第十一届数学中国数学建模网络挑战赛

地址：数学中国数学建模网络挑战赛组委会　　网址：www.tzmcm.cn
电话：0471-4969085　　邮编：010021　　Email：service@tzmcm.cn

第十一届“认证杯”数学中国

数学建模网络挑战赛
承　诺　书

我们仔细阅读了第十届“认证杯”数学中国数学建模网络挑战赛的竞赛规则。

我们完全明白，在竞赛开始后参赛队员不能以任何方式（包括电话、电子邮件、网上咨询等）与队外的任何人（包括指导教师）研究、讨论与赛题有关的问题。

我们知道，抄袭别人的成果是违反竞赛规则的, 如果引用别人的成果或其他公开的资料（包括网上查到的资料），必须按照规定的参考文献的表述方式在正文引用处和参考文献中明确列出。

我们郑重承诺，严格遵守竞赛规则，以保证竞赛的公正、公平性。如有违反竞赛规则的行为，我们接受相应处理结果。

我们允许数学中国网站(www.madio.net)公布论文，以供网友之间学习交流，数学中国网站以非商业目的的论文交流不需要提前取得我们的同意。

我们的参赛队号为：4770

参赛队员：

队员 1：

队员 2：

队员 3：

参赛队教练员：

参赛队伍组别：研究生组

第十一届数学中国数学建模网络挑战赛

地址：数学中国数学建模网络挑战赛组委会　　网址：www.tzmcm.cn
电话：0471-4969085　　邮编：010021　　Email：service@tzmcm.cn

第十一届“认证杯”数学中国

数学建模网络挑战赛
编 号 专 用 页

参赛队伍的参赛队号：（请各个参赛队提前填写好）：

4770

竞赛统一编号（由竞赛组委会送至评委团前编号）：

竞赛评阅编号（由竞赛评委团评阅前进行编号）：

第十一届数学中国数学建模网络挑战赛

地址：数学中国数学建模网络挑战赛组委会　　网址：www.tzmcm.cn
电话：0471-4969085　　邮编：010021　　Email：service@tzmcm.cn

2017 年第十一届“认证杯”数学中国数学建模网络挑战赛第一阶段论文

题　　目　基于轮廓特征的机械零件位置识别研究

关 键 词　轮廓特征、Hough 检测、聚类分析、位置识别

摘　　要

图像位置识别（机器视觉）在工业生产中应用广泛。本文首先通过霍夫变换提取给定零件轮廓中发直线和圆特征，进而利用峰值检测和 K-means 聚类分析方法准确识别有效的直线和圆特征；依次建立零件轮廓的绝对坐标系和零件标准位置模板坐标系；分别采用基于零件全特征（随机抽样）和基于零件主要特征的方式提取零件轮廓的有效像素点，并通过模拟退火算法求解了不同迭代次数下采用这两种方式识别零件位置的速度（时间）和精度。采用目标轮廓分割法提取零件的二维轮廓像素矩阵，之后通过模拟退火算法求解多个零件轮廓的位置和识别速度（时间）。

问题 1：对于被测零件几何特征未知的情况，以 DATA1 中的初始图像像素起始点为原点，沿行增加与列增加的方向建立绝对直角坐标系，得到了零件轮廓中矩形的四个顶点坐标，即（182, 172）、（266, 317）、（334, 84）和（414, 230），三个圆的圆心坐标为（247, 288）、（188, 253）、（183, 329），且半径均为 26，最低识别精度为 99.1%。对于几何特征已知情况，在初始图像中某一确定位置处建立零件的标准位置模板，分别随机提取标准位置模板的有效像素点和被测零件的有效像素点，以被测零件轮廓图像的有效像素抽样点与标准位置模板的有效像素抽样点的最短距离和最小为目标，以平移量和旋转角度为自变量，通过模拟退火算法(迭代 5000 次)得到被测零件相对中心位置为(296.0209, 200.6858)，逆时针旋转 28.0433°，识别时间为 2.306s，最低识别精度为 95%。

问题 2：将多个零件的识别分成零件分割和零件定位两步。在 DATA2 的初始图像中建立直角坐标系，采用问题 1 的方法获得被测零件轮廓组的 6 个圆心坐标，并对其进行聚合分类；以几何中心为圆心，以矩形轮廓对角线为直径做圆，划分成 2 个独立领域；并提取独立领域内的二维图像信息，采用问题 1 方法，求解得到零件 a 和零件 b 中心相对于标准位置模板中心的位置分别为（239.9272, 400.0000）和（305.9829, 199.5862），对应的逆时针旋转角度为 60.64°和 29.31°。

模型改进：为提高识别速度，建立基于零件几何特征（3 个圆心）的快速识别模型，即将目标函数简化为被测零件的圆心与标准位置模板圆心的最短距离和最小，通过模拟退火算法（迭代为 5000 次）进行求解。零件 a 的识别时间为 0.312s，最低识别精度为 98.8%，零件 b 的识别时间为 0.292s，最低识别精度为 97.6%。

参赛队号： 4770

所选题目： C　题

参赛密码 _____
（由组委会填写）

第十一届数学中国数学建模网络挑战赛

地址：数学中国数学建模网络挑战赛组委会　　网址：www.tzmcm.cn
电话：0471-4969085　　邮编：010021　　Email：service@tzmcm.cn

Abstract

Image location recognition has been widely used in industry automation production line. In this paper, several methods were used to solve the question of parts' image location recognition. First, the Hough transform was used to extract the linear and circular characteristics of a given part, and peak detection and k-means clustering analysis method were applied to accurately identify the effective line and circle characteristics; second, we established the absolute coordinate system of the part as well as the standard position template coordinate system of the same part relatively; third, the effective pixel points of the part were extracted by random sampling and main feature sampling; then through simulated annealing algorithm, the speed and precision of the position of the part were identified based on different maximum iterations; finally, according to the characteristics of the parts contour information, the target contour segmentation method was used to extract parts' contour pixel matrix, and also several parts contour position and recognition speed were solved by simulated annealing algorithm.

Question1: In case of unknown geometrical features of the part being tested, we established the absolute coordinate system of the part image, and used Hough transform, peak detection and k-means clustering analysis to extract the linear and circular characteristics of a given part. Results are as follows. The four vertex coordinates of the rectangular shape of the part are （182,172）,（266,317）,（334,84）,（414,230）relatively. The center coordinates of the three circles are （247,288）,（188,253）,（183,329）relatively. The radiuses of the three circles are all 26. The minimum recognition accuracy is 99.1%. In case of unknown geometrical features of the parts being tested, we established the standard position template, used random sampling to extract effective pixel points of the part, and also used simulated annealing algorithm to obtain the best answer about the figure for translation and rotation angles. Finally we obtained that the relative center position of the measured part is (296.0209, 296.0209), 28.0433° counterclockwise, recognition time is 2.306 s, lowest recognition accuracy is 95%.

Question2: The recognition of several parts was divided into two steps, including part segmentation and part positioning. Through k-means clustering analysis, six coordinate of circle center and two center positions of two parts were obtained to separate parts districts. The relative center positions of part a and part b obtained through the method of Question 1 are (239.9272, 239.9272), (305.9829, 305.9829), and the corresponding counterclockwise rotation angles are 60.64 ° and 29.31 °.

Model improvement: to improve the recognition speed, we put forward a rapid identification model based on three geometry feature points (centers of three circles) of the part, through the simulated annealing algorithm to solve this problem (iteration for 5000).The identification time of part a is 0.312s, the minimum recognition accuracy is 98.8%, and the identification time of part b is 0.292S, the minimum recognition accuracy is 97.6%.

1. 问题背景和重述

我国实现“中国制造 2025”，完成从制造大国向制造强国的转变，智能制造是未来的主攻方向。在智能制造过程中，机器视觉主要用计算机来模拟人或动物的视觉功能，即对客观事物的图像进行信息提取和处理，实现检测、测量和控制等功能。

随着智能化发展的愈演愈烈，市场对于机器视觉的需求也将逐渐增多。机器视觉可说是工业自动化系统的灵魂之窗，从物件/条码辨识、产品检测、外观尺寸测量、智能装配到机械手臂/传动设备定位，都是机器视觉技术发挥能量的舞台。

目前消费者对工业生产的效率和产品的质量要求越来越高，应用机器视觉可大幅提高工业生产的自动化率，降低工人劳动强度和误操作率，对于提高产品的生产效率和加工质量具有重要的作用。本论文根据图像处理的基本方法，利用计算机来实现工业制造生产线的加工、装配、包装等工序中机械加工零件的智能识别，并由机械手将零件自动搬运到特定位置。本题给出了某零件轮廓图（见图 1），同时给出了该零件搬运前后的位置示意图（如图 2 所示）。

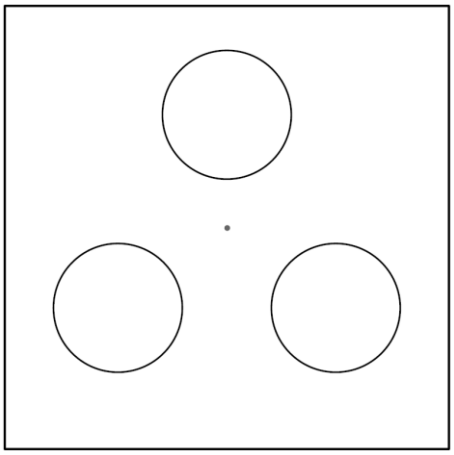

图 1　零件轮廓示意图

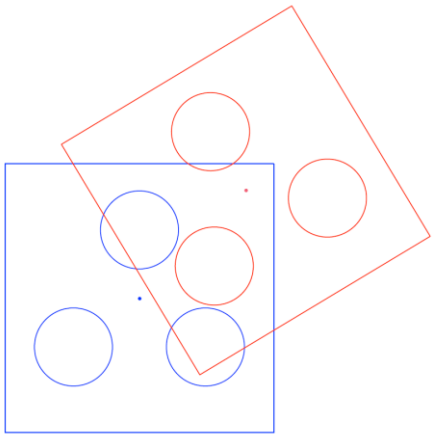

图 2　零件搬运前后位置示意图

建立数学模型，研究并分析以下问题：

问题 1：根据附件 DATA1 中给出的零件轮廓数据，请建立数学模型，识别计算出给定零件的位置坐标，并分析评价求解零件位置的算法是否快速高效。

问题 2：问题 1 讨论的是单个零件放置于平面操作台上的情况。有时我们需要处理多个零件显示在同一图像中的情况，请根据附件 DATA2 中的数据，建立数学模型，识别不同零件的位置。

2. 问题的分析

2.1 问题 1 的分析

本问题要求根据附件 DATA1 中给出的零件轮廓数据，识别并计算出给定零件的位置坐标，DATA1 的零件轮廓如图 3 所示。零件的识别分类主要包括特征提取和识别。颜色特征、纹理特征、形状特征、和几何特征是图像的四大特征[1]。从图 1 可以看出本问题给出的图是机械零件的二维轮廓图，可知该零件具有不可形变的特点，并且为白色（颜色特征不明显），为此本文考虑采用几何、形状特征对其进行识别。常见的几何特征为轮廓个数、面积、周长，而形状特征一般有点、直线、圆、椭圆等。根据本题给出

的零件轮廓图，可以很容易地看出其主要包括 3 个圆、4 条直线且两两垂直（或矩形）。为此问题 1 中的轮廓识别就变成了圆的识别和直线（或矩形）的识别问题。霍夫变换[2]（Hough Transform）是由图像空间的边缘点去计算参数空间中参考点的可能轨迹，从而在计算机的累加器中给计算出的参考点计数，最后选出峰值。因为它将直角坐标系中的线变为极坐标中的点，故一般常将霍夫变换称为线、点变换。根据这个原理，可以用霍夫变换提取直线、圆等常规的几何特征。为此问题 1 可以利用霍夫变换识别已知零件轮廓的圆特征和直线特征（矩形）。

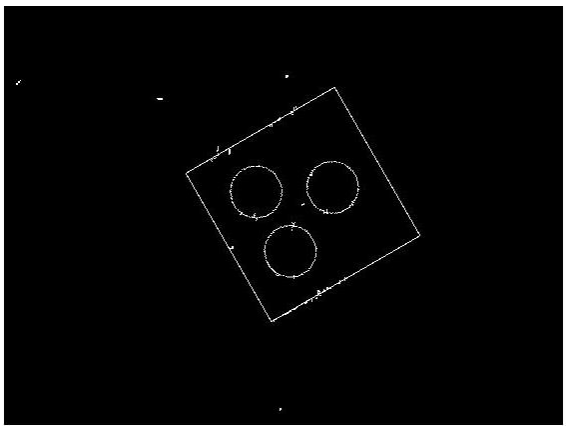

图 3　DATA1 的零件轮廓

判断某一图像的位置通常需要定义参考坐标系或者参考物，之后根据参考坐标系或参考物来判断图形的相对位置。而零件的位置信息又可分为零件的绝对位置（与参考坐标系或者参考物相比）及零件本身几何特征之间的相对位置（如本题中的圆、直线之间的相对位置）。在二维坐标系中（假如已知参考坐标系），判断某一个点通常只需知道其横向坐标（x 向）和纵向坐标（y 向），而定位线段不仅需要知道该线段特征（如中心、起点或终点）的坐标，还需知道其相对于横轴（x 向）或纵轴（y 向）的角度，具体可见图 4 所示。

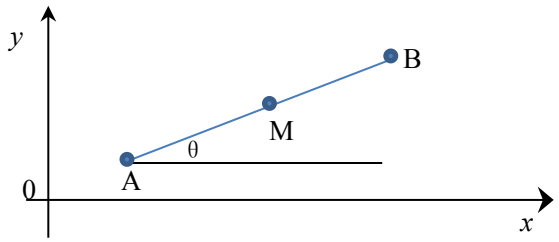

图 4　几何特征在二维坐标系中的定位

为此，要想准确定位问题 1 给出的零件轮廓图，即确定其位置坐标，可以通过相应的几何特征进行平移及旋转（即通过平移和旋转后使其恢复到参考物状态）。可以推知，问题 1 可具体化为零件轮廓（可取其几何中心点）的坐标平移与轮廓自身旋转的问题。通过上述分析可知，位置的定位可以通过图像的旋转和平移来实现[3]。

就问题 1 而言，可在初始图像（DATA1 中的零件轮廓图）中建立直角坐标系，通过霍夫变换来检测轮廓中的几何特征（直线与圆），之后结合 K-means 聚类分析确定零件轮廓中准确的三个圆和四条直线（矩形）的信息，即三个圆的半径及方程、直线长度，为此，零件的位置信息也可相应地计算得到[4-7]。

考虑到问题 1 中零件轮廓的几何特征和形状特征信息，如矩形轮廓边长、圆的半径及圆心相对位置均可通过 Matlab 简单计算得到（假设长度单位为 1），因而可在某一确

定的位置处建立零件的标准模板，见图 5。从 DATA1 中的零件轮廓和零件的标准模板轮廓中随机取部分像素点坐标，依次计算 DATA1 提取的每一像素点坐标到标准模板轮廓所提取的像素点坐标的最小距离，即以求解距离和最小为目标，以给定零件轮廓的平移至标准模板位置的移动量和旋转角度为设计变量，建立数学模型，并通过优化算法（可选用模拟退火方法）求解该模型，即可求得的移动量与旋转角度，从而反求零件的位置与角度（相对于标准模板）。该方法可称为模板匹配法。

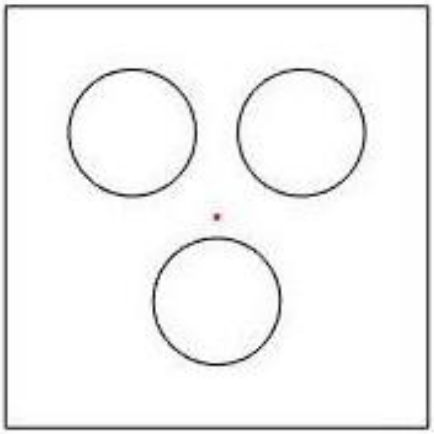

图 5　零件轮廓的标准模板及其位置

2.2 问题 2 的分析

本问题要求根据附件 DATA2 中给出的多个零件轮廓，识别一幅图像中多个零件的不同位置信息。利用 Matlab 可将 DATA2 中的数据进行图像再现，如图 6 所示。从图 6 可以看出，图中包含 2 个相同的零件轮廓（每个轮廓与 DATA1 中的轮廓相同，仅姿态不同），即总共包含 6 个圆和 8 条直线。如采用问题 1 的方法直接对图 6 中的 2 个零件轮廓进行位置求解则会发生轮廓特征重复的问题，即无法获取所需轮廓的难题。为此，解决问题 2 的关键在于分割 2 个零件轮廓，即使 2 个零件轮廓独立显示在图像中。分割完 2 个轮廓后，分别采取问题 1 的办法即可实现 2 个零件轮廓的位置定位。

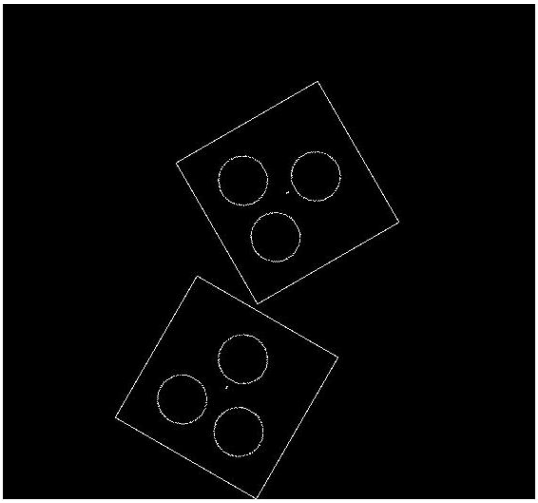

图 6　多个零件轮廓

从图像中分离并提取目标（也称轮廓分割）可以看作是基元轮廓检测的一种推广。通常有基于目标轮廓和基于目标区域两种方法用于轮廓分割。目标轮廓分割法：考虑目标与图像中其他部分的界限，如果能确定目标轮廓界限，就可将目标与图像的其他部分

割开。在这种方法中，除了检测出轮廓点将其连接起来，还可以边检测轮廓点边的连接。而基于目标区域法是考虑所有属于目标区域的像素，如果能确定出每个属于目标的像素就可以获得完整的目标。通常是假设构成目标区域的像素的灰度值大于某个特定的阈值，确定这个阈值把目标区分开。考虑到问题 2 中的 2 个零件轮廓为简单的圆和直线轮廓，为了提高识别速度，本文采用目标轮廓分割法[8]将 2 个零件轮廓分割开。

为此问题 2 中轮廓分割的具体方案为：在初始图像中建立直角坐标系，通过霍夫变换进行几何特征圆的检测，结合 K-means 聚类分析方法，获取 DATA2 中 6 个圆心的坐标，随后对所得的 6 圆心坐标再次进行 K-means 聚类分析，从而可得到每个零件轮廓的近似中心点坐标，以近似中心点坐标为中心，零件模板矩形的对角线一半为半径，通过 Matlab 画一轮廓，即可将该区域数据的零件信息分割出来。

针对于分割出来的零件轮廓的位置坐标及旋转角度的确定，依然可采用问题 1 的方法对其进行以求解。即求解每个给定零件与标准位置模板部分位置坐标的和最小为目标函数，建立数学模型的方法，根据平移的移动量与旋转的角度，对其反求即可得到图像中 2 个零件轮廓的位置坐标和角度。综上，零件轮廓位置定位的技术路线如图 7 所示。

图 7　零件轮廓位置定位技术路线

考虑到本问题 1 和 2 中的噪点对零件轮廓位置的定位没有影响，因此，本文在进行零件轮廓定位之前并未进行图像预处理。

3. 模型假设及符号说明

3.1 模型假设

（1）零件轮廓为二维轮廓。

（2）多个零件在一副图像中不会发生重叠现象。

（3）图像位置的偏差在机械零件加工中处于合理误差范围（即机器手或者夹具是可以准确定位和夹紧，比如采用锥形定位或半销定位）。

（4）图像中的噪声（非圆及非矩形上的点不会影响位置的定位，因此不必进行滤波或图像平滑处理）。

3.2 符号说明

符号	意义
x_s， y_s	被测零件二值化图像中第 s 个 1 值的行号 x_s 和列号 y_s
θ， d	Hough 变换直线特征表示参数
Z	水平状态下被测零件的二值化图像中有效像素点的集合
a_l，b_l，r_l	Hough 变换圆检测计算所得圆心横坐标 a_l 、纵坐标 b_l 和半径 r_l
$[A,B,R]$	K-means 聚类分析所确定的圆坐标参数集合
KP	元素为 0 的空白模板矩阵
ZP， n'， m'	ZP 为零件标准位置模板的矩阵，其行数为 n'，其列数为 m'
ZU	标准位置模板的有效像素点集合
TS	标准位置模板的有效像素抽样点集合
(e_1, f_1)， (e_2, f_2)	标准位置矩形左上角点坐标(e_1, f_1)和右下角点坐标(e_2, f_2)
TT	被测零件图像有效像素抽样点集合
(a_q, b_q)	被测零件的平移起始点坐标
$TT\_move(v)$	第 v 次平移后的被测零件图像有效像素抽样点
sum_S_0	第 0 次平移后的所有被测零件图像有效像素抽样点与标准位置模板有效像素抽样点的最短距离和
$move\_x, move\_y, move\_angle$	被测零件图像有效像素抽样点当前横向平移参数 $move\_x$ 、纵向平移参数 $move\_y$ 和旋转角度 $move\_angle$

4.模型的建立与求解

4.1 问题 1：模型的建立与求解

将问题 1 中给定的零件轮廓数据（见附件 DATA1）转化为二值化图像，如图 8 所示。本文将零件轮廓称为被测零件。

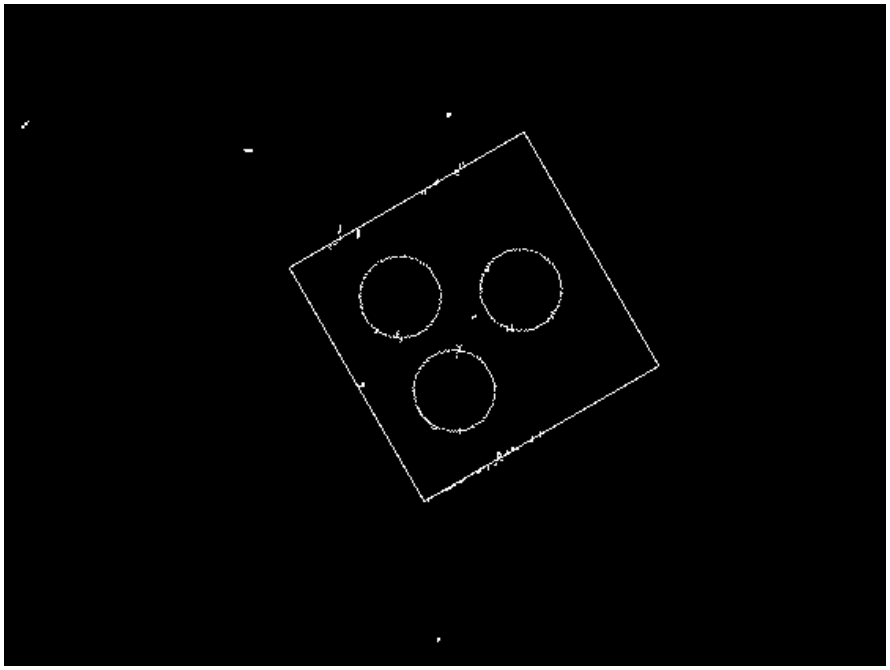

图 8　DATA1 的二值化图像（零件轮廓）

由前文 2.1 问题 1 的分析可知，为计算图 8 中零件轮廓的位置，本文定义了两种位置，一种是绝对位置（即被测零件在通过 Hough 变换后在所建立坐标系中的位置），另外一种是相对于某一零件标准模板的位置。对于被测零件的绝对位置可直接通过 Hough 变换得到，即对二值化矩阵进行 Hough 变换之后，再进行峰值检验和聚类分析（聚类分析的目的是准确获得被测零件轮廓中 3 个圆的半径及其圆心位置）。在 Hough 变换同时，也得到了被测零件几何特征（圆、矩形）的相对位置。

4.1.1 基于 Hough 变换与聚类分析的位置识别模型

以被测零件轮廓的二值化图像的左上角作为绝对坐标原点，将被测零件位置求解的问题转换为直线特征位置求解和圆特征位置求解问题。主要分为如下步骤：

第一步：被测零件轮廓数据的预处理

统计被测零件的二值化矩阵中 1 值的行号 x_s、列号 y_s 以及其数量 n，建立有效像素点的集合 U，并表示为 $U=\{X,Y\}$，$X=x_1,x_2,...,x_s,...,x_n$，$Y=y_1,y_2,...,y_s,...,y_n$。

第二步：识别被测零件轮廓的几何特征

利用 Hough 变换提取直线特征的基本原理在于利用点与线的对偶性，将图像空间的线条变为参数空间的聚集点，从而检测给定图像是否存在给定性质的曲线。

（1）建立 Hough 参数空间 $\theta-d$，其中 $\theta\in[0,2\pi],d\in[d_1,d_2]$，$d_1,d_2$ 范围根据经验设定。按步长 $\Delta\theta$ 和 Δd 分别将参数空间 $\theta-d$ 划分为 $P\times Q$ 个单元，记为 $V_{\theta\times d}$。对每个单元设置累加器 $A(i,j)$，并将其初值设定为 0，从而得到累加器矩阵 $A_{p\times q}$。

（2）将 $\theta\in[0,2\pi]$ 与有效像素集合点 $U=\{X,Y\}$ 作为输入量，将其带入直线极坐标公式（1），从而可求解得到 p 组 (θ_i,d_i) 值。

$$\begin{aligned} &x_s\cos\theta_i+y_s\sin\theta_i=d_i \\ &\theta_i=\mathrm{i}\times\Delta\theta,\ i=1,2,...,p \end{aligned} \qquad (1)$$

（3）将求解得到的 p 组 (θ_i,d_i) 值带入到 Hough 参数空间 $\theta-d$，便可找到 θ_i、d_i 等值的 $V_{\theta\times d}$ 单元所对应的累加器 $A(i,j)$，此时该累加器 $A(i,j)$ 加 1，即

$$A(i,j)=A(i,j)+1 \qquad (2)$$

（4）当有效像素集合点U中的所有的有效像素点都完成上述（3）、（4）步骤后，对累加器矩阵$A_{p\times q}$中的各个累加器值进行峰值检验，找到累加器值最大的四个单元所对应的（θ,d），并依次按累加器值大小从高到低排列，分别记为$(\theta_1,d_1),(\theta_2,d_2),(\theta_3,d_3),(\theta_4,d_4)$，即为被测零件轮廓的矩形边框的四组直线的极坐标参数。

（5）求解四组直线所围成的矩形边框的边长，其长宽分别记为c和d。Hough 直线识别的效果图下图所示：

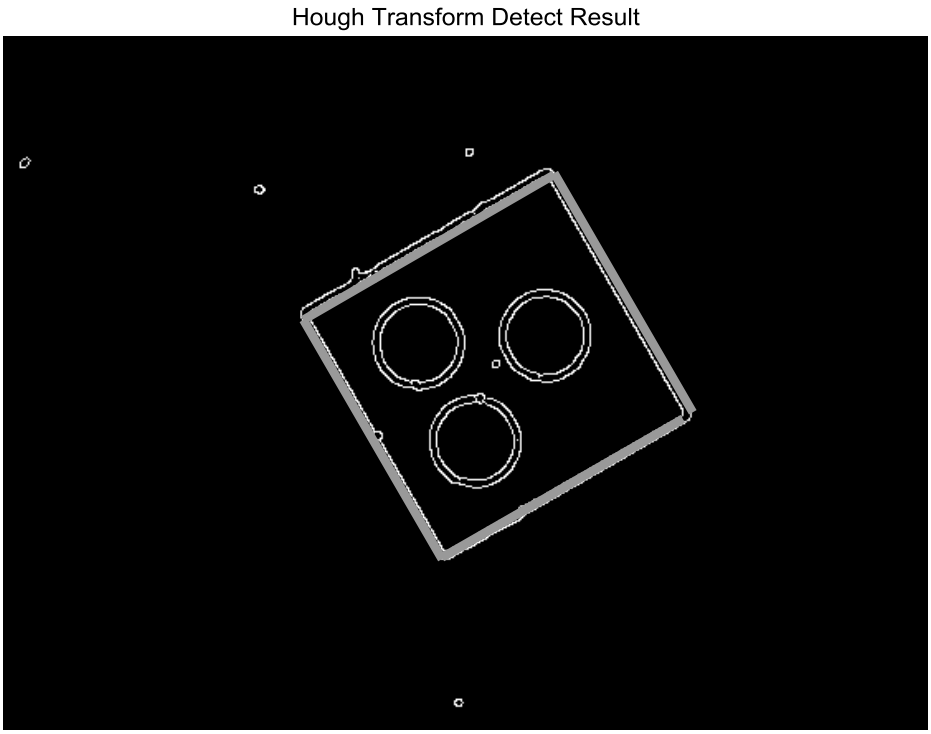

图 9　Hough 变换的直线识别效果

利用 Hough 变换识别出的四条直线的顶点坐标及其旋转角度如表 1 所示：

表 1　四条直线的顶点坐标及其旋转角度

直线	端点 1	端点 2	相对绝对坐标偏转角度（极坐标角度）	相对绝对坐标原点距离（极坐标距离）
直线一	[182, 172]	[266, 317]	30	71.5
直线二	[334, 83]	[418, 228]	30	247.5
直线三	[182, 172]	[334, 84]	-60	238.5
直线四	[266, 316]	[412, 232]	-60	405.5

通过对四条直线各个顶点的坐标（取平均值）进行拟合，近似得到了被测零件的矩形各个顶点的绝对坐标分别为（182, 172）、（266, 317）、（334, 84）和（414, 230）。由此计算出矩形的长和宽分别为$c=167.573$和$d=175.636$

第三步：旋转被测零件图像

为快速定位被测零件几何特征圆形轮廓相对于矩形边框的相对位置，选取累加器最大值对应的直线段的端点作为旋转中心 O，整体旋转被测零件（包括有效像素点和无效像素点）$-\theta_1=-30°$度，旋转公式如式（3）所示。使该直线段处于水平位置，从而得到

了水平状态下的被测零件二值化图像素矩阵 P 。

$$P=\begin{bmatrix}\cos\theta_1 & -\sin\theta_1\\ \sin\theta_1 & \cos\theta_1\end{bmatrix}Q \tag{3}$$

第四步：识别旋转后被测零件的几何特征——圆

通过类似于前文介绍的 Hough 变换方法，建立 Hough 三维参数空间 $a-b-r$ ，将零件轮廓中圆的边缘点映射到 Hough 三维参数空间 $a-b-r$ 中。

（1）统计水平状态下的被测零件二维图像矩阵 P 中 1 值的行号 x_t 、列号 y_t 及其数量 n，建立有效像素点集合 Z，并表示为 $Z=\{X,Y\}$，$X=x_1,x_2,...,x_t,...,x_n$，$Y=y_1,y_2,...,y_t,...,y_n$ 。

（2）建立 Hough 三维参数空间 $a-b-r$ ，$r\in[0,\min(c,d)/2]$，按步长 $\Delta a,\Delta b,\Delta r$ 将三维参数空间 $a-b-r$ 分别划分为 $e\times f\times g$ 个单元，记为 $W_{a\times b\times r}$ 。对每个单元设置累加器 $B(i,j,k)$，其初值设为 0，从而得到累加器矩阵 $B_{e\times f\times g}$ 。

（3）将 $\alpha\in[0,2\pi],r\in[0,\min(c,d)/2]$ 与有效像素集合点 $Z=\{X,Y\}$ 作为输入量，带入圆极坐标公式（4）中，可求解得到 $g\times z$ 组 (a_l,b_l,r_l) 。

$$\begin{aligned}&a_l=x_t-r_i\times\cos\alpha_j\\&b_l=y_t-r_i\times\sin\alpha_j\\&r_i=i\times\Delta r,\ i=1,2,...,g\\&\alpha_j=j\times\Delta\alpha,\ j=1,2,...,z\end{aligned} \tag{4}$$

（4）将 $g\times z$ 组 (a_l,b_l,r_l) 值带入 Hough 三维参数空间 $a-b-r$ 中，可找到与 (a_l,b_l,r_l) 等值的 $B_{e\times f\times g}$ 单元所对应的累加器 $B(i,j,k)$，并使该累加器 $B(i,j,k)$ 加 1，即

$$B(i,j,k)=B(i,j,k)+1 \tag{5}$$

（5）当有效像素集合点 z 中所有有效像素点都完成上述（3）和（4）的步骤后，对累加器矩阵 $B_{e\times f\times g}$ 中的各个累加器值进行峰值检验。由于被测二维图像的有效线段的粗细不同，因此会存在累加器峰值被削弱的现象，即使同一个圆，由于线条粗细不同的问题，经 Hough 变换后，也会识别出多个绕某一中心波动的直径近似相同的圆。Hough 变换识别的圆效果图如图 10 所示。

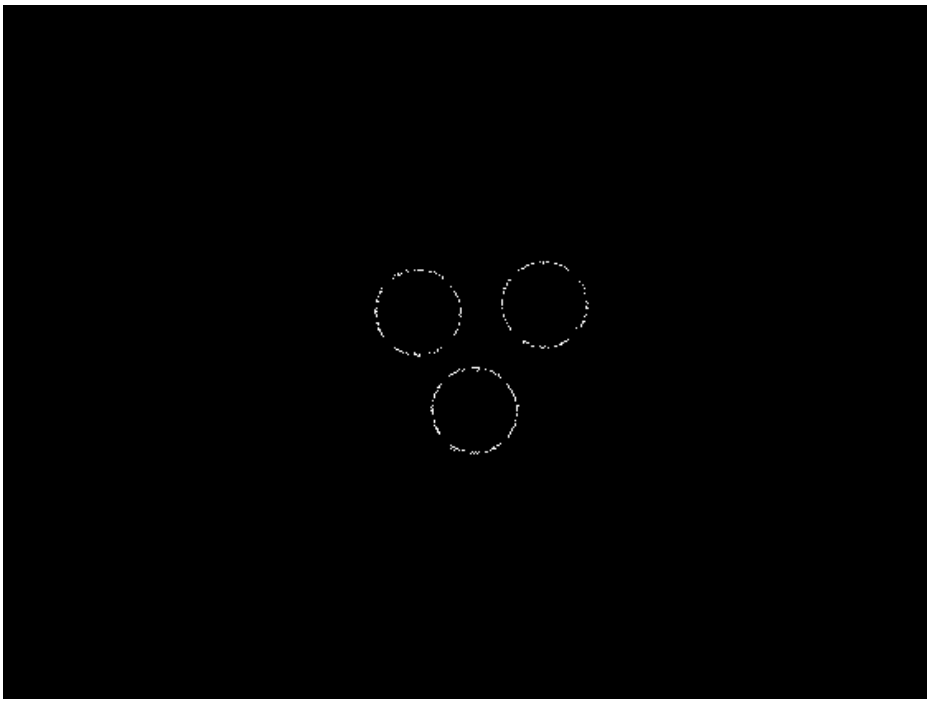

图 10　Hough 变换识别的圆效果

通过 Hough 变换识别的各个圆的圆心位置坐标及半径如表 2 所示。

表 2　圆心位置坐标及半径

组号	位置坐标（x, y）	半径/r	组号	位置坐标（x, y）	半径值 r
1	（187, 253）	25	10	（246, 287）	26
2	（246, 287）	25	11	（247, 287）	26
3	（247, 288）	25	12	（248, 287）	26
4	（268, 288）	25	13	（246, 288）	26
5	（182, 329）	25	14	（247, 288）	26
6	（183, 329）	25	15	（183, 329）	26
7	（187, 253）	26	16	（182, 330）	26
8	（188, 253）	26	17	（183, 330）	26
9	（188 , 254）	26			

通过上述 Hough 变换的原理来准确获取圆形的几何特征存在一定的问题，为减少识别误差，本文采用 K-means 聚类分析，对 17 组峰值较高的(a_l,b_l,r_l)进行分类。

第五步：对 17 组圆的坐标参数(a_l,b_l,r_l)进行聚类分析

K-means 聚类分析基本原理：

（1）假设有 K 个圆，从第四步所得的 N 个圆的坐标参数(a_l,b_l,r_l)中任意选择 K 个坐标参数(a_{ck},b_{ck},r_{ck})，并将其视为初始的聚类中心。

（2）计算各个圆的坐标参数与聚类中心坐标的距离

$$D=\sqrt{(a_1-a_{ck})^2+(b_1-b_{ck})^2+(r_1-r_{ck})^2} \tag{6}$$

根据最小距离原则，对每个圆的坐标参数重新进行类别划分。

（3）重新计算每个类别的坐标参数均值，并将其赋值给(a_{ck},b_{ck},r_{ck})。

（4）循环步骤（2）、（3），直至每个类别的坐标参数均值不再发生变化。这种划分使得各个圆的坐标参数与聚类中心坐标的距离最小，可减少通过 Hough 变换识别的圆参数的误差。

（5）最终得到的 K 个坐标参数均值，即被测零件轮廓中圆的信息，记为$[A,B,R]$。

令 K=3，聚类分析效果图如图 11 所示：

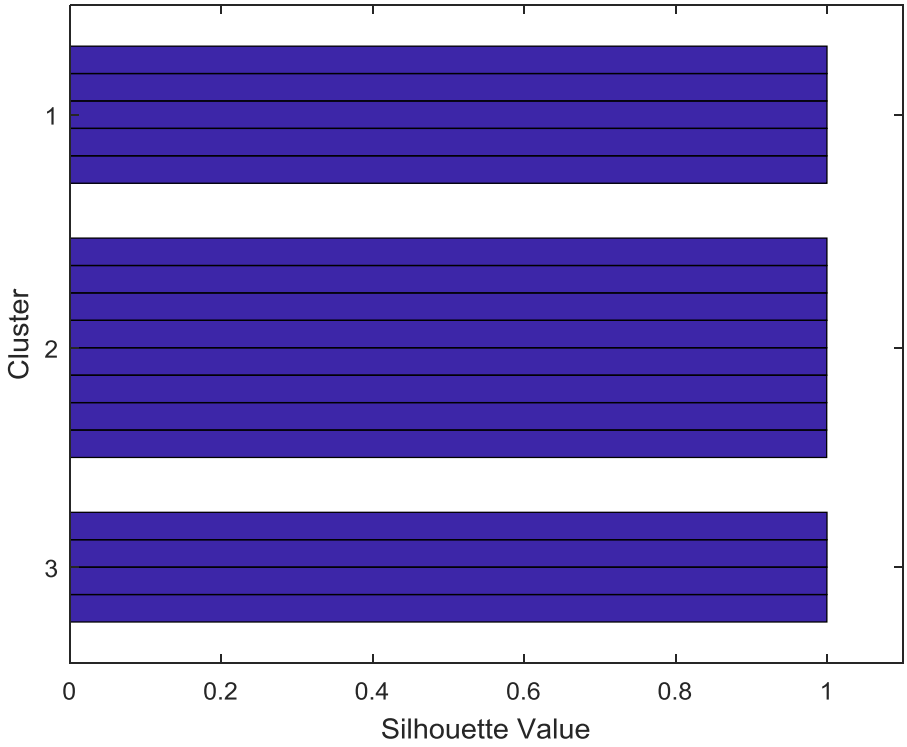

图 11　被测零件轮廓中圆识别的 K-means 聚类分析效果

通过聚类分析可得到被测零件轮廓中圆的参数如表 3 所示。

表 3　被测零件轮廓中圆的参数

类别	位置坐标	半径/r
圆一	（247, 288）	26
圆二	（188, 253）	26
圆三	（183, 329）	26

因此，在绝对坐标系中，被测零件的矩形外廓的四个顶点坐标分别是（182, 172）、（266, 317）、（334, 84）、（414, 230），矩形的长和宽分别为$c=167.573$和$d=175.636$；三个圆心点的坐标分别为（247, 288）、（188, 253）和（183, 329），其半径均为$r=26$。从而确定了被测零件在绝对坐标系中的位置。

4.1.2 基于模拟退火算法的被测零件相对位置识别模型

在工业生产线过程中，被测零件的特征尺寸可以很方便地通过图纸或测量获得，实际应用更关注的是如何将被测零件通过机械手移动到标准位置（或目标位置，如机床工作台或车床主轴卡盘）。若需识别相对位置，则需要测量出零件相对于标准模板位置的坐标。此时，若再利用 4.1.1 节中基于 Hough 变换与聚类分析的识别模型，会重复提取被测零件内的几何特征信息，不仅会使数据处理冗杂和低效，而且所得到的基于绝对坐标的位置数据并不能使机械手将被测零件准确移动到标准位置。因此，本文提出一种基

于模拟退火算法的被测零件的位置坐标提取方法。首先，建立被测零件的标准模板，去除重复提取被测零件几何特征步骤，将标准模板的位置作为最终的移动位置。以被测零件相对于标准模板位置移动的 2 个参数（x 向的移动量和 y 向的移动量）和旋转的 1 个参数作为变量，对其进行求解，找到使被测零件能准确移动和旋转到标准模板位置的三参数值。

第一步：零件标准位置模板的建立

参考 4.1.1 节获得的水平状态下的被测零件二维图像矩阵 P，建立零件的标准位置模板。为降低后续模拟退火算法求解标准位置模板与被测零件的最小距离的计算量，希望得到一个轮廓线极细的零件标准位置模板。模板建立的具体过程为：

（1）建立零件标准位置的空白模板矩阵

参考水平状态下的被测零件的二维图像矩阵 P，建立一个与 P 同等大小的元素为 0 的空白模板矩阵 KP。

（2）建立模板的矩形轮廓像素点

以水平状态下的被测零件二维图像矩阵 P 的行标值 i 为自变量，寻找水平方向上有效像素为最多的两行，将其作为模板矩形轮廓的行像素。

以水平状态下的被测零件二维图像矩阵 P 的列标值 j 为自变量，寻找竖直方向上有效像素最多两列，将其作为模板矩形轮廓的列像素。

将行像素和列像素信息写入空白模板的矩阵 KP 中。

（3）建立模板的圆形轮廓像素点

根据 4.1.1 节第五步所得的被测零件的圆形几何特征信息 $[A,B,R]$，空白模板矩阵 KP 中对满足圆形坐标公式（7）的像素点置 1，建立模板圆轮廓像素点。

$$(X-A)^2+(Y-B)^2=R^2 \tag{7}$$

从而得到零件标准位置模板的矩阵 ZP，其中设 ZP 矩阵共有 n 行、m 列。标准位置模板如图 12 所示：

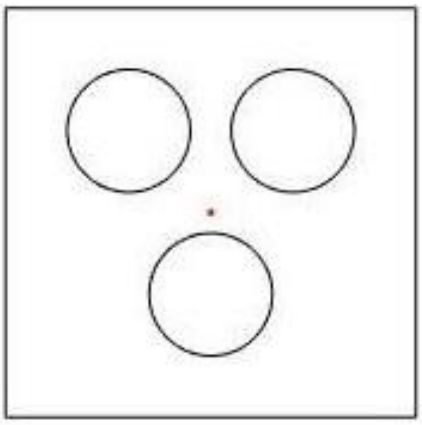

图 12　标准位置模板

第二步：计算被测零件与标准位置模板的相对位置

（1）随机提取被测零件与标准位置模板各自有效像素点

假设零件的拍摄位置、角度、像素均不改变，即被测零件与标准位置模板的二维图像提取的底板完全一致，那么显然被测零件图像二维像素矩阵 Q 也有 n 行，m 列。

统计标准位置模板矩阵 ZP 中 1 值的行号 x_a、列号 y_a 以及数量 b，建立标准位置模板的有效像素点集合 ZU，表示为 $ZU=\{ZX,ZY\}$，$ZX=x_1,x_2,...,x_a,...,x_b$，$ZY=y_1,y_2,...,y_a,...,y_b$。

统计被测零件图像二维像素矩阵 Q 中元素为 1 的行号 x_c、列号 y_c 及数量 d，建立被

测零件图像有效像素点集合 QU， $QU=\{QX,QY\}$，$QX=x_1,x_2,...,x_c,...,x_d$，$QY=y_1,y_2,...,y_c,...,y_d$。

在标准位置模板有效像素点集合 ZU 中随机提取 Ta 个不重样的像素点，记为标准位置模板的有效像素抽样点集合 $TS=\{TSX,TSY\}$，$TSX=x_1,x_2,...,x_{ta},...,x_{tb}$，$TSY=y_1,y_2,...,y_{ta},...,y_{tb}$；在被测零件图像有效像素点集合 QU 中随机提取 Tb 个不重样的像素点，记为被测零件图像有效像素抽样点集合 $TT=\{TTX,TTY\}$，$TTX=x_1,x_2,...,x_{tc},...,x_{td}$，$TTY=y_1,y_2,...,y_{tc},...,y_{td}$。

（2）平移被测零件图像的有效像素抽样点至起始点

为加快计算速度，以被测零件二维图像的像素起始点作为绝对坐标系的原点，记被测零件的起始点坐标为 (a_q,b_q)，平移被测零件的图像有效像素抽样点，得到第 0 次平移后的被测零件图像有效像素抽样点为 $TT\_move(0)=\{TTX-a_q,TTY-b_q\}$。

（3）计算第 0 次平移后的所有被测零件图像有效像素抽样点与标准位置模板有效像素抽样点的最短距离，同时寻找距第 0 次平移后的被测零件图像有效像素抽样点 $TT\_move(0)$ 中任意点 (TTX_i-a_q,TTY_i-b_q) 与标准位置模板有效像素抽样点集合 TS 的最短距离记为 S_{i0}，即

$$\begin{aligned} &S_{i0}=\min(\sqrt{(TTX_i-a_q-x_{ta})^2+(TTY_i-b_q-y_{ta})^2})\\ &ta=1,2,...,tb\\ &i=1,2,...,td \end{aligned} \tag{8}$$

求得第 0 次平移后的所有被测零件图像有效像素抽样点与标准位置模板有效像素抽样点的最短距离和 sum_S_0。

$$sum\_S_0=\sum_{i=1}^{td}S_{i0} \tag{9}$$

（4）基于模拟退火算法，平移并旋转被测零件图像的有效像素抽样点，使被测零件图像的有效像素抽样点与标准位置模板的有效像素抽样点的最短距离和最小。

以被测零件图像的有效像素抽样点与标准位置模板的有效像素抽样点的最短距离和最小为目标，以被测零件图像的有效像素抽样点当前平移位置和旋转角度 $(move\_x,move\_y,move\_angle)$ 为自变量，建立如下模型，如式（10）所示。通过模拟退火算法求得最优解。

$$\begin{aligned} \min\quad & sum\_S_0\\ s.t.\quad & 0\le \text{move\_x}\le \max(\text{ZX,QX})\\ & 0\le \text{move\_y}\le \max(\text{ZY,QY})\\ & -\pi\le \text{move\_angle}\le \pi \end{aligned} \tag{10}$$

（5）模拟退火算法的求解

第一步：令温度 $T_1=T_0$，即开始退火的初始温度，随机生成一个初始解 $W_0=(move\_x_0,move\_y_0,move\_angle_0)$，并计算相应的目标函数值 $sum\_S_0(W_0)$。

第二步：根据退火因子 q 确定下一个温度，即令 $T_{i+1}=qT_i$，并根据当前解 W_i 进行扰动，产生一个新解 W_{i+1}，并计算相应目标函数值 $sum\_S_0(W_{i+1})$，从而得到 $\Delta=sum\_S_0(W_{i+1})-sum\_S_0(W_i)$。

第三步：若 $\Delta<0$，则新解 W_{i+1} 被接受，作为新的当前解；若 $\Delta>0$，则新解 W_{i+1} 按概率 $\exp(-\Delta/T_i)$ 接受。

第四步：在温度 T_i 下，重复 L_k 次的扰动和接受过程，即再次运行第三步、第四步。

第五步：判断 T 是否已到达 T_f，如已到达，则终止计算，否则转到第二步继续运行。

模拟退火算法的初始参数设置如下：

设计变量为给定零件轮廓的矩形中心相对于标准模板的位置坐标 $(move\_x, move\_y)$ 以及旋转角度 $move\_angle$ 的初始位置分别为 $move\_x_0 = 100$、$move\_y_0 = 100$，$move\_angle_0 = 100$

设定初始温度为 $T_0 = 100$，最终温度为 $T_f = 3$，退火因子为 q=0.8；

初始解 Z0，为在给定零件及标准模板轮廓上分别随机均匀提取的 n 和 m 个点的位置坐标，给定零件平移及旋转的上限为 $move\_x_{\max} = 400$、$move\_y_{\max} = 400$，$move\_angle_{\max} = \pi$。

给定零件平移及旋转的下限为 $move\_x_{\min} = -400$、$move\_y_{\min} = -400$，$move\_angle_{\min} = -\pi$。

目标函数可允许误差为 $1e^{-9}$；最大迭代次数 $K_{\max} = 50000$。

当取 $K_{\max} = 50000$ 该方法的运算效果图如图 13 所示：

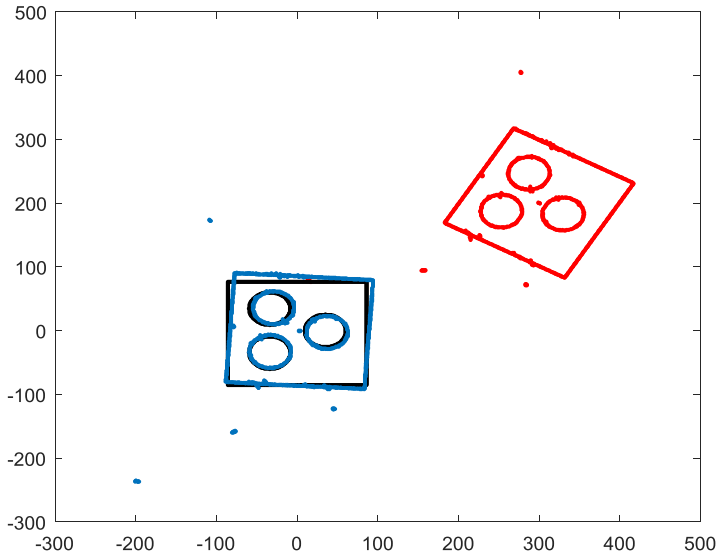

图 13　基于模拟退火的相对位置识别（最大迭代次数 $K_{\max} = 50000$）

程序的运行时间为如下图 14 所示：

开始探查(P) 运行此代码(R): 探查时间: 7 秒

探查摘要

基于performance时间于 16-Apr-2018 12:18:00 生成。

函数名称	调用次数	总时间	自用时间*	总时间图 (深色条带 = 自用时间)
Opt_Simu	1	5.729 s	0.323 s	
test_demo2>@(x)myfun2(x,d,D)	25295	5.224 s	0.150 s	
myfun2	25295	5.108 s	5.108 s	
test_demo2	1	1.000 s	0.709 s	
Mu_Inv	25294	0.183 s	0.183 s	
imshow	3	0.176 s	0.075 s	
subplot	2	0.084 s	0.057 s	
newplot	8	0.038 s	0.010 s	
basicImageDisplay	3	0.033 s	0.020 s	
newplot>ObserveAxesNextPlot	8	0.024 s	0.003 s	
subplot>addAxesToGrid	2	0.023 s	0.006 s	
cla	7	0.021 s	0.004 s	
imrotate	1	0.020 s	0.001 s	

图 14　最大迭代次数为 50000 的程序运行时间

其中黑色为标准模板的位置，红色为被测零件的初始位置，蓝色为优化后（平移和旋转）所得模型，依据算法所输出的位置信息为 $move\_x=-297.818$、$move\_y=-202.731$，$move\_angle=0.517386$。即被测零件中心相对于标准模板中心的位置为（297.818, 202.731），逆时针旋转 29.6590°程序运行时间为 5.729s。

若取最大迭代次数 kmax=5000，该方法的运算效果图如图 15 所示

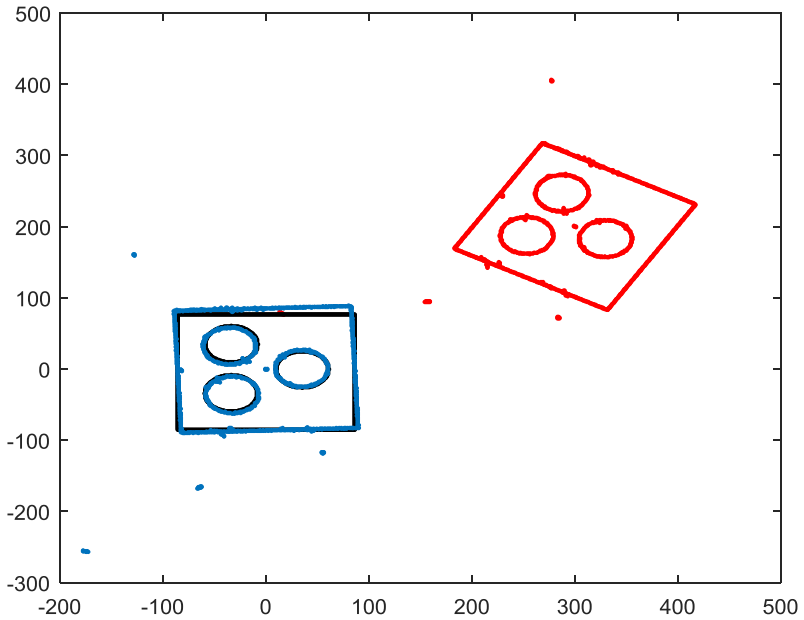

图 15　基于模拟退火的相对位置识别（最大迭代次数 kmax=5000）

运行时间图如下：

函数名称	调用次数	总时间	自用时间*	总时间图 (深色条带 = 自用时间)
test_demo2	1	2.306 s	0.840 s	
Opt_Simu	1	1.126 s	0.066 s	
test_demo2>@(x)myfun2(x,d,D)	5002	1.023 s	0.021 s	
myfun2	5002	1.002 s	1.002 s	
imshow	3	0.197 s	0.082 s	
subplot	2	0.097 s	0.066 s	
newplot	9	0.044 s	0.012 s	
Mu_Inv	5001	0.038 s	0.038 s	
basicImageDisplay	3	0.035 s	0.021 s	
imrotate	1	0.033 s	0.003 s	
newplot>ObserveAxesNextPlot	9	0.028 s	0.004 s	
subplot>addAxesToGrid	2	0.028 s	0.008 s	
cla	7	0.025 s	0.004 s	

图 16　最大迭代次数为 5000 的程序运行时间

依据算法所输出的位置信息为 $move\_x=-296.0209$、$move\_y=-200.6858$，$move\_angle=0.489386$，被测零件中心相对于标准模板中心的位置为（296.0209, 200.6858），逆时针旋转 28.0433°，运行时间为 2.306s。

由图可以看出，如果迭代次数取 50000 次左右时，该算法可以精确地计算出被测零件相对于标准位置模板的位置信息，很好地实现了零件的定位。该方法针对被测零件上所有轮廓点随机选取几百个点进行模型优化，优化过程虽然具有高精度，然而求解过程所用时间相对较慢（5.729s），虽然减少迭代次数可以在一定程度上缩短零件轮廓的定位时间，然而为之付出的是牺牲很大的定位精度（如果迭代次数取 5000 时，定位时间为 2.306s，x 向的平移误差为 0.6%、y 向的平移误差为 1%、旋转误差为 5.44%）。虽然说误差在机械加工的自动化定位中是可接受的，但是定位时间相对而言依然不低，一定程度上还会影响生产效率。

为此，本文考虑通过减小随机点选取的数量或者提取被测零件中的某些特征点，并只对这几个特征点进行模型优化（平移和旋转），从而实现零件定位得更迅速、更精确。具体定位方法详见第 5 节改进模型及求解。

4.2 问题 2：模型的建立求解

通过前文对问题 2 的分析，对于确定同一图像中多个零件位置的问题首先要实现多个图像的分割，即将每个零件单独地置于某一区域；之后再分别对所分割地图像进行相应的平移和旋转，从而得到各个零件的位置坐标与旋转角度。其中位置的确定与问题 1 中的方法相类似，本节不在赘述。

4.2.1 按轮廓分割现有图像

第一步：识别零件图像圆几何特征

类似于前文 4.1.1 的第四步、第五步，通过 Hough 变换圆检测方法得到 17 个圆的圆心及半径如下表

表 4 Hough 变换圆检测所得圆心坐标及半径

组号	位置坐标（x, y）	半径/r	组号	位置坐标（x, y）	半径值 r
1	（452 248）	25	10	（188 253）	26
2	（452 249）	25	11	（375 253）	26
3	（247 288）	25	12	（376 253）	26
4	（ 417 189）	25	13	（375 254）	26
5	（418 189）	25	14	（247 287）	26
6	（418 190）	25	15	（247 288）	26
7	（452 248 ）	26	16	（183 329 ）	26
8	（452 249）	26	17	（183 330）	26
9	（187 253）	26			

经过 K-means 聚类分析，可求得 DATA2 中六个圆的圆心坐标如表 5 所示

表 5　DATA2 中各圆心点坐标值

圆心序号	1	2	3	4	5	6
坐　　标	(183,330)	(247,288)	(188,253)	(418,189)	(375,253)	(452,249)

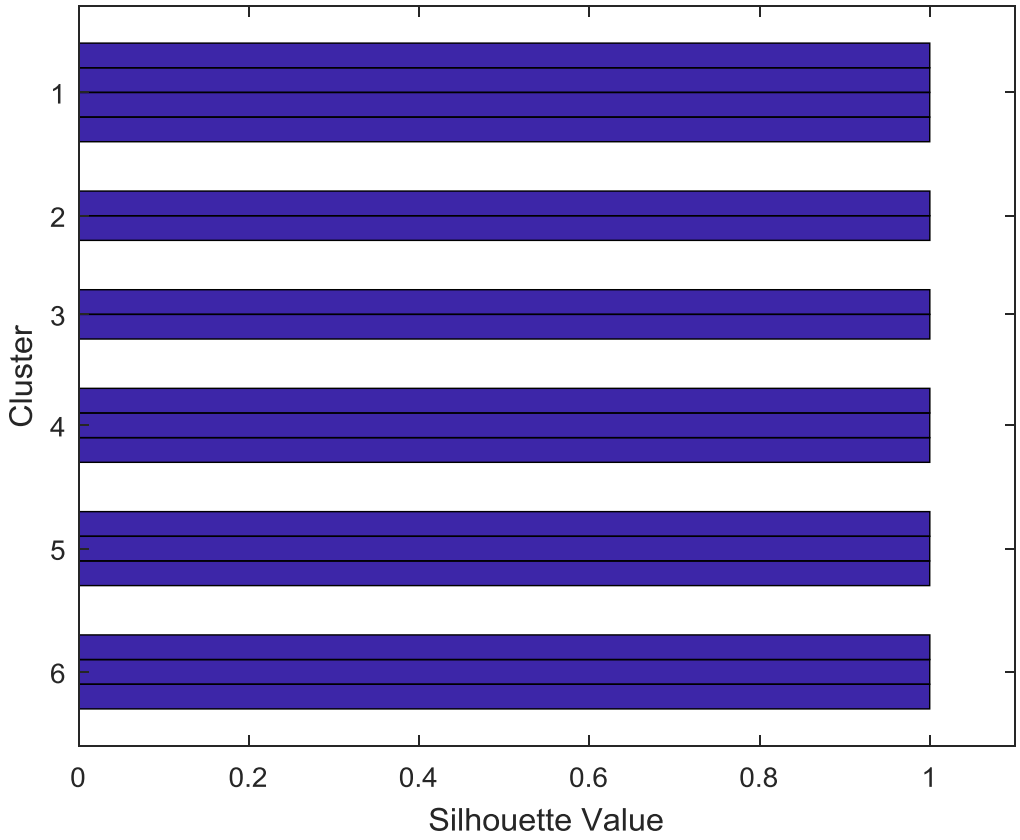

图 17　多个被测零件轮廓中圆识别的 K-means 聚类分析效果

通过识别后得到圆的半径均为 26。圆识别后所得到的图像如图 18 所示。

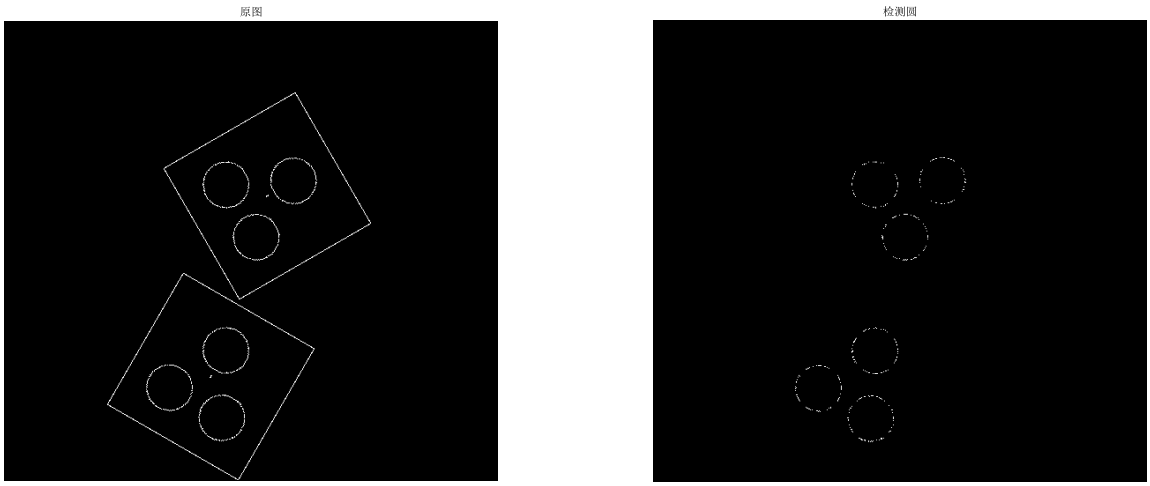

图 18　DATA2 Hough 变换圆检测结果图

第二步：按零件对圆的坐标参数值进行分组

（1）假设圆的个数已知，设为 K' 个，从第一步所得的 N' 个圆的坐标参数 (a_l', b_l') 中任意选择 K' 个坐标参数 (a_{ck}', b_{ck}') 作为初始聚类中心。

（2）计算各圆的坐标参数与聚类中心坐标的距离

$$D'=\sqrt{(a_l' - a_{ck}')^2 + (b_l' - b_{ck}')^2} \tag{11}$$

并根据最小距离重新对每个圆的坐标参数进行类别划分，即将其赋给最近的类别。

（3）重新计算每个类别的坐标参数均值，并重新赋值给（a_{ck}', b_{ck}'）。

（4）循环（2）（3）直至每个类别的坐标参数均值不再发生变化。这种划分使得各圆的坐标参数与聚类中心坐标的距离最小。

（5）最终得到 K' 组分类信息，每组内有圆心得坐标参数 $\frac{N'}{K'}$ 个，即每个零件中圆的个数 K。

通过聚类识别，可将识别得到的六个圆心点坐标分为两组，即第一组为 1 号点、2 号点和 3 号点，为零件一中的三个圆，第二组为 4 号点、5 号点和 6 号点，确定为零件二中的三个圆。

第三步：划分每个零件的区域

（1）根据零件标准模板中被测零件圆心坐标（a_{ck}, b_{ck}）和矩形轮廓中心的坐标（a_r, b_r），可分别计算出标准模板中。

（2）根据零件标准模板中矩形轮廓的对角点（e_1, f_1）和（e_2, f_2）计算出矩形轮廓对角线的长度。

（3）以 K' 组中的一组 K 个圆心为圆心，步骤（1）中求出的标准模板 K 个圆的圆心到矩形轮廓中心的距离为半径画圆，圆的交点即为图像中相应零件矩形轮廓的中心点；其他各组圆心坐标也进行相同的处理，获得相应零件矩形轮廓的中心点。

（4）以零件矩形轮廓的中心点为圆心，矩形对角线的长度为直径画圆，此圆即为该零件对应所属的区域。

通过采用目标轮廓分割方法，可以近似得到图像中每个零件对应的区域，具体区域

见图 19 和图 20。

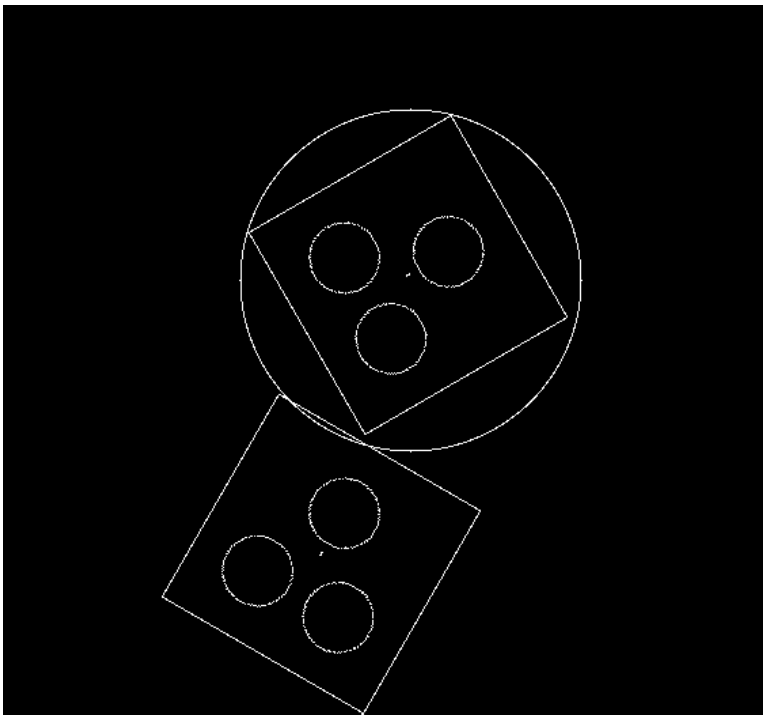

图 19　零件 a 对应的区域图（大圆轮廓内）

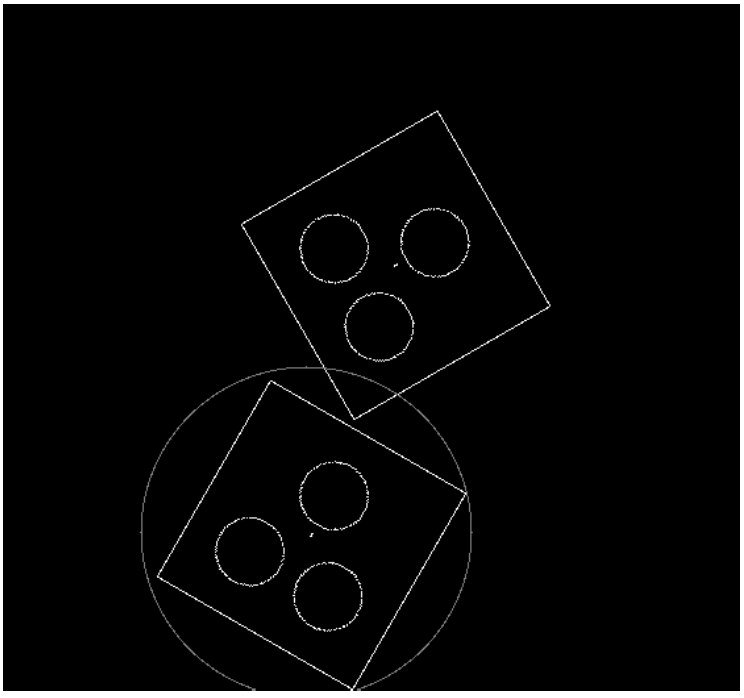

图 20　零件 b 对应的区域图（大圆轮廓内）

（5）分别屏蔽零件 b 和零件 a，从而得到零件的目标图像如图 21 和图 22 所示。

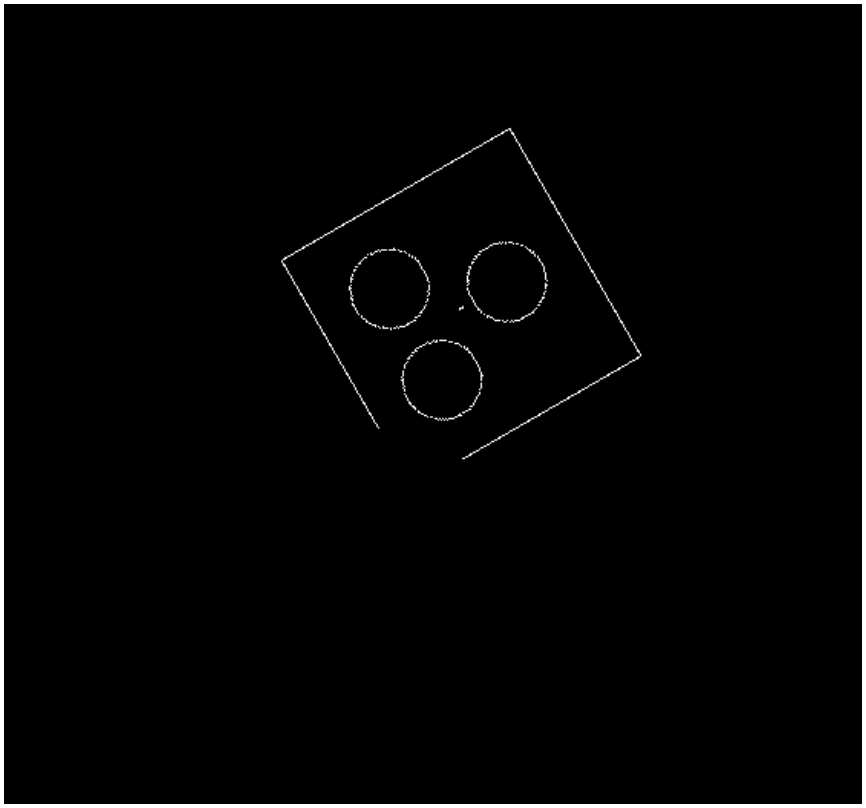

图 21　零件 a 的目标图像

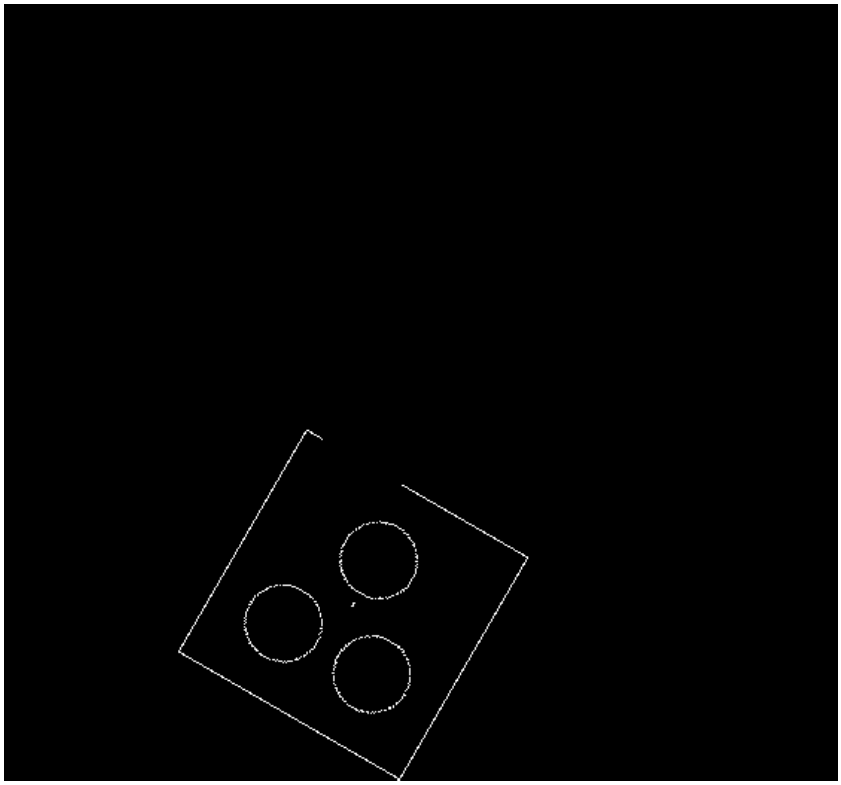

图 22　零件 b 的目标图像

可以看出两个零件的二值化图像被分割为两个仅包含一个零件的图像，因此问题 2

既转变为求解问题 1 中被测零件与目标模板相对位置关系的问题。

4.2.2 确定每个零件相对于目标模板的位置信息

第一步：屏蔽干扰零件图像

将 K' 个零件图像中的一个零件图像取为目标图像，屏蔽其他零件的所属区域，即在该区域内全部置 0。

第二步：确定目标图像的相对位置坐标与旋转角度

参考问题一 4.1.2 章节的基于模拟退火算法的被测零件相对位置识别模型的方法，获得零件 a 的目标图像的位置坐标与旋转角度。

第三步：循环进行第一步、第二步，获得零件 b 的目标图像的位置坐标与旋转角度。

采用 4.2.2 中的方法，令 kmax=5000 可求得零件 a 的位置信息，如表 6 所示。

表 6　零件 a 位置信息 （基于轮廓分割的位置识别）

move_x	move_y	move_angle
-239.9272	-400.0000	1.057831

零件 a 优化后与标准位置模板对比图如图 23 所示：

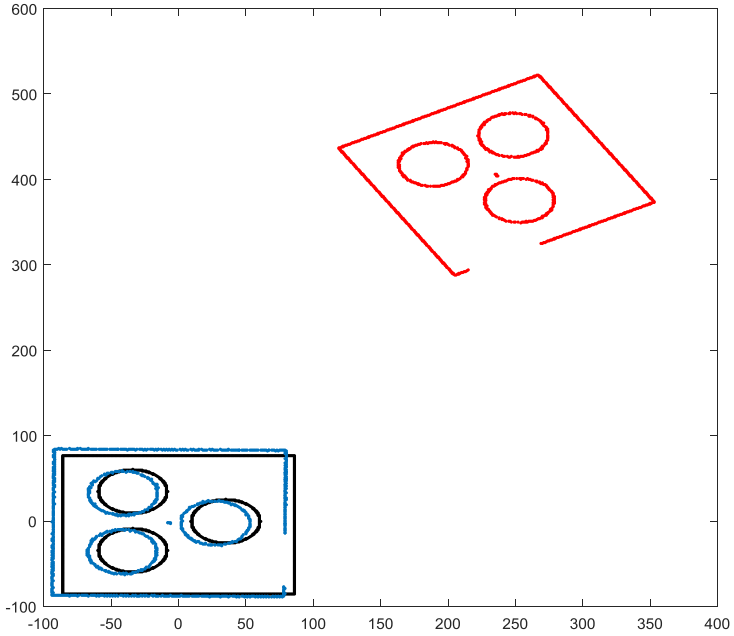

图 23　零件 a 与标准位置模板比对图（基于轮廓分割的位置识别）

零件 a 所用优化模型的运行时间如下图所示。


开始探查(P) 运行此代码(R): 探查时间: 3 秒

探查摘要

基于performance时间于 15-Apr-2018 23:45:49 生成。

函数名称	调用次数	总时间	自用时间*	总时间图 (深色条带 = 自用时间)
PositionBelow	1	2.815 s	0.521 s	
Opt_Simu	1	1.987 s	0.476 s	
PositionBelow>@(x)myfun2(x,d,D)	50002	1.227 s	0.143 s	
myfun2	50002	1.084 s	1.084 s	
Mu_Inv	50001	0.284 s	0.284 s	
imshow	1	0.276 s	0.041 s	
initSize	1	0.166 s	0.008 s	
movegui	1	0.142 s	0.135 s	
newplot	8	0.037 s	0.015 s	
imageDisplayParseInputs	1	0.021 s	0.002 s	
newplot>ObserveAxesNextPlot	8	0.017 s	0.004 s	
basicImageDisplay	1	0.015 s	0.008 s	
imageDisplayValidateParams	1	0.014 s	0.006 s	

图 24　零件 a 优化模型的运行时间

通过 4.2.2 中的方法令 kmax=5000 可求得零件 b 的位置信息，如表 7 所示

表 7　零件 b 位置信息 （基于轮廓分割的位置识别）

move_x	move_y	move_angle
-305.9829	-199.5862	0.511297

零件 b 优化后与标准位置模板对比图如图 25 所示：

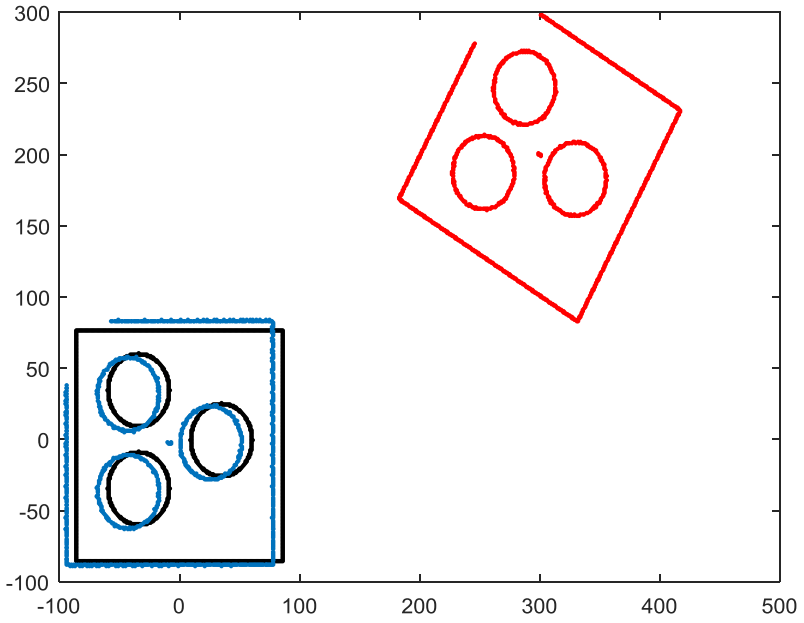

图 25　零件 b 与标准位置模板比对图（基于轮廓分割的位置识别）

零件 b 所用优化模型的运行时间如下图所示。


开始探查(P) 运行此代码(R):

探查时间: 3 秒

探查摘要

基于performance时间于 16-Apr-2018 00:01:24 生成。

函数名称	调用次数	总时间	自用时间*	总时间图 (深色条带 = 自用时间)
PositionUp	1	2.951 s	0.487 s	
Opt_Simu	1	2.119 s	0.494 s	
PositionUp>@(x)myfun2(x,d,D)	50002	1.318 s	0.149 s	
myfun2	50002	1.169 s	1.169 s	
imshow	1	0.317 s	0.047 s	
Mu_Inv	50001	0.307 s	0.307 s	
initSize	1	0.183 s	0.013 s	
movegui	1	0.148 s	0.142 s	
newplot	8	0.035 s	0.015 s	
isSingleImageDefaultPos	1	0.023 s	0.013 s	
imageDisplayParseInputs	1	0.020 s	0.001 s	
basicImageDisplay	1	0.018 s	0.011 s	
imageDisplayValidateParams	1	0.017 s	0.006 s	

图 26　零件 b 优化模型的运行时间

由以上两次运行结果可以看出当迭代次数为 k=5000 时，依据算法所输出的零件 a 的位置信息为 $move\_x=-305.9829$、$move\_y=-199.5862$，$move\_angle=0.511297$，被测零件中心相对于标准模板中心的位置为(305.9829, 199.5862)，逆时针旋转 29.31°，运行时间为 2.951s.依据算法所输出的零件 b 的位置信息为 $move\_x=-239.9272$、$move\_y=-400.0000$，$move\_angle=-1.057831$，被测零件中心相对于标准模板中心的位置为（239.9272,400.0000)，逆时针旋转 60.64°，运行时间为 2.815s。

5.改进模型建立与求解

5.1 基于零件几何特征的快速位置识别模型

在 4.1.2 节中未考虑被测零件的几何特征，如圆心、正方形等，直接通过随机抽样提取被测零件与标准位置模板的有效像素点，通过采用模拟退火算法求解最优的被测零件平移位置和旋转角度。而这种方法在进行 $Tb\times Ta$ 次（通常比较大）的被测零件图像有效像素抽样点与标准位置模板有效像素抽样点的最短距离和的最小解迭代时，大大影响了模拟退火算法的求解速度。

通过观察被测零件几何特征，发现被测零件由 4 条线段和 3 个圆组成，通过标准位置模板，直接得到了 4 条线段与 3 个圆的相对位置关系和圆的半径，因此直接求得 3 个圆的圆心位置即可得到整个被测零件的相对位置关系。因此，本文提出一种基于零件几何特征的被测零件位置快速识别模型。将目标函数简化为仅考虑被测零件的圆心与标准位置模块圆心的最短距离平方和最小。利用该方法进行位置识别的主要步骤如下：

第一步：提取被测零件和标准位置模板的圆心像素点信息。

参考 4.1.1 节第四步方法，直接利用霍夫变换识别被测零件的 3 个圆心的位置，利用 K-means 进行圆心归类逼近，得到被测零件的 3 个圆心在二维像素空间中的位置及对应圆的半径，记被测零件圆心像素点的坐标值为 $CT_1(x_{ct_1},y_{ct_1})$，$CT_2(x_{ct_2},y_{ct_2})$，$CT_3(x_{ct_3},y_{ct_3})$。

根据 4.1.1 节第五步所得到的旋转后的被测零件的圆特征信息，等价于标准位置模板圆心像素点的坐标值，记为 $CS_1(x_{cs_1},y_{cs_1})$，$CS_2(x_{cs_2},y_{cs_2})$，$CS_3(x_{cs_3},y_{cs_3})$。

第二步：平移被测零件图像圆心点至起始点。

为提高计算速度，以被测零件二维图像的像素起始点作为绝对坐标原点，记被测零

件起始点坐标为(a_{pq},b_{pq})，平移被测零件图像的 3 个圆心点，得到第 0 次平移后的被测零件图像的 3 个圆心点为$IT_move(0)=\{x_{ct_i}-a_{pq},y_{ct_i}-b_{pq}\}$。

第三步：计算第 0 次平移后的被测零件图像 3 个圆心点与标准位置模板 3 个圆心点的最短距离和，寻找距第 0 次平移后的被测零件图像的 3 个圆心点$IT_move(0)$中任意点$(x_{ct_i}-a_{pq},y_{ct_i}-b_{pq})$最短的标准位置模板的 3 个有效点集合 IS 距离记为IS_{i0}；

$$\begin{aligned} &IS_{i0}=\min(\sqrt{(x_{ct_i}-a_{pq}-x_{cs_i})^2+(x_{ct_i}-b_{pq}-y_{cs_i})^2}) \\ &ct_i=1,2,3 \\ &cs_i=1,2,3 \end{aligned} \tag{12}$$

求得第 0 次平移后的所有被测零件图像的 3 个圆心点与标准位置模板的 3 个圆心点的最短距离和sum_IS_0

$$sum\_IS_0=\sum_{i=1}^{3}IS_{i0} \tag{13}$$

第四步：基于模拟退火算法，平移并旋转被测零件图像的 3 个圆心点，使被测零件图像的 3 个圆心点与标准位置模板的 3 个圆心点的最短距离和最小。为此，识别被测零件与标准位置模板之间的距离及角度。

5.2 改进模型的求解

5.2.1 问题 1 改进模型的求解

上文 4.1.1 中求出的基于霍夫变换与聚类分析的位置识别模型三个圆心点的坐标分别为（247, 288）、（188, 253）和（183, 329），以此三个圆心作为特征点求解如下

表 8　问题 1 中零件轮廓的位置信息（基于特征的位置识别）

项目	move_x	move_y	move_angle
位置信息	-299.5761	-200.5846	0.5298
误差	0.59%	1%	2.4%

问题一改进模型优化图如下：

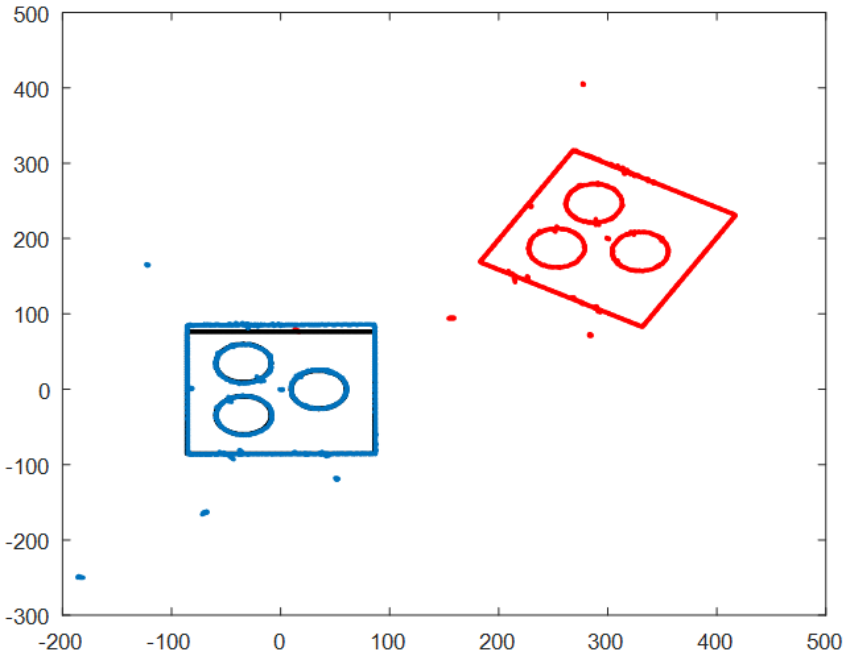

图 27　问题一改进模型优化设计求解图

该种方法的运行时间结果如图 28 所示。

图 28　问题一程序运行时间图

由表 8 及时间运行图可以看出，当迭代次数取 5000 时，依据算法所输出的位置信息为 $move\_x=-299.5761$、$move\_y=-200.5846$，$move\_angle=0.5298$。即被测零件中心相对于标准模板中心的位置为（299.5761, 200.5846），逆时针旋转 29.6590°，程序运行时间仅为 0.653s，x 向的平移误差为 0.59%、y 向的平移误差为 1%、旋转误差为 2.4%，基本每一项指标都高于 4.1.2 中基于模拟退火算法的被测零件相对位置识别模型所提方法。可以看出基于零件几何特征的快速位置识别模型可以很好地实现问题 1 中所要求的精准快速的目标，更适合于实际工况。

5.2.2 问题 2 改进模型的求解

根据 4.2.1 按轮廓分割现有图像小节第一步通过 Hough 变换及 K-means 聚类两个零件的六个圆心位置点坐标分别为（183, 330）、（247, 288）、（188, 253）、（418, 189）（375, 253）和（452,249）。

零件 a 中三个圆的圆心分别为（418, 189）、（375, 253）、（452, 249），零件 b 中三个圆的圆心分别为（183, 330）、（247, 288）、（188, 253）检测结果如下：

表 9　问题 2 中零件 a 位置信息（基于特征的位置识别）

项目	move_x	move_y	move_angle
位置信息	-235.7780	-400.0000	1.0672
误差	1.75%	0.85%	0.83%

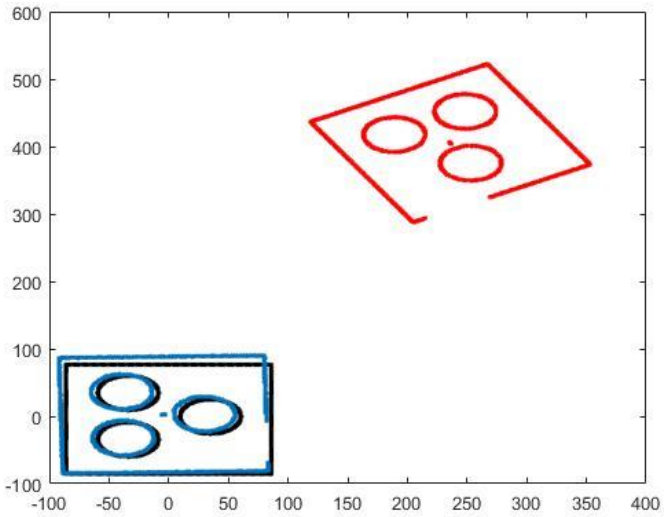

图 29　改进模型中零件 a 与标准位置模板比对图（基于特征的位置识别）

运行时间效果图如下。

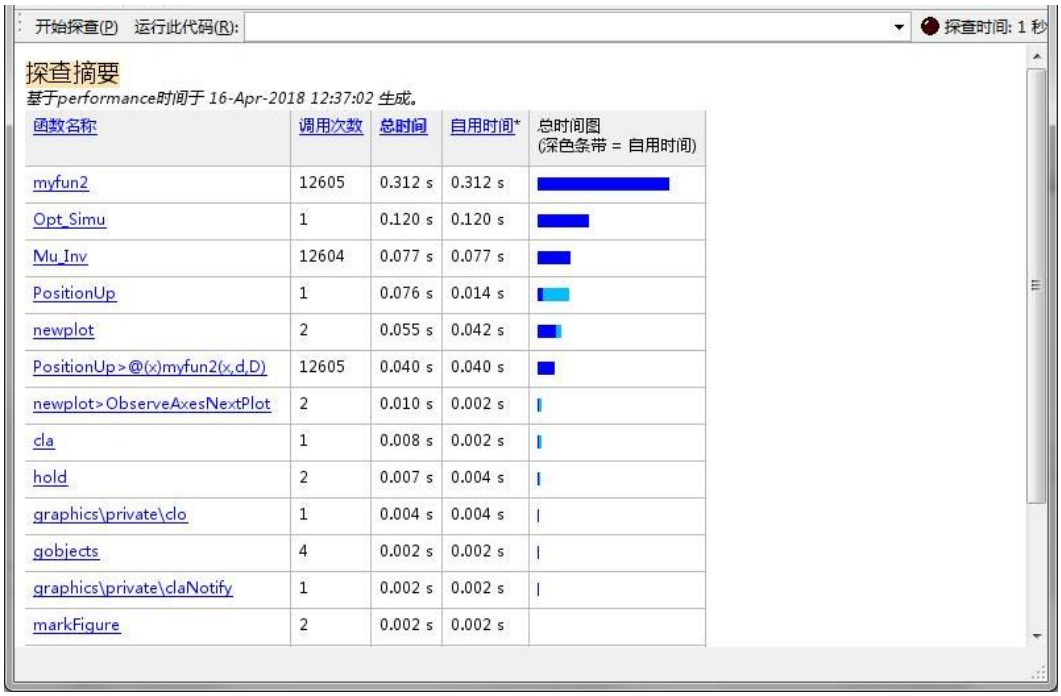

图 30　零件 a 定位程序运行时间图

由表 9 及时间运行图可以看出，当迭代次数取 5000 时，依据算法所输出的零件 a 位置信息为 $move\_x=-235.7780$、$move\_y=-400.0000$、$move\_angle=1.0672$。即被测零件中心相对于标准模板中心的位置为（235.7780,400.0000），逆时针旋转 61.15°，程序运行时间仅为 0.312s，x 向的平移误差为 1.75%、y 向的平移误差为 0.85%、旋转误差为 0.83%，最低识别精度为 98.8%。基本每一项指标都高于 4.1.2 中基于模拟退火算法的被测零件相对位置识别模型所提方法。

表 10　零件 b 位置信息（基于特征的位置识别）

项目	move_x	move_y	move_angle
位置信息	-299.9323	-200.6254	0.5256
误差	2.01%	0.51%	2.66%

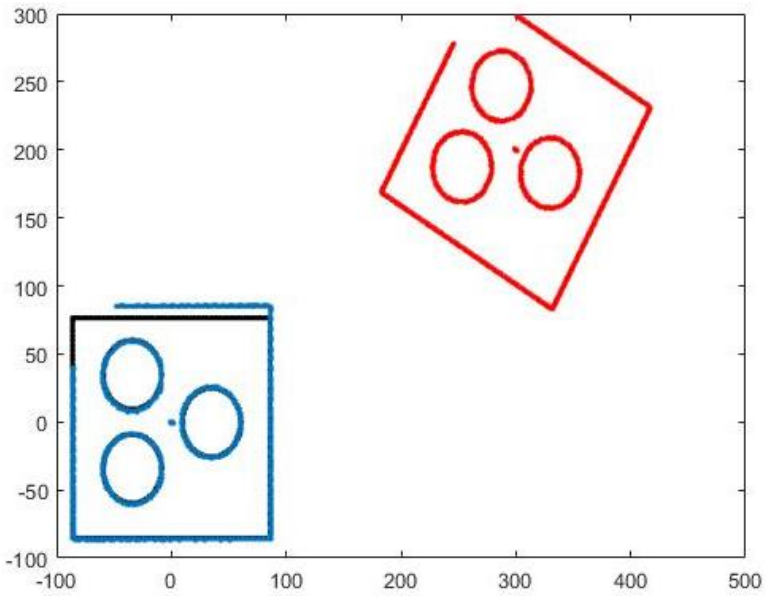

图 31　改进模型中零件 b 与标准位置模板的比对图（基于特征的位置识别）

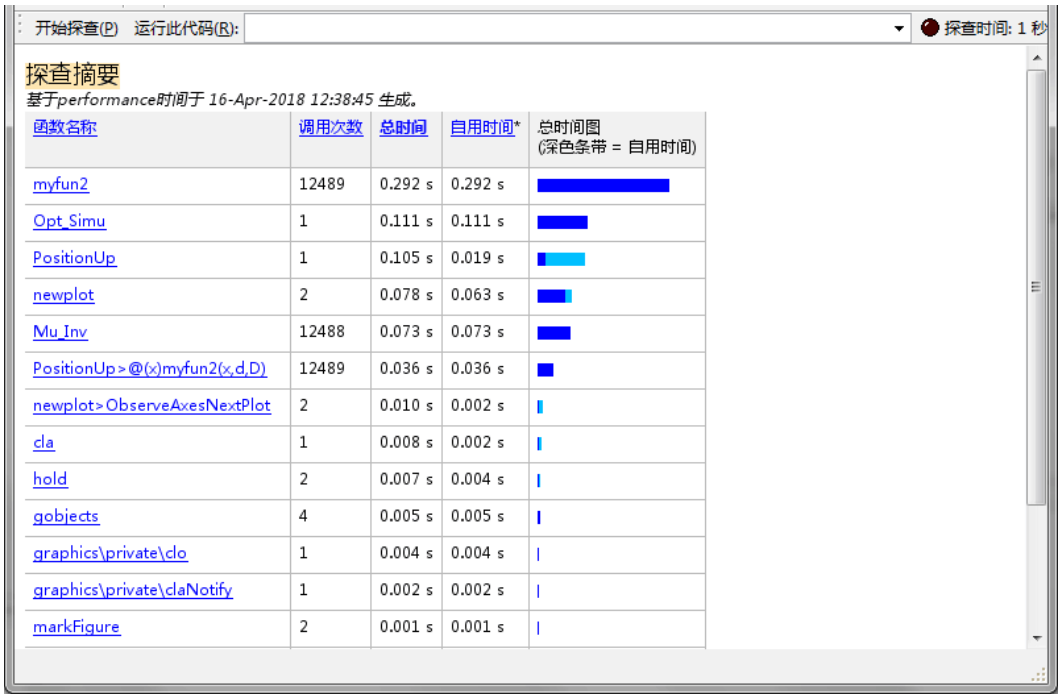

图 32　零件 b 定位程序运行时间图

由表 10 及时间运行图可以看出，当迭代次数取 5000 时，依据算法所输出的位置信息为 $move\_x=-299.9233$、$move\_y=-200.6254$、$move\_angle=0.5256$。即被测零件中心相对于标准模板中心的位置为（299.9233,200.6254），逆时针旋转 30.11°，程序运行时间仅未 0.292s，x 向的平移误差为 2.01%、y 向的平移误差为 0.51%、旋转误差为 2.66%，最低识别精度为 97.6%。基本每一项指标都高于 4.1.2 节中基于模拟退火算法的被测零件相对位置识别模型所提方法。

由表 9 和表 10 可以看出，采用改进方法确定的零件的位置与前文中所用方法相比误差都在 3%以内，位置精度明显提高，且定位时间可缩短 3 倍以上。由于改进模型有

效提取了实际零件的特征值，使得确定零件位置的算法的时间复杂度和空间复杂度相比前文所述方法明显提高，更具有实际价值。但是相比于前文所述方法，其缺点主要在于该方法不适用于所有零件，当实际零件不具备可以表征其位置的特征时，便不能使用此方法。

6.模型的评价

6.1 本文模型优点

（1）针对零件几何特征参数未知的情况，提出了基于 Hough 变换与聚类分析的位置识别模型。通过在图像坐标系内求解零件各几何特征的方程，既可快捷准确地获得零件的位置信息，又可获得零件几何特征的参数，如圆的半径、矩形边长等，可谓一举两得。

（2）针对多数实际工况中零件几何特征参数已知的情况，提出了基于模拟退火算法的被测零件相对位置的识别模型。根据已知零件信息建立标准模板，将标准模板的位置作为最终移动位置。以被测零件相对于标准模板位置移动的两个参数和旋转的一个参数作为变量，进行优化设计求解。利用两个方向的移动量和一个旋转量作为确定零件的位置信息，具有简洁明了的优点。此外，本模型还得到了被测零件到其标准位置的位移向量与旋转角度，更加适合于实际应用中搬运零件到指定位置的情况。

（3）为了提高识别与计算速度，本文提出了改进的基于零件几何特征的快速位置识别模型。本模型提取了零件图像中的三个特征点来表征整个零件，而后对被测零件的三特征点与标准位置的三特征点进行类似于前一模型的比对优化设计，求解两个移动参数和一个旋转参数。通过特征点的选取，可大大减小识别时间，使得计算速度大幅度提高。

（4）对于同一图像中存在多个零件的情况，将问题求解分为两个步骤：一是通过划分区域使零件独立处于某一区域，方便单独处理；二是利用前文提出的模型分别确定每个零件的位置。此模型中根据零件的区域划分使同一图像中不同零件的信息得以区分，避免了计算过程中可能出现的数据混乱现象，所得位置信息更为精确。

6.2 本文模型改进方向

（1）本模型目前仅适用于较简单的零件几何形状，对于结构复杂的零件，本模型的迭代次数可能会迅速增加，计算规模增加，计算速度降低。

（2）在改进模型中，提取三点作为零件特征点，但无法排除三点构成等边三角形这种极端情况，此时需要加入更多的特征点来表征零件，即本文中的零件特征点的选取还未建立更加成熟的模型。

参考文献

[1] 李海涛, 柳健明, 德烈,等. 一种统计特征点网格分布的表格图像识别方法[J]. 华中科技大学学报(自然科学版), 2002, 30(9):60-63.

[2] 徐立云, 李霄峰, 张斌,等. 基于 Hough 变换的模糊建模研究及其应用[J]. 系统仿真学报, 2001, 13(z1):66-68.

[3] 张博, 唐文彦, 黄勇. 采用改进的几何算法快速估计图像旋转角度[J]. 计算机仿真, 2009, 26(6):263-266.

[4] 李亚娣, 黄海波, 李相鹏,等. 基于 Canny 算子和 Hough 变换的夜间车道线检测[J]. 科学技术与工程, 2016, 16(31):234-237.

[5] 刘雪梅, 贾勇琪, 陈祖瑞,等. 缸体类零件加工特征识别方法[J]. 计算机集成制造系统, 2016, 22(5):1197-1204.

[6] 司小婷. 基于视觉的零件特征识别与分类方法研究与实现[D]. 中国科学院研究生院(沈阳计算技术研究所), 2016.

[7] 史思琦. 基于轮廓特征的目标识别研究[D]. 西安电子科技大学, 2012.

[8] 李旭超, 刘海宽, 王飞,等. 图像分割中的模糊聚类方法[J]. 中国图象图形学报, 2012, 17(4):447-458.

附　录

附录 1

1. Hough 变换——直线检测

```
clc
clear
load('DATA1.mat')
BW=D1;
thresh=[0.01,0.17];
sigma=2;                                    %定义高斯参数
f = edge(double(BW),'canny',thresh,sigma);
figure(1),imshow(f,[]);
title('canny  边缘检测');
[H, theta, rho]= hough(f,'RhoResolution', 0.5);
rho=rho;
peak=houghpeaks(H,4);
hold on
lines=houghlines(f,theta,rho,peak);
figure,imshow(f,[]),title('Hough Transform Detect Result'),hold on
for k=1:length(lines)
    xy=[lines(k).point1;lines(k).point2];
    plot(xy(:,1),xy(:,2),'LineWidth',4,'Color',[.6 .6 .6]);
end
```

2. Hough 变换——圆检测

```
%%首先导入像素矩阵 D1,再设置如下参数
clc,clear
load('DATA1.mat')
step_r = 1;
step_angle = 0.1;
r_min = 20;
r_max = 30;
p = 0.7;
% step_r:检测的圆半径步长
% step_angle:角度步长，单位为弧度
% r_min:最小圆半径
% r_max:最大圆半径
% p:阈值，0，1 之间的数

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% output
% hough_space:参数空间，h(a,b,r)表示圆心在(a,b)半径为 r 的圆上的点数
% hough_circl:二值图像，检测到的圆
% para:检测到的圆的圆心、半径
[m,n] = size(D1);
size_r = round((r_max-r_min)/step_r)+1;
size_angle = round(2*pi/step_angle);
hough_space = zeros(m,n,size_r);
[rows,cols] = find(D1);
```

```
ecount = size(rows);
% Hough 变换
%  将图像空间(x,y)对应到参数空间(a,b,r)
% a = x-r*cos(angle)
% b = y-r*sin(angle)
for i=1:ecount
    for r=1:size_r
        for k=1:size_angle
            a = round(rows(i)-(r_min+(r-1)*step_r)*cos(k*step_angle));
            b = round(cols(i)-(r_min+(r-1)*step_r)*sin(k*step_angle));
            if(a>0&&a<=m&&b>0&&b<=n)
                hough_space(a,b,r) = hough_space(a,b,r)+1;
            end
        end
    end
end
%  搜索超过阈值的聚集点
max_para = max(max(max(hough_space)));
index = find(hough_space>=max_para*p);
length = size(index);
hough_circle=zeros(m,n);
for i=1:ecount
    for k=1:length
        par3 = floor(index(k)/(m*n))+1;
        par2 = floor((index(k)-(par3-1)*(m*n))/m)+1;
        par1 = index(k)-(par3-1)*(m*n)-(par2-1)*m;
```

```
if((rows(i)-par1)^2+(cols(i)-par2)^2<(r_min+(par3-1)*step_r)^2+5&&(rows(i)-par1)^2+(cols(i)-par2)^2>(r_min+(par3-1)*step_r)^2-5)
                    hough_circle(rows(i),cols(i)) = 1;
                end
            end
        end
% 处理结果
    for k=1:length
      par3 = floor(index(k)/(m*n))+1;
      par2 = floor((index(k)-(par3-1)*(m*n))/m)+1;
      par1 = index(k)-(par3-1)*(m*n)-(par2-1)*m;
      par3 = r_min+(par3-1)*step_r;
      fprintf(1,'Center %d %d radius %d\n',par1,par2,par3);
      para(:,k) = [par1,par2,par3]';
    end
    %%找出了很多组数据，这些数据可以分成三类

X=para'; %%使用聚类分析得到三个 x y r 信息
[cidx2,cmeans2,sumd2,D2] = kmeans(X,3,'dist','sqEuclidean');
P2 = figure;clf;
[silh2,h2] = silhouette(X,cidx2,'sqeuclidean');
% A 即为三个圆的中心坐标
A=round(cmeans2);
% 绘制圆形
figure,imshow(hough_circle)
```

3. 优化求解：

```
%% 建立模板
% 将实际零件旋转角度
clc
clear
load('DATA1');
subplot(121)
imshow(D1)
I=imrotate(D1,-30);
%将旋转后的矩阵切割成 420*560
D=I(112:531,67:626);
%画出了两张图 一张为旋转前，一张为旋转后
subplot(122)
imshow(D)
%count_row 为每一行的像素计数
for i=1:420
    count_row(i,1)=sum(D(i,:)==1);
end
%找出像素最多的两行 n_row 为行号
[n1,o1]=sort(count_row,'descend');
n_row=o1(1:2,:);
%对于列同理 n_line 为列号 最后得出第 127 和 289
for i=1:560
    count_line(1,i)=sum(D(:,i)==1);
end
[n2,o2]=sort(count_line,'descend');
```

```
n_line=o2(:,1:2);
% 建立模板图形
% 画矩形 矩形的行列号由 rotate 中得出
model_p=zeros(420,560);
model_p(127,218:390)=1;
model_p(289,218:390)=1;
model_p(127:289,218)=1;
model_p(127:289,390)=1;
% 圆的半径及中心点
r=26;
c1=[247,270];
c2=[212,339];
c3=[178,270];
% 420 行，560 列像素点
for i=1:420
    for j=1:560
        point=[i,j];                    % 描述像素点的像素坐标
        l1=norm(c1-point);              % 像素点距离圆心 c1 的距离
        l2=norm(c2-point);              % 像素点距离圆心 c2 的距离
        l3=norm(c3-point);              % 像素点距离圆心 c3 的距离
        % 寻找距离圆心 c1 为 r 的像素点，并使其为 1
        if r-1<=l1 && l1<=r
            model_p(i,j)=1;
        end
        % 寻找距离圆心 c2 为 r 的像素点，并使其为 1
        if r-1<=l2 && l2<=r
```

```
                model_p(i,j)=1;
            end
            % 寻找距离圆心 c3 为 r 的像素点，并使其为 1
            if r-1<=l3 && l3<=r
                model_p(i,j)=1;
            end
        end
end
% 画三个圆
imshow(model_p);
% 求对角线长度
% 左上角
A1(1)=n_line(1);        % 横坐标
A1(2)=n_row(2);         % 纵坐标
% 右下角
B1(1)=n_line(2);        % 横坐标
B1(2)=n_row(1);         % 纵坐标
% 求对角线长
dis=norm(B1-A1);
% 求半对角线长
dis=dis/2;
%% 确定实际零件位置
% D1 是 DATA1 中的数据
[Y,X]=find(D1==1);                  % 实际零件的离散点，Y 表示行号，X 表示列号
%*************零件模板信息***************
```

```
% 通过模板建立过程得到
I=model_p;                          % 导入模板
% imshow(I)                         % 显示模板图像
[y,x]=find(I==1);                   % 模板中的离散点（即白点）
% 已标定模板中心点
tx=304;
ty=212.5;
% 对比模板
figure;
plot(X,Y,'R.')                      % 绘制实际零件图像，显示红色
hold on
plot(x-tx,y-ty,'k.')                % 绘制平移之后的模板图像
% 将模板移动到坐标轴的原点位置，这一步目前看来是有必要的，
% 因为后文乘以旋转矩阵 T 的时候，旋转的基准点是坐标原点
d=[x-tx y-ty];
% 创建一个实际零件离散点列数的随机序列
r=randperm(length(x));
% 创建实际零件的像素坐标点
D=[X Y];
% 用创建的随机序列在 D1 中选取 150 个点
D=D(r(1:150),:);
% 用创建的随机序列在模板 I 中选 150 个点，这部分在论文里可以直接说 D1 里选 150 点
d=d(r(1:150),:);
% 在模板 I 中选了 150 个点，选取的点的数量越多，算法越准，运算速度越慢
% 确定优化边界条件
```

```
l = [-400 -400 -pi];
u = [400    400    pi];
x0 = [100    100    0];
TolFun = 1e-9;
TolX=1e-5;
kmax =5000;
  q =0.8;
  %设定退火算法的各个参数
[xo_sa,fo_sa] =Opt_Simu(@(x)myfun2(x,d,D),x0,l,u,kmax,q,TolFun)
t=xo_sa(3);
a=xo_sa(1);
b=xo_sa(2);
%退火算法得出了 x y theta 三个参数
T=[cos(t) sin(t)
        -sin(t) cos(t)];
%得出旋转矩阵
temp=[X+a Y+b]*T;
X=round(temp(:,1));
Y=round(temp(:,2));
%将 D1 通过得出的 x y theta  变换到模板位置
plot(X,Y,'.')
%做第三个图
```

Myfun2(s,d,D):

```
function f=myfun2(s,d,D)
% s(1) X 平移量
% s(2) Y 平移量
% s(3)  旋转角
a=s(1);
b=s(2);
t=s(3); %在图像中是顺时针旋转
T=[cos(t) sin(t)
        -sin(t) cos(t)];
X=D(:,1);
Y=D(:,2);
D=[X+a Y+b]*T;            %平移加旋转变换
x=d(:,1)';
y=d(:,2)';
X=D(:,1);%更新
Y=D(:,2);
n=length(d);
N=length(D);
X=repmat(X,1,n);
Y=repmat(Y,1,n);
x=repmat(x,N,1);
y=repmat(y,N,1);
DS=(X-x).^2+(Y-y).^2; %所有 D 到所有 d 的距离矩阵
f=sum(min(DS'));
```

Opt_Simu(f,x0,l,u,kmax,q,TolFun):

```
function [xo,fo] = Opt_Simu(f,x0,l,u,kmax,q,TolFun)
% 模拟退火算法求函数 f(x)的最小值点， 且 l <= x <= u
% f 为待求函数，x0 为初值点，l，u 分别为搜索区间的上下限，kmax 为最大迭代次数
% q 为退火因子，TolFun 为函数容许误差
%%%%算法第一步根据输入变量数，将某些量设为缺省值
if nargin < 7
    TolFun = 1e-8;
end
if nargin < 6
    q = 1;
end
if nargin < 5
    kmax = 100;
end
%%%%算法第二步，求解一些基本变量
N = length(x0); %自变量维数
x = x0;
fx = feval(f,x); %函数在初始点 x0 处的函数值
xo = x;
fo = fx;
%%%%%算法第三步，进行迭代计算，找出近似全局最小点
for k =0:kmax
    k;
    Ti = (k/kmax)^q;
```

```
    mu = 10^(Ti*100); %  计算 mu
    dx = Mu_Inv(2*rand(size(x))-1,mu).*(u - l);%步长 dx
    x1 = x + dx; %下一个估计点
    x1 = (x1 < l).*l +(l <= x1).*(x1 <= u).*x1 +(u < x1).*u; %将 x1 限定在区间[l,u]上
    fx1 = feval(f,x1);
    df = fx1- fx;
    if df < 0|rand < exp(-Ti*df/(abs(fx) + eps)/TolFun) %如果 fx1<fx 或者概率大于随机数
z
        x = x1;
        fx = fx1;
    end
    if fx < fo
        xo = x;
        fo = fx1;
    end
end
```

Mu_Inv(y, mu)：

```
%模拟退火法中的 mu^（-1）定理
function x = Mu_Inv(y,mu)
x = (((1+mu).^abs(y)- 1)/mu).*sign(y);
```

问题 1：基于特征提取的位置求解

```
%% **************通过取三个圆的中心点**************
clc,clear
load('DATA1.mat')
load('model.mat')
aaaa=zeros(420,560);%aaaa 是模板
bbbb=aaaa;%bbbb 是样本
%  模板圆心
aaaa(247,270)=1;
aaaa(212,339)=1;
aaaa(178,270)=1;
%  样本圆心
bbbb(183,329)=1;
bbbb(188,253)=1;
bbbb(247,288)=1;
[Y X]=find(bbbb==1);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%零件模板信息
I=aaaa; %导入新模板 model_p2  大了一圈
% imshow(I)                  %%显示原始图像
[y x]=find(I==1); %模板中的离散点
%已标定模板中心点
tx=304;
ty=212.5;
%  对比模板
plot(X,Y,'R.')
```

```
hold on

plot(x-tx,y-ty,'k.')

hold on

d=[x-tx y-ty];

%将模板移动到坐标轴的原点位置，这一步目前看来是有必要的，因为后文乘以旋转矩阵 T 的时候，旋转的基准点是坐标原点

%创建一个随机序列

D=[X Y];

%用创建的随机序列在模板 I 中选 100 个点，这部分在论文里可以直接说 D1 里选 100 个点

%在模板 I 中选了 500 个点，选取的点的数量越多，算法越准，运算速度越慢

l = [-400 -400 -pi];

u = [400    400    pi];

x0 = [100    100    0];

TolFun = 1e-9;

TolX=1e-5;

kmax =50000;

 q =0.8;

 %设定退火算法的各个参数

[xo_sa,fo_sa] =Opt_Simu(@(x)myfun2(x,d,D),x0,l,u,kmax,q,TolFun)

t=xo_sa(3);

a=xo_sa(1);

b=xo_sa(2);

%退火算法得出了 x y theta 三个参数

T=[cos(t) sin(t)

        -sin(t) cos(t)];

%得出旋转矩阵
```

```
temp=[X+a Y+b]*T;
X=round(temp(:,1));
Y=round(temp(:,2));
%将 D1 通过得出的 x y theta 变换到模板位置
plot(X,Y,'.')
%做第三个图
%% ****************绘制原始图形******************
model_pd=D1;
r=125;
p=[204,302];
for i=1:420
    for j=1:560
        point=[i,j];
        l=norm(p-point);
        if l<=r
            model_pd(i,j)=0;
        end
    end
end
%以零件 1 的中心为原点，125 为半径进行擦除
figure,imshow(D1);
% 绘制圆形
hold on
[Y0 X0]=find(D1);
%[Y0 X0]=find(model_pd==1); %擦除后样本中的离散点，此时为零件 1，若使用零件 2，则将所有的 model_pd 改为 model_pd2
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%零件模板信息
I=model_p; %导入新模板 model_p2  大了一圈
% imshow(I)                    %%显示原始图像
[y0 x0]=find(I==1); %模板中的离散点
%已标定模板中心点
tx0=304;
ty0=212.5;
%  对比模板
figure
plot(X0,Y0,'R.')
hold on
plot(x0-tx0,y0-ty0,'k.')
hold on
T=[cos(t) sin(t)
          -sin(t) cos(t)];
temp=[X0+a Y0+b]*T;
%将 D1 通过得出的 x y theta  变换到模板位置
X0=round(temp(:,1));
Y0=round(temp(:,2));
%做第三个图
plot(X0,Y0,'.')
```

问题 2：求解 B 零件位置

PositionBelow：

```
%% ************hough 圆检测,求圆的坐标*****************
%首先导入像素矩阵 D1,再设置如下参数
clc,clear
load('DATA2')
step_r = 1;
step_angle = 0.1;
r_min = 20;
r_max = 30;
p = 0.7;

% step_r:检测的圆半径步长
% step_angle:角度步长，单位为弧度
% r_min:最小圆半径
% r_max:最大圆半径
% p:阈值，0，1 之间的数

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% output
% hough_space:参数空间，h(a,b,r)表示圆心在(a,b)半径为 r 的圆上的点数
% hough_circl:二值图像，检测到的圆
% para:检测到的圆的圆心、半径
[m,n] = size(D2);
size_r = round((r_max-r_min)/step_r)+1;
```

```
size_angle = round(2*pi/step_angle);
hough_space = zeros(m,n,size_r);
[rows,cols] = find(D2);
ecount = size(rows);
% Hough 变换
%  将图像空间(x,y)对应到参数空间(a,b,r)
% a = x-r*cos(angle)
% b = y-r*sin(angle)
for i=1:ecount
    for r=1:size_r
        for k=1:size_angle
            a = round(rows(i)-(r_min+(r-1)*step_r)*cos(k*step_angle));
            b = round(cols(i)-(r_min+(r-1)*step_r)*sin(k*step_angle));
            if(a>0&&a<=m&&b>0&&b<=n)
                hough_space(a,b,r) = hough_space(a,b,r)+1;
            end
        end
    end
end
%  搜索超过阈值的聚集点
max_para = max(max(max(hough_space)));
index = find(hough_space>=max_para*p);
length = size(index);
hough_circle=zeros(m,n);
for i=1:ecount
    for k=1:length
```

```
            par3 = floor(index(k)/(m*n))+1;
            par2 = floor((index(k)-(par3-1)*(m*n))/m)+1;
            par1 = index(k)-(par3-1)*(m*n)-(par2-1)*m;

if((rows(i)-par1)^2+(cols(i)-par2)^2<(r_min+(par3-1)*step_r)^2+5&&(rows(i)-par1)^2+(cols
(i)-par2)^2>(r_min+(par3-1)*step_r)^2-5)
                hough_circle(rows(i),cols(i)) = 1;
            end
        end
end

% 处理结果
     for k=1:length
       par3 = floor(index(k)/(m*n))+1;
       par2 = floor((index(k)-(par3-1)*(m*n))/m)+1;
       par1 = index(k)-(par3-1)*(m*n)-(par2-1)*m;
       par3 = r_min+(par3-1)*step_r;
       fprintf(1,'Center %d %d radius %d\n',par1,par2,par3);
       para(:,k) = [par1,par2,par3]';
     end
     %%找出了很多组数据，这些数据可以分成六类
subplot(121),imshow(D2),title('原图')
subplot(122),imshow(hough_circle),title('检测圆')
X=para'; %%使用聚类分析得到六个 x y r 信息
[cidx2,cmeans2,sumd2,E2] = kmeans(X,6,'dist','sqEuclidean');
figure;clf;
[silh2,h2] = silhouette(X,cidx2,'sqeuclidean');
```

```
A=round(cmeans2);
%%A 即为三个圆的中心坐标
D22=D2;
hough_circle2=hough_circle;
for i=1:6
hough_circle2(A(i,1),A(i,2))=1;
D22(A(i,1),A(i,2))=1;
end
subplot(121),imshow(D22),title('包含圆心的原图')
subplot(122),imshow(hough_circle),title('包含圆心的检测圆')

%% ******************求零件中心坐标******************
figure
%模板中心点
o=[208,304];
%模板中三个圆的圆心位置
c1=[247,270];
c2=[212,339];
c3=[178,270];
%模板中三个圆和中心点的距离
l1=norm(o-c1);
l2=norm(o-c2);
l3=norm(o-c3);
%计算出三个圆的半径值
model_p2=zeros(522,560);
%创建一个空矩阵
```

```
p1=[183,330];
p2=[188,253];
p3=[247,288];
%三个圆的圆心
for i=1:522
    for j=1:560
        point=[i,j];
        s1=norm(p1-point);
        s2=norm(p2-point);
        s3=norm(p3-point);
    if l2-2<=s1&&s1<=l2+2
            model_p2(i,j)=model_p2(i,j)+0.1;
    end
    if l1-2<=s2&&s2<=l1+2
            model_p2(i,j)=model_p2(i,j)+0.1;
    end
    if l3-2<=s3&&s3<=l3+2
            model_p2(i,j)=model_p2(i,j)+0.1;
    end
    end
end
%遍历矩阵，将在每一个圆上的点都自增 0.1，因此自增了 0.3 的点即为所求点
[a1,a2]=find(model_p2>0.2);
a1=mean(a1);
a2=mean(a2);
C1=[a1,a2];
```

```
C1=round(C1);
%求出了中心点的坐标 C1
%重复以上步骤来求第二个零件的中心位置
q1=[375,253];
q2=[452,249];
q3=[418,189];
model_p3=zeros(522,560);
for i=1:522
    for j=1:560
        point=[i,j];
        s1=norm(q1-point);
        s2=norm(q2-point);
        s3=norm(q3-point);
    if l2-2<=s1&&s1<=l2+2
            model_p3(i,j)=model_p3(i,j)+0.1;
    end
    if l1-2<=s2&&s2<=l1+2
            model_p3(i,j)=model_p3(i,j)+0.1;
    end
    if l3-2<=s3&&s3<=l3+2
            model_p3(i,j)=model_p3(i,j)+0.1;
    end
    end
end
%遍历矩阵，将在每一个圆上的点都自增 0.1，因此自增了 0.3 的点即为所求点
[a1,a2]=find(model_p3>0.2);
```

```
a1=mean(a1);
a2=mean(a2);
C2=[a1,a2];
C2=round(C2);
%求出了中心点的坐标 C2
subplot(121),imagesc(model_p2)
subplot(122),imagesc(model_p3)
%作图
%% ****************擦除上面的图形*******************
model_pd2=D2;
r=125;
p=[403,232];
for i=1:522
    for j=1:560
        point=[i,j];
        l=norm(p-point);
        if l<=r
            model_pd2(i,j)=0;
        end
    end
end
figure,imshow(model_pd2);
%以零件 1 的中心为原点，125 为半径进行擦除，最终得到擦除后图像 model_pd2

%% ******************进行最优化求解*****************
[Y X]=find(model_pd2==1); %擦除后样本中的离散点，此时为零件 1，若使用零件 2，则
```

将所有的 model_pd 改为 model_pd2

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%零件模板信息

load('model_new.mat')

I=model_p2;　　%导入新模板 model_p2 大了一圈

%imshow(I)　　%%显示原始图像

[y x]=find(I==1); %模板中的离散点

%已标定模板中心点

tx=304;

ty=212.5;

% 对比模板

figure

plot(X,Y,'R.')

hold on

plot(x-tx,y-ty,'k.')

d=[x-tx y-ty];

%将模板移动到坐标轴的原点位置，这一步目前看来是有必要的，因为后文乘以旋转矩阵 T 的时候，旋转的基准点是坐标原点

% r=randperm(length(X)-200);

% r=randperm(1000);

[row1,col1]=size(X);

r=randperm(row1);

%创建一个随机序列

D=[X Y];

D=D(r(1:300),:);

%用创建的随机序列在 D1 中选取 150 个点

d=d(r(1:500),:);

```
%用创建的随机序列在模板 I 中选 100 个点，这部分在论文里可以直接说 D1 里选 100 个点

%在模板 I 中选了 500 个点，选取的点的数量越多，算法越准，运算速度越慢

l = [-400 -400 -pi];

u = [400    400    pi];

x0 = [100    100    0];

TolFun = 1e-7;

TolX=1e-5;

kmax =50000;

  q =0.8;

  %设定退火算法的各个参数

[xo_sa,fo_sa] =Opt_Simu(@(x)myfun2(x,d,D),x0,l,u,kmax,q,TolFun)

t=xo_sa(3);

a=xo_sa(1);

b=xo_sa(2);

%退火算法得出了 x y theta 三个参数

T=[cos(t) sin(t)

          -sin(t) cos(t)];

%得出旋转矩阵

temp=[X+a Y+b]*T;

X=round(temp(:,1));

Y=round(temp(:,2));

%将 D1 通过得出的 x y theta  变换到模板位置

plot(X,Y,'.')

%做第三个图
```

问题 2：求解 A 零件位置

PositionUp：

```
%% ************hough 圆检测,求圆的坐标*****************
%首先导入像素矩阵 D1,再设置如下参数
clc
clear
load('DATA2')
step_r = 1;
step_angle = 0.1;
r_min = 20;
r_max = 30;
p = 0.7;
% step_r:检测的圆半径步长
% step_angle:角度步长，单位为弧度
% r_min:最小圆半径
% r_max:最大圆半径
% p:阈值，0，1 之间的数
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% output
% hough_space:参数空间，h(a,b,r)表示圆心在(a,b)半径为 r 的圆上的点数
% hough_circl:二值图像，检测到的圆
% para:检测到的圆的圆心、半径
[m,n] = size(D2);
size_r = round((r_max-r_min)/step_r)+1;
size_angle = round(2*pi/step_angle);
```

```
hough_space = zeros(m,n,size_r);
[rows,cols] = find(D2);
ecount = size(rows);
% Hough 变换
%  将图像空间(x,y)对应到参数空间(a,b,r)
% a = x-r*cos(angle)
% b = y-r*sin(angle)
for i=1:ecount
    for r=1:size_r
        for k=1:size_angle
            a = round(rows(i)-(r_min+(r-1)*step_r)*cos(k*step_angle));
            b = round(cols(i)-(r_min+(r-1)*step_r)*sin(k*step_angle));
            if(a>0&&a<=m&&b>0&&b<=n)
                hough_space(a,b,r) = hough_space(a,b,r)+1;
            end
        end
    end
end
%  搜索超过阈值的聚集点
max_para = max(max(max(hough_space)));
index = find(hough_space>=max_para*p);
length = size(index);
hough_circle=zeros(m,n);
for i=1:ecount
    for k=1:length
```

```
            par3 = floor(index(k)/(m*n))+1;
            par2 = floor((index(k)-(par3-1)*(m*n))/m)+1;
            par1 = index(k)-(par3-1)*(m*n)-(par2-1)*m;

if((rows(i)-par1)^2+(cols(i)-par2)^2<(r_min+(par3-1)*step_r)^2+5&&(rows(i)-par1)^2+(cols
(i)-par2)^2>(r_min+(par3-1)*step_r)^2-5)
                hough_circle(rows(i),cols(i)) = 1;
            end
        end
end
%  处理结果
     for k=1:length
       par3 = floor(index(k)/(m*n))+1;
       par2 = floor((index(k)-(par3-1)*(m*n))/m)+1;
       par1 = index(k)-(par3-1)*(m*n)-(par2-1)*m;
       par3 = r_min+(par3-1)*step_r;
       fprintf(1,'Center %d %d radius %d\n',par1,par2,par3);
       para(:,k) = [par1,par2,par3]';
     end
    %%找出了很多组数据，这些数据可以分成六类
subplot(121),imshow(D2),title('原图')
subplot(122),imshow(hough_circle),title('检测圆')
X=para'; %%使用聚类分析得到六个 x y r 信息
[cidx2,cmeans2,sumd2,E2] = kmeans(X,6,'dist','sqEuclidean');
figure;clf;
[silh2,h2] = silhouette(X,cidx2,'sqeuclidean');
A=round(cmeans2);
```

```
%%A 即为三个圆的中心坐标
D22=D2;
hough_circle2=hough_circle;
for i=1:6
hough_circle2(A(i,1),A(i,2))=1;
D22(A(i,1),A(i,2))=1;
end
subplot(121),imshow(D22),title('包含圆心的原图')
subplot(122),imshow(hough_circle),title('包含圆心的检测圆')
%% ******************求零件中心坐标******************
figure
%模板中心点
o=[208,304];
%模板中三个圆的圆心位置
c1=[247,270];
c2=[212,339];
c3=[178,270];
%模板中三个圆和中心点的距离
l1=norm(o-c1);
l2=norm(o-c2);
l3=norm(o-c3);
%计算出三个圆的半径值
model_p2=zeros(522,560);
%创建一个空矩阵
p1=[183,330];
p2=[188,253];
```

```
p3=[247,288];
%三个圆的圆心
for i=1:522
    for j=1:560
        point=[i,j];
        s1=norm(p1-point);
        s2=norm(p2-point);
        s3=norm(p3-point);
    if l2-2<=s1&&s1<=l2+2
            model_p2(i,j)=model_p2(i,j)+0.1;
    end
    if l1-2<=s2&&s2<=l1+2
            model_p2(i,j)=model_p2(i,j)+0.1;
    end
    if l3-2<=s3&&s3<=l3+2
            model_p2(i,j)=model_p2(i,j)+0.1;
    end
    end
end
%遍历矩阵，将在每一个圆上的点都自增 0.1，因此自增了 0.3 的点即为所求点
[a1,a2]=find(model_p2>0.2);
a1=mean(a1);
a2=mean(a2);
C1=[a1,a2];
C1=round(C1);
%求出了中心点的坐标 C1
```

```
%重复以上步骤来求第二个零件的中心位置
q1=[375,253];
q2=[452,249];
q3=[418,189];
model_p3=zeros(522,560);
for i=1:522
    for j=1:560
        point=[i,j];
        s1=norm(q1-point);
        s2=norm(q2-point);
        s3=norm(q3-point);
    if l2-2<=s1&&s1<=l2+2
            model_p3(i,j)=model_p3(i,j)+0.1;
    end
    if l1-2<=s2&&s2<=l1+2
            model_p3(i,j)=model_p3(i,j)+0.1;
    end
    if l3-2<=s3&&s3<=l3+2
            model_p3(i,j)=model_p3(i,j)+0.1;
    end
    end
end
%遍历矩阵，将在每一个圆上的点都自增 0.1，因此自增了 0.3 的点即为所求点
[a1,a2]=find(model_p3>0.2);
```

```
a1=mean(a1);
a2=mean(a2);
C2=[a1,a2];
C2=round(C2);
%求出了中心点的坐标 C2
subplot(121),imagesc(model_p2)
subplot(122),imagesc(model_p3)
%作图
%% *****************擦除下面的零件图形***********************
model_pd=D2;
r=125;
p=[204,302];
for i=1:522
    for j=1:560
        point=[i,j];
        l=norm(p-point);
    if l<=r
            model_pd(i,j)=0;
    end
    end
end
%以零件 1 的中心为原点，125 为半径进行擦除
figure
imshow(model_pd);
%% ******************进行最优化求解*****************
[Y X]=find(model_pd==1); %擦除后样本中的离散点，此时为零件 1，若使用零件 2，则
```

将所有的 model_pd 改为 model_pd2

%%零件模板信息

load('model_new.mat')

I=model_p2;　　%导入新模板 model_p2 大了一圈

%imshow(I)　　%%显示原始图像

[y x]=find(I==1); %模板中的离散点

%已标定模板中心点

tx=304;

ty=212.5;

% 对比模板

figure

plot(X,Y,'R.')

hold on

plot(x-tx,y-ty,'k.')

d=[x-tx y-ty];

%将模板移动到坐标轴的原点位置，这一步目前看来是有必要的，因为后文乘以旋转矩阵 T 的时候，旋转的基准点是坐标原点

%r=randperm(length(X)-200);

r=randperm(1000);

%创建一个随机序列

D=[X Y];

D=D(r(1:200),:);

%用创建的随机序列在 D1 中选取 150 个点

d=d(r(1:500),:);

%用创建的随机序列在模板 I 中选 100 个点，这部分在论文里可以直接说 D1 里选 100

```
个点
%在模板 I 中选了 500 个点，选取的点的数量越多，算法越准，运算速度越慢
l = [-400 -400 -pi];
u = [400    400    pi];
x0 = [100    100    0];
TolFun = 1e-9;
TolX=1e-5;
kmax =5000;
  q =0.8;
  %设定退火算法的各个参数
[xo_sa,fo_sa] =Opt_Simu(@(x)myfun2(x,d,D),x0,l,u,kmax,q,TolFun)
t=xo_sa(3);
a=xo_sa(1);
b=xo_sa(2);
%退火算法得出了 x y theta 三个参数
T=[cos(t) sin(t)
        -sin(t) cos(t)];
%得出旋转矩阵
temp=[X+a Y+b]*T;
X=round(temp(:,1));
Y=round(temp(:,2));
%将 D1 通过得出的 x y theta  变换到模板位置
plot(X,Y,'.')
%做第三个图
```

绘制 B 零件区域：

```
%% ****************擦除上面的图形*********************
load('DATA2')
model_pd2=D2;
r=125;
p=[403,232];
for i=1:522
    for j=1:560
        point=[i,j];
        l=norm(p-point);
%           if l<=r
%               model_pd2(i,j)=0;
%           end
        if l>=r && l<=r+1
            model_pd2(i,j)=0.5;
         end
    end
end
figure
imshow(model_pd2);
%以零件 1 的中心为原点，125 为半径进行擦除，最终得到擦除后图像 model_pd2
```

绘制 A 零件区域：

```
%% ****************擦除下面的零件图形**********************
model_pd=D2;
r=125;
p=[204,302];
for i=1:522
    for j=1:560
        point=[i,j];
        l=norm(p-point);
%           if l<=r
%               model_pd(i,j)=0;
%           end
        if l>=r && l<=r+1
            model_pd(i,j)=1;
         end
    end
end
%以零件 1 的中心为原点，125 为半径进行擦除
figure
imshow(model_pd);
```

问题 2——基于零件特征提取的方法：

求 B 零件区域

```
%% **************通过取三个圆的中心点**************
clc,clear
load('DATA2.mat')
load('model_new.mat')
aaaa=zeros(522,560);%aaaa 是模板
bbbb=aaaa;%bbbb 是样本
%  模板圆心
aaaa(247,270)=1;
aaaa(212,339)=1;
aaaa(178,270)=1;
%  样本圆心
bbbb(183,330)=1;
bbbb(188,253)=1;
bbbb(247,288)=1;


%擦除后样本中的离散点，此时为零件 1，若使用零件 2，则将所有的 model_pd 改为 model_pd2
[Y X]=find(bbbb==1);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%零件模板信息
I=aaaa; %导入新模板 model_p2  大了一圈
% imshow(I)                  %%显示原始图像
[y x]=find(I==1); %模板中的离散点
%已标定模板中心点
```

```
tx=304;

ty=212.5;

% 对比模板

plot(X,Y,'R.')

hold on

plot(x-tx,y-ty,'k.')

hold on

d=[x-tx y-ty];

%将模板移动到坐标轴的原点位置，这一步目前看来是有必要的，因为后文乘以旋转矩阵 T 的时候，旋转的基准点是坐标原点

%创建一个随机序列

D=[X Y];

%用创建的随机序列在模板 I 中选 100 个点，这部分在论文里可以直接说 D1 里选 100 个点

%在模板 I 中选了 500 个点，选取的点的数量越多，算法越准，运算速度越慢

l = [-400 -400 -pi];

u = [400    400    pi];

x0 = [100    100    0];

TolFun = 1e-9;

TolX=1e-5;

kmax =50000;

 q =0.8;

 %设定退火算法的各个参数

[xo_sa,fo_sa] =Opt_Simu(@(x)myfun2(x,d,D),x0,l,u,kmax,q,TolFun)

t=xo_sa(3);

a=xo_sa(1);

b=xo_sa(2);
```

```
%退火算法得出了 x y theta 三个参数
T=[cos(t) sin(t)
        -sin(t) cos(t)];
%得出旋转矩阵
temp=[X+a Y+b]*T;
X=round(temp(:,1));
Y=round(temp(:,2));
%将 D1 通过得出的 x y theta  变换到模板位置
plot(X,Y,'.')
%做第三个图
%% ****************绘制原始图形*******************
model_pd2=D2;
r=125;
p=[403,232];
for i=1:522
    for j=1:560
        point=[i,j];
        l=norm(p-point);
    if l<=r
            model_pd2(i,j)=0;
    end
    end
end
figure,imshow(model_pd2);
%以零件 1 的中心为原点，125 为半径进行擦除，最终得到擦除后图像 model_pd2
```

```
hold on

[Y0 X0]=find(model_pd2==1); %擦除后样本中的离散点，此时为零件 1，若使用零件 2，则将所有的 model_pd 改为 model_pd2

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%零件模板信息

I=model_p2; %导入新模板 model_p2  大了一圈

% imshow(I)                  %%显示原始图像

[y0 x0]=find(I==1); %模板中的离散点

%已标定模板中心点

tx0=304;

ty0=212.5;

%  对比模板

figure

plot(X0,Y0,'R.')

hold on

plot(x0-tx0,y0-ty0,'k.')

hold on

T=[cos(t) sin(t)

        -sin(t) cos(t)];

temp=[X0+a Y0+b]*T;

X0=round(temp(:,1));

Y0=round(temp(:,2));

%将 D1 通过得出的 x y theta  变换到模板位置

plot(X0,Y0,'.')

%做第三个图
```

求 A 零件区域：

```
%% *************通过取三个圆的中心点**************
clc,clear
load('DATA2.mat')
load('model_new.mat')
aaaa=zeros(522,560);%aaaa 是模板
bbbb=aaaa;%bbbb 是样本
%  模板圆心
aaaa(247,270)=1;
aaaa(212,339)=1;
aaaa(178,270)=1;
%  样本圆心
bbbb(375,253)=1;
bbbb(452,249)=1;
bbbb(418,189)=1;
%擦除后样本中的离散点，此时为零件 1，若使用零件 2，则将所有的 model_pd 改为
model_pd2
[Y X]=find(bbbb==1);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%零件模板信息
I=aaaa; %导入新模板 model_p2  大了一圈
% imshow(I)                  %%显示原始图像
[y x]=find(I==1); %模板中的离散点
%已标定模板中心点
tx=304;
ty=212.5;
```

```
% 对比模板

plot(X,Y,'R.')

hold on

plot(x-tx,y-ty,'k.')

hold on

d=[x-tx y-ty];

%将模板移动到坐标轴的原点位置，这一步目前看来是有必要的，因为后文乘以旋转矩阵 T 的时候，旋转的基准点是坐标原点


%创建一个随机序列

D=[X Y];

%用创建的随机序列在模板 I 中选 100 个点，这部分在论文里可以直接说 D1 里选 100 个点

%在模板 I 中选了 500 个点，选取的点的数量越多，算法越准，运算速度越慢

l = [-400 -400 -pi];

u = [400    400    pi];

x0 = [100    100    0];

TolFun = 1e-9;

TolX=1e-5;

kmax =50000;

 q =0.8;

 %设定退火算法的各个参数

[xo_sa,fo_sa] =Opt_Simu(@(x)myfun2(x,d,D),x0,l,u,kmax,q,TolFun)

t=xo_sa(3);

a=xo_sa(1);

b=xo_sa(2);

%退火算法得出了 x y theta 三个参数
```

```
T=[cos(t) sin(t)
        -sin(t) cos(t)];
%得出旋转矩阵
temp=[X+a Y+b]*T;
X=round(temp(:,1));
Y=round(temp(:,2));
%将 D1 通过得出的 x y theta  变换到模板位置
plot(X,Y,'.')
%做第三个图


%% ****************绘制原始图形******************
model_pd=D2;
r=125;
p=[204,302];
for i=1:522
    for j=1:560
        point=[i,j];
        l=norm(p-point);
    if l<=r
            model_pd(i,j)=0;
    end
    end
end
%以零件 1 的中心为原点，125 为半径进行擦除
figure,imshow(model_pd);
```

```
% 绘制圆形
hold on
[Y0 X0]=find(model_pd==1); %擦除后样本中的离散点，此时为零件 1，若使用零件 2，则将所有的 model_pd 改为 model_pd2
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%零件模板信息
I=model_p2; %导入新模板 model_p2 大了一圈
% imshow(I)                %%显示原始图像
[y0 x0]=find(I==1); %模板中的离散点
%已标定模板中心点
tx0=304;
ty0=212.5;
% 对比模板
figure
plot(X0,Y0,'R.')
hold on
plot(x0-tx0,y0-ty0,'k.')
hold on
T=[cos(t) sin(t)
        -sin(t) cos(t)];
temp=[X0+a Y0+b]*T;
%将 D1 通过得出的 x y theta 变换到模板位置
X0=round(temp(:,1));
Y0=round(temp(:,2));
%做第三个图
plot(X0,Y0,'.')
```