基于误差控制的点阵图形矢量化方法研究

栗全庆
(郑州航空工业管理学院工业工程系, 郑州 450005)

摘 要 点阵图形的矢量化是计算机图形学的经典问题,最小二乘法是点阵图矢量化的主要方法,它用拟合误差是否在阈值之内来判别矢量化结果的正确性,因而阈值的确定是关键.本文分析了用最小二乘法将理想点阵图形识别为矢量图的误差,科学地确定了阈值,提出了评价识别结果的判别式.为自动实现矢量化,还提出了点阵图形矢量化的新方法 —— 双向滚动最小二乘识别法,导出了递推公式.本文给出的评判式科学准确,提出的矢量化方法快捷高效,具有通用性。

关键词 点阵图形 矢量化 特征识别 最小二乘法
中图法分类号: TP391 **文章标识码**: A **文章编号**: 1006-8961(2002)10-1048-06

A Study on the Vectorization of Dot Matrix Image Based on Error Controlling

LI Quan-qing
(*Department of Industrial Engineering, Zhengzhou Institute of Aeronautical Industry Management, Zhengzhou* 450005)

Abstract The vecorization of dot matrix image is a classical question of computer graphics. The least-square algorithm, which distinguishes the result of vectorization right or not by defining whether the fitting errors is in the range of threshold or not, is a main method of vectorization of dot matrix image. So how to define the threshold is a key. These errors are analyzed while dot matrix image is recognized as vector diagram by the least-square algorithm, the threshold is defined scientifically, and the judgement formulas to evaluate recognition resul are put forward to in this thesis. In order to complete vectorization automatically, a new method of the vectorization of dot matrix image —— the double direction roll least-square algorithm is presented, and some recurence formulas are deduced. The judgement formulas are scientific and accurate, the mothod of the vectorization is rapid and efficient. The formulas and the method are all quite well and universal.

Keywords Dot matrix image, The vectorization, Feature recognition, Least-square algorithm

0 引 言

点阵图形的矢量化不但是计算机图形学的经典问题,而且在机械制造领域也得到了深入的研究.例如,工程图纸扫描图象的识别、数控加工图形自动编程、反求工程基于层析图形的三维重构等,都需进行点阵图形的矢量化,且其研究对各自的系统都有着举足轻重的作用.在机械制造领域,又把点阵图形矢量化称为点阵图形的特征线段识别.

在点阵图形的自动矢量化中,需解决的问题是:选择合适的点群,将其拟合为合适的线段.点阵图形矢量化的算法已有许多[1~3],有的是先采用计算近似曲率、点与邻近点连线的夹角等方式来确定特征点(拐点),以选择点群;有的是用识别圆模式的方法确定点群,然后再将点群识别为特征线段[4~7],这种方法称为间接识别法.而在将点群识别为特征线段时,一般是采用拟合误差来判断识别的正确性,即首先给定一阈值,然后再将选定的点群拟合为直线或圆弧,并计算误差,若误差在阈值内,则识别成功;反

收稿日期:2001-02-05;改回日期:2002-01-09

之则失败. 由此可知，阈值的科学选取，是保证识别结果正确的关键. 但目前阈值的选取仍是靠经验[8].

1　点阵图形矢量化的误差分析

1.1　最小二乘法的基本思想

目前在工程中应用的点阵图形矢量化，基本上都是将点群拟合为直线和圆弧，并且大部分采用最小二乘法[1,2,9].

1.1.1　圆弧最小二乘拟合的误差

已知点群 $P_i(x_i,y_i)(i=1,2,\cdots,n)$，用最小二乘法将其拟合为圆(圆弧)$(x_i-x_0)^2+(y_i-y_0)^2-R^2=0$($(x_0,y_0)$为圆心，$R$ 为半径)时，有

$$x_0=\frac{\alpha}{\Delta},y_0=\frac{\beta}{\Delta},x_0^2+y_0^2-R^2=\frac{\gamma}{\Delta} \tag{1}$$

其中

$$\Delta=\begin{vmatrix}A_1 & B_1 & C_1\\ A_2 & B_2 & C_2\\ A_3 & B_3 & C_3\end{vmatrix},\alpha=\begin{vmatrix}D_1 & B_1 & C_1\\ D_2 & B_2 & C_2\\ D_3 & B_3 & C_3\end{vmatrix}$$
$$\beta=\begin{vmatrix}A_1 & D_1 & C_1\\ A_2 & D_2 & C_2\\ A_3 & D_3 & C_3\end{vmatrix},\gamma=\begin{vmatrix}A_1 & B_1 & D_1\\ A_2 & B_2 & D_2\\ A_3 & B_3 & D_3\end{vmatrix} \tag{2}$$

式中，$A_i,B_i,C_i,D_i(i=1,2,3)$为方程系数.

点群拟合所产生的误差为

$$E=\sum_{i=1}^{n}[(x_i-x_0)^2+(y_i-y_0)^2-R^2]^2 \tag{3}$$

1.1.2　直线最小二乘拟合的误差

已知点群 $P_i(x_i,y_i)(i=1,2,\cdots,n)$，用最小二乘法将其拟合为直线 $y_i=kx_i+b$(k 为斜率，b 为截距)时，有

$$k=\frac{\alpha}{\Delta},b=\frac{\beta}{\Delta} \tag{4}$$

其中

$$\Delta=\begin{vmatrix}A_1 & B_1\\ A_2 & B_2\end{vmatrix},\alpha=\begin{vmatrix}D_1 & B_1\\ D_2 & B_2\end{vmatrix},\beta=\begin{vmatrix}A_1 & D_1\\ A_2 & D_2\end{vmatrix} \tag{5}$$

式中，$A_i,B_i,C_i,D_i(i=1,2,3)$为方程系数.

点群拟合所产生的误差为

$$E=\sum_{i=1}^{n}[kx_i-y_i+b]^2 \tag{6}$$

直线几乎平行于 y 轴时，应采用直线方程：$x_i=ky_i+b$，并计算各值.

1.2　拟合误差的定义

将已知点群 $P_i(x_i,y_i)$拟合为某种特征线段，虽然也可以用式(1)、式(2)或式(4)、式(5)求出相应的各参数(x_0,y_0)、R、k、b. 但拟合结果未必正确. 例如，如果将本来位于一条直线上的点群 P_i 拟合为圆弧(一般来讲，拟合工作是可以进行的)，则结果就是错误的. 这时式(3)决定的 E 值就特别大，因此，E 值的大小就成为判断拟合结果是否正确的依据. 但是，E 值是绝对误差，且 E 值随圆半径 R 的增大(参加拟合的点数增多)而增大. 显然，用其作为判断拟合结果的依据是不合适的. 在直线拟合时，也有这个问题. 这就使得用 E 值无法判断点群的选择是否合理，因而也无法判断拟合结果是否正确. 因此，为了能描述误差的相对性，且考虑到 E 是误差的平方和，定义特征线段的拟合误差为点均方根误差

$$e=\frac{\sqrt{E}}{n} \tag{7}$$

式中，n 是被拟合点群的点数. E 分别由式(3)或式(6)确定.

1.3　拟合阈值的确定

1.3.1　圆弧拟合阈值的确定

如果一组连续分布的点可以拟合为圆弧，那么它们所组成的图形与几何参数相同的圆弧的理想点阵图形是一样的，所以，可以从分析理想点阵图形圆的拟合误差来分析所选点群的拟合结果. 显然，在拟合中，应尽可能使本来属于同一特征线段的点选为一个点群，并进行该特征线段的拟合，如把位于同一圆弧的所有点组成点群去拟合圆弧. 但实际上，点阵图形圆是由长度很小的不等长的直线来组成的，而直线的长度随圆弧半径的不同而变化，即对同一半径的圆弧，因其位置不同，组成圆弧的直线段长度也不同. 如图 1 所示，在同一方位，组成大圆弧的直线段就较长(A 与 B 比较)；而在同一圆弧上，位于垂直和水平位置的部分直线段就较长(A 与 C 比较). 也就是说，如

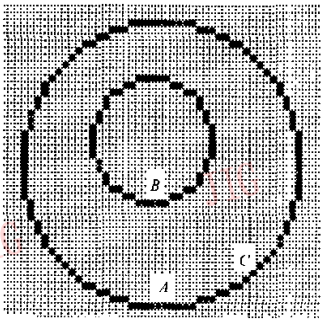


图 1　圆的点阵图形

果圆弧长度和半径不同,拟合误差 e 就不同.

为分析半径和弧长对拟合误差的影响,首先,用 Windows 的画图程序画出若干个半径 R 不等的圆.这些圆是没有误差的理想点阵图形圆,求得每一圆上各像素点以像素为单位表示的坐标值;然后分别取出每一圆的全部点或不同数目的点组成点群,用最小二乘方法将其拟合为矢量圆,即用式(1)~(3)求出圆心(x_0,y_0)、半径 R 和误差 E,再用式(7)计算拟合误差 e.R,e 值如表 1 所示.

表 1 理想圆弧识别为圆弧的误差

点数	R	e	R	e	R	e	R	e
全部	254	3.47	122	2.14	55	1.58	19	0.85
20	11	4 010.53	18	2 310.12	29	198.17	18	3.81
40	106	2 442	92	330.33	54	7.19	19	0.96
60	192	1 025	116	61.31	54	7.99	19	0.85
100	239	250	121	4.07	55	2.32	19	0.85
140	251	46.16	122	4.25	55	1.58	19	0.85
180	253	18.86	122	3.97	55	1.59	19	0.85

表中第 1 行是取理想位图的全部点而计算出的半径和拟合误差 e,是理想圆的真实半径和最小拟合误差.

从表中可知,圆弧的拟合误差及所取点群的点数目与圆弧的半径有关.观察各列,当所取点数增加时,e 减小,R 趋于真实半径;观察各行,当 R 减小时,e 趋于最小拟合误差.为防止把直线识别为圆弧,也分析了将直线识别为圆弧的拟合误差.结论是,如果直线可以硬性"识别"为圆弧,则由式(7)计算出的拟合误差 $e>60$.考虑到各种情况,根据拟合误差 e 而确定的圆弧识别阈值应该为

$$\varepsilon \leq \begin{cases} 1.0 & R \leq 20 \quad n \geq 40 \\ 2.0 & R \leq 60 \quad n \geq 60 \\ 5.0 & R \leq 125 \quad n \geq 100 \\ 20.0 & R > 125 \quad n > 180 \end{cases} \tag{8}$$

该式也表明,对半径较大,但弧长较短(可以取出的点数少)的圆弧,是无法将其正确识别为一段圆弧的,因为从表 1 中可以看出,此时的 e 将远大于最小拟合误差.

1.3.2 直线拟合阈值的确定

采用以上的方法分析理想直线拟合为直线的误差,即用 Windows 的画图程序画出斜率不同的若干条理想直线,分别取不同数目的点,组成点群,用最小二乘方法来拟合矢量直线,即用式(4)~式(6)计算参数和误差,用式(7)计算拟合误差.水平、垂直和 45°斜直线的拟合误差不因取点数目和位置的变化而变化,其他可以拟合为直线的连续点的数目和位置(直线斜率)对拟合误差则有影响,但影响较小,故可得直线拟合的判别式为

$$\varepsilon \leq 0.08 \tag{9}$$

2 基于误差控制的特征线段识别

式(8)、式(9)已解决了可否将选定点群识别为特征线段的问题,下面来讨论在自动识别时,如何确定点群的问题.

工程中被识别的点阵图形,是零件的边界轮廓图(不考虑工程图纸识别时的字符、尺寸线、引出线等).点阵图形矢量化就是识别出图中的特征线段.目前的研究多限于识别出直线和圆弧,为此,提出一种新的识别直线和圆弧的方法——双向滚动识别算法.

设点阵图形被识别的边界轮廓已被分离为互不关联的轮廓图形,对每一轮廓图按某种顺序取出各点组成为环链 PtList.对点阵图形的识别,就是对每一环链 PtList 的识别.如果 PtList 不能从整体上识别为圆,就识别其中的直线和圆弧.为了避免将组成圆弧的短直线识别为直线,每轮识别时,总是先识别圆弧,失败后才进行直线识别.

2.1 双向滚动识别的基本算法

2.1.1 基链的滚动构造

双向滚动识别的基本思路是:

(1) 从 PtList 中取出连续的一组点(如果可能,点数为 140 点)组成基段 Glist.

(2) 如果 Glist 的长度(点数)大于 20,转入对基段 Glist 的圆弧识别流程,若成功,回到第 1 步,开始下轮识别;若失败,缩短 Glist,再回到第 2 步.

(3) 如果 Glist 的长度不大于 20,转入对基段 Glist 的直线识别流程.直线识别一定可以成功.

(4) 回到第 1 步,开始下轮识别,直到 PtList 链尾.

第 1 轮识别构造基段 Glist 时,不取 PtList 中的全部点(因为 PtList 不能整体识别为圆).每次从 PtList 中取点时,若不足一组,则取出其全部点.构造 Glist 时,从 PtList 首端取点;而缩短 Glist 时,从其尾部取点,并将这些点联入 PtList 的首端,详细流程如图 2 所示.其中,A、B 分别为对基段 Glist 的圆弧、直线识别流程.

2.1.2 圆弧的滚动识别

对构造出的基段 Glist,如其长度大于 20,则采用双向滚动识别方法来拟合圆弧.基本思路为:(1)如果 Glist 不能识别为圆弧(拟合误差 e 大于阈值 ε),

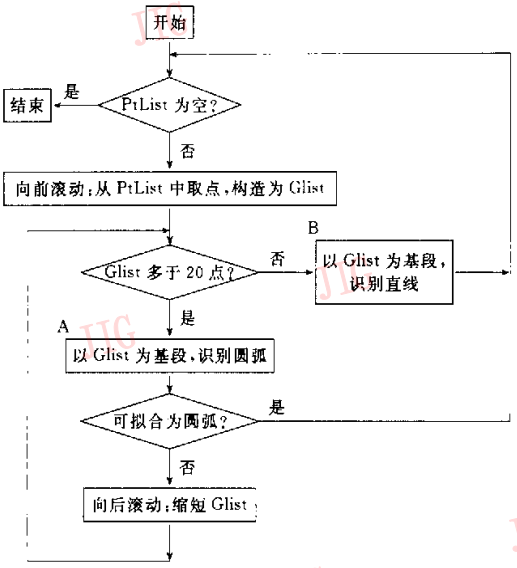


图 2　滚动识别的流程

返回失败,否则:(2)向前滚动,即每次将 Glist 增加一组点,并识别其为圆弧,直到失败;然后,再向后滚动:每次从 Glist 尾部减少一点,并识别圆弧,直到成功.返回成功.详细流程见图 3.图 3 是图 2 中框图 A 的流程.

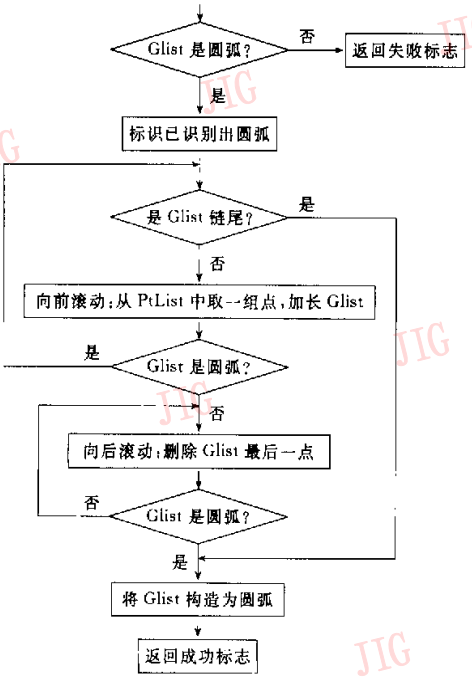


图 3　将 Glist 滚动识别为圆弧的流程

2.1.3　直线的滚动识别

如果对基段 Glist 的圆弧识别失败,则对其进行直线识别.基本思路为:依次加长 Glist,并识别为直线,直到识别失败;依次缩短 Glist,直到识别成功或 Glist 的长度小于某一值(如 6).当其长度小于 6 点时,指定其为由首尾 2 点确定的直线,返回成功.详细流程见图 4.它是图 2 中 B 框的流程.

识别时,到达 PtList 链尾,就结束了一条边界轮廓环链的识别.对其他环链进行识别,即可完成对整个点阵图形的矢量化.

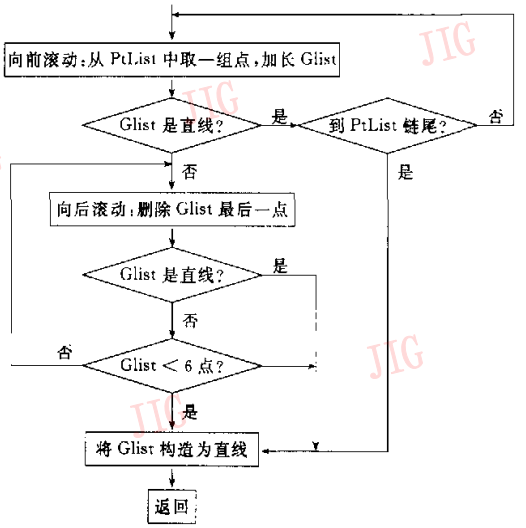


图 4　将 Glist 滚动识别为直线的流程

2.2　直线最小二乘滚动识别的递推算法

在式(2)、式(5)中,系数矩阵 $\boldsymbol{A}_i$、$\boldsymbol{B}_i$、$\boldsymbol{C}_i$、$\boldsymbol{D}_i$($i=1,2$ 或 $i=1,2,3$)为仅与点群 P_i 有关的系数.显然,在计算几何参数时,若每向前滚动一组或向后滚动一点,均从第 1 点到最后一点来计算各系数,这显然是不合理的,故需推导滚动识别时的递推算法.

2.2.1　直线的向前滚动递推算法

为了推导递推公式,设第 J 次滚动以后,点链 Glist 中有 $m=Js$ 个点,其中,s 为步长,即每次增加的点数,令其对应的 A_1 系数为 $A_1^{(J)}$.对第 $J+1$ 次滚动而言,点链 Glist 中的点数增加至 n 个,其中,后 $n-1$ 个点($m+1\sim n$)是当前 Glist 中新增加的点,它们引起了各系数的变化.其中,A_1 系数的变动量为 $\sum_{i=m+1}^{n} x_i^2$.规定:$A_1^{(0)}=0$,则可得向前滚动时,系数 A_1 的递推公式为

$$A_1^{(J+1)}=A_1^{(J)}+\sum_{i=Js+1}^{n}x_i^2$$

类似地,可得第 J 次向前滚动其他系数的递推公式为

$$A_2^{(0)}=0,A_2^{(J+1)}=A_2^{(J)}+\sum_{i=Js+1}^{n}x_i$$

$$B_1^{(J+1)}=A_2^{(J+1)},B_2^{(J+1)}=n$$

$$D_1^{(0)}=0,D_1^{(J+1)}=D_1^{(J)}+\sum_{i=Js+1}^{n}x_iy_i$$

$$D_2^{(0)}=0,D_2^{(J+1)}=D_2^{(J)}+\sum_{i=Js+1}^{n}y_i$$

以上各式中,均有 $J=0,1,2,\cdots;n\leq(J+1)s$,$n$ 为 Glist 的点数,若新增加的点数为 s 时,取等号,若不足 s 时,取实际值. 计算出第 J 次向前滚动的各系数以后,可计算出对应的 Δ、α 和 β,并计算出此时的直线参数 k 和 b. 然后用式(2)计算 E 值,并用式(4)判断拟合结果.

2.2.2 直线的向后滚动递推算法

设向后滚动的次数用 K 表示. 第 K 次滚动对应的 A_1 的系数用 $A_1^{(K)}$ 表示. 当 $K=0$(即向后滚动尚未开始)时,系数为第 $J+1$ 次向前滚动后得到的值,即

$$A_1^{(K=0)}=A_1^{(J+1)}$$

向后滚动的步长为 $s=1$(即每次删减一点). 第 K 次向后滚动结束以后,设 Glist 中有 n 个点,第 $K+1$次滚动以后,Glist 的第 n 点被删除,它引起的 A_1 的变化为 x_n^2. 故系数 A_1 向后滚动的递推公式为

$$A_1^{(K+1)}=A_1^{(K)}-x_n^2$$

同理,其他系数的递推公式为

$$A_2^{(K=0)}=A_2^{(J+1)},A_2^{(K+1)}=A_1^{(K)}-x_n$$

$$B_1^{(K=1)}=A_2^{(K+1)},B_2^{(K+1)}=n-1$$

$$D_1^{(K=0)}=D_1^{(J+1)},D_1^{(K+1)}=D_1^{(K)}-x_ny_n$$

$$D_2^{(K=0)}=D_2^{(J+1)},D_2^{(K+1)}=D_2^{(K)}-y_n$$

2.3 圆弧最小二乘滚动识别的递推算法

和直线的滚动识别相似,推导圆弧的递推公式.

2.3.1 圆弧的向前滚动

圆弧向前滚动时,各系数的递推公式为

$$A_1^{(0)}=0,A_1^{(J+1)}=A_1^{(J)}+2\sum_{i=Js+1}^{n}x_i^2$$

$$A_2^{(0)}=0,A_2^{(J+1)}=A_2^{(J)}+2\sum_{i=Js+1}^{n}x_iy_i$$

$$A_3^{(0)}=0,A_3^{(J+1)}=A_3^{(J)}+2\sum_{i=Js+1}^{n}x_i$$

$$B_1^{(J+1)}=A_2^{(J+1)}$$

$$B_2^{(0)}=0,B_2^{(J+1)}=B_2^{(J)}+2\sum_{i=Js+1}^{n}y_i^2$$

$$B_3^{(0)}=0,B_3^{(J+1)}=B_3^{(J)}+2\sum_{i=Js+1}^{n}y_i$$

$$C_1^{(J+1)}=-A_3^{(J+1)}/2$$

$$C_2^{(J+1)}=-B_3^{(J+1)}/2,C_3^{(J+1)}=-n$$

$$D_1^{(0)}=0,D_1^{(J+1)}=D_1^{(J)}+\sum_{i=Js+1}^{(J+1)s}(x_i^3+x_iy_i^2)$$

$$D_2^{(0)}=0,D_2^{(J+1)}=D_2^{(J)}+\sum_{i=Js+1}^{(J+1)s}(x_i^2y_i+y_i^3)$$

$$D_3^{(J+1)}=(A_1^{(J+1)}+B_2^{(J+1)})/2$$

各式中,$J=0,1,2,\cdots;n$ 为 Glist 的点数,$n\leq(J+1)s$. 若新增加的点数为 s 时,取等号,若不足 s 时,取实际值.

2.3.2 圆弧的向后滚动

圆弧向后滚动的递推公式为

$$A_1^{(K=0)}=A_1^{(J+1)},A_1^{(K+1)}=A_1^{(K)}-2x_n^2$$

$$A_2^{(K=0)}=A_2^{(J+1)},A_2^{(K+1)}=A_2^{(K)}-2x_ny_n$$

$$A_3^{(K=0)}=A_3^{(J+1)},A_3^{(K+1)}=A_3^{(K)}-2x_n$$

$$B_1^{(K+1)}=A_2^{(K+1)}$$

$$B_2^{(K=0)}=B_2^{(J+1)},B_2^{(K+1)}=B_2^{(K)}-2y_n^2$$

$$B_3^{(K=0)}=B_3^{(J+1)},B_3^{(K+1)}=B_3^{(K)}-2y_n$$

$$C_1^{(K+1)}=-A_3^{(K+1)}/2$$

$$C_2^{(K+1)}=-B_3^{(K+1)}/2,C_3^{(K+1)}=1-n$$

$$D_1^{(K=0)}=D_1^{(J+1)},D_1^{(K+1)}=D_1^{(K)}-(x_n^3+x_ny_n^2)$$

$$D_2^{(K=0)}=D_2^{(J+1)},D_2^{(K+1)}=D_2^{(K)}-(x_n^2y_n+y_n^3)$$

$$D_3^{(K+1)}=(A_1^{(K+1)}+B_2^{(K+1)})/2$$

2.4 最小二乘滚动识别方法的特点

对点阵图形——圆和圆弧的识别是一难题[7]. 由于点阵图形的圆弧是由不等长的短小直线所组成,故很难用传统的差分法求出图形的真正拐点,而用间接法需两次遍历边界轮廓,因而计算量大、速度低. 最小二乘滚动识别方法基本上一次遍历就可实现矢量化(是一种直接法),故计算量小、效率高、速度快. 对同一组数据用两种方法在同一台计算机上进行了对比试验,当识别精度基本相当时,运算时间的比较如表 2 所示.

表 2 两种识别方法的速度对比

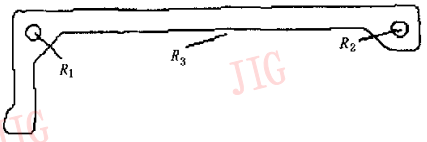
	时间(ms)	时间比
间接法	2570	1
直接法	548	0.21

该组实验表明,用双向滚动递推方法,速度可提

高 80%左右.

3 实　例

图 5(a)所示为某真实零件的层析图. 在对其进行矢量化之前，已进行了数据处理和轮廓分离，形成了 R_1、R_2 和 R_3，它们互不关联，边界点已按一定顺序依次排列，且为单像素. 识别时，首先将轮廓的所有点放入 PtList，并从整体上识别为圆，用式(8)判别结果. R_1 和 R_2 可识别为圆，而 R_3 不行. 对 R_3，按前述的方法识别其中的直线和圆弧. 图 5(b)为识别结果. 其中，C_1 和 C_2 为圆，ab、cd、ef 和 gh 各段为圆弧，bc、de、fg 和 ha 各段为直线(每段含一条或多条直线).



(a) 某零件的层析点阵图形

(b) 矢量化结果

图 5　点阵图形矢量化实例

4 结　论

在分析理想点阵图形矢量化误差的基础上，提出了评判特征线段识别结果的判别式(式(8)和式(9))，其是科学评判点群特征线段拟合结果的根据. 由于该式是从分析特征线段的计算机显示机理而提出的，故该式及双向滚动最小二乘识别法具有通用性，其不仅解决了单点点阵轮廓的矢量化问题，且具有精度高、速度快的优点，为点阵图形矢量化提供了一种新方法.

参 考 文 献

1　方卫宁，梁锡昌. 机械图纸的自动处理与识别[J]. 机械工程学报，1995，31(1)：8～14.

2　朱林，常明，李小涛. 工程图纸中的图素识别方法[J]. 华中理工大学学报，1995，23(2)：120～124.

3　李冬青，冯雷，徐庆鸿等. 数控加工自动编程中的边缘矢量化技术[J]. 计算机辅助设计与图形学学报，1998，10(6)：516～519.

4　高玮，吴中奇，童红卫. 工程图纸轮廓线自动跟踪新方法[J]. 计算机应用与软件，1996，6：49～52.

5　郭丙炎，常明，朱林等. 工程图形扫描后的智能识别方法[J]. 中国机械工程，1992，3(6)：5～7.

6　李冬青，冯雷，徐庆鸿等. 数控加工自动编程中的边缘矢量化技术[J]. 计算机辅助设计与图形学学报，1998，10(6)：516～519.

7　李伟青，彭群生. 一种基于模式的圆的识别算法[J]. 软件学报，1999，10(2)：129～132.

8　沈力，张晨曦. 黑白图象的矢量化[J]. 计算机辅助设计与图形学学报，2000，12(3)：170～173.

9　谢红. 基于 ICT 的 CAD 建模技术研究[D]. 西安：西北工业大学 CAD 研究所，1997.

栗全庆　1956 年生，1982 年获洛阳工学院机械制造工艺及其设备专业工学学士学位，1993 年获洛阳工学院机械制造工艺及其设备专业工学硕士学位，1999 年获西安交通大学机械工程专业工学博士学位，现为郑州航空工业管理学院工业工程系副教授. 主要从事 CAD/CAM、反求工程、现代生产管理等方面的研究.