机器人视觉全自动测量系统的设计

赵思源

（南京理工大学 机械工程学院，江苏 南京 210000）

摘 要：**目的：**为了实现对工件上孔型部分尺寸的全自动、高精度测量，建立了全自动工件测量系统，对该系统中像素当量值的标定以及圆孔的识别与测量方法展开分析。**方法：**首先，使用边缘检测算法提取图像的轮廓信息。接着，使用霍夫变换的方式提取出图中圆孔的大致信息，并综合使用此信息与通过闭运算消除噪声点后的图片边缘，提取图片上圆孔边缘的组成点。最后使用双线性插值的方法得到小孔的拟合圆的亚像素级信息，结合标定获得的像素当量值，获取零件的测量结果。**结果：**实验结果表明，各小孔的直径分别为9.6541、9.7074、9.7990、9.8477、9.8287、9.9484mm，各孔之间的距离分别为49.8345、49.6885、49.9114、49.8388、50.0044、49.9458mm，外围六个小孔所在中心圆的直径为997.411mm。**结论：**满足了机器人视觉全自动测量系统稳定性高，测量精度高，抗干扰能力强的特点。

关键词：机器人视觉；边缘检测；曲线拟合；亚像素

Design of Robot Vision Automatic Measuring System

Zhao Siyuan

（College of Mechanical Engineering, Nanjing Univ. of Sci &Tech., Nanjing 210000, China）

Abstract: Objective: In order to realize the automatic and high-precision measurement of the size of the hole pattern on the workpiece, a fully automatic workpiece measurement system was established, and the calibration of the pixel equivalent value in the system and the identification and measurement of the circular hole were analyzed. **Method:** First, use edge detection algorithm to extract the contour information of the image. Next, use the Hough transform to extract the general information of the hole in the picture, and use this information comprehensively with the edge of the picture after the noise point is eliminated by the closing operation, and extract the composition points of the edge of the hole in the picture. Finally, the bilinear interpolation method is used to obtain the sub-pixel level information of the fitting circle of the small hole, and the measurement result of the part is obtained by combining the pixel equivalent value obtained by the calibration. **Results:** The experimental results show that the diameters of the small holes are9.6541、9.7074、9.7990、9.8477、9.8287、9.9484mm , the distances between the holes are 49.8345、49.6885、49.9114、49.8388、50.0044、49.9458mm: , and the diameter of the central circle where the six peripheral small holes are located is997.411mm . **Conclusion:** It meets the characteristics of high stability, high measurement accuracy and strong anti-interference ability of the robot vision automatic measurement system.

Key words: robot vision; edge detection; curve fitting; subpixel

1. 引言

在全球经济激烈竞争的环境下，各类零件加工企业都在探索不同的方法来保证他们出厂产品的质量，其中对生产出来的零件进行形状与尺寸检测是零件产品质量保证的重要环节。目前，国内零件加工需求密集且质量要求高的零件制造行业中(如飞机制造业)，零件检测全部依赖于三坐标量测仪，然而其操作复杂，不仅检测速度无法满足在线实时检测的要求，而且检测精度与操作人员的熟练程度有关。

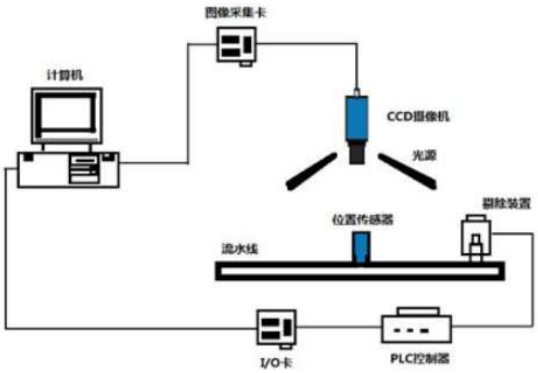
工业自动化的快速发展，使生产效率大幅提升，从而对检测效率提出了更高的要求。人工检测效率是在一个固定区间，无法大幅提升，而在流水线重复且机械化的检测过程中，检察人员很容易出现疲劳而导致检测效率降低;而机器视觉能够更快的检测产品，特别是在生产线检测高速运动的物体时，机器能够提高检测效率，速度甚至能够到达人工10-20倍；

由于人员有物理条件的限制，即使是依靠放大镜或显微镜来检测产品，也会受到主观性方面的影响，精度无法得到保证，而且不同的检测人员的标准也会存在有差异；在精确性上机器有明显的优点，它的精度能够达到千分之一英寸。而且机器不受主观控制，只要参数设置没有差异，相同配置的多台机器均能保持相同精度

视觉测量方法及系统的研究开发可以弥补当前检测方法的不足，进一步解决工业检测部门的实际问题。虽然国外视觉测量技术已经达到了实用阶段，多种产品已经投入使用，但是国内视觉测量技术尚处于发展阶段，因此自主研究相应的视觉测量方法，开发性能价格比高、操作方便的实用化系统是当前必要和迫切的要求。

2. 测量系统的构成及工作原理

2.1 测量系统的硬件组成



一个完整的机器人视觉系统包括：照明光源、光学镜头、 CCD 摄相机、图像采集卡、图像检测软件、监视器、通讯单元等。系统的大致构成如下图所示：

图 1 机器人测量硬件系统组成

Figure 1: Composition of the robot measuring system

2.2 测量系统的工作流程

当传感器探测到被检测物体接近运动至摄像机的拍摄中心，将触发脉冲发送给图像采集卡；图像采集卡根据已设定的程序和延时，将启动脉冲分别发送给照明系统和摄像机；一个启动脉冲送给摄像机，摄像机结束当前的拍照，重新开始一副新的拍照，或者在启动脉冲到来前摄像机处于等待状态，检测到启动脉冲后启动，在开始新的一副拍照前摄像机打开曝光构件（曝光时间事先设定好）；另一个启动脉冲送给光源，光源的打开时间需要与摄像机的曝光时间匹配；摄像机扫描和输出一幅图像；图像采集卡接收信号并通过A/D转换将模拟信号数字化，或者是直接接收摄像机数字化后的数字视频数据；图像采集卡将数字图像存储在计算机的内存中；计算机对图像进行处理、分析和识别，获得检测结果；处理结果控制流水线的动作、进行定位、纠正运动的误差等。

其工作流程如下图所示：

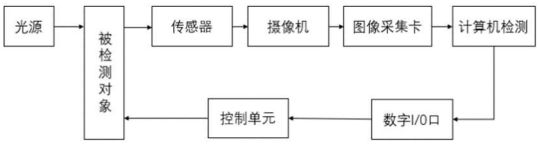


图 2：机器人测量系统流程

Figure 2: Robotic measurement system flow

2.3 图像处理与分析

图像处理与检测部分是机器人视觉测量的核心，其常规流程如下所示：

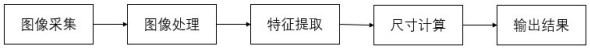


图 3：图像处理与检测流程

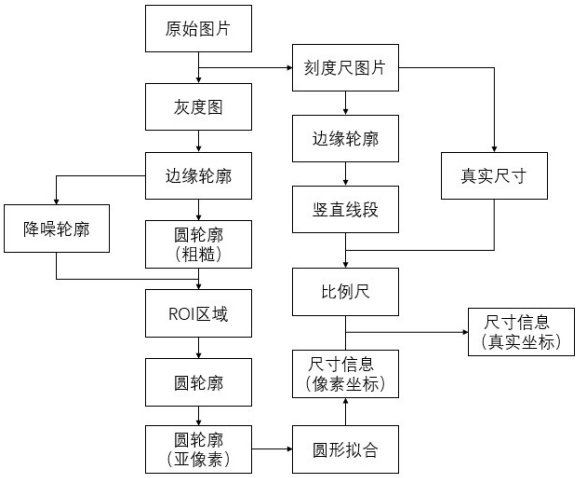
Figure 3: Image processing and inspection process

该部分首先针对需要实现的测量功能对图片进行预处理，综合运用灰度转换、滤波除噪等技术，完成初步处理，在通过边缘粗定位、亚像素边缘检测获取待测量部分的准确信息；再综合利用上述信息，完成直线、圆等图形的拟合，并进行测量，最终获取待测量工件的准确尺寸信息，完成图片测量任务。

本文的主要研究部分即为对此部分的分析与设计。

3. 测量系统软件的关键算法

3.1 总览



本文设计的测量系统总体流程图如下：

图 4：测量系统整体流程
Figure 4: The overall process of the measurement system

首先将原始图片转换为灰度图，对其进行边缘检测，获取图像的边缘信息；再利用霍夫检测高容忍度的特点，获取图像中圆的大致信息，由此确定图像中的重点检测区域，确定该区域后，综合闭运算后的图像边缘，消除掉刻度尺等非检测目标带来的影响，为后续线性插值提供了条件。使用封闭区域检测获得圆轮廓的具体像素点信息，并对其进行双线性插值和二次拟合，获得亚像素级别的高精度像素坐标；再使用圆形拟合获取圆孔像素坐标下的尺寸，结合利用霍夫直线检测刻度尺得到的真实尺寸与像素尺寸的比例，求得待检测圆孔的真实尺寸。由此完成机器人视觉测量系统的构建。

3.2 边缘检测算法

图像边缘是图像最基本的特征，边缘是指图像局部特性的不连续性。灰度或结构等信息的突变处称之为边缘。例如，灰度级的突变、颜色的突变,、纹理结构的突变等。边缘是一个区域的结束，也是另一个区域的开始，利用该特征可以分割图像。本文使用了Perwitt和canny算子分别对整体图像和刻度尺部分图像进行了边缘识别与提取。

图像中的边缘部分像素值分布大致如下图所示：

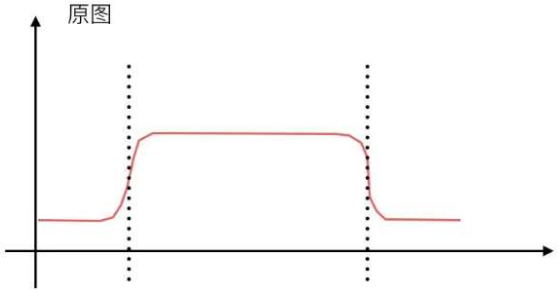


图 5：包含边缘的图像像素分布
Figure 5: Image pixel distribution with edges

其中，两条虚线的位置是边缘，变化剧烈，非边缘的部分较为平缓。

因此，对原始图像的像素值进行一阶微分：

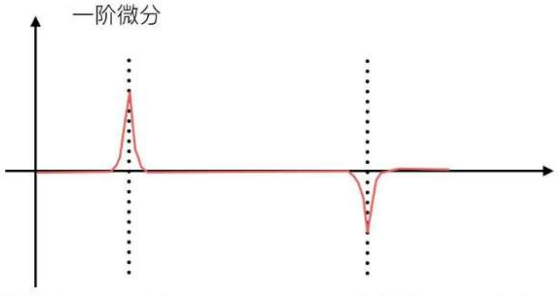


图 6：包含边缘的图像像素一阶微分值
Figure 6: Image pixel first-order differential values

可观察得知，经过微分后，平缓区域的微分值接近于0，而边缘区域的微分值绝对值较大，基于此原理，可通过计算图像各点的梯度值获得图像中的边缘区域。完成图片的边缘检测任务

假设待检测图像为 f，待检测算子采用计算图像的梯度值来寻找其在 (x, y) 处的边缘强度与方向，则梯度可表示为：

$$\nabla f(x,y)=\begin{pmatrix} g_x \\ g_y \end{pmatrix}=\left(\frac{\partial f}{\partial x},\frac{\partial f}{\partial y}\right)^T$$

该向量指出了图像 f 在 (x,y) 处最大变化率的方向与大小，将其大小和方向分别记为:

$$G(x,y)=\max(\nabla f)=\sqrt{{g_x}^2+{g_y}^2}$$

$$\alpha(x,y)=\arctan(\frac{g_y}{g_x})$$

边缘检测的任务转变成了对图片中像素点的梯度的计算。由于此测量系统的识别任务较为简单，使用一阶算子即可完成梯度的计算。一阶算子主要有以下四种形式。

3.2.1 Roberts 算子

Roberts算子分为水平部分算子和垂直部分算子：

$$G_x=\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},G_x=\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

该算子的详细公式如下:

$$G_x(i,j)=f(i+1,j+1)-f(i,j)$$

$$G_y(i,j)=f(i,j+1)-f(i+1,j)$$

对于该算法来说，当图像边缘接近于正 45° 或负 45° 时，该算法处理效果最为理想。然而该算法的一些不足在于边缘的定位不太准确，图像的噪声污染没有被排除以及检测出来的边缘轮廓较粗等。

3.2.2 Prewitt 算子

Prewitt 算子采用3x3的滤波算子不同的是Prewitt算子采用了3x3的算子来对图像特定区域中的像素值进行计算，算子形式如下:

$$G_x=\begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix},G_y=\begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{pmatrix}$$

因此使用此算子检测边缘轮廓比Roberts算子检测结果更为明显，但过于单一化和绝对化的判断形式会造成一些像素点的误判。

3.2.3 Sobel 算子

Sobel算子结合了高斯模糊和一阶微分并计算图像明暗程度的近似值，通过比较图像边缘的明暗程度把该区域内超过阈值的特定像素点记为边缘点。该算法在Prewitt算子的基础上增加了距离权重的概念，距离越近的像素点对于当前的像素点的影响越大其算子的形式如下：

$$G_x=\begin{pmatrix}-1 & 0 & 1\\ -2 & 0 & 2\\ -1 & 0 & 1\end{pmatrix},G_y=\begin{pmatrix}1 & 2 & 1\\ 0 & 0 & 0\\ -1 & -2 & -1\end{pmatrix}$$

该算法的缺点是没有将图像的主题和背景严格的划分开，这可能导致边缘轮廓模糊。

以上三种算子，都可以实现边缘检测，但都有部分缺点，最终选用了Canny算子，

3.2.4 Canny 算子

（1）首先对图片进行高斯滤波消除噪声，在本测量系统中，通过对高斯曲面进行离散采样和归一化，可获得尺寸为3×3，标准差为1的高斯核：

$$\begin{pmatrix}0.0751136 & 0.1238414 & 0.0751136\\ 0.1238414 & 0.2041799 & 0.1238414\\ 0.0751136 & 0.1238414 & 0.0751136\end{pmatrix}$$

将其与图像卷积，消除图像噪声。

（2）使用Sobel算子计算像素梯度

（3）非极大值抑制：将当前像素梯度强度与沿正负梯度方向上的相邻像素的梯度强度进行比较，若其最大（即为极值），则保留该像素为边缘点，若不是最大，则对其进行抑制，不将其作为边缘点。

（4）滞后阈值处理：定义一个高阈值和一个低阈值。梯度强度低于低阈值的像素点被抑制，不作为边缘点；高于高阈值的像素点被定义为强边缘，保留为边缘点；处于高低阈值之间的定义为弱边缘，

留 n 待进一步处理。

（5）孤立弱边缘抑制：通常而言，由真实边缘引起的弱边缘像素点将连接到强边缘像素点，而噪声响应则未连接。通过查看弱边缘像素及其8个邻域像素，可根据其与强边缘的连接情况来进行判断。一般，可定义只要其中邻域像素其中一个为强边缘像素点，则该弱边缘就可以保留为强边缘，即真实边缘点。

3.3 霍夫变换

霍夫变换是一个特征提取技术，可用于识别图像中特定的形状特征（直线、圆），其通过在参数空间内进行的投票程序在特定类型的形状内找到对象的实例，由于此算法可以容忍特征边界描述中的间隙，并且相对不受图像噪声的影响，在本文机器人视觉测量系统中，使用该算法粗略地寻找灰度图像中的圆形边界以及刻度尺标记。

3.3.1 霍夫直线检测

根据Hesse法线式，可将直线写为如下形式：

$$r=x\cos\theta+y\sin\theta$$

其中r是原点到直线上最近点的距离，θ是x轴与连接原点和最近点直线之间的夹角。如下图所示：

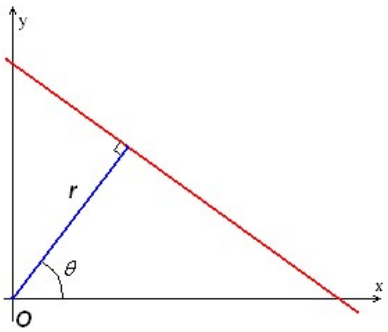


图 7：Hesse 法线式图像的符号表示

Figure 7: Symbolic representation of a Hesse normal image

因此，图像上的直线均可使用与之唯一对应的(r,θ)表示，由此便可将该直线映射至$r-\theta$霍夫空间上的一点。对于$x-y$平面上的一点，则可以映射为霍夫空间内的一条线。在此条件下，图像中的直线识别问题转换为了霍夫空间内多条线的交点问题。

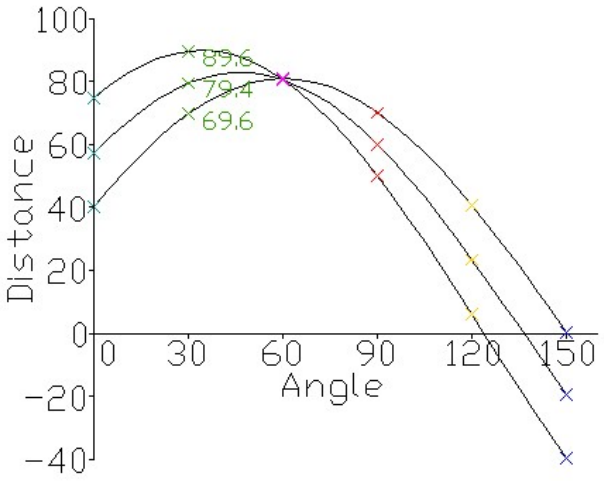


图 8：由三个点组成的霍夫空间图示例，粉色交点对应坐标即为三点确定直线

Figure 8: Example of a Hove spatial diagram consisting of three points, where the coordinates corresponding to the pink intersection points correspond to the three-point deterministic line

3.3.2 霍夫圆检测

霍夫圆检测的原理与霍夫直线检测类似，根据圆的一般方程：

$$(x-a)^2+(y-b)^2=r^2$$

任意圆均可映射至 $a-b-r$ 的三维霍夫空间中的一点。同时，$x-y$ 图像坐标系下的一点映射至霍夫空间呈一以 (x,y) 为顶点的圆锥，相同圆上的点所映射的圆锥会产生一交点，该交点即为圆的对应点。

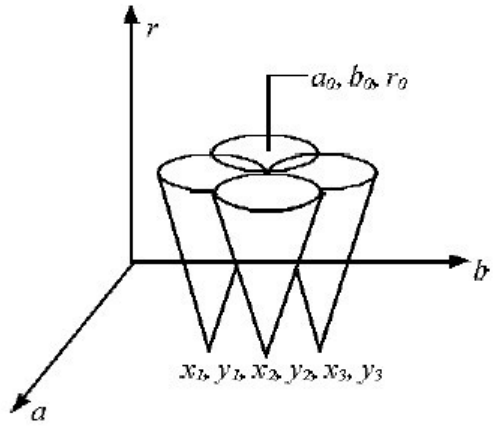


图 9：同一圆上四个点的霍夫变换结果

Figure 9: The result of the Hough transformation at four points on the same circle

3.4 闭运算

闭运算可以看作对图片先进行膨胀、再进行腐蚀。此运算可以消除图片中的微小间隙。

3.4.1 结构元

结构元可以为任意形状的矩阵，其各元素数值可以为0或1，通常将其中心定义为锚点。

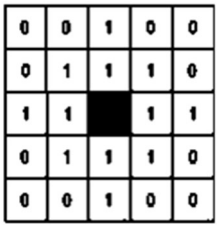


图 10：5×5十字形结构元

Figure 10: 5×5 Cross-shaped structural elements

通过在图像上滑动结构元，可实现对图像的膨胀与腐蚀操作。

3.4.2 膨胀

在图像上滑动结构元，使结构元的锚点遍历整幅图像，把结构元锚点位置对应图像像素点位置，目标像素点的灰度值设置为结构元值为1的区域对应图像区域像素的最大值，此操作即为图像的膨胀。使用 (x,y) 表示锚点的位置，(x',y') 表示结构元元素不为0部分对应的图像像素最大点相对锚点的偏移量。并用 s 表示结构元，f 表示图像，将图像的膨胀操作记为 $f\oplus s$，则输出图像 dst 和输入图像 src 像素点间的关系为：

$$f\oplus s=dst(x,y)=src(x+x',y+y')$$

其原理可简单概括为求局部最大值。结果会使得图像的高亮区域增长，直观视觉就是图像中较亮的物体尺寸会变大，较暗的物体尺寸会减小。，

3.4.3 腐蚀

腐蚀的算法流程与膨胀类似，与膨胀不同的是，腐蚀是将锚点处图像像素值改为局部最小值，若使用 (x',y') 表示像素最小值对应点坐标相对锚点的坐标偏差，并将图像的腐蚀操作记为 $f\odot s$，其余数学符号与膨胀操作保持一致，则:

$$f\odot s=dst(x,y)=src(x+x',y+y')$$

腐蚀的结果也与膨胀相对应，会使得图像的高亮区域减少，引起图像中的物体变得更为瘦小的直观视觉感受。

3.4.4 闭运算

图像的闭运算即为先对图像进行膨胀操作，再对图像进行腐蚀操作，可将该运算表示为如下形式:

$$f\bullet s=(f\oplus s)\odot s$$

图 11：闭运算效果展示

Figure 11: The closed operation effect is demonstrated

通过闭运算可以消除图像内部的小孔，在本测量系统中，上方的两个圆形小孔由于刻度尺的遮拦，在完成边缘检测后获得的边缘像素点无法连接在一起，因此使用闭运算“填补”被遮拦的像素点，使得小孔边缘的像素点可以彼此连接，构成封闭区域，方便后续检测与测量。

3.5 亚像素边缘提取

3.5.1 双线性插值

在获取图像中小圆的边缘部分的像素点坐标后，对这些像素点进行双线性插值，获取亚像素级别的边缘坐标，提高检测的准确率。

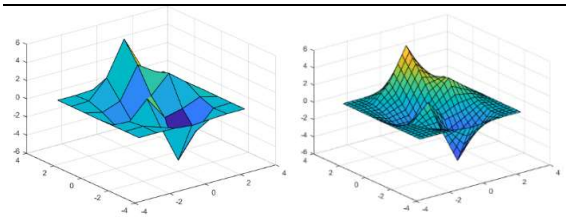


图 12：双线性插值效果演示

Figure 12: Demonstration of the bilinear interpolation effect

对于单线性插值，假定已知两点坐标 (x_1, y_1) (x_0, y_0)，应用三点共线的关系即可得：

$$\frac{y-y_0}{x-x_0}=\frac{y_1-y_0}{x_1-x_0}$$

利用此式便可得到位置 x 在直线上的值

$$y=\frac{x_1-x}{x_1-x_0}y_0+\frac{x-x_0}{x_1-x_0}y_1$$

双线性插值为两个变量的线性插值的扩展，在已知 $Q_{11}=(x_1,y_1),Q_{12}=(x_1,y_2),Q_{21}=(x_2,y_1),Q_{22}=(x_2,y_2)$ 的情况下，为获取 $P(x,y)$ 处的值，首先通过但线性插值获得 $R_1(x,y_1),R_2(x,y_2)$ 的值

$$f(R_1)=\frac{x_2-x}{x_2-x_1}f(Q_{11})+\frac{x-x_1}{x_2-x_1}f(Q_{21})$$

$$f(R_2)=\frac{x_2-x}{x_2-x_1}f(Q_{12})+\frac{x-x_1}{x_2-x_1}f(Q_{22})$$

再次使用线性插值，通过 $R_1(x,y_1),R_2(x,y_2)$ 的值获得：

$$f(P)=\frac{x_2-x}{x_2-x_1}f(R_1)+\frac{x-x_1}{x_2-x_1}f(R_2)$$

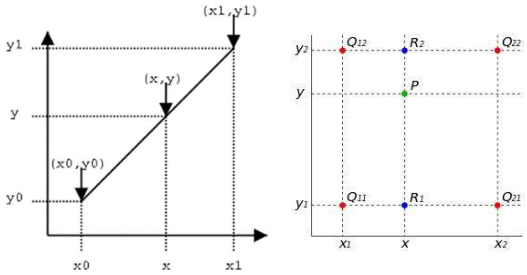


图 13：单线性插值与双线性插值

Figure 13: Unlinear interpolation versus bilinear interpolation

3.5.2 二次曲线拟合

设二次曲线的形式为 $y=Ax^2+Bx+C$,根据方形孔径采样定理，每个像素的输出值为：

$$y(n)=\int_{\frac{n-1}{2}}^{\frac{n+1}{2}}(Ax^2+Bx+C)dx$$

假定像素差分（梯度）值最大点的序号为0，相邻点序号为-1和1，对应的差分值用 y_{-1} 和 y_1 表示，根据上述公式可计算得：

$$y_{-1}=\frac{13}{12}A-B+C$$

$$y_0=\frac{1}{12}A+C$$

$$y_1=\frac{13}{12}A+C+C$$

抛物线的顶点即可求得

$$x=-\frac{B}{2A}=\frac{y_1-y_{-1}}{2(2y_0-y_1-y_{-1})}$$

利用上式，即可完成对圆孔边缘像素点的高精度拟合。

4. 测量实验与结果

下图为一张摄像机拍摄的工件图片，该工件上方两小孔被刻度尺遮挡，通过提取和分析此图片，测量图中七个圆孔的直径和圆心，各小孔圆心间的距离以及小孔圆心组成圆的直径和圆心，以此展示该测量系统的效果。

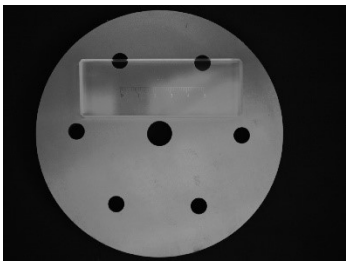


图 14：工件的原始图像

Figure 14: Original image of the workpiece

分别使用Roberts、Sobel、Perwitt和canny算子作用于图像，可获得图像边缘检测结果如下所示：

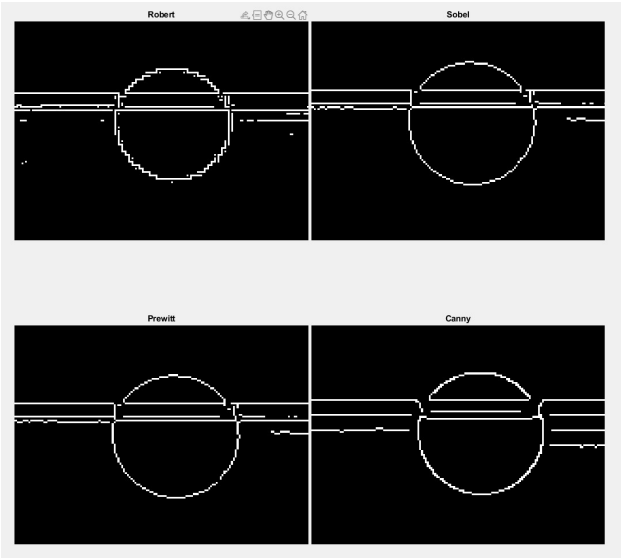


图 15：边缘检测局部效果对比，从左至右，从上至下分别为 Roberts、Sobel、Perwitt 和 canny 算子

Figure 15: Comparison of local effects of edge detection, from left to right, from top to bottom, roberts, Sobel, Perwitt, and cannon operators

由于Canny算子很好的考虑了噪声的影响，经过对比，canny算子的表现最佳，因此选用canny边缘检测后的图像作为原始图片的边缘检测结果。

对边缘图像进行霍夫圆检测，由于霍夫圆检测具有较强的抗噪声能力，因此即便上方两小孔被刻度值遮蔽，仍可以实现圆的检测，将检测出的图像用蓝色的线画出，结果如下：

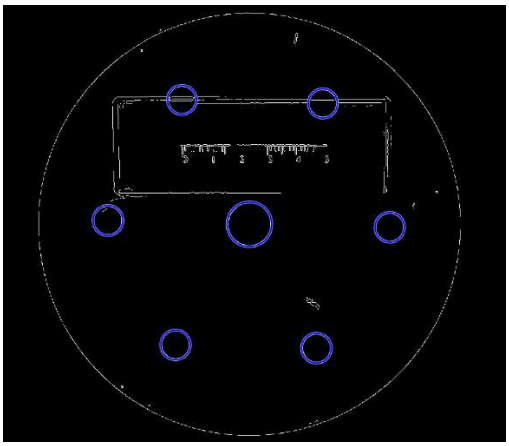


图 16：霍夫圆检测结果
Figure 16: Hough circle test results

霍夫圆检测虽可以检测出圆，但其精度不高，无法满足工业测量的需求，因此对边缘图像进行闭运算。填补由于刻度尺遮挡而无法识别的部分圆孔边缘。

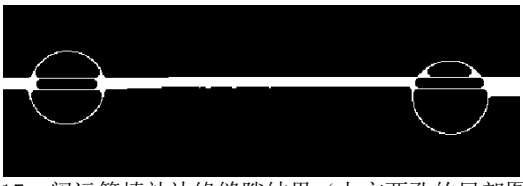


图 17：闭运算填补边缘缝隙结果（上方两孔的局部图）
Figure 17: Closed operation fills in the edge gap result (local view of the upper two holes)

结合之前霍夫圆检测获得的粗略圆信息，提取闭运算处理后图像的小孔部分，将其他部分的像素改为0：

图 18：利用霍夫圆圈定部分区域保留的闭运算结果（上方两孔的局部图）
Figure 18: Closed operation result of a partial area reserved using a Hove circle (local view of the upper two holes)

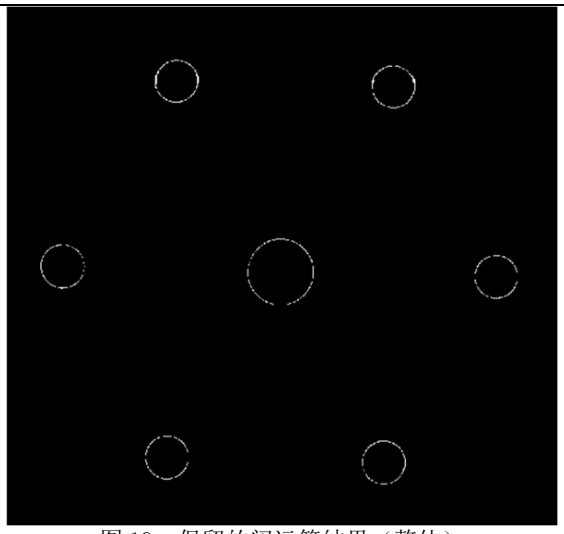


图 19：保留的闭运算结果（整体）
Figure 19: Closed result of the reserved operation

接下来进行封闭区域检测，提取出最外圈像素点的信息，将这些像素点的坐标记为各圆的边缘坐标，并将其进行双线性插值和二次曲线拟合，获得精确的拟合圆信息：

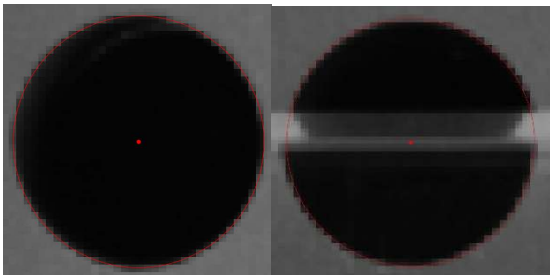


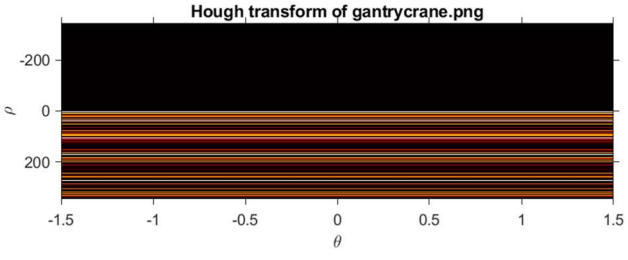
图 20：亚像素级圆拟合结果
Figure 20: Subpixel-level circle fitting results

完成圆的拟合后，对刻度尺部分进行边缘检测

图 21：刻度尺原始图像
Figure 21: Original image of a scale ruler

图 22：刻度尺 Perwitt 边缘检测结果
Figure 22: Scale Perwitt edge detection results

对刻度尺的边缘检测结果图片进行霍夫变换，将图片中各像素点转换至霍夫空间，并投票提取出长度与角度均合适的直线



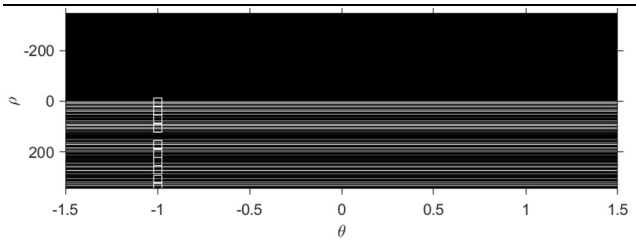


图 23：刻度尺边缘图像的霍夫空间（仅垂直直线）
Figure 23: Hove space for scale edge image (vertical straight lines only)

图 24：刻度尺霍夫直线检测结果。绿色线代表刻度尺上较长的刻度线，除第 4、5 条刻度线外，相邻两条刻度线间隔 0.5mm
Figure 24: Scale ruler Hove straight line test results. The green line represents the longer tick marks on the scale ruler, and in addition to the 4th and 5th tick marks, the two adjacent tick marks are spaced 0.5mm apart

通过刻度尺的像素长度（384个像素点）与真实长度（50mm）比例可标定得本张图片的像素当量为0.1488，标定精度为2.6μm，假定以左上方圆孔为起点，顺时针旋转。可计算得出：

（1）刹车盘边缘六小孔的直径分别为9.6541、9.7074、9.7990、9.8477、9.8287、9.9484mm，中心小孔直径为15.0265mm。

（2）刹车盘边缘六小孔的圆心像素坐标分别为(227.129,474.2099)、(235.509,808.9932)、(529.0107,968.2116)、(815.7673,794.2377)、(807.7462,459.4147)、(512.9186,298.1591)。

（3）边缘六相邻孔间的距离分别为49.8345、49.6885、49.9114、49.8388、50.0044、49.9458mm。

（4）边缘六孔与中心孔距离分别为49.9224、49.8090、49.5813、49.7803、49.9779、50.1524mm。

（5）外围六小孔所在圆周直径为997.411mm。

（6）外围六小孔所在圆心像素坐标(521.526,633.4808)

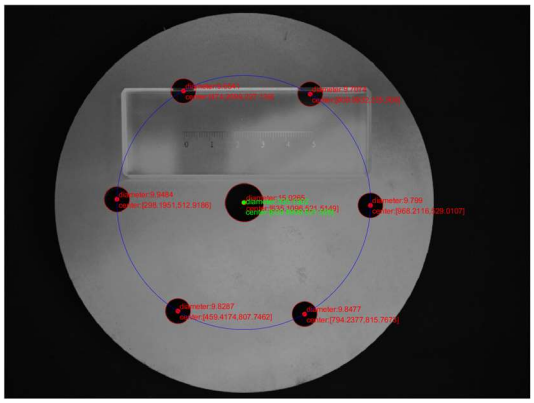


图 24：最终结果展示
Figure 24: Final results

5. 结论

本文根据现代化机器人视觉自动测量的高精度需求，对目前现有的图像处理算法进行了推导和比较，综合使用了边缘、霍夫变换、线性插值、膨胀腐蚀等多种视觉相关处理技术，提出了一种亚像素级别的自动测量方法，系统比例尺的标定精度为2.6μm。基本满足了自动影像测量的要求。

参考文献：

[1] 陈明晶，方源敏，陈杰．最小二乘法和迭代法圆曲线拟合[J]．测绘科学，2016,41(01):194-197+202.DOI:10.16251/j.cnki.1009-2307.2016.01.038.

[2]代勤,王延杰,韩广良.基于改进Hough变换和透视变换的透视图像矫正[J].液晶与显示,2012,27(04):552-556.

[3]王森,杨克俭.基于双线性插值的图像缩放算法的研究与实现[J].自动化技术与应用,2008(07):44-45+35.

[4]段瑞玲,李庆祥,李玉和.图像边缘检测方法研究综述[J].光学技术,2005(03):415-419.

[5]潘兵,续伯钦,陈丁,冯娟.数字图像相关中亚像素位移测量的曲面拟合法[J].计量学报,2005(02):128-134.

[6]马艳,张治辉.几种边缘检测算子的比较[J].工矿自动化,2004(01):54-56.

[7]曾接贤,张桂梅,储珺,鲁宇明.霍夫变换与最小二乘法相结合的直线拟合[J].南昌航空工业学院学报(自然科学版),2003(04):9-13+40.

[8]李红俊,韩冀皖.数字图像处理技术及其应用[J].计算机自动测量与控制,2002(09):620-622.DOI:10.16526/j.cnki.11-4762/tp.2002.09.022.

附录

项目地址：

https://github.com/zhaosiyuan1098/robotVision_homework/tree/test/2

源代码：

1.main.m

```
clear;
%% load images
imagePath='..\homework2\';
imageName='toMeasure.bmp';
fullfile(imagePath,imageName);
originImg=imread(fullfile(imagePath,imageName));
[rows, columns, channelNum] = size(originImg);
if channelNum > 1
    outputMessage="the number of channel is bigger than 3,convertion to gray will start";
    disp(outputMessage);
    grayImg = rgb2gray(originalImg);
else
    grayImg=originImg;
end

%% compare four methods of edge detection

BW1=edge(grayImg,'Roberts',0.03);
BW2=edge(grayImg,'Sobel',0.03);
BW3=edge(grayImg,'Prewitt',0.03);
BW4=edge(grayImg,'Canny',0.2);
edgeImg=BW4;

%% find circles roughlly for next step

[foreCenters, radii, metric] = imfindcircles(edgeImg,[20,80],'EdgeThreshold',0.02);

%Eliminate the tiny unconnected parts of the picture
se = strel('disk',5);
edgeCloseImg=imclose(edgeImg,se);

roiImg=zeros(size(originImg));
for curCol=1:size(roiImg,1)
    for curRow=1:size(roiImg,2)
        for i =1:size(foreCenters,1)
            dis=sqrt(abs((curCol-foreCenters(i,1)))^2+abs((curRow-foreCenters(i,2))^2));
            if    dis>radii(i)-3 && dis<radii(i)
                roiImg(curRow,curCol)=255;
```

```
                    roiImg(curRow,curCol)=roiImg(curRow,curCol)&&edgeCloseImg(curRow,curCol);
                end
            end
        end
end

%% detect and save pixels of circles

[labelImg,num]=bwlabel(roiImg);
circleEdges={};
minEdgesLength=200;
maxEdgesLength=400;

for i=1:num
    Edge=[];
    [r,c]=find(labelImg==i);
    if length(r)>minEdgesLength && length(r)<maxEdgesLength
        Edge(end+1:end+length(r),1:2)=[r,c];
        circleEdges{end+1}=Edge;
    end
end
%% resolution pixels and calculate accurate circles

subCenters=zeros(size(foreCenters));
subRadiums=zeros(size(foreCenters,1),1);
for i=1:length(circleEdges)
    subCircle=Subpixel_Extraction(circleEdges{i},originImg);
    abc=[subCircle(:,2),subCircle(:,1),ones(size(subCircle,1),1)]\-(subCircle(:,2).^2+subCircle(:,1).^2);
    a=abc(1);b=abc(2);c=abc(3);
    xc = -a/2;
    yc = -b/2;
    subCenters(i,:)=[yc,xc];
    subRadiums(i)=sqrt((xc^2 + yc^2) - c);
end
SubCenters_6=subCenters;
SubCenters_6(4,:)=[];
abc=[SubCenters_6(:,2),SubCenters_6(:,1),ones(size(SubCenters_6,1),1)]\-
(SubCenters_6(:,2).^2+SubCenters_6(:,1).^2);
a=abc(1);b=abc(2);c=abc(3);
xc = -a/2;
yc = -b/2;
center_6=[yc,xc];
radium_6=sqrt((xc^2 + yc^2) - c);
```

```
%% calculate distances in image coordinates of each circle centers

centersSize=length(subCenters);
distanceMatrix=zeros(centersSize);
for i =1:centersSize
    for j =i:centersSize
        distanceMatrix(i,j)=sqrt(abs(subCenters(i,1)-subCenters(j,1))^2+abs(subCenters(i,2)-subCenters(j,2))^2);
        distanceMatrix(j,i)=distanceMatrix(i,j);
    end
end

% distanceMatrix

%% identify ruler and calculate rate between pixels and turth

startCutPoint=[472,339];
endCutPoint=[817,355];
rulerOriginImg =imcrop(originImg,[startCutPoint(1),startCutPoint(2),abs(startCutPoint(1)-
endCutPoint(1)),abs(startCutPoint(2)-endCutPoint(2))]);
rulerEdgeImg=edge(rulerOriginImg,'Prewitt',0.01);
[H,T,R] = hough(rulerEdgeImg,'Theta',-1:1);
P=houghpeaks(H,10,'Threshold',0.1*max(H(:)));
lines = houghlines(rulerEdgeImg,T,R,P,'FillGap',1,'MinLength',13);
startX=lines(1).point1(1);
endX=startX;
for k = 1:length(lines)
    if lines(k).point1(1)>endX
        endX=lines(k).point1(1);
    end
    if lines(k).point1(1)<endX
        startX=lines(k).point1(1);
    end
end
convRate=50/abs(endX-startX);
%% calculate real size
trueDistanceMatrix=distanceMatrix.*convRate;
trueSubDiameters=subRadiums.*convRate*2;
trueDiameter_6=radium_6*convRate*2;

%% draw
figure;
subplot(2,2,1);
imshow(BW1);
title('Robert')
```

```
subplot(2,2,2);
imshow(BW2);
title('Sobel')
subplot(2,2,3);
imshow(BW3);
title('Prewitt');
subplot(2,2,4);
imshow(BW4);
title('Canny')

figure;
imshow(edgeImg);
viscircles(foreCenters, radii,'EdgeColor','b');

figure;
imshow(edgeCloseImg);

figure;
imshow(roiImg);

figure;
imshow(originImg);
hold on;
circleTheta=-pi:0.01:pi;
for i =1:length(subCenters)
    xfit=subRadiums(i)*cos(circleTheta)+subCenters(i,2);
    yfit=subRadiums(i)*sin(circleTheta)+subCenters(i,1);
    plot(xfit,yfit,'r');
    plot(subCenters(i,2),subCenters(i,1),'ro-','MarkerFaceColor','r');
    str={['diameter:',num2str(trueSubDiameters(i))],...
        ['center:[',num2str(subCenters(i,2)),',',num2str(subCenters(i,1)),']']...
        num2str(i)};
    text(subCenters(i,2)+5,subCenters(i,1),str,'color','red');
end
xfit=radium_6*cos(circleTheta)+center_6(2);
yfit=radium_6*sin(circleTheta)+center_6(1);
plot(xfit,yfit,'b-');
plot(center_6(2),center_6(1),'ro-','MarkerFaceColor','g');
str={['diameter:',num2str(radium_6)],...
    ['center:[',num2str(center_6(2)),',',num2str(center_6(1)),']']};
text(center_6(2)+5,center_6(1)+10,str,'color','g');

figure;
subplot(2,1,1);
```

```
imshow(imadjust(rescale(H)),'XData',T,'YData',R,...
    'InitialMagnification','fit');
title('Hough transform of gantrycrane.png');
xlabel('\theta'), ylabel('\rho');
axis on, axis normal, hold on;
colormap(gca,hot);
subplot(2,1,2);
imshow(H,[],'XData',T,'YData',R,'InitialMagnification','fit');
xlabel('\theta'), ylabel('\rho');
axis on, axis normal, hold on;
plot(T(P(:,2)),R(P(:,1)),'s','color','white');

figure;
subplot(3,1,1);
imshow(rulerOriginImg);
subplot(3,1,2);
imshow(rulerEdgeImg)
subplot(3,1,3);
imshow(rulerEdgeImg), hold on
for k = 1:length(lines)
    xy = [lines(k).point1; lines(k).point2];
    plot(xy(:,1),xy(:,2),'LineWidth',2,'Color','green');

    % Plot beginnings and ends of lines
    plot(xy(1,1),xy(1,2),'x','LineWidth',2,'Color','yellow');
    plot(xy(2,1),xy(2,2),'x','LineWidth',2,'Color','red');
end
```

2.Subpixel_Extraction.m

```
function sub_edge=Subpixel_Extraction(edge,originImg)

if size(originImg,3)==3
    originImg=rgb2gray(originImg);
end

originImg=double(originImg);

mask_r=2;
mask=fspecial('gaussian',2*mask_r+1);
gaussianImg=imfilter(originImg,mask,'replicate');


w=fspecial('sobel');
```

```
gradient_h_im=imfilter(gaussianImg,w,'replicate');
w=w';
gradient_w_im=imfilter(gaussianImg,w,'replicate');
gradient_amp_im=sqrt(gradient_h_im.^2+gradient_w_im.^2);

R=[-2 -1 0 1 2 ];
Len=length(edge);
Neigh_H_Matrix=zeros(Len,5);
Neigh_W_Matrix=zeros(Len,5);
Gradient_Thea=zeros(Len,1);
for i=1:Len
    h=edge(i,1);
    w=edge(i,2);
    dh=gradient_h_im(h,w);
    dw=gradient_w_im(h,w);
    thea=atan2(dh,dw);
    Neigh_H_Matrix(i,:)=R*sin(thea)+h;
    Neigh_W_Matrix(i,:)=R*cos(thea)+w;
    Gradient_Thea(i)=thea;

end

Neigh_Gray_Diff=interp2(gradient_amp_im,Neigh_W_Matrix,Neigh_H_Matrix);

Ln_Neigh_Gray_Diff=log(Neigh_Gray_Diff);

V=[4 -2 1;
   1 -1 1;
   0  0 1;
   1  1 1;
   4  2 1];
E=(V'*V)\V';
sub_pixel_edge=zeros(Len,2);
for i=1:Len
    U=Ln_Neigh_Gray_Diff(i,:)';
    A=E(1,:)*U;
    B=E(2,:)*U;
    x=-B/(2*A);
    sub_pixel_edge(i,1)=edge(i,1)+x*sin(Gradient_Thea(i));
    sub_pixel_edge(i,2)=edge(i,2)+x*cos(Gradient_Thea(i));
end

sub_edge=sub_pixel_edge;
```