图 3-16 Laplacian 算子边缘检测效果

3.4.5 LoG 算子

由于利用图像强度二阶导数的零交叉点来求边缘点的算法对噪声十分敏感，所以希望在边缘检测前滤除噪声。为此 Marr 和 Hildreth 提出了马尔算子，因为是基于高斯算子和拉普拉斯算子的，所以也称高斯-拉普拉斯（Laplacian of Gaussian, LoG）边缘检测算子，简称 LoG 算子。

该算子首先采用高斯函数对原始图像 $f(x,y)$ 进行平滑，高斯函数的表达式为：

$$h(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{3-39}$$

其中 σ 表示标准差，通过对 σ 进行调整可以对图像的平滑效果进行控制，平滑可以表示为原始图像和高斯函数的卷积，即：

$$g(x,y)=f(x,y)*h(x,y) \tag{3-40}$$

平滑后的图像再采用 Laplacian 算子进行边缘检测，即：

$$\nabla^2 g(x,y)=\nabla^2\left[f(x,y)*h(x,y)\right]=\nabla^2\left[h(x,y)\right]*f(x,y)=\left(\frac{r^2-\sigma^2}{\sigma^4}\right)e^{-\frac{r^2}{2\sigma^2}}*f(x,y) \tag{3-41}$$

其中 r 表示离原点的径向距离，即 $r^2=x^2+y^2$，$\nabla^2\left[h(x,y)\right]$称为高斯-拉普拉斯滤波算子，它的轴截面如图 3-17 所示，它是一个轴对称函数，各向同性，通过将该函数与原始图像进行卷积，再利用二阶导数算子过零点的性质，，便可检测出图像中的边缘像素。

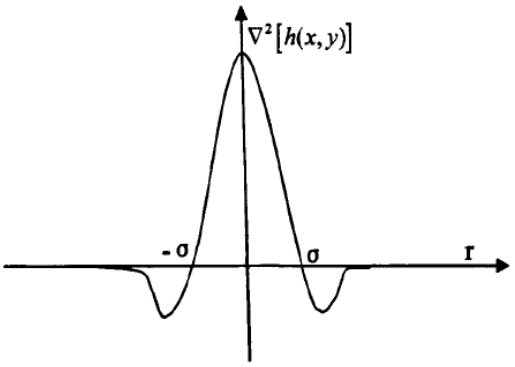


图 3-17 高斯-拉普拉斯滤波算子的轴截面图

由图 3-17 可知，σ越小，滤波算子对图像的平滑程度越小，但是图像的边缘像素失真较少，边缘定位精度越高。实际使用时，需要根据实际图像来调节σ的大小。对于本文中的图像，本文将采用5×5大小的 LoG 算子模板，如式(3-42)所示：

$$\begin{bmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & -2 & -1 & 0 \\ -1 & -2 & 16 & -2 & -1 \\ 0 & -1 & -2 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{bmatrix} \tag{3-42}$$

利用 LoG 算子对中值滤波后的图像进行边缘检测的效果如图 3-18 所示。

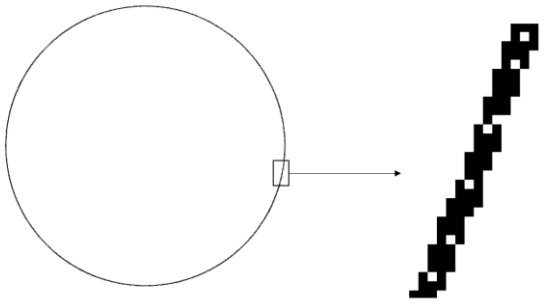


图 3-18 LoG 算子边缘检测效果

3.4.6 Canny 算子

Canny边缘检测算子是Canny在1986年提出来的，它是高斯函数的一阶导数，是基于最优化算法的检测算子，具有良好的边缘检测精度和信噪比[26,27]。Canny 算子将边缘检测问题转化成为通过检测单位函数的极大值问题，从而实现边缘提取的目的。评价边缘检测性能优劣的准则有以下三个[28,29]：

（1）低误判率：尽可能减少噪声所产生的边缘像素的误报，检测到尽可能多的边缘像素点.

（2）高定位精度：把边缘点准确地定位在灰度变化最大的像素点上，检测到的边缘像素点要与实际的边缘像素点尽可能相近。

（3）最小响应：图像的边缘像素点只能标识一次，并且可能存在的噪声不应标识为边缘。

Canny 边缘检测的主要步骤如下：

（1）用高斯滤波器平滑图像

对输入图像进行平滑处理，可减少噪声对边缘检测的影响，同时有利于减少边缘的断裂。高斯滤波的数学表达式如下：

$$f_s(x,y)=G(x,y)*f(x,y) \tag{3-43}$$

其中，$f(x,y)$表示输入图像，$G(x,y)$表示高斯函数，$*$表示卷积运算，$f_s(x,y)$为滤波器输出图像。

（2）计算输入图像 $f(x,y)$ 的梯度

如前面所述，梯度描述了边缘的基本特征，梯度的幅度表示边缘的强度，梯度的方向与边缘的方向垂直。梯度的幅值和方向角度的计算公式为：

$$M(x,y)=\sqrt{g_x^2+g_y^2} \tag{3-44}$$

$$\theta(x,y)=\arctan\frac{g_y}{g_x} \tag{3-45}$$

其中，$g_x=\partial f_s/\partial x$，$g_y=\partial f_s/\partial y$。

（3）对梯度幅值进行非极大值抑制

在之前介绍的基本边缘检测算子都是利用图像的梯度幅值来判定边缘像素的，并没有结合图像的梯度方向信息进行综合分析，而 Canny 算子则补充了前面基本边缘检测算子的缺陷。

非极大值抑制方法首先将图像的梯度方向离散化，一般选 0°，45°，90°，135°，如图 3-19 所示，其中 0 表示 0°方向，1 表示 45°方向，2 表示 90°，3 表示 135°方向。

通过式(3-45)计算得到的中心像素点的梯度方向角度，比较中心像素点和其梯度方向上与之相邻的其他两个像素的梯度幅值，若中心像素点的梯度幅值不比其他两个像素点的梯度幅值大，则将该中心像素点的灰度值置零，表示该点被抑制，即判断该点是非边缘像素。使用非最大抑制能够使检测的边缘更细，去除伪边缘，提高边缘定位的精度。

3	2	1
0		0
1	2	3

图 3-19 图像梯度方向划分

（4）双阈值检测和连接边缘

对上一步处理完成的图像作用两个阈值T_L（低阈值）和T_H（高阈值）。高阈值和低阈值的比率通常在 2：1 与 3：1 之间，定义$g_N(x,y)$为进行非最大抑制处理后的图像，则对该图像的每一像素点，用以下标准判断是否为边缘点。

①若$g_N(x,y)\geq T_H$，则该像素点保留为边缘像素。

②若$g_N(x,y)\leq T_H$，则该像素点不属于边缘像素，被排除。

③若$T_L\leq g_N(x,y)\leq T_H$，还需进一步判断该点是否连接到有效的边缘点上，该像素仅仅在连接到一个高于高阈值的像素时被保留。

利用 Canny 算子对中值滤波后的图像进行边缘检测的效果如图 3-20 所示。

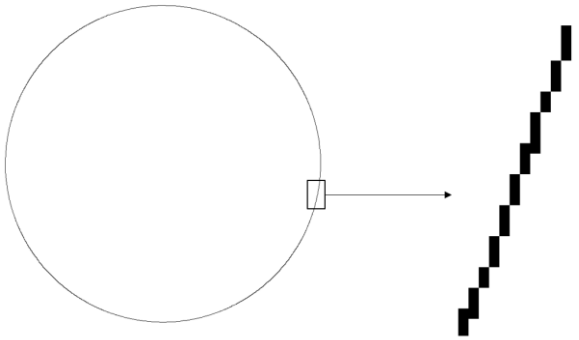


图 3-20 Canny 算子边缘检测效果

3.4.7 六种边缘检测算子的比较

经过对上述几种经典边缘检测算子的计算原理进行分析，并对图像进行处理后的效果对比，本文得出如下结论：Roberts 算子处理图像的速度快，在水平和垂直方向的边缘检测效果较好，但其本身不具备抑制噪声的能力。Prewitt 算子和 Sobel 算子都考虑了图像像素点的邻域信息，首先对图像进行加权平滑运算，继而进行微分计算处理，所以其本身具有一定抑制噪声的能力，两者的区别在于平滑部分的权值差异，但这两种算子对本文图像提取的边缘较粗，不能完全排除处理结果中存在的虚假边缘信息。对比图 3-13 至图 3-16 可以看出，Laplacian 算子对本文图像提取的边缘比前面三种算子更细，但由于 Laplacian 算子是通过二阶微分运算得到边缘位置的，所以它对噪声更加敏感，可能会把噪声当边缘点检测出来，而真正的边缘点被噪声淹没而未检测出来。另外，因为 Laplacian 算子是各向同性的，所以它检测不到边缘的方向。LoG 算子是针对 Laplacian 算子的缺陷改进获得的，该算子采用高斯滤波器进行滤波，提高了算子抑制噪声的能力，所以它的边缘检测效果比 Laplacian 算子更佳，图 3-18 提取出的边缘比图 3-16 更细也验证了这一点。而 Canny 算子抑制噪声的能力较强，能够准确地提取出劣化图像的边缘，实用性较强。由于 Canny 算子抑制了虚假边缘，所以它提取的边缘往往具有单像素宽，从而定位精度也就更高。图 3-20 的 Canny 算子边缘检测效果是这六种算子中最佳的。

综上所述，上文介绍的六种边缘检测算子有各自的特点和应用领域，在实际应用中，需要根据图像的实际情况选择合适的边缘检测算子。从边缘检测原理和图像处理的直观效果来看，Canny 算子对本文零件图像的适用性是最好的，因此本文将采用 Canny 算子对零件图像进行像素级边缘检测。

3.5 亚像素边缘定位

对零件图像进行 Canny 算子边缘检测之后，检测到的零件尺寸和位置参数只能达到像素级精度。在轴孔零件机器视觉检测时，由于零件的尺寸要求较高，所以大多数时候，边缘检测仅达到像素级精度是无法满足测量要求的。为了提高边缘检测的精度，可以从提高相机的分辨率和提高边缘的定位精度这 2 个方面着手进行改进。

提高边缘检测精度最为直接的办法就是提高相机的分辨率，但是这种通过提高硬件分辨率来提高边缘检测精度的方法是非常昂贵的，并且对于工业相机，提高硬件分辨率是非常有限的。与之同时，如果能够通过某种算法使得图像中边缘的定位精度达到亚像素级别，同样也可以提高测量系统的分辨率。而且亚像素定位精度提升空间很大，一般常用的亚像素定位算法的精度可以达到 0.1~0.5 个像素，有些算法在理想的情况下可以达到 0.01 个像素左右的精度[30]，因此，基于上述这些优势，亚像素定位算法在边缘检测中就具有非常重要的理论和实际意义。

亚像素定位算法的使用条件可以归纳成两个方面[30]。首先，边缘由多个像素点组成，并具有一定的几何形状特征和灰度分布特征；其次，必须明确边缘定位基准点的具体位置。上一节已经通过 Canny 算子得出零件图像边缘的像素级位置坐标，然后根据边缘轮廓的几何形状特征和灰度分布特征，确定与边缘轮廓最吻合的位置，从而使得边缘定位于更加准确的位置。

亚像素定位的方法有很多种，其中较为简单的方法为形心法和中心法[31]，较为常用的还有基于矩方法[32]，拟合法[33]，插值法[34]，此外还有基于概率论法[35]，滤波重建法[36]和数字相关法等应用不同原理的亚像素提取算法。本文根据轴孔零件的几何形状特征和灰度分布特征，重点介绍矩方法亚像素定位算法。

矩方法是计算机视觉与模式识别中广泛使用的方法。在机器视觉检测图像处理中，根据一个物体的矩特性在成像前后保持不变的原理，常常将矩方法应用于椭圆、圆和矩形等中心对称目标，以及边缘和角点等目标的亚像素定位中。

矩方法作为数学上的完备描述，相当于原函数在新的坐标空间上展开，即一个分段连续有界函数可用矩族唯一表示。矩方法一般可以分为根据灰度值信息计算的灰度矩、根据灰度位置信息的空间矩以及 Zernike 矩等。由于 Zernike 矩的定位精度较高，且相对空间矩的运算量要小，故本文将采用 Zernike 矩对边缘进行亚像素定位。下面对 Zernike 矩亚像素边缘定位算法理论进行分析与实践。

3.5.1 Zernike 矩亚像素边缘定位

设一幅给定图像为 $f(x,y)$，则它的 n 阶 m 次 Zernike 矩定义为：

$$A_{nm}=\frac{n+1}{\pi}\iint_{x^2+y^2\le 1}V_{nm}^{*}(\rho,\theta)f(x,y)dxdy \tag{3-46}$$

其中$V_{nm}(\rho,\theta)$是在极坐标系的单位圆内的正交 n 阶 m 次 Zernike 多项式，$*$表示复共轭。当图像为离散形式时，Zernike 矩表示为：

$$A_{nm}=\sum_{x}\sum_{y}f(x,y)V_{nm}^{*}(\rho,\theta),x^2+y^2\le 1 \tag{3-47}$$

为了计算一幅给定图像的 Zernike 矩，必须将图像的中心移到坐标原点，将图像像素点映射到单位圆内。Zernike 矩是积分型算子，因此在有噪声的情况下它对噪声不敏感。利用 Zernike 矩进行亚像素边缘定位，建立理想边缘阶跃模型如图 3-21 所示。

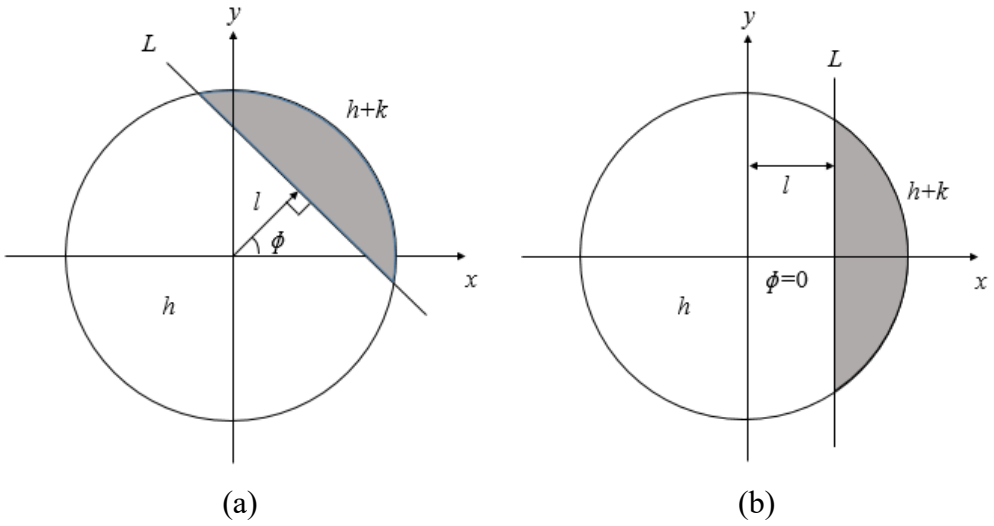


图 3-21 理想边缘阶跃模型。(a)原始边缘图像；(b)旋转后的边缘图像

其中，图(a)为原始边缘图像，圆是单位圆，直线 L 被单位圆包含的部分代表理想边缘，L 两侧的灰度值分别为 h 和 $h+k$，k 为灰度差，l 是圆心到边缘的垂直距离，ϕ 是 l 和 x 轴的夹角。如果把边缘顺时针旋转 ϕ，则边缘将与 y 轴平行，如图(b)所示。因为 Zernike 矩具有旋转不变性，图像旋转前的 Zernike 矩 A_{nm} 与旋转后的 Zernike 矩 A'_{nm} 的变换关系为：

$$A'_{nm}=A_{nm}e^{-jm\phi} \tag{3-48}$$

利用 Zernike 矩的旋转不变性，可以计算出边缘检测所需的三个参数：k、l、ϕ，进而实现对边缘的精确定位，下面将给出具体的推导过程。

由图 3-21(b)可以看出，旋转后的图像 $f'(x,y)$ 关于 x 轴对称，即质心落在 x 轴上，因此其一阶几何矩 A_{01} 满足：

$$A_{01}=\iint_{x^2+y^2\le 1} f'(x,y)ydxdy=0 \tag{3-49}$$

因为A'_{11}的 Zernike 多项式为$x+jy$，由 Zernike 矩的定义式(3-46)可以看出，A'_{11}的虚部为 0，这一点非常重要。对于A'_{11}，我们有：

$$\begin{aligned}A'_{11}&=A_{11}e^{-j\phi}\\&=[\mathrm{Re}(A_{11})+j\,\mathrm{Im}(A_{11})][\cos(\phi)-j\sin(\phi)]\\&=[\mathrm{Re}(A_{11})\cos(\phi)+\mathrm{Im}(A_{11})\sin(\phi)]+j[\mathrm{Im}(A_{11})\cos(\phi)-\mathrm{Re}(A_{11})\sin(\phi)]\\&=\mathrm{Re}(A'_{11})+j\,\mathrm{Im}(A'_{11})\end{aligned} \tag{3-50}$$

其中，Re( )、Im( )分别表示复数的实部和虚部。已知$\mathrm{Im}(A'_{11})=0$，故有：

$$\mathrm{Im}[A_{11}]\cos(\phi)-\mathrm{Re}[A_{11}]\sin(\phi)=0 \tag{3-51}$$

因此边缘旋转的角度为：

$$\phi=\tan^{-1}\left(\frac{\mathrm{Im}[A_{11}]}{\mathrm{Re}[A_{11}]}\right) \tag{3-52}$$

对图 3-21 所示的边缘模型计算 Zernike 矩A'_{11}、A'_{20}可得：

$$A'_{11}=\iint_{x^2+y^2\le 1} f'(x,y)(x-jy)dydx=\iint_{x^2+y^2\le 1} f'(x,y)xdydx=\frac{2k(1-l^2)^{\frac{3}{2}}}{3} \tag{3-53}$$

$$A'_{20}=\iint_{x^2+y^2\le 1} f'(x,y)(2x^2+2y^2-1)dydx=\frac{2kl(1-l^2)^{\frac{3}{2}}}{3} \tag{3-54}$$

由以上两式，可以导出另外两个边缘参数 l 和 k：

$$l=\frac{A_{20}}{A'_{11}}=\frac{A_{20}}{\mathrm{Re}[A'_{11}]}=\frac{A_{20}}{\mathrm{Re}(A_{11})\cos(\phi)+\mathrm{Im}(A_{11})\sin(\phi)} \tag{3-55}$$

$$k=\frac{3A'_{11}}{2(1-l^2)^{\frac{3}{2}}}=\frac{3\mathrm{Re}[A'_{11}]}{2(1-l^2)^{\frac{3}{2}}}=\frac{3[\mathrm{Re}(A_{11})\cos(\phi)+\mathrm{Im}(A_{11})\sin(\phi)]}{2(1-l^2)^{\frac{3}{2}}} \tag{3-56}$$

设图 3-21(a)中的圆心坐标为(x,y)，垂点坐标为(x',y')，则有：

$$\begin{bmatrix}x'\\y'\end{bmatrix}=\begin{bmatrix}x\\y\end{bmatrix}+l\begin{bmatrix}\cos(\phi)\\\sin(\phi)\end{bmatrix} \tag{3-57}$$

利用式(3-57)确定的垂点是否作为最终的边缘点，这取决于 k 和 l 与设定的阈值间的大小关系。由于 l 的取值范围为[-1，-1]，因此边缘定位的精度是亚像素级的。对于数字图像，Zernike 矩的计算可以通过模板与图像的卷积实现，设模板大

小为 $N\times N$，则在定位边缘点时应将 l 放大 $N/2$ 倍，于是有：

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \frac{N}{2}\cdot l \begin{bmatrix} \cos(\phi) \\ \sin(\phi) \end{bmatrix} \tag{3-58}$$

在本文中，选取的模板大小为7×7，由于参数 k、l、ϕ 的确定需要 A_{11}、A_{20} 两个 Zernike 矩，故需要构造两个对应的模板 A_{11}、A_{20}。下面将给出模板的构造过程。如图 3-22 所示，用 7×7 的均匀网格分割单位圆，记第 i 行第 j 列正方形区域为 S_{ij}，C 为单位圆区域，则该正方形区域对应的模板系数 $A_{nm}(i,j)$ 为：

$$A_{nm}(i,j) = \iint_{S_{ij}\,\mathrm{I}\,C} V_{nm}^{*}\,dxdy \tag{3-59}$$

上述定义式实际是由 Zernike 矩的定义式演化而来的，只是将积分区域缩小，并取 $f(x,y)=1$。

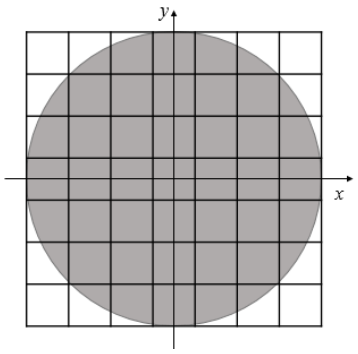


图 3-22 Zernike 矩 7×7 模板构造示意图

表 3-2、3-3、3-4 是根据式(3-59)构造出的模板，用于求亚像素边缘定位所需的 Zernike 矩 A_{11} 和 A_{20}。由于 A_{11} 包含实部和虚部，故将其拆分成两个独立的模板使用。

表 3-2 A_{11} 实数模板

0	-0.015	-0.019	0	0.019	0.015	0
-0.0224	-0.0466	-0.0233	0	0.0233	0.0466	0.0224
-0.0573	-0.0466	-0.0233	0	0.0233	0.0466	0.0573
-0.069	-0.0466	-0.0233	0	0.0233	0.0466	0.069
-0.0573	-0.0466	-0.0233	0	0.0233	0.0466	0.0573
-0.0224	-0.0466	-0.0233	0	0.0233	0.0466	0.0224
0	-0.015	-0.019	0	0.019	0.015	0

表 3-3 A_{11} 虚数模板

0	-0.0224	-0.0573	-0.069	-0.0573	-0.0224	0
-0.015	-0.0466	-0.0466	-0.0466	-0.0466	-0.0466	-0.015
-0.019	-0.0233	-0.0233	-0.0233	-0.0233	-0.0233	-0.019
0	0	0	0	0	0	0
0.019	0.0233	0.0233	0.0233	0.0233	0.0233	0.019
0.015	0.0466	0.0466	0.0466	0.0466	0.0466	0.015
0	0.0224	0.0573	0.069	0.0573	0.0224	0

表 3-4 A_{20} 实数模板

0	0.0225	0.0394	0.0396	0.0394	0.0225	0
0.0225	0.0271	-0.0128	-0.0261	-0.0128	0.0271	0.0225
0.0394	-0.0128	-0.0528	-0.0661	-0.0528	-0.0128	0.0394
0.0396	-0.0261	-0.0661	-0.0794	-0.0661	-0.0261	0.0396
0.0394	-0.0128	-0.0528	-0.0661	-0.0528	-0.0128	0.0394
0.0225	0.0271	-0.0128	-0.0261	-0.0128	0.0271	0.0225
0	0.0225	0.0394	0.0396	0.0394	0.0225	0

3.5.1 Zernike 矩亚像素检测精度验证

为验证Zernike矩亚像素边缘定位的精度，本文人工生成一幅棋盘格二值图像，图像的大小为400×400个像素，如图 3-23 所示。设置每个棋盘格的边长为 100 像素，则第一个格点区域（第 1 行至第 100 行，第 1 列至第 100 列区域）的灰度值为 255，第二个格点区域（第 1 行至第 100 行，第 101 列至第 200 列区域）的灰度值为 0。那么，这两个格点区域边缘的列坐标应在第 100 列和第 101 列中间，亚像素坐标为 100.5。以此类推，选取第 50 行、第 150 行、第 50 列、第 150 列，这四条线在图中用黄色线条标注，这四条线与黑白格边界各有三个交点，交点用红色圆圈标出，用本节提出的 Zernike 矩亚像素边缘定位算法求取这 12 个边缘点处的亚像素坐标，计算结果如表 3-5 所示。从表中数据可以看出，Canny 算子的检测精度只能精确到单个像素，与边缘点边缘的实际位置存在 0.5 个像素的误差。而 Zernike 矩亚像素边缘定位算法的定位精度可以达到 0.07 个像素，检测结果精确可靠，较 Canny 算子有了明显的提高。

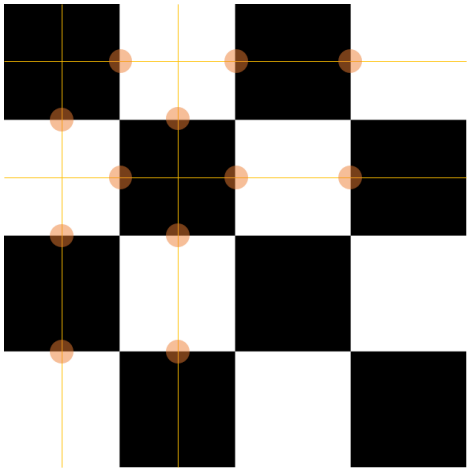


图 3-23 400×400 棋盘格二值图像

表 3-5 图像坐标检测结果

行、列位置	实际坐标	Canny 算子检测结果	Zernike 矩算法检测结果
第 50 行	100.5	100	100.42917
	200.5	200	200.42824
	300.5	300	300.42917
第 150 行	100.5	100	100.42824
	200.5	200	200.42917
	300.5	300	300.42824
第 50 列	100.5	100	100.42917
	200.5	200	200.42824
	300.5	300	300.42917
第 150 列	100.5	100	100.42824
	200.5	200	200.42917
	300.5	300	300.42824

利用 Zernike 矩亚像素边缘定位算法对本文的零件图像进行边缘检测后，得到的图像如图 3-24 所示。

由于 Zernike 矩算法是在 Canny 算子对边缘进行粗提取的基础上再进行亚像素定位，因此 Zernike 矩算法进一步去除了一些噪声和伪边缘，使提取出的边缘更细，定位更精确。

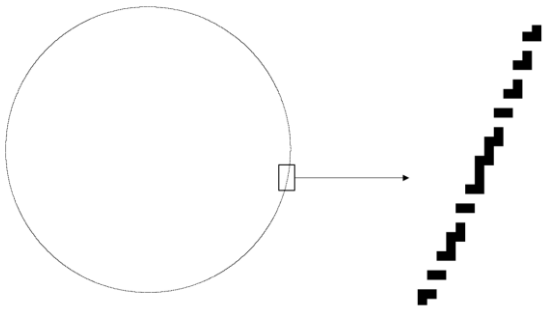


图 3-24 Zernike 矩算法边缘检测效果

3.6 像素尺寸测量

通过上述的加权平均值灰度变换、中值滤波去噪、Canny 算子边缘检测、Zernike 矩亚像素边缘定位后，零件图像的边缘已经提取完成，但要获得零件的像素尺寸，还要对提取的亚像素边缘坐标进行统计。

由于本文要测量对象为主活塞的轴径和导向盘的孔径，其中主活塞的轴在图像中呈现为两条平行的直线，导向盘的孔在图像中呈现为圆形，两者均具有明确的轮廓形状和灰度分布，因此适宜采用拟合法来得到直线和圆的函数表达式，从而计算出零件的像素尺寸。下面将介绍最小二乘拟合的方法步骤，它的思想是通过最小化误差的平方和来找到拟合函数的最佳参数匹配。

3.6.1 圆形拟合

对于导向盘的孔图像 $f(x,y)$，设孔的圆心为 (x_0,y_0)，半径为 r，则该圆的函数表达式为：

$$(x-x_0)^2+(y-y_0)^2=r^2 \tag{3-60}$$

将其展开改写为：

$$x^2+y^2+Ax+By+C=0 \tag{3-61}$$

其中，$A=-2x_0$，$B=-2y_0$，$C=x_0^2+y_0^2-r^2$。只需要求出 A、B、C，就可以求得圆的参数，得到圆心坐标与半径：

$$x_0=-\frac{A}{2} \tag{3-62}$$

$$y_0=-\frac{B}{2} \tag{3-63}$$

$$r=\frac{\sqrt{A^2+B^2-4C}}{2} \tag{3-64}$$

其中圆形轮廓上的点集(x_i,y_i)为上一节 Zernike 矩边缘检测得到的亚像素坐标，各亚像素点到圆心的距离的平方D_i^2为：

$$D_i^2=(x_i-x_0)^2+(y_i-y_0)^2 \tag{3-65}$$

因此各亚像素点到圆心的距离与圆的半径的平方差为：

$$\delta_i^2=D_i^2-r^2=x_i^2+y_i^2+Ax_i+By_i+C \tag{3-66}$$

令$G(A,B,C)=\sum_{i=1}^{n}\delta_i^2$，使目标平方差值的和$G(A,B,C)$最小，由极值条件得到：

$$\begin{cases}\dfrac{\partial G(A,B,C)}{\partial A}=\sum_{i=1}^{n}2(x_i^2+y_i^2+Ax_i+By_i+C)x_i=0\\ \dfrac{\partial G(A,B,C)}{\partial B}=\sum_{i=1}^{n}2(x_i^2+y_i^2+Ax_i+By_i+C)y_i=0\\ \dfrac{\partial G(A,B,C)}{\partial C}=\sum_{i=1}^{n}2(x_i^2+y_i^2+Ax_i+By_i+C)=0\end{cases} \tag{3-67}$$

解以上方程组可得：

$$A=\frac{NH-PQ}{MQ-N^2} \tag{3-68}$$

$$B=\frac{MH-PN}{N^2-MQ} \tag{3-69}$$

$$C=\frac{\sum_{i=1}^{1}(x_i^2+y_i^2)+Ax_i+B\sum_{i=1}^{1}y_i}{n} \tag{3-70}$$

其中，

$$M=n\sum_{i=1}^{n}x_i^2-\sum_{i=1}^{n}x_i\sum_{i=1}^{n}x_i \tag{3-71}$$

$$N=n\sum_{i=1}^{n}x_iy_i-\sum_{i=1}^{n}x_i\sum_{i=1}^{n}y_i \tag{3-72}$$

$$P=n\sum_{i=1}^{n}x_i^3+n\sum_{i=1}^{n}x_iy_i^2-\sum_{i=1}^{n}(x_i^2+y_i^2)\sum_{i=1}^{n}x_i \tag{3-73}$$

$$Q=n\sum_{i=1}^{n}y_i^2-\sum_{i=1}^{n}y_i\sum_{i=1}^{n}y_i \tag{3-74}$$

$$H = n\sum_{i=1}^{n} x_i^2 y_i + n\sum_{i=1}^{n} y_i^3 - \sum_{i=1}^{n}(x_i^2 + y_i^2)\sum_{i=1}^{n} y_i \tag{3-75}$$

因此，结合式(3-62)、(3-63)和式(3-64)，对本文的导向盘的孔图像进行圆形拟合，得到图 3-25 的标准圆形图像。其圆心在图像中的像素行列坐标为（472.99321，477.45716），圆的半径为 701.22588 个像素，这就是我们需要求的像素尺寸，下文可以利用标定系数可将像素尺寸转化为世界坐标系中的实际尺寸。

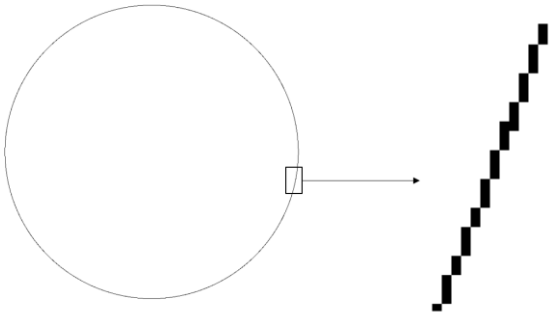


图 3-25 拟合边缘得到的圆

3.6.2 直线拟合

直线的拟合是用误差分析的方法来进行最小二乘线性回归的。对于主活塞的轴图像 $f(x,y)$，设其中一条直线的线性回归方程为：

$$\hat{y} = c_1 x + c_0 \tag{3-76}$$

其中，c_0、c_1为回归方程的常数和系数。将通过上一节 Zernike 矩边缘检测得到的亚像素坐标作为测量值点集，则误差函数为：

$$S = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{n}[y_i - (c_1 x_i + c_0)]^2 \tag{3-77}$$

由最小二乘法原理可以知道，要使回归直线与全部测量值最接近，则要求误差函数最小，根据极值条件得到：

$$\begin{cases} \dfrac{\partial S}{\partial c_0} = 2\sum_{i=1}^{n}(c_0 + c_1 x - y_i) = 0 \\ \dfrac{\partial S}{\partial c_1} = 2\sum_{i=1}^{n}(c_0 + c_1 x - y_i)x_i = 0 \end{cases} \tag{3-78}$$

解以上方程组可得：

$$c_0 = \frac{AD - BC}{nA - B^2} \tag{3-79}$$

$$c_1 = \frac{nC - BD}{nA - B^2} \tag{3-80}$$

其中，

$$A = \sum_{i=1}^{n} x_i^2, B = \sum_{i=1}^{n} x_i, C = \sum_{i=1}^{n} x_i y_i, D = \sum_{i=1}^{n} y_i \tag{3-81}$$

运用以上计算方法对本文的主活塞的轴图像进行直线拟合，得到如下图 3-26 的两条平行直线。

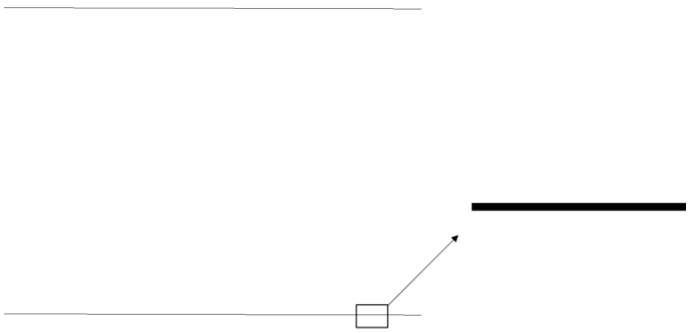


图 3-26 拟合边缘得到的两条直线

由计算得到的参数c_0，c_1可知两条直线方程分别为：

$$\begin{cases} y_1 = 2.75178 \times 10^{-3} x_1 + 125.56355 \\ y_2 = 2.77878 \times 10^{-3} x_2 + 821.47558 \end{cases} \tag{3-82}$$

可以看出，两条直线的斜率几乎都接近于零，但是却不完全相等，这说明两条直线并不严格地平行，这是由零件检测时的定位偏差和镜头的畸变等因素造成的微小误差。为尽量把平均误差降到最小，本文取其中一条直线段的中点，然后计算该点到另一条直线的垂直距离作为两直线间的平均距离。通过上述方法计算得到轴的像素尺寸为 695.92216 个像素。

3.7 本章小结

本章对图像处理与测量流程中各步骤的不同处理方法进行理论分析与实践，通过比较选出对本文轴孔零件图像有最佳处理效果的处理方法。在灰度变换中，本文选用了实用性更强的加权平均值法；在图像平滑中，本文选用了去噪效果最好的中值滤波法；在图像二值化中，本文选用了能自适应确定阈值的最大类间方差法；在像素级边缘检测中，本文选用了提取的边缘具有单像素宽、定位精度更高的

Canny 边缘检测算子。

为了进一步提高机器视觉尺寸测量系统的测量精度，本文在 Canny 算子提取出的像素级边缘的基础上，利用 Zernike 矩实现了边缘的亚像素定位。通过实验验证得出该方法可达到 0.07 个像素的定位精度。最后，本章通过最小二乘法拟合出孔图像的圆形和轴图像的两条直线，得到孔径和轴径的精确像素尺寸。

第四章 系统标定与零件测量

在图像采集过程中，由于所使用的相机属于非量测相机，对其内部参数并不完全知晓，相机本身存在一定的机械误差，镜头也会出现畸变，这些因素都会对成像质量造成一定影响，因此必须通过标定的方法来确定相机的内部参数和外部参数。

相机的内部参数指的是相机成像的基本参数，如主点（理论上是图像帧存的中心点，但实际上，由于相机制作的原因，图像实际中心与帧存中心并不重合）、设计焦距（与标称焦距有一定差距）、径向镜头畸变、切向镜头畸变以及其他系统误差参数；而相机外部参数指的是相机相对于外部世界坐标系的方位[37]。

在图像测量中，相机所获取的三维物体的二维图像是以像素为单位的，如何确定物体的三维空间坐标和二维图像的对应关系是相机标定工作所要解决的问题。空间物体表面某点的三维几何位置与其在图像中对应点之间的相互关系是由相机成像的几何模型决定的。这些几何模型参数就是相机参数，为了得到这些参数而进行的实验与计算的过程称为相机标定。

相机标定技术作为机器视觉测量与校正中的关键技术，主要包括相机成像模型建立与相机标定方法两个研究方向。针孔成像模型线性地表示了三维空间坐标和图像坐标之间的关系，是理想的相机成像数学模型，该模型计算方便，运算速度快，然而由于镜头的加工制造误差以及相机与镜头的安装误差导致相机成像过程中可能存在畸变，线性模型很难完整地表达镜头的像差等误差以及相机复杂的成像过程，使得所求三维空间点的坐标产生误差。若采用非线性模型，则可以模拟和补偿相机存在的畸变误差，但是在求解模型时，需要采用非线性优化算法来求解模型。

本章将首先建立相机成像的数学模型，接着对相机标定方法进行分析，选用合适的标定方法对相机进行标定，从而获得成像畸变校正所需的相机内部参数。

4.1 相机成像模型的建立

4.1.1 线性成像几何模型

在计算机视觉领域，通常线性成像几何模型是光学成像几何的简化，选用针孔模型作为相机成像的基本模型，其物理上相当于薄透镜成像。它的最大优点是成像关系式线性的，即假设物体表面的发射光都经过针孔投射到相机成像平面上，其成像模型如图 4-1 所示。

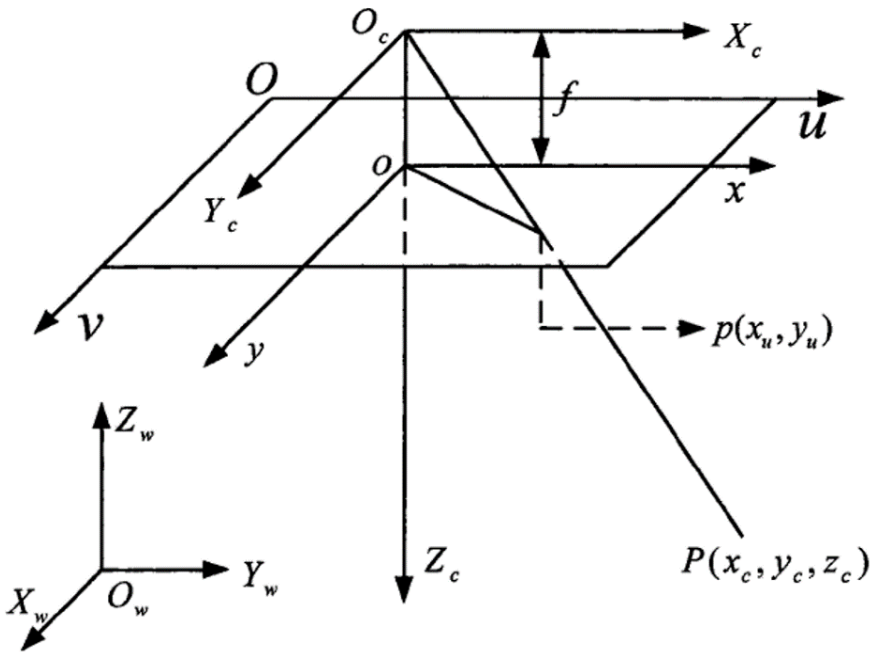


图 4-1 相机线性成像几何模型

其中，f为图像平面到光学中心的距离，即等效焦距；$(x_c, y_c, z_c, 1)$为 P 点在相机坐标系下的齐次坐标；$(x_u, y_u, 1)$是理想针孔相机下 P 点在图像物理坐标系下的齐次坐标，单位为 mm。图像物理坐标系到图像像素坐标系的变换关系如图 4-2 所示。

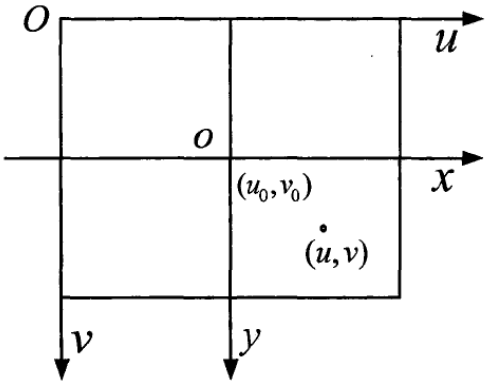


图 4-2 图像物理坐标系与图像像素坐标系之间的关系

在图像中，像素点的坐标(u, v)表示的是像素位于数字图像数组中的行数和列数，并没有用物理单位表示该像素在图像中的位置，为此建立了以 mm 为单位来表示的图像物理坐标系$o-xy$。对于相机来说，像个坐标系之间的关系依赖于像素的尺寸和形状，以及成像平面在相机中的位置。图像物理坐标系到图像像素坐标系的变换关系式为：

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{dx} & 0 & u_0 \\ 0 & \frac{1}{dy} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_u \\ y_u \\ 1 \end{bmatrix} \tag{4-1}$$

其中，$(u,v,1)$为P点成像在图像像素坐标系$O-uv$下的齐次坐标；dx、dy为图像像素坐标系中每个像素在x轴和y轴方向的物理尺寸（mm）；(u_0,v_0)为相机光轴与图像平面的交点，称为主点坐标。

相机坐标系到图像物理坐标系的变换关系式为：

$$Z_c \begin{bmatrix} x_u \\ y_u \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} \tag{4-2}$$

其中，$(X_c,Y_c,Z_c,1)$为P点成像在相机坐标系$O-X_cY_cZ_c$下的齐次坐标。

世界坐标系到相机坐标系的变换关系为：

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} R & t \\ O^T & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{4-3}$$

其中，$(X_w,Y_w,Z_w,1)$为P点成像在相机坐标系$O-X_wY_wZ_w$下的齐次坐标；R和T分别为世界坐标系到相机坐标系的旋转和平移变换矩阵，反映的是相机坐标系与世界坐标系之间的位置关系，因此成为外部参数。R是一个3×3的单位正交旋转矩阵，它只有三个自由度，由θ（俯仰角）、ϕ（旋转角）、ψ（侧倾角）三个角度决定，T为3×1的平移向量，R和T分别表示为：

$$R = \begin{bmatrix} \cos\phi\cos\theta & \cos\phi\sin\theta\sin\psi-\sin\phi\cos\psi & \cos\phi\sin\theta\cos\psi+\sin\phi\sin\psi \\ \sin\phi\cos\theta & \sin\phi\sin\theta\sin\psi+\cos\phi\cos\psi & \sin\phi\sin\theta\cos\psi-\cos\phi\sin\psi \\ -\sin\theta & \cos\theta\sin\psi & \cos\theta\cos\psi \end{bmatrix} \tag{4-4}$$

$$T^T = [T_x \quad T_y \quad T_z] \tag{4-5}$$

联立式(4-1)、式(4-2)和式(4-3)得：

$$
\begin{aligned}
Z_c\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} &= \begin{bmatrix} \frac{1}{dx} & 0 & u_0 \\ 0 & \frac{1}{dy} & v_0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} R & T \\ 0^T & 1 \end{bmatrix}\begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \\
&= \begin{bmatrix} f_u & 0 & u_0 & 0 \\ 0 & f_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} R & T^T \\ 0^T & 1 \end{bmatrix}\begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = M_1 M_2 W = MW
\end{aligned}
\tag{4-6}
$$

其中，$f_u = f/dx$，$f_v = f/dy$ 称为图像像素坐标系 u 轴和 v 轴的尺度因子；M 为 3×4 矩阵，称为透视变换矩阵；M_1 只与相机内部结构有关，称为相机内部参数矩阵；M_2 只与相机相对于世界坐标系的位置与姿态有关，称为相机外部参数矩阵。

4.1.2 非线性成像几何模型

在相机成像过程中，由于相机制造工艺等原因，相机的光学成像系统与理论模型之间存在着差异，使得二维图像存在这不同程度的非线性变形，如入射光线在通过各个透镜时的折射误差和成像平面点阵的位置误差等。相机非线性成像的的几何模型如图 4-3 所示。

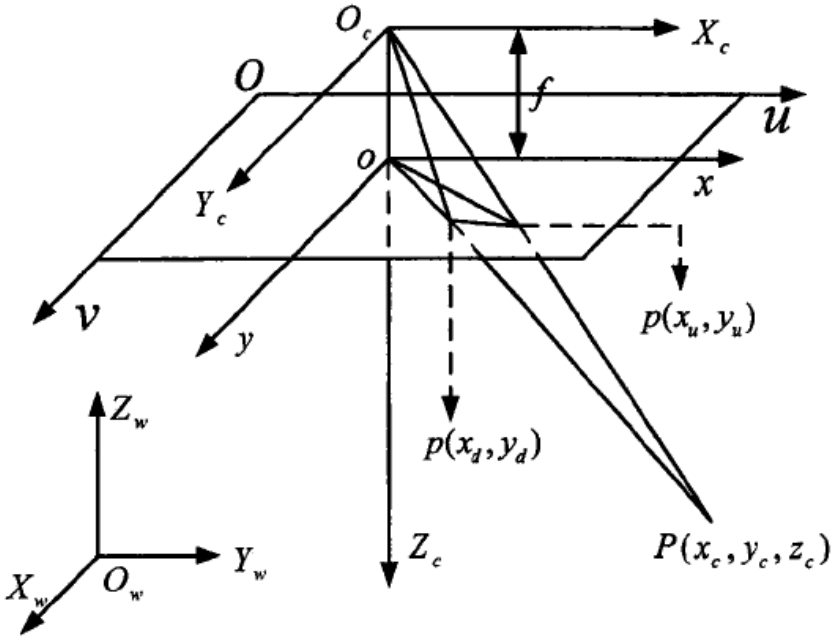


图 4-3 相机非线性成像几何模型

光学系统存在着的非线性几何失真，会使得目标像点与理论成像点相比存在

着多种类型的几何畸变。模型中的点 P 本应成像在 $p(x_u, y_u)$，却由于存在畸变发生了偏移，成像在了 $p(x_d, y_d)$。

常见的相机畸变包括径向偏移的径向畸变、切方向偏移的偏心畸变以及薄棱镜畸变等，如图 4-4 所示。

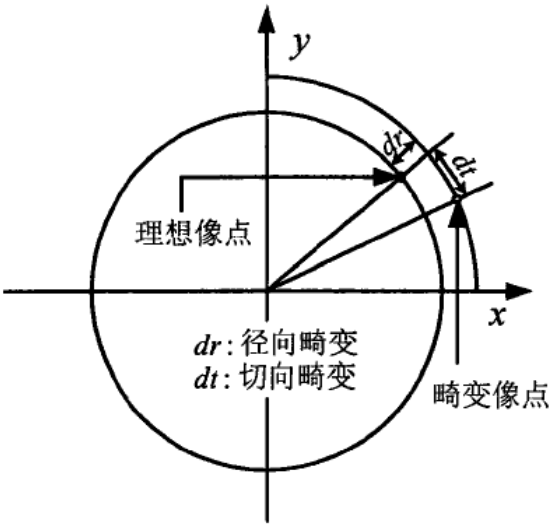


图 4-4 镜头畸变示意图

（1）径向畸变

径向畸变主要是由镜头形状缺陷造成的，是关于相机的主光轴对称的，其特点是像点的位置误差与它到光心的距离有关[38]。其数学模型为：

$$\begin{cases} \delta_{xr} = k_1 x(x^2 + y^2) + O[(x, y)^5] \\ \delta_{yr} = k_1 y(x^2 + y^2) + O[(x, y)^5] \end{cases} \tag{4-7}$$

其中，k_1 为径向畸变系数。

（2）偏心畸变

偏心畸变主要是由光学系统光心与几何中心不一致造成的，即各透镜的光轴中心不能严格共线，造成成像过程中的径向畸变和切向畸变。这种偏心畸变的数学模型为：

$$\begin{cases} \delta_{xd} = p_1(3x^2 + y^2) + 2p_2 xy + O[(x, y)^4] \\ \delta_{yd} = p_2(x^2 + 3y^2) + 2p_1 xy + O[(x, y)^4] \end{cases} \tag{4-8}$$

其中，p_1，p_2 为偏心畸变误差系数。

（3）薄棱镜畸变

薄棱镜畸变是由于镜头设计、制造缺陷和加工安装误差所造成的，如镜头与相

机成像面有很小的倾角。这类畸变相当于在光学系统中附加了一个薄棱镜，不仅会引起径向偏差，而且会引起切向偏差。薄棱镜畸变误差在工轴和轴上的分量模型为：

$$\begin{cases}\delta_{xp}=s_1(x^2+y^2)+O[(x,y)^4]\\ \delta_{yp}=s_2(x^2+y^2)+O[(x,y)^4]\end{cases} \tag{4-9}$$

其中，s_1，s_2为薄棱镜畸变误差系数。

综合考虑上述的几何畸变，其畸变误差模型表示为：

$$\begin{cases}\delta_x=k_1x(x^2+y^2)+2p_1xy+p_2x(3x^2+y^2)+s_1(x^2+y^2)\\ \delta_y=k_1y(x^2+y^2)+p_1y(x^2+3y^2)+2p_2xy+s_2(x^2+y^2)\end{cases} \tag{4-10}$$

实际光学系统成像模型的建立需要考虑几何畸变对成像过程的影响，对针孔成像模型进行修正。图像物理坐标系的理想坐标$P_u=(x_u,y_u)^T$与有畸变的实际图像点的的坐标$P_d=(x_d,y_d)^T$之间的关系为：

$$P_u=\begin{bmatrix}x_u\\ y_u\end{bmatrix}=\begin{bmatrix}x_d+k_1x_d(x_d^2+y_d^2)+2p_1x_dy_d+p_2x_d(3x_d^2+y_d^2)+s_1(x_d^2+y_d^2)\\ y_d+k_1y_d(x_d^2+y_d^2)+p_1y_d(x_d^2+3y_d^2)+2p_2x_dy_d+s_2(x_d^2+y_d^2)\end{bmatrix} \tag{4-11}$$

4.2 相机标定方法分析

近年来，相机标定方法得到了深入的研究，新的标定方法被不断研究出来，现有的相机标定方法可以归纳三类，即标定精度高的传统法、标定灵活的自标定法以及基于主动视觉法。

采用传统的相机标定方法时，必须知道标定物的大小，通过建立两个坐标系下点的相对联系，根据相关算法获得反映相机本身属性的内部参数以及反映相机空间位置的外部参数，标定物可以为二维的，也可以是三维的[39]。由于前者的制造与保存较容易，因此采用从不同的角度获得多幅图像进行标定的方法来求出所有参数。当应用场合要求的精度较高且相机参数不经常改变时，传统的相机标定方法可以获得较高的标定精度。

自标定相机标定法是一种十分灵活的标定方法，它可以根据多幅确定运动信息的拍摄对象的图像来完成标定。O. D. Faugeras，Q. T. Luong，S. T. Maybank 等人第一次研究出自标定的相关理论[40]，并成功在现场未知以及相机运动任意的情况下的标定相机。但是自标定方法也有其不足的地方，如该方法的算法稳定性差，标定精度不高，需要直接或间接地求解复杂的 Kruppa 方程[41]。

在采用基于主动视觉的相机标定方法标定相机时，根据运动相机平台的运动参数来标定。该方法不需要已知标定板，根据相机确定的运动获得的多幅图像来确

定相机参数，该方法过程简单，易操作[42]，但是对实验条件要求较高。

表 4-1 中表示了上述三类标定方法在四个方面的比较：

表 4-1 三类标定方法的比较

标定方法	精度	标定过程	对装置依赖程度	对标定物依赖程度
传统标定法	高	繁琐	低	高
基于主动视觉法	较高	较繁琐	一般	一般
自标定法	低	简单	高	低

基于上述分析比较，本文将采用介于传统标定法和自标定法之间的张正友标定法对相机进行标定。张正友标定法是指张正友教授在 1998 年提出的单平面棋盘格的相机标定方法[43]。这种标定法克服了传统标定法需要的高精度标定物的缺点，而仅需要使用一个打印出来的棋盘格就可以作为标定物。同时，它相对于自标定法而言，提高了标定精度，而且便于操作。因此，张正友标定法被广泛应用于机器视觉方面。

4.3 张正友标定法

张正友标定法主要考虑径向畸变以及偏心畸变，其畸变模型比式(4-11)的模型少了薄棱镜畸变的误差成分。同时，由于张正友标定法在考虑径向畸变时保留了更高阶次的误差成分，因此其畸变模型表达式为：

$$\begin{cases} x_u = x_d + x_d(k_1 r^2 + k_2 r^4 + k_3 r^6) + 2p_1 x_d y_d + p_2 x_d (r^2 + 2x_d^2) \\ y_u = y_d + y_d(k_1 r^2 + k_2 r^4 + k_3 r^6) + p_1 y_d (r^2 + 2y_d^2) + 2p_2 x_d y_d \end{cases} \tag{4-12}$$

式中，$r = \sqrt{x_d^2 + y_d^2}$，k_1、k_2、k_3 为径向畸变参数，p_1、p_2 为偏心畸变参数。

本文相机标定所要获得的相机参数主要包括内部参数矩阵中的 f_u、f_v、u_0、v_0 以及畸变系数 k_1、k_2、k_3、p_1、p_2。下面对张正友平面标定法的核心原理和计算步骤进行简要分析。

（1）设已知点的像素坐标为 $\tilde{m} = (u, v, 1)^T$，对应的三维空间点的坐标为 $\tilde{M}(X, Y, Z, 1)$，根据相机线性成像模型可得：

$$\lambda \tilde{m} = K \begin{bmatrix} R & t \end{bmatrix} \tilde{M} \tag{4-13}$$

式中，λ 为一个缩放因子标量，K 表示内部参数矩阵，$\begin{bmatrix} R & t \end{bmatrix}$ 为外部参数矩阵，R、t 分别表示旋转和平移矩阵。

令 Z 坐标为 0，将 R 表示为 $\begin{bmatrix} r_1 & r_2 & r_3 \end{bmatrix}$，则式(4-13)可以表示为：

$$\lambda\begin{bmatrix}u\\v\\1\end{bmatrix}=K\begin{bmatrix}r_1 & r_2 & r_3 & t\end{bmatrix}\begin{bmatrix}X\\Y\\0\\1\end{bmatrix}=K\begin{bmatrix}r_1 & r_2 & t\end{bmatrix}\begin{bmatrix}X\\Y\\1\end{bmatrix}=H\begin{bmatrix}X\\Y\\1\end{bmatrix} \tag{4-14}$$

其中，H 是一个3×3的单应性矩阵，H 可以表示为：

$$H=[h_1 \quad h_2 \quad h_3]=sK[r_1 \quad r_2 \quad t] \tag{4-15}$$

s 表示比例因子，根据标定物及其图像可以计算出相关的单应性矩阵 H，因为 R 是单位正交矩阵，所以 r_1，r_2 满足下式：

$$r_1^T r_2=0 \tag{4-16}$$

$$\|r_1\|=\|r_2\|=1 \tag{4-17}$$

将式(4-15)代入(4-16)和(4-17)并移项，可得：

$$h_1^T K^{-T}K^{-1}h_2=0 \tag{4-18}$$

$$h_1^T K^{-T}K^{-1}h_1=h_2^T K^{-T}K^{-1}h_2 \tag{4-19}$$

（3）令 $C=K^{-T}K^{-1}$，还需解出绝对二次曲线，才能计算出相机内部参数，由于 C 是一个对称矩阵，也可以表示成下式：

$$C=[C_{11} \quad C_{12} \quad C_{22} \quad C_{13} \quad C_{23} \quad C_{33}]^T \tag{4-20}$$

令 H 矩阵的第 i 列 $h_i=[h_{i1} \quad h_{i2} \quad h_{i3}]^T$，则联立式(4-18)、式(4-19)可得：

$$\begin{bmatrix}v_{12}^T\\(v_{11}-v_{22})^T\end{bmatrix}c=0 \tag{4-21}$$

式中，$v_{ij}=[h_{i1}h_{j1} \quad h_{i1}h_{j2}+h_{i2}h_{j1} \quad h_{i2}h_{j2} \quad h_{i3}h_{j1}+h_{i1}h_{j3} \quad h_{i3}h_{j2}+h_{i2}h_{j3} \quad h_{i3}h_{j3}]$。

若有 n 幅标定板的平面图像，则可以得到：

$$VC=0 \tag{4-22}$$

式中，V 是一个 $2n\times6$ 的矩阵，当 $n\geq3$，则 C 可以位移确定，求解出 C 后，再计算下列内部参数。

$$\begin{cases} v_0 = \dfrac{C_{12}C_{13} - C_{11}C_{23}}{C_{11}C_{22} - C_{12}^2} \\ \lambda = C_{33} - \dfrac{C_{13}^2 + v_0(C_{12}C_{13} - C_{11}C_{23})}{C_{11}} \\ f_u = \sqrt{\dfrac{\lambda}{C_{11}}} \\ f_v = \sqrt{\dfrac{\lambda C_{11}}{C_{11}C_{22} - C_{12}^2}} \\ s = -\dfrac{C_{12} f_u^2 f_v}{\lambda} \\ u_0 = \dfrac{sv_0}{f_v} - \dfrac{C_{13} f_u^2}{\lambda} \end{cases} \tag{4-23}$$

根据内部参数矩阵 K 以及单应性矩阵 H，由式(4-15)可以计算出对应图像的外部参数：

$$\begin{cases} r_1 = \lambda K^{-1} h_1 \\ r_2 = \lambda K^{-1} h_2 \\ r_3 = \lambda K^{-1} h_3 \\ T = \lambda K^{-1} h_3 \end{cases} \tag{4-24}$$

式中，$\lambda = 1/\left\| K^{-1}h_1 \right\| = 1/\left\| K^{-1}h_2 \right\|$ 为尺寸因子。

（4）考虑畸变，采用最大似然准则对上述参数进行求解。若对相机拍摄有 n 幅图像，每幅图像具有 m 个特征点，则最大似然估计值可以通过式(4-25)进行优化：

$$C = \sum_{i=1}^{n}\sum_{j=1}^{m}\left\| m_{ij} - m(K, k_1, k_2, k_3, p_1, p_2, R_i, t_i, M_i) \right\|^2 \tag{4-25}$$

其中，m_{ij} 为第 j 个点在第 i 幅图像中的像点，R_i 为第 i 幅图像的旋转矩阵，t_i 为第 i 幅图像的平移向量，M_j 为图像中第 j 个点的空间坐标。相机内部参数矩阵 K 和畸变系数 k_1、k_2、k_3、p_1、p_2 是本文需要求得的。

4.4 相机标定实验

为了得到拍摄零件图像的相机的参数信息，我们首先利用该相机在不同角度下拍摄 20 张棋盘格图片，图 4-5 展示了所拍摄的其中 6 张图片。

然后，我们根据上一节介绍的张正友平面标定法原理编写程序算法，对相机的参数信息进行计算，计算结果如表 4-2 所示。

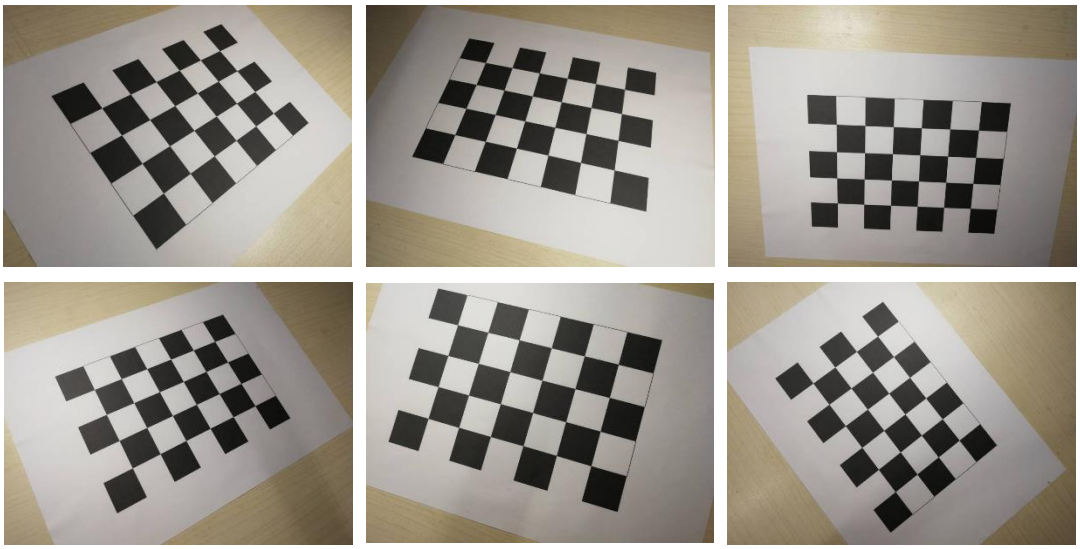


图 4-5 张正友标定法标定图像

表 4-2 相机参数标定结果

基于张正友法相机参数	张正友标定结果
f_u	3275.820677
f_v	3278.933371
u_0	2000.336430
v_0	1477.123576
k_1	0.373533
k_2	-3.733266
k_3	10.800031
p_1	2.148453×10^{-4}
p_2	6.413584×10^{-4}

经过标定校正后的导向盘孔图像如图 4-6 所示。

此外，本文计算了每张图片重投影坐标和提取的亚像素角点坐标之间的偏差，得到总体平均误差为 0.166502 个像素。

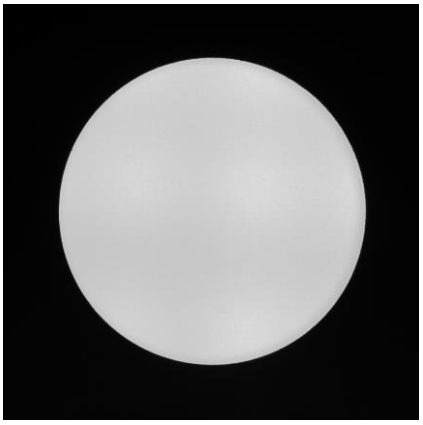


图 4-6 导向盘孔图像标定校正后的效果

4.5 标定系数的确定

在对零件图像的处理和测量后，我们得到的是零件在像素坐标系中的尺寸，即像素尺寸。虽然它反映了零件尺寸的特征，但并不是实际尺寸值。为了得到目标零件的实际尺寸，还需知道单个像素在世界坐标系中的物理尺寸。

在用上一节的张正友标定法进行标定后，我们只是求得了用来校正图像畸变的相机相关参数，还不能得到图像像素的标定系数。所谓标定系数，是指物像之间的尺寸比例关系，其定义式为：

$$k = \frac{l}{n} \tag{4-26}$$

其中，l 为被测物体的实际尺寸，n 为该物体在成像平面上所占有的像素个数，k 表示一个像素所对应的实际尺寸。

为了得到标定系数，本文采用了实验标定的方法，即通过对标准零件的测量实验来确定。由于标定过程也会引入误差，为了去掉系统误差，采用多次标定法来确定比例系数 k。试验证明，被测物体的实际尺寸和对应的像素个数之间满足如下关系：

$$l = kn + b \tag{4-27}$$

其中，b 为测量中的系统误差。通过两次标定即可确定 k 和 b 的值，这样就可消除系统误差对测量精度的影响。

本文中被测对象有两个，分别为导向盘的孔径和主活塞的轴径。由于它们被拍摄图片时与相机和镜头的距离是不一样的，所以它们的标定系数也是不相同的，因此要通过实验得到他们各自的标定系数。

我们首先取两个已知具有标准尺寸的的导向盘，其孔径实际尺寸分别为

16.020mm 和 16.030mm。通过第三章介绍的零件图像处理与测量方法，我们得到了两者的像素尺寸分别为 701.1862 个像素和 701.6247 个像素。由式(4-27)可得：

$$\begin{cases}16.020 = 701.1862k + b \\ 16.030 = 701.6247k + b\end{cases} \tag{4-28}$$

解以上方程组得到 $k = 2.2527\times10^{-2}\,\text{mm/pixel}$，$b = 0.2245\,\text{mm}$。于是导向盘孔径的实际尺寸表达式为：

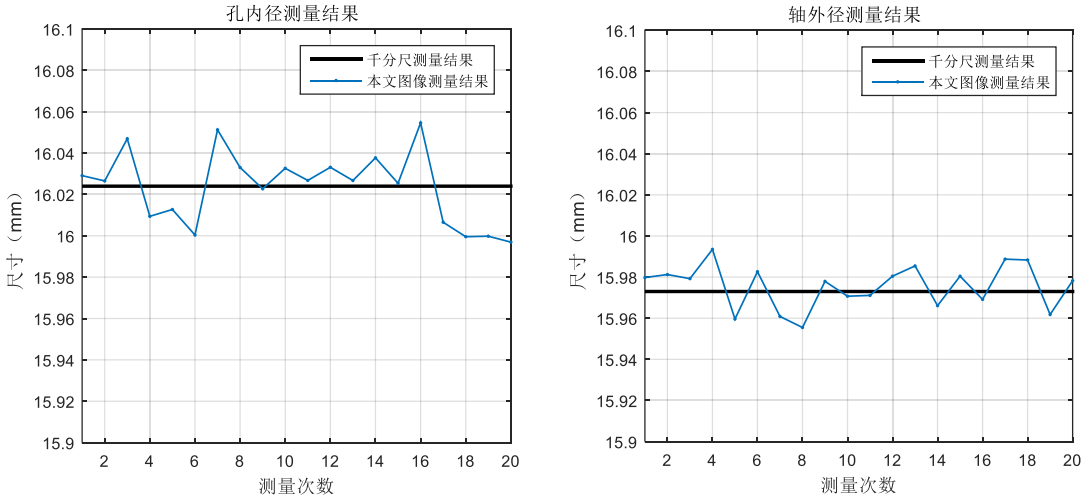
$$l = 2.2527\times10^{-2}n + 0.2245 \tag{4-29}$$

同理可得主活塞轴径的实际尺寸表达式为：

$$l = 2.2671\times10^{-2}n + 0.1968 \tag{4-30}$$

4.6 实际尺寸测量

为了充分验证本文尺寸测量系统的性能和测量精度，本文对轴孔零件进行了多次重复测量实验。通过用千分尺测得一导向盘的孔径尺寸为 16.024mm，一主活塞的轴径尺寸为 15.973mm，对这两个零件用相机分别拍摄 20 张图片。对这些零件图片成像校正后再进行图像处理与测量，最后将像素尺寸转换为实际尺寸。其测量结果的统计图如图 4-7 所示。



(a) (b)

图 4-7 轴孔零件尺寸测量结果。(a)孔径尺寸测量结果；(b)轴径尺寸测量结果

分析以上两子图可以得知，本文图像测量结果与用千分尺测量的结果非常接近。其中，本文经过图像测量所得到的孔径最大尺寸为 $L_{1\max}=16.0546\text{mm}$，最小尺寸为 $L_{1\min}=15.9969\text{mm}$，轴内径最大尺寸为 $L_{2\max}=15.9935\text{mm}$，最小尺寸为

$L_{2\min}=15.9555\text{mm}$，而 $L_1=16.024\text{mm}$，$L_2=15.973\text{mm}$ 为千分尺所测得的结果。因此，孔径尺寸的最大测量误差为：

$$\Delta E_1=\max\left\{\left|L_{1\max}-L_1\right|,\left|L_{1\min}-L_1\right|\right\}=0.0306\text{mm} \tag{4-31}$$

轴径尺寸的最大测量误差为：

$$\Delta E_2=\max\left\{\left|L_{2\max}-L_2\right|,\left|L_{2\min}-L_2\right|\right\}=0.0205\text{mm} \tag{4-32}$$

由上述测量结果可知，本文图像测量系统的测量精度基本控制在 0.03mm 以内，达到了预期设定的 0.05mm 检测精度要求。同时，本文还对每张图片处理与测量所需的时间进行了记录，其平均用时约为 537ms，这也达到了预期设定的 1s/件的检测效率要求。因此，本文基于机器视觉的零件图像测量系统具有较高的检测精度和检测效率。

4.7 误差分析

本文系统的测量结果与目标的实际尺寸相比误差在 0.03mm 范围内，误差产生的原因大致有如下几点：

（1）本文系统有固定的照明光源，但测量时依然受到环境光影响，光照的变化造成目标边缘亮度和阴影发生变化，影响系统成像的稳定程度。

（2）在图像采集过程中，硬件设备对系统产生了噪声干扰，噪声反映在图像上形成噪点。虽然通过滤波去噪方法降低了噪声影响，但滤波等处理过程本身也会造成边缘锐化程度减弱，导致图像发生模糊，影响图像质量。

（3）图像采集系统中的摄像机和光学镜头虽然经过了标定，但仍具有一定的误差，致使采集到的原始图像与目标实际形状有所偏差。

（4）图像测量精度受到相机分辨率的制约，本文通过改进的 Zernike 算子进行亚像素边缘检测，精度能够达到 0.07 个像素，与理想情况依然存在误差，经过算子处理的边缘信息随之产生误差。

除了上述几点，如搭建机器视觉检测平台时，轴孔的定位出现偏差，投影平面与镜头光轴没有完全垂直造成透视畸变等人为因素，同样影响系统测量精度，在后续的研究工作中，需要继续分析误差产生的原因，提高系统的测量精度和稳定性。

4.8 本章小结

本章首先介绍了相机的成像模型，继而分析了传统标定法、基于主动视觉法、自标定法的优劣，选定张正友标定法对相机进行标定，得到了成像校正所需的相机

内部参数。通过实验标定方法分别确定了轴和孔的标定系数，并利用标定系数将第三章中得到的零件像素尺寸转化为实际尺寸。多次的重复测量实验表明，本文的轴孔零件尺寸测量系统达到了 0.03mm 的测量精度，效率为 0.557s/件，满足预期设计要求。最后，本章对可能造成系统测量误差的因素进行了分析。

第五章 轴孔零件自动选配算法设计

目前在活门的装配中，主活塞的轴和导向盘的孔的选配都由人工在装配过程中完成，这样带来的问题是显而易见的。一方面，人工的选配效率较低，不适宜大批量零件的快速装配要求，也容易造成待选配零件积压严重，浪费工厂的存储空间。另一方面，对于精密零件的装配，人工选配难以达到最佳的匹配效果，甚至会出现零件装配不成功，或没有达到规定的装配精度要求，从而影响产品的功能和质量。因此，有必要通过设计零件的自动选配算法解决人工选配带来的问题，同时提高活门装配生产线的自动化程度。而本设计的轴孔零件选配系统正是活门智能装配生产线中的关键一环。

5.1 算法准备

对于已有的一批数量相等的主活塞（轴类零件）和导向盘（孔类零件），本文首先以 10 个零件为一组（数量可适量调整）把它们分成若干组，分别装在各自的专用物料盒中。然后，本文通过第二章介绍的机器视觉检测平台依次采集一组主活塞的轴图像和一组导向盘的孔图像。接着，本文利用第三章中的零件图像处理与测量算法得到了主活塞的轴径和导向盘的孔径的像素尺寸后，再通过第四章的系统标定将零件的像素尺寸转化为精密的实际尺寸，并将它们存储在数据库里面。为了节省存储空间，本文设计算法采用的策略是检测完一组轴孔零件的尺寸后，马上对这一组零件进行自动选配，将配对成功的轴和孔放在同一个输出物料盒内，以便后续的装配。而对尚未配对成功的轴和孔，则将它们放入配对失败的缓存物料盒，等待与下一组零件进行选配。

为了说明本文轴孔零件自动选配算法的步骤流程，本文首先给出了如表 5-1 的一组待选配的轴和孔尺寸。

已知轴的直径尺寸为$16^{-0.025}_{-0.040}$mm，孔的直径尺寸$16^{+0.040}_{+0.025}$mm，两者在装配时为间隙配合，配合间隙范围为 0.050~0.075mm，即如果孔径尺寸比轴径尺寸大 0.05~0.075mm 就可配对成功。在一组轴孔零件中，会存在多种配对关系，如一个轴类零件与多个孔类零件存在配合关系，或一个孔类零件与多个轴类零件存在配合关系。以表 5-1 给出的一组轴和孔尺寸数据为例，它们在配对时如果仅仅将编号相同的零件一对一地试配，则配对成功的结果如表 5-2 所示。

表 5-1 一组待选配的轴和孔尺寸

轴编号	轴尺寸（mm）	孔编号	孔尺寸（mm）
0001	15.9680	0001	16.0234
0002	15.9602	0002	16.0398
0003	15.9662	0003	16.0172
0004	15.9696	0004	16.0285
0005	15.9731	0005	16.0352
0006	15.9757	0006	16.0200
0007	15.9698	0007	16.0307
0008	15.9618	0008	16.0250
0009	15.9843	0009	16.0258
0010	15.9828	0010	16.0268

表 5-2 用一对一试配法配对成功的轴和孔

轴编号	轴尺寸（mm）	孔编号	孔尺寸（mm）	配合间隙（mm）
0001	15.9680	0001	16.0234	0.0554
0003	15.9662	0003	16.0172	0.0510
0004	15.9696	0004	16.0285	0.0589
0005	15.9731	0005	16.0352	0.0621
0007	15.9698	0007	16.0307	0.0609
0008	15.9618	0008	16.0250	0.0632

由上表可知，用一对一试配法配对成功的轴孔只有 6 对，配对失败的有 4 对。这种选配方法简单直接，但配对成功的概率较低，配对效率也较低。如果继续选配多组零件，将会导致配对失败的缓存物料盒积压越来越多的待选配零件，最终导致零件存放空间的不足。

5.2 算法流程

为解决上述选配方法的缺陷，本文以最大数量配对为准则，设计了配对成功概率高，且配对效率快的轴孔零件自动选配算法。该算法运用了贪婪法的核心思想，对当前待选配的 10 个轴和 10 个孔，穷举它们所有可能的配对情况，总共有10!种，

然后计算每种配对情况中满足配合间隙要求的对数，最终输出配对成功对数最大的那种配对情况。

值得注意的是，要遍历10!=3628800种配对情况，那样的运算量是非常大的。为了减小算法运算量，应当在多层嵌套循环中适当添加判断语句，当满足判断条件才进行计算或及时跳出循环，这样便极大减少了不必要的运算，同时缩短了运行所需的时间。以上面给出的一组数据为例，原本需要3628800次的运算量，经过算法优化，最终只需282843次运算便可得出同样正确的结果，运算量仅为原来的0.078倍，相应地，运行时间也减小为原来的十几分之一。可见自动选配算法的优化对算法性能的提升有很大的帮助。

运用本文的最大数量选配法对包含表5-1的一组轴孔零件尺寸在内的10组数据（100个轴径尺寸和100个孔径尺寸）进行最大数量配对，具体算法步骤如下：

（1）导入一组（10个）轴类零件的编号和尺寸、一组（10个）孔类零件的编号和尺寸，分别存储在数组$bh1[10]$、$a[10]$、$bh2[10]$和$b[10]$，将总配对成功对数sum初始化为零；

（2）将每次最大配对数量max初始化为零，利用$a[10]$、$b[10]$构造零一数组$c[10][10]$用于标记轴孔的配对关系，$c[i][j]$代表第i+1个孔与第$j+1$个轴的配对关系。计算轴孔两两组合时的配合间隙，若在0.050~0.075mm范围内，则两者满足配对要求，将数组中相对应的元素置为1，否则置为0。如第五个轴与第一个孔满足配对要求，则将$c[0][4]$置为1；

（3）对于数组$c[10][10]$，每行只取其中一个元素，且各元素所在的列数各不相同，从而得到10个$c[i][j]$，这就代表了一种配对情况。将当前配对情况的各$c[i][j]$值存储在数组$m[10]$中，并计算它们的和t。当t=10时，说明该组轴孔已达到最大数量配对，将当前配对情况的各$c[i][j]$的列下标存储在$n[10]$中，并执行第（4）步；当$t > max$，令max=t，将当前配对情况的各$c[i][j]$的列下标存储在$n[10]$中，并重复执行第（3）步，直到遍历完所有配对情况；

（4）输出该组最大数量配对结果中配对成功的max对轴孔编号、尺寸以及它们的配合间隙；

（5）令$sum = sum + max$，若$100 - sum < 10$则结束运行，否则，将未配对成功的$10 - max$对轴孔与下一组的前max对轴孔的编号和尺寸组成新的$bh1[10]$、$a[10]$、$bh2[10]$和$b[10]$，进行下一轮次的配对，跳回执行第（2）步。

算法流程图如图5-1所示。算法的具体代码详见附录1。

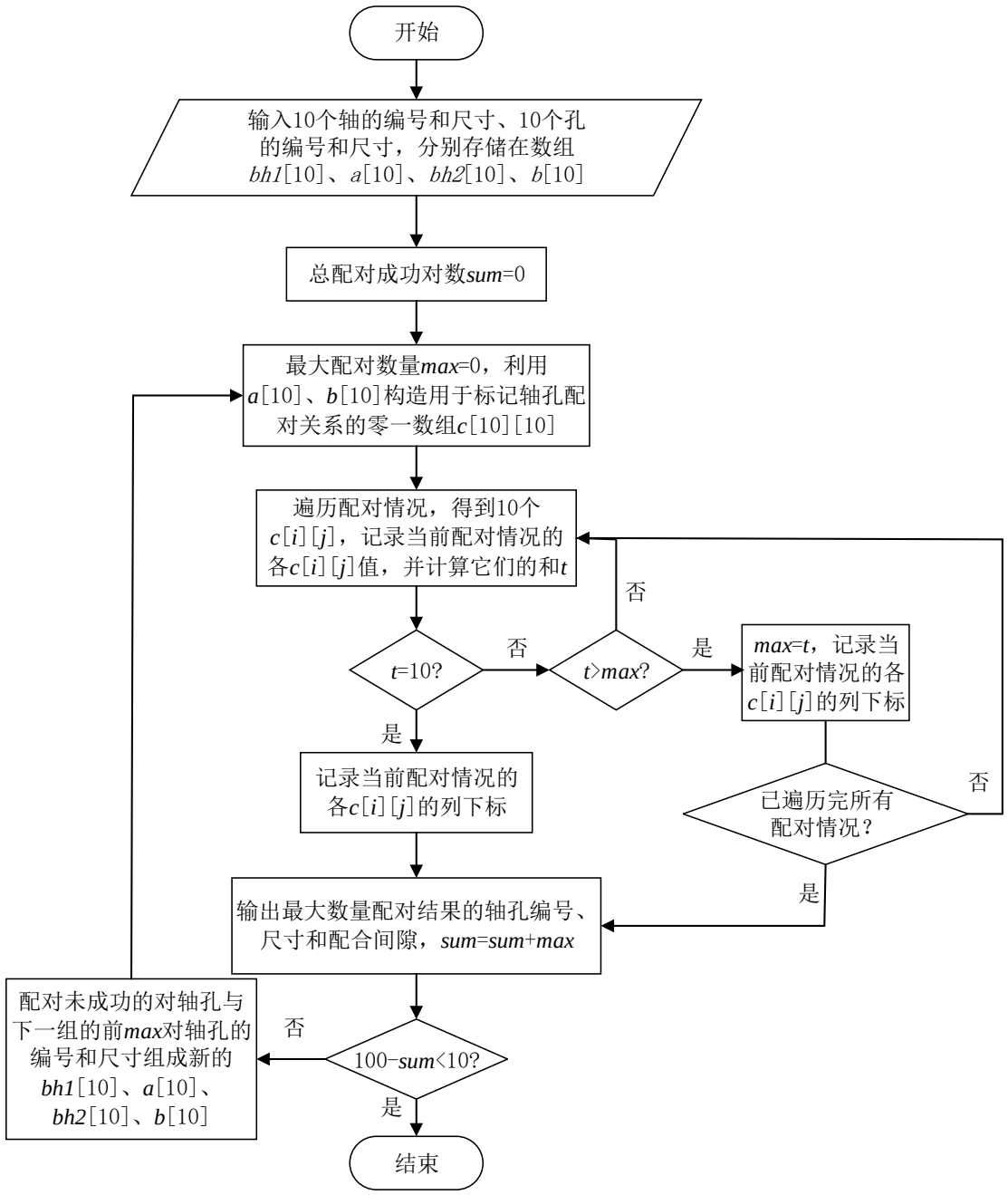


图 5-1 轴孔自动选配算法流程图

5.3 算法实现

运行以上算法程序，输出结果显示，最终配对成功的有 96 对轴孔零件，共分为 12 轮次最大数量选配，各选配轮次中配对成功的轴孔对数如表 5-3 所示。其中第 1 轮次为表 5-1 中的一组轴孔零件的选配，其配对结果为如表 5-4 所示。

表 5-3 各最大数量选配轮次中配对成功的轴孔对数

选配轮次数	配对成功的轴孔对数
1	10
2	10
3	10
4	6
5	7
6	6
7	5
8	6
9	8
10	9
11	10
12	8

表 5-4 第 1 轮次最大数量选配中配对成功的轴和孔

轴编号	轴尺寸（mm）	孔编号	孔尺寸（mm）	配合间隙（mm）
0001	15.9680	0001	16.0234	0.0554
0009	15.9843	0002	16.0398	0.0555
0002	15.9602	0003	16.0172	0.0570
0003	15.9662	0004	16.0285	0.0623
0010	15.9828	0005	16.0352	0.0524
0004	15.9696	0006	16.0200	0.0504
0005	15.9731	0007	16.0307	0.0576
0007	15.9698	0008	16.0250	0.0552
0006	15.9757	0009	16.0258	0.0501
0008	15.9618	0010	16.0268	0.0650

对比表 5-1 和表 5-4，我们可以看出本文的轴孔零件选配算法实现了最大数量配对，让配对的轴和孔的配合间隙均达到了很好的匹配效果。同时，本文对以上 12 轮次选配所用的时间进行了记录，约为 3.373s，平均每轮次的选配时间仅为 0.281s，可见本文选配算法的效率非常高，可实际运用到活门智能装配生产线中。

5.4 本章小结

本文首先利用一对一试配法对一组已知尺寸的轴孔零件进行选配，指出了该方法配对成功率较低、效率较低的缺点，进而以最大数量配对准则设计本文的轴孔零件自动选配算法，并给出了算法的具体流程。对本文最大数量选配算法进行运行仿真实验，结果表明该算法能对给出的轴孔零件实现了最大数量配对，平均每轮次的选配时间仅为 0.281s，配对成功率与配对效率都非常高，满足轴孔零件自动选配的实际应用需求。

第六章 轴孔零件测量选配系统软件设计与开发

本章将介绍基于机器视觉的轴孔零件测量选配系统的设计与开发，包括系统软件设计要求、系统的整个工作流程，以及软件功能和界面设计等。

6.1 系统软件设计要求简介

本系统软件的设计目标是能够根据轴孔零件图像精确测量出轴径和孔径的实际尺寸，并对满足装配精度要求的轴和孔进行自动选配。由这一目标，系统需要实现如下性能指标：

（1）能处理具有较强噪声和干扰的图片；

（2）对轴孔零件图片的测量精度达到亚像素级；

（3）每次对图片的处理时长低于 1s；

（4）对轴孔实现最大数量选配，每轮次选配时间不超过 1s；

（5）系统可以在较长时间内稳定健康运行；

（6）系统操作简单，有较好的用户体验和界面友好度。

为能够实现上述的性能指标，实现系统的快速、灵活处理，在 Windows10 系统下基于 Microsoft Visual Studio 2017 软件开发平台编写 MFC 对话框程序，并配合利用了 OpenCV 库用于图像的处理。

OpenCV（Open Source Computer Vision Library）是一个开源的计算机视觉和机器学习的软件库，它具有 C++，C，Python，Java 和 MATLAB 接口，可以在桌面（Windows，Linux，Android，MacOS，FreeBSD，OpenBSD）和移动设备（Android，Maemo，iOS）上运行。在模式识别中具有先进的算法，对人脸识别，物体检测，识别对象，无人机飞行，用户监视与安全系统，图像匹配等领域应用广泛。

OpenCV 通过运用优化的 C/C++代码编写出一系列跨平台的中高层 API，极大的发挥了多核处理器的优势，显著提升了图像处理的执行速度和数据的实时性。OpenCV 库中具有 20 多种模块，本系统软件运用到的模块包括 imgproc、core 和 highgui 等模块，它们的介绍如下：

（1）imgproc 模块：图像处理库，包括一些算法函数，主要实现对图像的滤波、图像的转换、图像的结构分析以及特征的检测等功能，是图像处理一系列过程中的基础库。

（2）core 模块：核心功能库，图像在像素层中处理时所用到，其中包含一些基本数据结构和动态数据结构，提供一系列函数与 OpenGL 配合操作。

（3）highgui 模块：高层 GUI 库，用于 HCI（人机交互）部分。

6.2 系统工作流程

总结前几章的主要内容，我们可以得到本文系统的总体工作流程如图 6-1 所示。我们首先对机器视觉检测平台的相机行进行标定，获取相机的内部参数，以此对成像进行校正。采集到轴孔零件的图像后，对图像进行灰度变换，使之由三通道的彩色图变换为单通道的灰色图，然后对图像进行滤波去噪，消除图像采集过程中产生的干扰信息，接着采用像素级边缘检测算子对图像边缘进行粗提取，继而再采用亚像素定位算法得到图像边缘的精确位置，利用亚像素坐标进行拟合便可求得零件的像素尺寸，最后通过标定系数将像素尺寸转化为世界坐标系中的实际尺寸，并将计算结果显示在用户界面上。

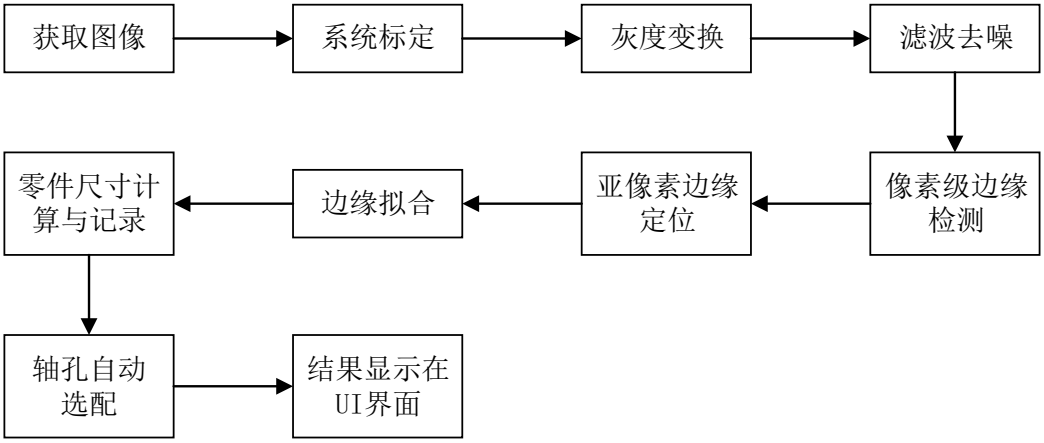


图 6-1 系统工作流程示意图

基于以上系统工作流程，本设计将系统软件的功能分为三个子对话框来实现，分别为图像处理与测量子对话框、尺寸测量结果子对话框以及轴孔选配结果子对话框，它们可通过 MFC 中的 Tab Control 控件实现自由切换。下面将介绍各子对话框所实现的功能和界面设计。

6.3 软件功能和界面介绍

6.3.1 图像处理与测量子对话框

图像处理与测量子对话框集成了第三章所介绍的多种图像处理功能，包括了灰度变换、图像二值化、滤波去噪、像素级边缘检测、亚像素边缘定位以及尺寸测量等模块，其软件界面如图 6-2 所示。

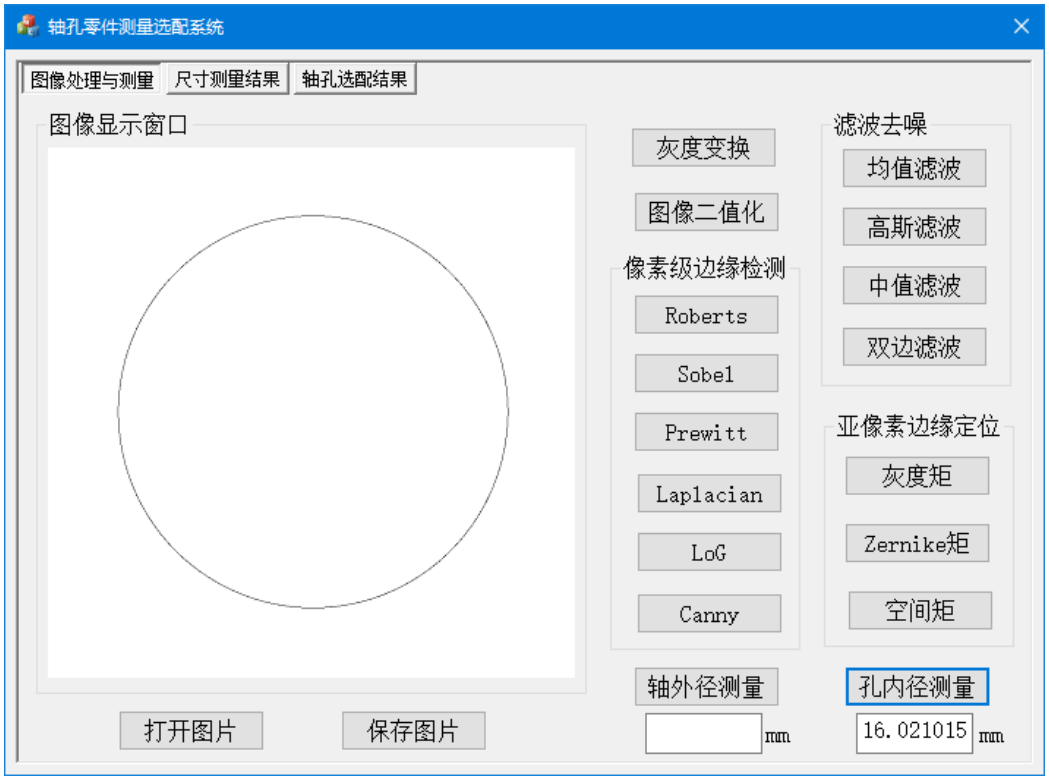


图 6-2 图像处理与测量子对话框的软件界面

在导入待处理的图像后，选定处理方式，点击相对应的按钮，系统会执行该算法处理，然后把处理后的图像显示在图像显示窗口。以下介绍各图像处理模块的实现方法。

（1）灰度变换模块

经第三章 3.1 节中对分量法、最大值法、平均值法和加权平均值法这四种灰度变换方法的分析比较，我们发现加权平均值法的实用性更强，故本模块默认采用加权平均值法对图像进行灰度变换。其核心算法语句为：

cvtColor(src, dst, COLOR_BGR2GRAY);

其中，cvColor( )为处理函数，src 为原始图像，dst 为处理后图像，COLOR_BGR2GRAY 为函数的设定参数。下文中出现的算法语句构成基本类似，只是处理函数不同，需要设定的参数个数和类型不同。

（2）图像二值化模块

经第三章 3.3 节中对直方图双峰法、迭代法和最大类间方差法的分析比较，我们发现最大类间方差法的二值化处理效果更佳，可以自适应确定阈值，无需人工干预，故本模块默认采用最大类间方差法对图像进行图像二值化。其核心算法语句为：

threshold(src, dst, 0, 255, CV_THRESH_BINARY | CV_THRESH_OTSU);

（3）滤波去噪模块

该模块包含了第三章 3.2 节中的四种图像滤波方法，如图 6-3 所示。它们对图像滤波有各自不同的适用范围。

图 6-3 滤波去噪模块

均值滤波、高斯滤波、中值滤波和双边滤波的核心算法语句依次如下：

```
blur(src, dst, Size(3, 3), Point(-1, -1));
GaussianBlur(src, dst, Size(3, 3), 1);
medianBlur(src dst, 3);
bilateralFilter(dst, dst1, 30, 30*2, 30/2);
```

（4）像素级边缘检测模块

该模块包含了第三章 3.4 节中的六种图像像素级边缘检测方法，如图 6-4 所示。它们对不同类型的图像有不同的边缘检测效果。在使用本系统软件时，可以用这六种算子分别对原始图像进行处理，观察比较图像显示窗口中的处理后图像，以决定选用对原始图像有最佳边缘检测效果的算子。

图 6-4 像素级边缘检测模块

Roberts 算子、Sobel 算子、Prewitt 算子、Laplacian 算子、LoG 算子和 Canny

算子的核心算法语句依次如下：

```
Roberts(src,dst,3);
Sobel(src,dst,3);
Prewitt(src,dst,3);
Laplacian(src,dst,3);
LoG(src,dst,5);
Canny(dst, dst1, 100, 40);
```

以上处理函数除 Canny 函数外全由自己定义，将各边缘检测算子的程序封装成相对应的函数，使程序变得更加简洁，提高了程序的可读性和可移植性。

（5）亚像素边缘检测模块

该模块包含了第三章 3.5 节中 Zernike 矩亚像素边缘定位方法，另外给出了灰度矩和空间矩方法作为比较，如图 6-5 所示。

图 6-5 亚像素边缘检测模块

灰度矩、Zernike 矩和空间矩亚像素边缘定位的核心算法语句依次如下：

```
Mat sub = GrayJu(src, dst);
Mat sub = ZernikeJu(src, dst);
Mat sub = SpaceJu(src, dst);
```

值得注意的是，以上函数的输出为边缘的亚像素坐标图像矩阵，可用于后续的边缘拟合处理。

（6）尺寸测量模块

该模块分为轴径尺寸测量和孔径尺寸测量，如图 6-6 所示。

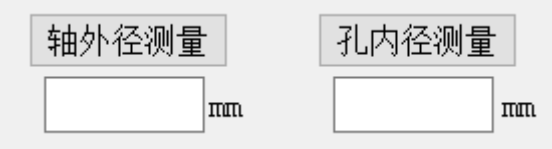


图 6-6 尺寸测量模块

在第三章中，我们通过分析比较已经选出各图像处理步骤中对本文轴孔零件图像处理效果最佳的方法。因此，我们可以得到如图 6-7 的图像处理与测量流程图。

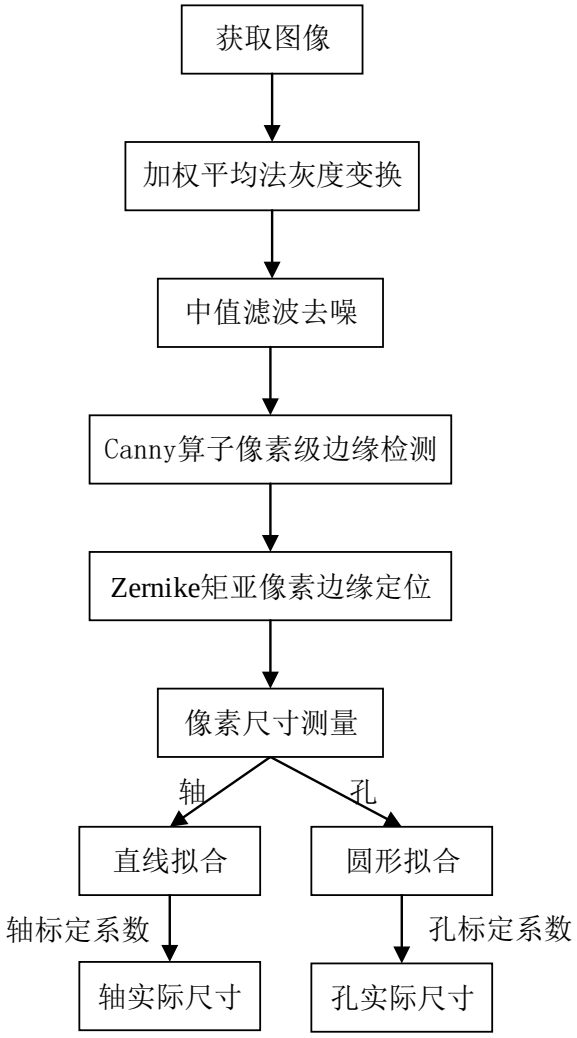


图 6-7 图像处理与测量流程图

于是，轴类零件图像处理与测量的核心算法语句为：

```
cvtColor(src, gray, COLOR_BGR2GRAY);
medianBlur(gray, dst, 3);
Canny(dst, dst1, 100, 40);
Mat sub = ZernikeJu(dst, dst1);
double D = LineFitting(sub);
double zhou = 0.022671*D + 0.1968;
```

孔类零件图像处理与测量的核心算法语句为：

```
cvtColor(src, src_gray, COLOR_BGR2GRAY);
```

```
medianBlur(src_gray, dst, 3);
Canny(dst, dst1, 100, 40);
Mat sub = ZernikeJu(dst, dst1);
double Radius = CircleFitting(sub);
double kong = 0.022527 * 2 * Radius + 0.2245;
```

其中，LineFitting( )为直线拟合函数，CircleFitting( )为圆形拟合函数，zhou 和 kong 则分别为轴类零件和孔类零件的实际尺寸，单位为 mm。当点击“轴径测量”或“孔径测量”按钮时，系统软件会运行以上程序，最终将它们的实际尺寸显示在对应的编辑框中。图像处理与测量的具体算法代码详见附录 2。

6.3.2 尺寸测量结果子对话框

尺寸测量结果子对话框用于存储图像处理与测量子对话框得到轴孔实际尺寸数据，其软件界面如图 6-8 所示。

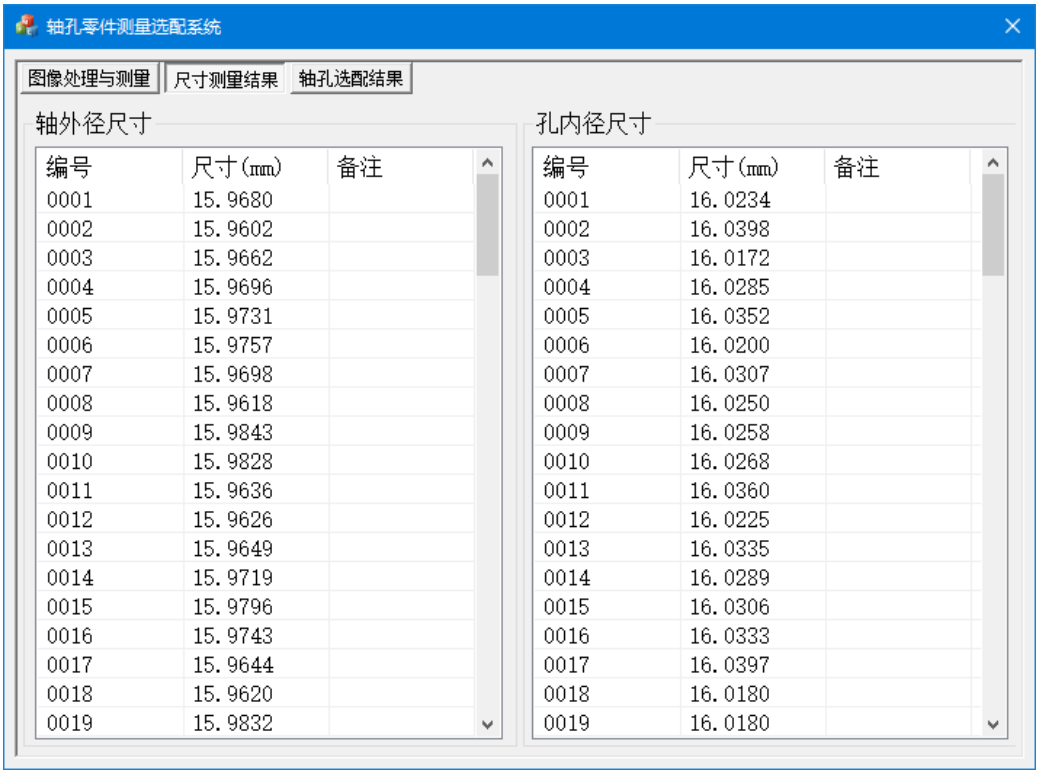


图 6-8 尺寸测量结果子对话框的软件界面

它由两个 List Control 控件组成，其中一个列表显示轴的编号及其外径尺寸，另一个列表显示孔的编号及其内径尺寸。当用户在图像处理与测量子对话框中点击“轴径测量”或“孔径测量”按钮时，系统软件将测得的实际尺寸添加到尺寸测量结果子对话框中的轴径尺寸或孔径尺寸列表中，便于用户随时查阅。

6.3.1 轴孔选配结果子对话框

轴孔选配结果子对话框由一个 List Control 控件组成，其软件界面如图 6-9 所示。

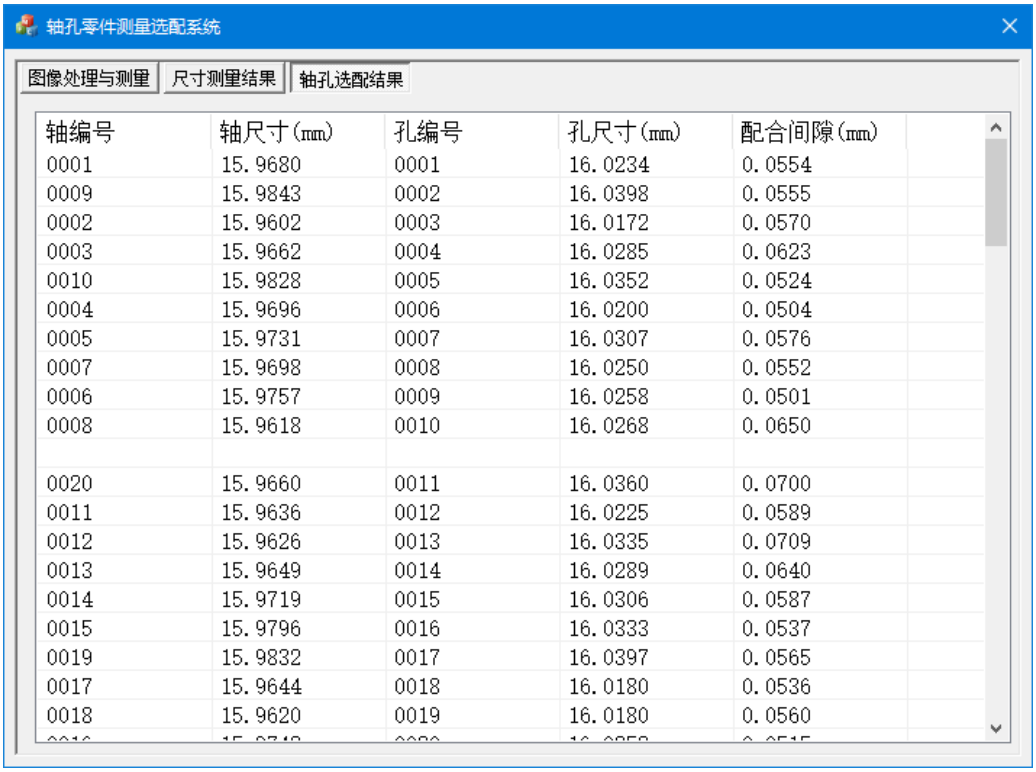


图 6-9 轴孔选配结果子对话框的软件界面

当用户由其他子对话框切换到轴孔选配结果子对话框，系统软件会根据尺寸测量结果子对话框中的数据用第五章中的轴孔自动选配算法对轴孔进行选配，并把配对成功的轴编号、轴径尺寸、孔编号、孔径尺寸以及轴孔的配合间隙显示在列表中，其中每轮次的配对结果用空行隔开。

6.4 本章小结

本章首先根据基于机器视觉的轴孔零件测量选配系统的设计目标分析了系统软件要实现的性能指标，并介绍开发本系统软件所需的编程环境平台和配合使用的 OpenCV 计算机视觉开源库。总结了系统的工作流程，基于此将系统软件的功能分为图像处理与测量、尺寸测量结果和轴孔选配结果三个子对话框来实现，展示了软件界面，并介绍了各模块的功能和工作机理。

第七章 总结与展望

7.1 总结

本论文以活门的主活塞的轴和导向盘的孔作为主要检测对象，针对这两种类型的零件特点，结合实际需求，选用合适的机器视觉检测平台硬件，利用机器视觉检测技术开发出了一套能够精密快速地对轴孔零件进行尺寸测量并实现自动选配的系统软件。

本论文的主要工作包括以下几点内容：

（1）通过文献资料的检索，总体介绍了机器视觉检测技术的概念、优点和关键技术，分析机器视觉检测技术的国内外研究现状和发展趋势。在机器视觉检测系统的通用模式的基础上，分析了各个硬件组成模块的功能和工作机理。

（2）对本文轴孔零件尺寸测量系统的需求和难点进行了详细分析，提出了合理的硬件和软件系统设计方案。在总体方案确定的基础上，针对系统的主要硬件组成部分进行了选型分析，选定采用带 COMS 传感器的工业相机作为摄像机，采用远心镜头作为光学镜头，采用 LED 光源作为照明光源。

（3）对图像处理与测量流程中各步骤的不同处理方法进行理论分析与实践，通过比较选出对本文轴孔零件图像有最佳处理效果的处理方法。在灰度变换中，本文选用了实用性更强的加权平均值法；在图像平滑中，本文选用了去噪效果最好的中值滤波法；在图像二值化中，本文选用了能自适应确定阈值的最大类间方差法；在像素级边缘检测中，本文选用了提取的边缘具有单像素宽、定位精度更高的 Canny 边缘检测算子。

（4）为了进一步提高系统尺寸测量的精度，本文在 Canny 算子提取出的像素级边缘的基础上，利用 Zernike 矩实现了边缘的亚像素定位。通过实验验证得出该方法可达到 0.07 个像素的定位精度。通过最小二乘法拟合出孔图像的圆形和轴图像的两条直线，得到孔径和轴径的精确像素尺寸。

（5）在建立相机的成像模型后，分析了传统标定法、基于主动视觉法、自标定法的优劣，选定张正友标定法对相机进行标定，得到了成像校正所需的相机内部参数。通过实验标定方法分别确定了轴和孔的标定系数，并利用标定系数将轴孔零件像素尺寸转化为实际尺寸。通过多次重复测量实验得出本文尺寸测量系统达到了 0.03mm 的测量精度，效率达到 0.557s/件，满足了预期设计要求。

（6）利用一对一试配法对一组已知尺寸的轴孔零件进行选配，指出了该方法配对成功率较低、效率较低的缺点，进而以最大数量配对准则设计本文的轴孔零件

自动选配算法，并给出了算法的具体流程。对本文最大数量选配算法进行运行仿真实验，结果表明该算法能对给出的轴孔零件实现了最大数量配对，平均每轮次的选配时间仅为 0.281s，配对成功率与配对效率都非常高，满足轴孔零件自动选配的实际应用需求。

（7）根据基于机器视觉的轴孔零件测量选配系统的设计目标分析了系统软件实现的性能指标和开发要求，总结了系统的工作流程，对系统软件的功能和界面进行了设计。

实际测量结果表明：本文提出的算法和设计的系统软件是有效可行的，能够精确检测轴孔零件的尺寸，同时整个系统的检测效率高，能够满足快速检测的要求，还实现了轴孔零件的最大数量配对。

7.2 展望

本文对于机器视觉检测技术中的图像处理、边缘检测、相机标定、轴孔自动选配等关键技术进行了研究，取得了一定的成果。在本文的研究基础上还有一些需要关注的问题值得在下一步工作中继续研究和探讨，具体包括：

（1）本文对测量目标的大小有一定的要求，原因在于相机成像平面尺寸和镜头的工作视野有限，而系统中没有加入图像拼接的相关技术，因而无法对大尺寸目标进行测量，在今后的研究中可通过图像拼接的方法生成目标的大尺寸图像。

（2）考虑在本系统中引入双目或多目测量技术，并使用传感器进行辅助测量，完成对目标三维，包括厚度和深度进行尺寸测量。

（3）由于矩方法是基于理想的阶跃边缘实现亚像素边缘定位的，本文轴零件图像的边缘存在灰度的过渡，并不是理想的阶跃边缘，因此定位精度还不能达到理想的效果。后续研究时，可以对本文涉及的算法深入研究，提出一种改进的亚像素边缘定位算法，使得检测速度更快，精度更高；

（4）本文采用的相机标定算法的标定过程由人工完成，较繁琐，若在生产过程中，相机或镜头被移动，则需要重新标定，在后续研究中，设计出标定精度高的相机自动标定软件系统，实现用户一键完成相机标定；

（5）本文机器视觉测量选配系统为实现轴孔零件尺寸测量和自动选配而设计，在今后应该实际应用到工业生产环节中，完成对工业制成品的在线实时检测和选配。因此，需要在系统中加入自动控制功能，提高系统的自动化程度。

综上所述，基于机器视觉的轴孔零件测量选配系统广泛实用化还有许多关键技术需要突破，需要我们不断总结和研究新的方法和技术，使系统达到更好的实际应用效果。

致　谢

四年的大学生活即将画上句号，在即将毕业之际，回顾本科求学的这四年，是不断成长、不断提升自己的过程。在李迅波老师的悉心指导下，本人漂亮地完成了本次毕业论文。李老师凭借丰富扎实的工程实践经验，以及对学术问题准确、深刻的分析和把握，使我在毕业设计过程中学到了很多与课题相关的专业知识，并得到了很好的科研训练。在此，我对李老师表示由衷的感谢。

其次，我要感谢教研室的师兄和同学们，他们在本人毕业设计过程中提供了很多帮助和建议，让教研室增添了许多笑声和温暖。同时，我要感谢我的室友，他们在本人的学习和生活中给予了很多关心和帮助。此外，我还要特别感谢我的家人，是他们多年来一直给予我物质上和精神上的支持和理解，让我有了坚强的后盾。

最后向所有关心、支持和帮助过我的老师、同学、亲人和朋友们再次致以诚挚的谢意！

参考文献

[1] 王中飞, 林茂松, 彭勇,等. 基于机器视觉的微片状物厚度检测[J]. 计算机测量与控制, 2013, 21(1):50-52.

[2] 黄晓波. 机械加工工艺对加工精度的影响[J]. 装备制造技术, 2012(9):143-145.

[3] 郭磊. 移动视觉精密测量关键技术研究[D]. 天津大学, 2011.

[4] Veni G, Regentova E.E, Zhang L. Detection of clustered microcalcifications[C]. International Conference on Image Analysis and Recognition, ICIAR, 2008:837-843.

[5] Kececi F, Nagel H H. Machine-vision-based estimation of pose and size parameters from a generic workpiece description[C] IEEE International Conference on Robotics and Automation, 2001. Proceedings. IEEE, 2001:2159-2164 vol.3.

[6] 韦光. 轴径测量的机器视觉技术研究[D]. 吉林大学, 2011.

[7] 周见行, 高红俐, 齐子诚,等. 基于摄像头自动跟踪定位的疲劳裂纹在线测量方法研究[J]. 中国机械工程, 2011, 22(11):1302-1306.

[8] 张平生, 张桂梅. 基于机器视觉的管孔类零件尺寸测量方法[J]. 机械设计与制造, 2012(12):139-141.

[9] 高琼. 基于 CMOS 机器视觉的圆筒形零件尺寸检测系统研究[D]. 中北大学, 2013.

[10] 卞晓东. 基于机器视觉的车辆几何尺寸测量系统研究[D]. 东南大学, 2005.

[11] 吴江. O 型密封圈几何尺寸视觉测量系统的关键技术研究[D]. 重庆大学, 2014.

[12] 李俊. 机器视觉照明光源关键技术研究[D]. 天津理工大学, 2006.

[13] 刘金桥, 吴金强. 机器视觉系统发展及其应用[J]. 机械工程与自动化, 2010(1):215-216.

[14] 孙志远. 插线机器人视觉定位算法研究[D]. 哈尔滨工程大学, 2012.

[15] 孙长胜, 吴云峰, 张传义,等. 智能相机发展及其关键技术[J]. 电子设计工程, 2010, 18(11):175-177.

[16] 张铮. 数字图像处理与机器视觉:Visual C++与 Matlab 实现[M]. 人民邮电出版社, 2014.

[17] 魏新华, 胡学同, 周杏鹏,等. 水果机器视觉自动分选机同步控制系统设计[J]. 农业机械学报, 2008, 39(11):99-103.

[18] Malamas E N, Petrakis E G M, Zervakis M, et al. A survey on industrial vision systems, applications and tools[J]. Image & Vision Computing, 2003, 21(2):171-188.

[19] 章毓晋. 图像理解与计算机视觉[M]. 北京: 清华大学出版社, 2000.

[20] 贾云得. 机器视觉[M]. 北京: 科学出版社, 2000, 49-53.

[21] 李了了, 邓善熙, 丁兴号. 基于大津法的图像分块二值化算法[J]. 微计算机信息, 2005(14):76-77.

[22] 刘蕊, 陈红卫. 一种改进的图像边缘检测算法[J]. 科学技术与工程, 2009, 9(21):6395-6398.

[23] 王丹. 基于各向异性高斯滤波的图像边缘检测方法[D]. 西安电子科技大学, 2010.

[24] 张元恒. 基于 CMOS 图像的陶瓷基片检测的算法研究与设计[D]. 山东理工大学, 2010.

[25] 初士博. 光学相关探测中物面滤波方法的研究[D]. 长春理工大学, 2008.

[26] 全太锋, 牟颖. 基于小波去噪的改进型 Canny 边缘检测法[J]. 四川大学学报(自然科学版), 2008, 45(6):1387-1389.

[27] 梅跃松, 杨树兴, 莫波. 基于 Canny 算子的改进的图像边缘检测方法[J]. 激光与红外, 2006, 36(6):501-503.

[28] 王植, 贺赛先. 一种基于 Canny 理论的自适应边缘检测方法[J]. 中国图象图形学报, 2004, 9(8):957-962.

[29] 朱秀昌, 刘峰, 胡栋. 数字图像处理与图像通信[M]. 北京邮电大学出版社, 2014.

[30] 杨文浩. 嵌入式机器视觉测控系统[D]. 江南大学, 2009.

[31] Elbaum M, Diament P. Estimation of image centroid, size, and orientation with laser radar[J]. Applied Optics, 1977, 16(9):2433.

[32] 付鹏, 高晓蓉. 基于矩的亚像素边缘检测算法的对比研究[J]. 微计算机信息, 2007, 23(18):264-265.

[33] 贺忠海, 王宝光, 廖怡白,等. 利用曲线拟合方法的亚像素提取算法[J]. 仪器仪表学报, 2003, 24(2):195-197.

[34] Jensen K, Anastassiou D. Subpixel edge localization and the interpolation of still images.[J]. IEEE Trans Image Process, 1995, 4(3):285-295.

[35] 郭彦珍, 尹国鑫. 应用概率论提高 CCD 尺寸测量的分辨力[J]. 仪器仪表学报, 1988(2):149-154.

[36] Seitz P. Optical Superresolution Using Solid-State Cameras And Digita; Signal Processing[J]. Optical Engineering, 1988, 27(7):277535.

[37] 赵小松, 张宏伟, 张国雄,等. 摄像机标定技术的研究[J]. 机械工程学报, 2002, 38(3):149-151.

[38] 袁野. 摄像机标定方法及边缘检测和轮廓跟踪算法研究[D]. 大连理工大学, 2002.

[39] 马伟. 计算机视觉中摄像机定标综述[J]. 价值工程, 2013(24):193-194.

[40] Faugeras O D, Luong Q T, Maybank S J. Camera self-calibration: Theory and experiments[M] Computer Vision — ECCV'92. Springer Berlin Heidelberg, 1992:321-334.

[41] Hartley R I. Kruppa's Equations Derived from the Fundamental Matrix[J]. Pattern Analysis & Machine Intelligence IEEE Transactions on, 1997, 19(2):133-135.

[42] 周明, 孙树栋. 遗传算法原理及应用:THEORY AND APPLICATIONS[M]. 国防工业出版社, 1999.

[43] Zhang Z.Y. A flexible New Technique for Camera Calibration[J]. IEEE Transactions on Patten Analysis and Machine Intelligence, 2000, 22(11):1330-1334.

附　录

1. 轴孔零件自动选配算法代码

```
clock_t startTime, endTime;
startTime = clock(); //记录程序开始运行时的时间
//读入轴零件的尺寸
ifstream fin1;
fin1.open("轴尺寸.txt");
double BH1, zhou[100];
for(int i = 0; i < 100; i++)
{
    fin1 >> BH1;
    fin1 >> zhou[i];
}
fin1.close();
//读入孔零件的尺寸
ifstream fin2;
fin2.open("孔尺寸.txt");
double BH2, kong[100];
for(int i = 0; i < 100; i++)
{
    fin2 >> BH2;
    fin2 >> kong[i];
}
fin2.close();
freopen("轴孔选配.txt", "w", stdout); //打开文件用于写入配对结果
int nn1 = 10, nn2 = 10, max = 0, idx = 1, sum = 0, bh1[10], bh2[10];
double a[10], b[10]; int c[10][10];
while (sum <= 100 - 10) //如果剩下待选配的轴孔不足 10 对，则退出循环
{
    //导入一组待选配的轴孔零件编号和尺寸
    if (max == 0)
```

```
{
    for (int i = 0; i < 10; i++)
    {
        a[i] = zhou[i];
        b[i] = kong[i];
        bh1[i] = i + 1;
        bh2[i] = i + 1;
    }
}
else
{
//上一组未配对成功的轴孔与下一组的轴孔组成新的编号数组和尺寸数组
    for (int i = 0; i < 10 - max; i++)
    {
    bh1[i] = temp3[i];
    bh2[i] = temp4[i];
    a[i] = temp1[i];
    b[i] = temp2[i];
    }
    for (int i = 10 - max; i < 10; i++)
    {
        a[i] = zhou[nn1++];
        b[i] = kong[nn2++];
        bh1[i] = sum + i + 1;
        bh2[i] = bh1[i];
    }
}
max = 0;
//构造用于标记轴孔配对关系的零一数组
for (int i = 0; i < 10; i++)
{
    for (int j = 0; j < 10; j++)
    {
```

```
            if (b[i] - a[j] >= 0.05 && b[i] - a[j] <= 0.075) //判断配合间隙
            {
                c[i][j] = 1;
            }
            else
            {
                c[i][j] = 0;
            }
        }
    }
    int m[10], n[10], p[10], q[10], temp3[10], temp4[10];
    double temp1[10], temp2[10];
    //遍历配对情况
    for(int j0 = 0; j0 < 10; j0++)
    {
        m[0] = c[0][j0];
        p[j0] = 1;
        for(int j1 = 0; j1 < 10; j1++)
        {
            if(p[j1] == 1) //若所选的轴已被配对，则继续该层循环
            continue;
            m[1] = c[1][j1];
            p[j1] = 1;
            for(int j2 = 0; j2 < 10; j2++)
            {
                if(p[j2] == 1)
                continue;
                m[2] = c[2][j2];
                p[j2] = 1;
                for(int j3 = 0; j3 < 10; j3++)
                {
                    if(p[j3] == 1)
                    continue;
```

```
                    m[3] = c[3][j3];
                    p[j3] = 1;
                    for(int j4 = 0; j4 < 10; j4++)
                    {
                        if(p[j4] == 1)
                        continue;
                        m[4] = c[4][j4];
                        p[j4] = 1;
for(int j5 = 0; j5 < 10; j5++)
{
    if(p[j5] == 1)
    continue;
    m[5] = c[5][j5];
    p[j5] = 1;
    for(int j6 = 0; j6 < 10; j6++)
    {
        if(p[j6] == 1)
        continue;
        m[6] = c[6][j6];
        p[j6] = 1;
        for(int j7 = 0; j7 < 10; j7++)
        {
            if(p[j7] == 1)
            continue;
            m[7] = c[7][j7];
            p[j7] = 1;
            for(int j8 = 0; j8 < 10; j8++)
            {
                if(p[j8] == 1)
                continue;
                m[8] = c[8][j8];
                p[j8] = 1;
                for(int j9 = 0; j9 < 10; j9++)
```

```
{
    if(p[j9] == 1)
    continue;
    m[9] = c[9][j9];
    p[j9] = 1;
    //计算该配对情况配对成功的对数
    int t = m[0] + m[1] + m[2] + m[3] + m[4] + m[5] + m[6]
        + m[7] + m[8] + m[9];
    //如果该配对情况已有 10 对轴孔配对成功，
    //保存该配对情况，并退出该组轴孔的遍历循环
    if(t == 10)
    {
        max = 10;
     //记录轴对应的列下标
     n[0] = j0; n[1] = j1; n[2] = j2; n[3] = j3; n[4] = j4;
     n[5] = j5; n[6] = j6; n[7] = j7; n[8] = j8; n[9] = j9;
     //记录轴孔配对关系
     q[0] = m[0]; q[1] = m[1]; q[2] = m[2]; q[3] = m[3];
     q[4] = m[4]; q[5] = m[5]; q[6] = m[6]; q[7] = m[7];
     q[8] = m[8]; q[9] = m[9];
        goto here;
    }
    //如果该配对情况可配对成功的对数为目前最大，
    //则保存该配对情况
    if(t > max)
    {
        max = t;
        //记录轴对应的列下标
        n[0] = j0; n[1] = j1; n[2] = j2; n[3] = j3; n[4] = j4;
        n[5] = j5; n[6] = j6; n[7] = j7; n[8] = j8; n[9] = j9;
        //记录轴孔配对关系
        q[0] = m[0]; q[1] = m[1]; q[2] = m[2]; q[3] = m[3];
        q[4] = m[4]; q[5] = m[5]; q[6] = m[6]; q[7] = m[7];
```

```
                        q[8] = m[8]; q[9] = m[9];
                    }
                                        p[j9] = 0;
                                    }
                                    p[j8] = 0;
                                }
                                p[j7] = 0;
                            }
                            p[j6] = 0;
                        }
                        p[j5] = 0;
                    }
                    p[j4] = 0;
                }
                p[j3] = 0;
            }
            p[j2] = 0;
        }
        p[j1] = 0;
    }
    p[j0] = 0;
}
here:
//如果有一组轴孔全都不能配对成功则退出程序
if (max == 0)
{
    fclose(stdout);
    return -1;
}
idx++;
sum += max; //累加当前已成功配对的总对数
for(int i = 0, j = 0; i < 10; i++)
{
```

```
            if(q[i] == 1) //输出配对成功的轴孔编号和尺寸
            {
                cout << setw(4) << setfill('0') << bh1[n[i]] << "          "
                 << setprecision(4) << setiosflags(ios::fixed) << a[n[i]]
                     << "          " << setw(4) << setfill('0') << bh2[i] << "          "
                 << setprecision(4) << setiosflags(ios::fixed) << b[i]
                     << "          " << setprecision(4) << setiosflags(ios::fixed)
                 << b[i] - a[n[i]] << endl;
            }
            else
            {
                temp3[j] = bh1[n[i]];
                temp4[j] = bh2[i];
                temp2[j] = b[i];
                temp1[j] = a[n[i]];
                j++;
            }
        }
        cout << endl;
    }
    fclose(stdout); //关闭输出文件
    endTime = clock();
    cout << "Totle Time : " << (double)(endTime - startTime) /
    CLOCKS_PER_SEC * 1000 << "ms" << endl; //输出程序运行用时
```

2. 图像处理与测量算法代码

```
    //轴径尺寸测量
    Mat src_gray, grad, dst, dst1;
    cvtColor(src, src_gray, COLOR_BGR2GRAY); //加权平均值法灰度变换
    medianBlur(src_gray, dst, 3); //中值滤波
    Canny(dst, dst1, 100, 40); //Canny 算子像素级边缘检测
    Mat sub = ZernikeJu(dst, dst1); //Zernike 矩亚像素边缘定位
    double D = LineFitting(sub); //直线拟合
    double zhou = 0.022671*D + 0.1968; //计算轴径实际尺寸
```

```
//孔径尺寸测量
Mat src_gray, grad, dst, dst1;
cvtColor(src, src_gray, COLOR_BGR2GRAY); //加权平均值法灰度变换
medianBlur(src_gray, dst, 3); //中值滤波
Canny(dst, dst1, 100, 40); //Canny 算子像素级边缘检测
Mat sub = ZernikeJu(dst, dst1); //Zernike 矩亚像素边缘定位
double Radius = CircleFitting(sub); //圆形拟合
double kong = 0.022527 * 2 * Radius + 0.2245; //计算孔径实际尺寸
//直线拟合函数定义
double LineFitting(Mat img)
{
    double XX1 = 0, XX2 = 0, X1 = 0, X2 = 0, XY1 = 0, XY2 = 0, Y1 = 0, Y2 = 0;
    int v1 = 0, v2 = 0;
    for (int i = 0; i < img.rows; i++)
    {
        double ff1 = img.at<double>(i, 1), ff2 = img.at<double>(i, 0);
        if (ff2 < src.rows / 2)
        {
            XX1 += ff1 * ff1;
            X1 += ff1;
            XY1 += ff1 * ff2;
            Y1 += ff2;
            v1++;
        }
        else
        {
            XX2 += ff1 * ff1;
            X2 += ff1;
            XY2 += ff1 * ff2;
            Y2 += ff2;
            v2++;
        }
    }
```

```
    //分别计算两条直线方程的参数
    double t = v1*XX1 - X1 * X1;
    double tt = v2*XX2 - X2 * X2;
    double c0 = (XX1*Y1 - X1 * XY1) / t;
    double c1 = (v1*XY1 - X1 * Y1) / t;
    double cc0 = (XX2*Y2 - X2 * XY2) / tt;
    double cc1 = (v2*XY2 - X2 * Y2) / tt;
    double x0 = src.cols / 2;
    double y0 = c1 * x0 + c0;
    double D = abs(cc1*x0 - y0 + cc0) / sqrt(cc1*cc1 + 1); //计算轴径像素尺寸
    return D;
}
//圆形拟合函数定义
double CircleFitting(Mat img)
{
    // TODO:  在此处添加实现代码.
    int i = 0;
    double X1 = 0, Y1 = 0, X2 = 0, Y2 = 0, X3 = 0;
    double Y3 = 0, X1Y1 = 0, X1Y2 = 0, X2Y1 = 0;
    for (i = 0; i < img.rows; i++)
    {
        double ff1 = img.at<double>(i, 1), ff2 = img.at<double>(i, 2);
        X1 = X1 + ff2;
        Y1 = Y1 + ff1;
        X2 = X2 + ff2 * ff2;
        Y2 = Y2 + ff1 * ff1;
        X3 = X3 + ff2 * ff2 * ff2;
        Y3 = Y3 + ff1 * ff1 * ff1;
        X1Y1 = X1Y1 + ff2 * ff1;
        X1Y2 = X1Y2 + ff2 * ff1 * ff1;
        X2Y1 = X2Y1 + ff2 * ff2 * ff1;
    }
    double C, D, E, G, H, N;
```

```
    double a, b, c;
    N = img.rows;
    C = N * X2 - X1 * X1;
    D = N * X1Y1 - X1 * Y1;
    E = N * X3 + N * X1Y2 - (X2 + Y2)*X1;
    G = N * Y2 - Y1 * Y1;
    H = N * X2Y1 + N * Y3 - (X2 + Y2)*Y1;
    a = (H*D - E * G) / (C*G - D * D);
    b = (H*C - E * D) / (D*D - G * C);
    c = -(a*X1 + b * Y1 + X2 + Y2) / N;
    //计算圆方程的参数
    double A, B, R;
    A = a / (-2);
    B = b / (-2);
    R = sqrt(a*a + b * b - 4 * c) / 2; //计算孔径像素尺寸
    return R;
}
//Zernike 矩亚像素边缘定位函数定义
Mat ZernikeJu(Mat img, Mat img1)
{
    //构造矩模板
    Mat M11R = (Mat_<double>(7, 7) << 0, -0.015, -0.019, 0, 0.019, 0.015, 0,
        -0.0224, -0.0466, -0.0233, 0, 0.0233, 0.0466, 0.0224,
        -0.0573, -0.0466, -0.0233, 0, 0.0233, 0.0466, 0.0573,
        -0.069, -0.0466, -0.0233, 0, 0.0233, 0.0466, 0.069,
        -0.0573, -0.0466, -0.0233, 0, 0.0233, 0.0466, 0.0573,
        -0.0224, -0.0466, -0.0233, 0, 0.0233, 0.0466, 0.0224,
        0, -0.015, -0.019, 0, 0.019, 0.015, 0);
    Mat M11I = (Mat_<double>(7, 7) << 0, -0.0224, -0.0573, -0.069, -0.0573, -
0.0224, 0,
        -0.015, -0.0466, -0.0466, -0.0466, -0.0466, -0.0466, -0.015,
        -0.019, -0.0233, -0.0233, -0.0233, -0.0233, -0.0233, -0.019,
        0, 0, 0, 0, 0, 0, 0,
```

```
        0.019, 0.0233, 0.0233, 0.0233, 0.0233, 0.0233, 0.019,
        0.015, 0.0466, 0.0466, 0.0466, 0.0466, 0.0466, 0.015,
        0, 0.0224, 0.0573, 0.069, 0.0573, 0.0224, 0);
    Mat M20 = (Mat_<double>(7, 7) << 0, 0.0225, 0.0394, 0.0396, 0.0394, 0.0225,
0,
        0.0225, 0.0271, -0.0128, -0.0261, -0.0128, 0.0271, 0.0225,
        0.0394, -0.0128, -0.0528, -0.0661, -0.0528, -0.0128, 0.0394,
        0.0396, -0.0261, -0.0661, -0.0794, -0.0661, -0.0261, 0.0396,
        0.0394, -0.0128, -0.0528, -0.0661, -0.0528, -0.0128, 0.0394,
        0.0225, 0.0271, -0.0128, -0.0261, -0.0128, 0.0271, 0.0225,
        0, 0.0225, 0.0394, 0.0396, 0.0394, 0.0225, 0);
    double minv = 0, maxv = 0;
    double* minp = &minv;
    double* maxp = &maxv;
    minMaxIdx(img, minp, maxp); //求图像最大的灰度值
    double lt, kt;
    lt = 0.7; kt = maxv / 3; //设定亚像素边缘的距离阈值和灰度值差阈值
    Mat m11I = Mat::zeros(img.rows, img.cols, CV_64F);
    Mat m11R, m20, theta, m11C, k;
    m11R.create(img.rows, img.cols, CV_64F);
    m20.create(img.rows, img.cols, CV_64F);
    theta.create(img.rows, img.cols, CV_64F);
    m11C.create(img.rows, img.cols, CV_64F);
    k.create(img.rows, img.cols, CV_64F);
    Mat l = Mat::ones(img.rows, img.cols, CV_64F);
    Mat sub1 = Mat::zeros(img.rows*img.cols, 2, CV_64F);
    Mat i1; int v = 0;
    i1.create(img.rows, img.cols, CV_16U);
    findNonZero(img1, i1); //找到 Canny 算子提取出的边缘像素的位置坐标
    double a, b, c, d, e, f, g;
    for (int i = 0; i < i1.rows; i++)
    {
        int ff1 = i1.at<int>(i, 1), ff2 = i1.at<int>(i, 0);
```

```
if (ff1 < 3 || ff1 > img.rows - 4 || ff2 < 3 || ff2 > img.cols - 4)
    continue;
//提取边缘像素点的周围领域的灰度值
Mat mu = (Mat_<double>(7, 7) << img.at<uchar>(ff1 - 3, ff2 - 3),
    img.at<uchar>(ff1 - 3, ff2 - 2), img.at<uchar>(ff1 - 3, ff2 - 1),
    img.at<uchar>(ff1 - 3, ff2), img.at<uchar>(ff1 - 3, ff2 + 1),
    img.at<uchar>(ff1 - 3, ff2 + 2), img.at<uchar>(ff1 - 3, ff2 + 3),
    img.at<uchar>(ff1 - 2, ff2 - 3),
    img.at<uchar>(ff1 - 2, ff2 - 2), img.at<uchar>(ff1 - 2, ff2 - 1),
    img.at<uchar>(ff1 - 2, ff2), img.at<uchar>(ff1 - 2, ff2 + 1),
    img.at<uchar>(ff1 - 2, ff2 + 2), img.at<uchar>(ff1 - 2, ff2 + 3),
    img.at<uchar>(ff1 - 1, ff2 - 3),
    img.at<uchar>(ff1 - 1, ff2 - 2), img.at<uchar>(ff1 - 1, ff2 - 1),
    img.at<uchar>(ff1 - 1, ff2), img.at<uchar>(ff1 - 1, ff2 + 1),
    img.at<uchar>(ff1 - 1, ff2 + 2), img.at<uchar>(ff1 - 1, ff2 + 3),
    img.at<uchar>(ff1, ff2 - 3),
    img.at<uchar>(ff1, ff2 - 2), img.at<uchar>(ff1, ff2 - 1),
    img.at<uchar>(ff1, ff2), img.at<uchar>(ff1, ff2 + 1),
    img.at<uchar>(ff1, ff2 + 2), img.at<uchar>(ff1, ff2 + 3),
    img.at<uchar>(ff1 + 1, ff2 - 3),
    img.at<uchar>(ff1 + 1, ff2 - 2), img.at<uchar>(ff1 + 1, ff2 - 1),
    img.at<uchar>(ff1 + 1, ff2), img.at<uchar>(ff1 + 1, ff2 + 1),
    img.at<uchar>(ff1 + 1, ff2 + 2), img.at<uchar>(ff1 + 1, ff2 + 3),
    img.at<uchar>(ff1 + 2, ff2 - 3),
    img.at<uchar>(ff1 + 2, ff2 - 2), img.at<uchar>(ff1 + 2, ff2 - 1),
    img.at<uchar>(ff1 + 2, ff2), img.at<uchar>(ff1 + 2, ff2 + 1),
    img.at<uchar>(ff1 + 2, ff2 + 2), img.at<uchar>(ff1 + 2, ff2 + 3),
    img.at<uchar>(ff1 + 3, ff2 - 3),
    img.at<uchar>(ff1 + 3, ff2 - 2), img.at<uchar>(ff1 + 3, ff2 - 1),
    img.at<uchar>(ff1 + 3, ff2), img.at<uchar>(ff1 + 3, ff2 + 1),
    img.at<uchar>(ff1 + 3, ff2 + 2), img.at<uchar>(ff1 + 3, ff2 + 3));
//卷积计算各边缘像素点的 Zernike 矩
m11I.at<double>(ff1, ff2) = M11I.dot(mu);
```

```
            m11R.at<double>(ff1, ff2) = M11R.dot(mu);
            m20.at<double>(ff1, ff2) = M20.dot(mu);
            a = m11I.at<double>(ff1, ff2);
            b = m11R.at<double>(ff1, ff2);
            c = m20.at<double>(ff1, ff2);
            theta.at<double>(ff1, ff2) = atan(a / b);
            d = theta.at<double>(ff1, ff2);
            m11C.at<double>(ff1, ff2) = b * cos(d) + a * sin(d);
            e = m11C.at<double>(ff1, ff2);
            l.at<double>(ff1, ff2) = c / e; //计算亚像素边缘的距离
            f = l.at<double>(ff1, ff2);
            k.at<double>(ff1, ff2) = 1.5 * e / pow(1 - pow(f, 2), 1.5); //计算亚像素边
缘的灰度值差
            g = k.at<double>(ff1, ff2);
            //筛除不满足阈值条件的亚像素边缘
            if (abs(f) < lt && abs(g) > kt)
            {
                double w1 = ff1 + 7 / 2 * f * sin(d);
                double w2 = ff2 + 7 / 2 * f * cos(d);
                J.at<uchar>(floor(w1), floor(w2)) = 255;
                sub1.at<double>(v, 0) = w1;
                sub1.at<double>(v, 1) = w2;
                v++;
            }
        }
        Mat sub = sub1.rowRange(0, v - 1).clone(); //得到亚像素边缘坐标位置
        return sub;
    }
```

外文资料原文

Image and Vision Computing 28 (2010) 1645–1658

Contents lists available at ScienceDirect

Image and Vision Computing

journal homepage: www.elsevier.com/locate/imavis

Sub-pixel edge detection based on an improved moment

Feipeng Da *, Hu Zhang

School of Automation, Southeast University, Nanjing 210096, China

ARTICLE INFO

Article history:
Received 7 January 2009
Received in revised form 19 March 2010
Accepted 16 May 2010

Keywords:
Edge detection
Sub-pixel edge location
Vision measuring

ABSTRACT

A novel moment-based method for sub-pixel edge location is proposed. Based on coarse edge-location by SOBEL operator, the geometric information of the target is used to reduce the number of moment-template to only one, which can largely save the time. Experimental results demonstrate that the proposed method is effective with more robustness, higher precision and speed.

1. Introduction

Edge is the basic character of image, and locating the edge fast and accurately takes important part in image processing and vision measuring. There are three methods in sub-pixel edge location: fitting-based method, interpolating-based method and moment-based method. Fitting-based method attempts to obtain sub-pixel edge location by fitting gray-scales of the image according to the given edge model [1–3]. It is high-accurate but time-consuming. Interpolating-based method performs the sub-pixel edge location by interpolating the image data in terms of gray-distribution in actual images [4–6]. It is time-saving but sensitive to noises. Since the moment is the integral operator which is non-sensitive to noises, the moment-based method became a widely used method.

Tabatabai and Mitchell firstly put the moment-based method into the sub-pixel edge location [7]. They obtained the final exact edges by computing four parameters of the edge model. After that, different methods based on Spatial-Gray Moment (SGM) were proposed [8–10]. Additionally, Cheng and Wu put the moment based method in color images [11]. As these methods based on SGM needed six moment-templates for convolution, efficiencies of the above moment-based methods were not so satisfactory. Lee et al. introduced an improved method with three moment-templates based on SGM. The grayscale of target and that of background were computed by bimodal property of histogram [12]. Once the two of four parameters in edge model were computed, only three moment-templates were needed. Ghosal proposed a method based on Zernike Orthogonal Moment (ZOM) which has orthogonality and rotation invariance for sub-pixel edge location [13]. The efficiency of ZOM method is higher than SGM method since it only needs three moment-templates for sub-pixel edge location. Recently, many authors have improved the moment-based methods of sub-pixel edge location. Bin et al. proposed a sub-pixel edge location method based on Orthogonal Fourier–Mellin Moment (OFMM) which has a better depicted ability for small shaped objects [14]. It pointed out that precision of OFMM method is higher than ZOM method. Li et al. took the amplification of moment-templates into account and selected more proper thresholds [15]. Qu et al. and Hu et al. proposed to coarsely locate edge points by SOBEL operator before using moment-based method in order to reduce computations [16,17].

In all the above moment-based methods, the number of moment-templates used in convolutions is not less than three, and therefore they may be computationally expensive. To improve the drawbacks of moment-based methods, a novel moment method for sub-pixel edge location is proposed in this paper. This method includes edge-location by using ROI and SOBEL, and most important it reduces the number of moment-template to be only one, which can largely save the computational time.

The paper will be discussed as following. A new method is introduced in Section 2. Experimental results and discussion are presented in Section 3 and conclusion is given in Section 4.

2. A novel moment-based method for sub-pixel edge location

In this section, a novel moment-based method for sub-pixel edge location is introduced. This method can be applied to locate the sub-pixel edge of objects which can be described by geometric languages. In the paper, the example of the object is the circular target which is one of representative geometric objects. For other geometric objects, the method can be slightly adjusted to its geometric parameters.

* Corresponding author. Tel.: +86 25 83794974; Fax: +86 25 83793000.
E-mail address: dafp@seu.edu.cn (F. Da).

0262-8856/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.imavis.2010.05.003

2.1. Preliminary steps

2.1.1. Initial solution for geometric parameters

For the original image, some basic image-processing steps, which include image filtering, image binary, boundary extraction and impurities removing, should be done. In these steps, the utilized filtering method in the paper is adaptive smoothing filtering, which could eliminate noises as well as maintain the original features of edges [18]. In addition, morphology method for the boundary extraction is taken [19]. After getting the boundary, the Least-Square method is used to fit these boundary points and the initial geometric parameters of objects can be obtained [20].

2.1.2. ROI (region of interest) extraction and edge detection

Usually, the geometric objects to be exacted are only one small part of the whole image. If we exact a ROI which is an area containing the geometric objects, the number of pixels participated in computations is largely reduced. This could make the algorithm much more effective and time-saving. In this paper, the geometric parameters give the solution to ROI-extraction. According to the geometric parameters of the object, one small area containing this object could be selected. This area is the ROI. The whole process for ROI-extraction is completely without any manual operations. Take circular targets as an example. The selected ROI is a square area whose center is the circular target's center, and the length of its side is the sum of circular target's diameter and a margin. The margin is necessary to prevent the influence of transition area, which is about 3–5 pixels.

Usually, the boundary points which are obtained by the morphology method are not quite accurate, and therefore SOBEL operator is utilized for edge detection in this paper as it has high speed and ability to smooth noises. Results of ROI extracting and edge detection are shown in Fig. 1.

The geometric parameters of objects are obtained by fitting the edge points detected by SOBEL operator using the Least-Square method.

2.2. Solution for parameters of the edge model and sub-pixel edge location

The edge of a continuous ideal image can be taken as a step-model, which is shown in Fig. 2, where h_1 is the background grayscale, h_2 is the object grayscale, l is the normalized distance between actual edge and the center, and θ is the angle between the edge and horizontal, which is satisfied as $\theta \in (-\frac{\pi}{2}, \frac{\pi}{2})$.

Furthermore, S_1 and S_2 respectively represent the area of target and background in the edge model. From the analyses of three moment-based methods in Section 1, it can be concluded that three moments of zero order are the same to each other. Here, we choose M_{00} as the mark for the moment of zero order.

$$M_{00} = \iint f(x,y)dxdy = S_1h_1 + S_2h_2 \tag{1}$$

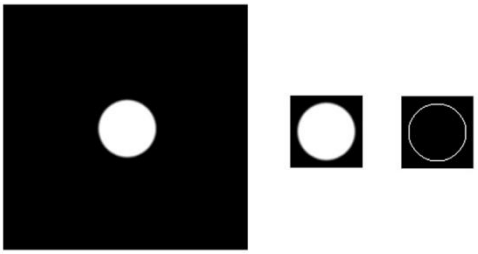


Fig. 1. ROI extraction and edge detection.

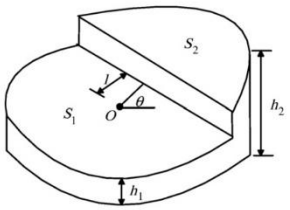


Fig. 2. 2D edge model.

Given the assumption that the edge model should be a unit circle, we have

$$M_{00} = (\pi - S_2)h_1 + S_2h_2 \tag{2}$$

2.2.1. Solution for θ

After rotating the edge model, the edge is vertical (along axis Y), so we have

$$\iint f(x,y)ydxdy = 0 \tag{3}$$

The left part of above equation could be converted to the expression which contains the moments, their combinations and θ. As a result of using the digitized template of moment, the accuracy of the results will be affected by some factors, which may include digitization, gray-scale sampling, the transition region of the edge and the number of pixels which participate in calculation of convolutions.

In this paper, the parameter θ is simply and easily calculated with the principle that gradient-orientation of the edge point is the orientation of normal line which just passes through it. Since the geometric parameters of the object are already calculated, the slope of the normal line could be obtained. Now θ is the inverse tangent of the slope. For a circular target, the orientation of the gradient can be solved in light of position of the center.

$$\theta = \tan^{-1}\frac{y_p - y_c}{x_p - x_c} \tag{4}$$

where (x_p, y_p) is the coordinate of the edge point, (x_c, y_c) is the coordinate of the center which can be obtained by geometric parameter of the circular target.

More widely, for quadratic curve $G(x, y)$,

$$G(x,y) = x^2 + Axy + By^2 + Cx + Dy + E = 0$$

the slope of the gradient is

$$k = \frac{\frac{\partial G(x,y)}{\partial y}}{\frac{\partial G(x,y)}{\partial x}} = \frac{Ax_p + 2By_p + D}{2x_p + Ay_p + C} \tag{5}$$

Apparently, the final precision of θ is influenced by initial geometric parameters. This will be discussed in Section 4.

2.2.2. Solution for h_1 and h_2

A method for separately solution to the gray-scale of target and background is presented in [12]. It utilizes the histogram of the whole image to choose the values of bimodal as h_1 and h_2. There are some drawbacks. First, it is time-consuming since it needs to add up the histogram and compare several times. Secondly, it is restricted to the cases for which the histogram of the image follows a bimodal

外文资料译文

基于一种改进矩的亚像素边缘检测

一. 引言

边缘是图像的基本特征，快速准确定位边缘在图像处理和视觉测量中占有重要地位。亚像素边缘定位有三种方法：基于拟合的方法，基于插值的方法和基于矩阵的方法。基于拟合的方法试图通过根据给定的边缘模型拟合图像的灰度来获得亚像素边缘位置，这种方法精度高但费时。基于插值的方法通过在实际图像中根据灰度分布内插图像数据来执行亚像素边缘位置，这种方法节省时间但对噪音敏感。由于矩是对噪声不敏感的积分算子，基于矩的方法成为广泛使用的方法。

Tabatabai 和 Mitchell 首先将基于矩的方法用于亚像素边缘定位。他们通过计算边缘模型的四个参数来获得最终的精确边缘。之后，提出了基于空间灰度矩（SGM）的不同方法。此外，Cheng 和 Wu 将这种基于矩的方法应用于彩色图像。由于基于上述 SGM 的这些方法需要六个卷积矩模板，所以上述基于矩的方法的效率并不令人满意。Lee 等人介绍了一种基于上下文模型的三个矩量模板的改进方法，目标和背景的灰度由直方图的双峰特性计算。一旦计算了边缘模型中的四个参数中的两个，就只需要三个矩量模板。Ghosal 提出了一种基于 Zernike Orthogonal Moment（ZOM）的方法，该方法对亚像素边缘位置具有正交性和旋转不变性。ZOM 方法的效率高于 SGM 方法，因为它仅需要三个矩阵用于亚像素边缘位置。最近，许多作者都改进了这一基于矩的亚像素边缘定位方法。Bin 等人提出了一种基于正交傅里叶梅林矩（OFMM）的亚像素边缘定位方法，该方法对小形状物体具有较好的描述能力，这种方法指出 OFMM 方法的精度高于 ZOM 方法。Li 等人考虑了矩模板的放大，并选择了更合适的阈值。Qu 和 Hu 等人提出在使用基于矩的方法之前为减少计量，用 Soble 算子粗略定位边缘点。

在所有上述基于矩的方法中，卷积中使用的矩量板的数目不小于 3，因此它们在计算量是很大的。为了改善基于矩的方法的缺点，本文提出了一种新矩方法用于亚像素定位。这种边缘定位方法包括使用 ROI 和 Soble 算子，最重要的是它将矩模板的数量减少到只有一个，这可以在很大程度上节省计算时间。

本文将作如下讨论。第二节介绍一种新方法。第三节给出该方法实验结果和讨论，第四节给出结论。

二．一种基于矩的亚像素边缘定位方法

在本节中，将介绍一种新的基于矩的亚像素边缘定位方法。该方法可用于定位可用几何语言描述的对象的亚像素边缘。在本文中，对象的例子是具有几何代表性的圆形目标对象。对于其他几何对象，该方法可以稍微调整为几何参数。

2.1 预备步骤

2.1.1 几何参数的初始解决方案

对于原始图像，应该进行一些基本的图像处理步骤，包括图像滤波，图像二值化，边界提取和杂质去除。在这些步骤中，所使用的滤波方法是自适应平滑滤波，它可以消除噪声并保持边缘的原始特征。此外，采用形态学方法进行边界提取。得到边界后，使用最小二乘法来拟合这些边界点，并且可以获得物体的初始几何参数。

2.1.2 ROI（感兴趣区域）提取和边缘检测

通常情况下，要显示的几何对象只是整个图像的一小部分。如果我们确定一个包含几何对象的区域的 ROI，则参与计算的像素数量将大大减少。这可以使算法更加有效和节省时间。在本文中，几何参数为解决 ROI 提取问题提供了解决方案。根据物体的几何参数，可以选择一个包含该物体的小区域，这个区域就是感兴趣区域。ROI 提取的整个过程完全没有任何手动操作。以圆形目标为例，选定的 ROI 是一个正方形区域，其中心是圆形目标的中心，其边长是圆形目标直径和边距之和。边距是必要的，可以防止过渡区域的影响，约为 3-5 像素的宽度。

通常，由形态学方法获得的边界点不是很精确，因此本文使用 Sobel 算子进行边缘检测，因为它具有很高的处理速度和平滑噪声的能力。ROI 提取和边缘检测的结果如图 1 所示。

物体的几何参数是通过使用最小二乘法对 Sobel 算子检测到的边缘点进行拟合来获得的。