

## 摘 要

数字地理信息是地理信息系统的基础,数字地理信息的获取是地理信息系统建立发展的关键技术之一。在地图扫描图像上识别和提取数字地理信息因其效率、精度、自动化程度和可靠性方面的显著优势,已经成为获取数字地理信息的主要途径之一。

本文在研究、分析现有扫描图像矢量化方法的基础上,与地图扫描二值图像的特点相结合,设计了一种基于图像细化算法的地图信息识别与提取的方法,并以此为理论指导在 Visual C++ 环境下编程实现了一个自动矢量化实验系统。

本文采用比较流行基于细化的矢量化方法作为整体设计思想。细化地图扫描二值图像后,对提取出来的骨架线进行识别。其中细化算法直接影响到识别的效果,本文提出的  $3 \times 3$  模板与  $4 \times 4$  模板相结合的并行细化方法,成功地保留原图的拓扑结构。图形的识别和提取是矢量化工作中的重点,文本将矢量外积与多边形识别中的最小角法则相结合,设计并实现了一种更高速的多边形提取方法。此外,还成功运用了矢量跟踪算法、直线分类识别方法、噪声滤除技术以及对二值图像的一些基本操作。所实现的实验系统能够较好的完成地图扫描图像中地理对象的自动识别。

**关键字:** 地理信息系统;数字地理信息;矢量化;细化;拓扑结构

## Abstract

Digital geographic information is the basis of Geography Information System (GIS), and to capture the information is one the keys to build and develop GIS. To recognize and capture digital geographic information in the scanned maps images has transcendent advantages, and has become the main method to capture the digital information.

After studying the nowadays methods, combining the characters of binary scanned maps images, the author designs a method to recognize and capture the digital geography information, which is based on thinning algorithm. A system is also realized by this method in the Visual C++ programming environment.

Vectorization method based on thinning algorithm used in the design prevails currently, whose progress is: after the binary scanned map images are thinned, recognize the skeleton captured. During the progress, the result after thinning affects the final effect of vectorization directly. The author designed a thinning algorithm combining  $3 \times 3$  template with  $4 \times 4$  templates, which keeps the topology structure of the primary maps successfully. In order to recognize and capture figures, vector exterior product is combined with minimal-angle principle, so this method can get polygons more rapidly. Otherwise, the author applies vector-searching algorithm, lines-recognizing algorithm, noise - removing algorithm and some basic operation towards binary images. The System built can get an excellent result after vectorization.

**Key Words:** GIS; digital geography information; vectorization; thinning; topology structure

## 0 前言

地理信息系统(GIS-Geography Information System)作为信息科学和信息产业的重要组成部分受到世界各国的普遍重视,并在近几年得到了快速的发展。数字地理信息是地理信息系统建立和应用的基础。现有各种比例尺的地图是对地理信息较完整的表示,现在以及将来相当长的时期内,地图数字化都是获取数字地理信息的主要途径之一。传统的地图手扶跟踪数字化方法有速度慢、精度低、可靠性差、自动化程度低和人力资源浪费等缺点,远远不能满足地理信息系统发展的需要。从地图扫描图像上识别和提取数字地理信息是一项正在发展之中的新技术,为地图数字化开辟了新的途径,对促进地理信息系统的建立和应用具有十分重要的意义。

地图信息的识别与提取技术涉及计算机、模式识别、数字图像处理、人工智能、计算机视觉、地理学等多个学科。地图信息的识别和提取技术主要解决地图上各种地理要素空间数据的获取问题。近些年来,国内外有关学者在这一领域进行了广泛而深入的研究,而且随着各相关学科的发展,各种新方法和新工具不断被引入到这项技术中。业已取得一定的成就,推动了地图信息识别与提取技术的发展。目前已有的地图信息识别与提取方法有基于细化的矢量化方法、基于轮廓线的方法、基于游码的方法、数学形态学方法以及基于人工智能技术的方法等。这些方法具有各自的特点、性能和适用对象。

本教研室曾有一名 1999 级同学采用基于整体拓扑结构的矢量化方法,设计实现了一个矢量化系统。本文在此基础上进行更深入的研究和尝试,采用了一种基于细化的方法,先利用细化方法提取图像骨架,再对骨架线进行识别。而且,对多边形和直线采用了更为有效的识别方法,避免了由于直线延伸法则带来的错误重建,基于方位角识别造成的大量运算。本系统能够更快、更好地重建地图图像的图形原貌从而取得了更佳的矢量化结果。

本文内容分为六章:第一章简要介绍了地理信息提取与识别技术,以及矢量化方法的现状。第二章介绍了利用数学形态学运算进行图像连贯性保持的实现。第三章介绍了系统采用的  $3 \times 3$  模板与  $4 \times 4$  模板相结合的细化方法。第四章介绍了图像提取与识别过程中采用的矢量跟踪法和对多边形、直线采用的识别方法。第五章介绍了实验系统的功能和功能实现的总体结构。第六章对所做工作进行总结,并对地图信息识别与提取的发展做以展望。

# 1 绪论

## 1.1 地理信息提取与识别技术概述

### 1.1.1 地理信息系统简介

地理信息系统(GIS)是一种集地理信息获取、存储、管理、分析和输出为一体的空间信息系统,它由计算机硬件、软件、地理信息和用户四大要素构成。是介于信息科学,空间科学和地球科学之间的交叉学科。在同地球资源与环境有关的各学科中,地理信息系统应用都极为广泛<sup>[1]</sup>。

地理信息系统按数据库管理系统,包括空间数据和属性数据。地理信息属于空间数据,是指与空间地理分布有关的信息<sup>[2]</sup>。数字地图及制图是地理信息系统的重要组成部分。数字地图是地理信息系统重要的数据源;地理信息系统中处理分析结果常以数字地图形式来表示和输出。

### 1.1.2 矢量化的提出

在计算机图形图像技术应用于地图制作之前,一般采用纸质地图或工程图纸记录地理信息。图纸在查阅、计算距离和标注地名符号等方面都是人工操作;不能对纸质地图进行修改、缩小比例和分层读图。

随着技术的发展,人们对地图的要求越来越高,通过GIS工具得到能在屏幕上显示的电子化地图的需求应运而生。但如今绝大部分地图资料是以纸为载体的形式存放的,于是将这些存储于纸介质上的地图信息转化成GIS的数据格式,已成为迫切需要解决并有重要意义的工作,已成为GIS中的图数据研究方向之一。

常用的数字化方法有两种方式:一是手扶跟踪数字化,采用数字化仪等交互式图形设备输入图形;另一种是用扫描仪将图样快速扫描输入,然后借助于软件对原图进行编辑处理或矢量化。前者这种半自动的输入方法工作繁琐,图纸利用率低。工程设计人员的精力都投入到绘图输入上了,影响设计效率的提高。后者随着扫描仪的应用越来越广泛,地图可以快速、准确地输入计算机。

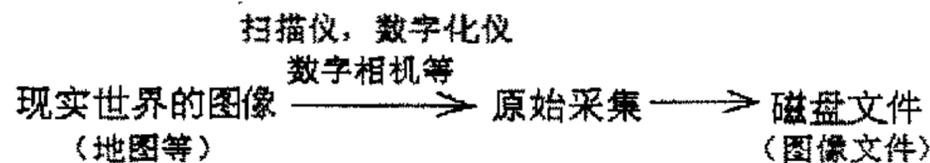


图 1.1.1 扫描输入过程

Fig.1.1.1 The scanning process

图像文件在计算机中是以数据描述的,图像文件的格式就是图像文件的数

据结构,也就是图像数据在计算机中存储的方式。图像文件的格式有两大类,一类是位图(光栅图)文件格式,另一类是矢量图文件格式。

光栅格式是以像素为单位描述图像。位图就是一组描述每个像素点的数据。光栅格式适用于有复杂颜色和灰度级变化的图,如照片、数字化视频图像等。

矢量格式是以图形元素为单位描述图形,统一的单元是点、线和多边形。每个地图要素(一个几何图形)与自身相关的属性数据相关联,更利于分析和处理。

矢量结构和光栅结构各有特点,光栅结构操作总的来说比较容易实现,更易于实现,如按空间坐标位置的搜索;在给定区域的统计指标运算;易于信息共享。矢量格式具有很大的灵活性,图形缩放、旋转等操作容易,还可以节省大量存储空间;而且矢量结构对于拓扑关系的搜索更高效。

一个 GIS 软件理想的方案是采用两种数据结构,提高系统的空间分辨率、数据压缩率、增强系统分析和输入输出的灵活性。矢量结构和光栅结构的相互转换,是 GIS 的基本功能之一。而从光栅格式到矢量格式的转换,特别是扫描图像的自动识别,仍是目前研究的重点。

### 1.1.3 矢量化意义和矢量化软件的设计要求

矢量化就是把光栅图转换成矢量图的过程。就是要从用像素点数据描述的位图文件中识别出线、圆、弧等基本几何图形,所以矢量化也称为识别。

矢量化工作对于 GIS 的发展有很大的意义,因为矢量化输出的矢量图有光栅图无法比拟的优点:

- (1) 存储空间小。同样一幅图矢量图所占的磁盘空间只是光栅图的 1/5-1/10。
- (2) 只有矢量图才能进入 GIS 系统。
- (3) 编辑、修改方便、容易、速度快。
- (4) 矢量图的编辑、修改要求计算机硬件环境低。
- (5) 对输出设备要求低。

矢量化的实现就是矢量化软件的设计,由于矢量化软件处理的对象非常复杂,矢量化软件必须是智能的识别软件。因为地图中图形对象的位置和形状是没有规则的,矢量化必须从图形互相重叠的点阵中识别出各类基本图形。而且矢量化大量处理的都是手工绘图,图面情况因人而异,图纸污损时有发生,污损也会经扫描仪带入光栅文件。

另外为提高工作效率,矢量化软件还应有友好的界面,方便灵活的显示控制。

## 1.2 地图信息识别与提取方法现状

随着计算机软硬件技术的不断提高,地图信息识别与提取工作取得了质的飞跃。从识别简单图元的低级矢量化发展到理解图纸内容的高级矢量化<sup>[3-5]</sup>。但目前低级矢量化的准确性和速度仍达不到预期的目标,仍有待于深入的研究。

目前,国内外对矢量化的研究以工程图纸为多,市场上大部分矢量化软件也是针对工程图设计的。地图扫描矢量化与之有着相同的本质,然而由于二者有着不同的特点,甚至有些特点是截然不同的。所以对地图信息识别与提取的研究可以借鉴一些工程图纸矢量化的技术,但针对地图特点进行专门研究是很有必要的。目前,矢量化方法大致可分为两类,基于细化的方法<sup>[6-9]</sup>和基于非细化的方法。在基于非细化的方法中,主要有:基于轮廓线的方法、整体矢量化的方法、基于网络栅格的方法<sup>[10]</sup>以及基于人工智能技术的方法。

### 1 基于细化的矢量化方法

对图像进行细化处理提取其骨架,在图像识别领域有着广泛的应用。一个好的细化结果能够在不改变原有拓扑结构的前提下即保持骨架的连通性又不过度腐蚀,同时还应具有一定的抗噪性能。

现在,基于细化方法和矢量后处理是当前矢量化方法的主流。其工作步骤是先由扫描仪将图纸扫描入计算机,然后通过线条细化,得到单点宽的连通线,再由线跟踪得到直线、圆和圆弧等图形元素,最后进行矢量化。

细化得到的骨架图基本保持了原图的线状结构和连通性,便于进一步的处理;算法简单。但多次迭代会造成处理时间长;骨架均为单点宽会导致信息遗失;而且细化对噪声敏感,容易引起骨架毛刺和交叉点畸变。

### 2 轮廓匹配方法

对于线性尺寸相同的区域,存储其周围轮廓比存储其所有像素,用的空间小得多。而且轮廓特征也可表达图像的有用信息。这类利用图像轮廓进行识别处理的方法就是轮廓匹配法。

轮廓匹配法的基本步骤是:(1)轮廓提取;(2)轮廓匹配和矢量化;(3)轮廓对骨架化;(4)骨架拼接,矢量重建图形。

轮廓匹配方法通常是与细化算法结合在一起的<sup>[11-14]</sup>。轮廓匹配方法是通过跟踪轮廓像素和匹配轮廓矢量,得到图线的中间骨架矢量,该方法有效地避免了缺陷的不利影响,所需数据量较少,但匹配过程复杂,正确性难以保证。而且,轮廓线跟踪受图纸质量退化影响极大。当轮廓线中有一条线与其它线或字符粘连时,成对的轮廓匹配线就不存在了,骨架线也就很难获得。

### 3 整体矢量化方法

整体矢量化方法是在近几年发展起来的，它克服了细化算法逐像素处理的局限性，是利用图像的整体信息进行矢量化的方法。力求从整体把握图像中各种线形及其关系，以扫描串或连通图段为基本单元对象素进行提取和识别。

整体识别的基本思想是：将图纸图像组织成图段数据结构；寻找符合各类线形特征的种子图段据此确定线形的初始几何参数；根据图段之间的位置关系，跟踪出组成同一线形的所有图段，并确定最终几何参数。整体识别方法具有识别方法简单，识别效果好，能快速提取图纸中的图元信息等特点。

#### (1). 基于扫描串的算法

基于扫描串的算法很适用于工程图纸扫描矢量化。扫描串又称游程编码，就是图像扫描数据某一行(列)中所有连通的黑像素的集合。扫描串法是介于单点像素和连通图段之间的一种方法，具有较大的灵活性。由于工程图中含有大量小像素、曲线和特殊符号，基于扫描段的识别方法在处理字符、小特征是有显著优势。

但是这种算法是以扫描串为单位的，出发点还是局部信息，无法充分利用图像的整体信息，制约了效率，更适用于工程图示的识别和矢量化。

#### (2). 基于图段的算法

基于连通图段的算法，国内在 1994 年就已经提出来了<sup>[15]</sup>。它以连通图段为单元作图像处理。连通图段是由一串上下相互连通的行程编码组成，包含了较大范围的图像信息；算法中直线、圆弧的识别过程就是图段的分裂、合并和匹配的过程。基于图段的算法在处理像素时有一定的局限性。地图，特别是大比例地形图具有与工程图完全不同的特点，它所包含的大量道路、房屋轮廓等较不规整的曲线。利用基于图段的算法对其进行自动识别和矢量化，并不合适。

大体步骤为：(1) 图像编码，用游程链表示图形；(2) 提取节点域；(3) 由节点确定矢量线段；(4) 由矢量线段依照拓扑关系重建图形。

在整体识别的前提下，有人提出了以矩形<sup>[16][17]</sup>或梯形<sup>[18]</sup>为单元，将图纸图像转化成以矩形块或梯形块为结点、以块与块之间的几何邻接关系为边的条块图，然后对节点进行分类识别。这些矩形块或梯形块是在扫描过程中按照一定的规则，由若干扫描串组合成的。这种方法适用于规则线形，是针对工程图纸设计的。而 GIS 道路中的地图的道，建筑物轮廓并不是规则的图形，所以不适合采用这种方法。

### 4 与人工智能技术的结合

随着计算机技术的发展，人工智能技术在越来越多的领域得到了应用。地图矢量化工作也不例外。已出现了与遗传算法、神经网络相结合的矢量化方法。

#### (1). 遗传算法的应用

遗传算法 (Genetic Algorithms, GA) 是基于达尔文进化论自然选择机制的一种搜索算法。它是一种全局搜索的方法, 避免了局部搜索技术的许多缺陷, 具有自适应性、鲁棒性、并行性等优点。

文献<sup>[19]</sup>提出了细化算法与遗传算法的结合。遗传算法应用于后处理工作中的骨架点确定, 这些骨架点用来连成线划去尽量逼近原图段。这实际上就是一个优化问题, 遗传算法可以有效的产生最优解。这种基于 GA 的矢量化算法于以往传统的矢量化算法有着很大的差异。

## (2). 神经网络的应用

神经网络方法的特点是稳健性、并行性、自适应和自学习性, 也适用于地图符号的识别。但由于神经网络本身的限制, 其处理问题的规模不能太大, 因而还不能用这种方法单独完成地图信息的识别。

文献<sup>[20]</sup>设计了一个自适应神经网络, 将工程图符号的特征矢量输入该神经网络, 实现符号的实现。文献<sup>[21]</sup>提出了细化算法与神经网络结合, 识别含有交叉线的图纸。图像在经过去噪、细化处理后, 映射到一个 Hopfield 神经网络上, 每一个神经元对应与一个像素, 神经元之间的连接权表示为像素间的关系函数。通过使能量函数最小, 将交叉线段分开。

## 1.3 本论文所写的内容

地图通过扫描仪等设备输入计算机, 获得图像信息。对图像进行矢量化的过程一般分为预处理、细化/轮廓化、矢量跟踪和图元识别四部分。预处理的主要目的是通过噪声滤除、形态学运算等改善输入图像的质量。细化或轮廓化的工作是把多线宽的原始图像处理成单线宽的轮廓或骨架图像。矢量跟踪的目的是通过对像素点连接关系的跟踪, 将复杂图形分割为一个个简单的结点之间的弧段, 便于后续识别。而图元识别的任务就是判断每个简单图形的几何形状, 并结合其相对位置、尺度等关系, 进行综合判别。

本文在研究典型地图、工程图扫描图像矢量化方法的基础上, 针对地图扫描图像的具体特点, 采用细化方法提取图像骨架, 再对骨架线进行矢量化, 实现了一个具有矢量化基本功能的实验系统。研究内容有:

(1) 利用矩形框平滑滤波进行噪声滤除; 利用数学形态学闭合运算保持图形连通性。

(2) 提出  $3 \times 3$  模板与  $4 \times 4$  模板相结合的细化方法, 取得了较为理想的效果, 提取出的骨架线基本保持了原图的拓扑特征。

(3) 采用一种基于查表的矢量转换算法, 不仅保持了原图的拓扑完整性, 而且构造简单, 复杂性只与被跟踪的象素点个数有关, 有利于软件实现。

(4) 将计算机几何学中矢量外积的概念引入多边形的提取中, 避免了以往算法中参数值计算和其它烦琐的数学运算, 生成算法明显优化了代码并提高

了算法的执行效率。

(5)将地图中直线分为一般直线和任意角度直线两种情况，分别进行识别；并把直线合并方法用于直线识别的纠错。

(6)使用 Visual C++软件集成开发环境开发了地图扫描图像矢量化实验系统。该系统主要功能包括：栅格图像的读取显示；图像预处理：噪声去除；图像细化，提取图像骨架线；保持骨架线的连通性；对骨架线的矢量化；对光栅图像的编辑处理：各种图形绘制、选取、删除、移动、缩放等。

与 1999 级同学所设计的实验系统相比，二者的主要差别在于：首先，采用的矢量化方法从根本上有着巨大的差异；其次，直线识别问题上本系统采用分类方法进行具体识别，避免了使用直线最长延伸原理带来的误差；最后在多边形的识别中，本系统采用了矢量外积与最小角法则相结合的新设想，较之方位角的方法运算更为简单。

## 2 扫描图像处理

噪声滤除是地图自动输入预处理的一个重要环节。地图经扫描仪扫描输出为二值图像，由于扫描过程的非理想性以及原有地图质量欠佳等因素，会引入各种噪声成分，如孤立噪声(地图背景中的孤立点、斑点、污点或麻点，其主要来源是图纸背景不干净引起的)、毛刺噪声(地图中线段边缘不应该有的小毛刺)、空洞和缺陷噪声(地图中线段内部出现的小洞或边缘出现的缺口)、断裂噪声(地图中个别线段出现的轻微断裂)等。这些噪声若不设法滤除，会影响后续的细化\轮廓提取、矢量化等环节的处理效果和正确性。

### 2.1 噪声滤除

消除图像噪声干扰的有效方法是采用局部平滑滤波。因为扫描输入的图像中的噪声主要为颗粒噪声和尖脉冲噪声，所以本系统采用矩形框平滑滤波的滤除噪声。

矩形框平滑滤波是根据二值线型图像的特征，总结出对于滤除二值图像麻点噪声及孤立噪声十分有效的方法，属于非线性局部平滑滤波的一种特殊形式。滤波过程如图 2.1.1 所示，采用一个矩形框模板，窗口尺寸为  $m \times n$  ( $m, n > 1$ )，对孤立噪声，矩形框模板一般选为  $3 \times 3$  模块，在图面中逐点移动该窗口，设该窗口中心像素值为  $f(i, j)$ ，设其周围 8 个像素的平均值为  $a$ ，若  $|f(i, j) - a|$  的值等于 1 时，该中心像素黑白反转，小于 1 则不变。本系统采用的就是  $3 \times 3$  矩形框模板。

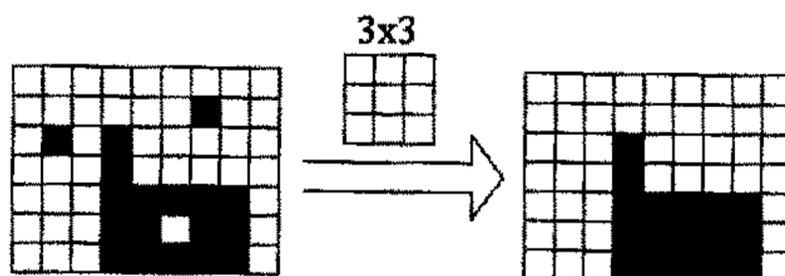


图 2.1.1 使用  $3 \times 3$  模板去噪

Fig.2.1.1  $3 \times 3$  template to remove noise

### 2.2 连通性保持

#### 2.2.1 数学形态学理论

数学形态学历史可回溯到 19 世纪的 Euler, Steiner, Crofton 和本世纪的 Minkowski, Matheron 和 Serra。1964 年法国的 Matheron 和 Serra 在积分几何

的研究成果上,将数学形态学引入图像处理领域,并研制了基于数学形态学的图像处理系统。1982年出版的专著《Image Analysis and Mathematical Morphology》是数学形态学发展的重要里程碑,表明数学形态学在理论上趋于完备及应用上不断深入。数学形态学蓬勃发展,由于其并行快速,易于硬件实现,已引起了人们的广泛关注。目前,数学形态学已在计算机视觉、信号处理与图像分析、模式识别、计算方法与数据处理等方面得到了极为广泛的应用。

数学形态学是一种非线性滤波方法。形态和差运算,即膨胀与腐蚀是数学形态学的基础。数学形态学可以用来解决抑制噪声、特征提取、边缘检测、图像分割、形状识别、纹理分析、图像恢复与重建、图像压缩等图像处理问题。

数学形态学中二值图像的形态变换是一种针对集合的处理过程<sup>[22]</sup>。数学形态学将二值图像看成是集合,并用结构元素来探察。结构元素是一个可以在图像上平移、且尺寸比图像小的集合。其形态算子的实质是表达物体或形状的集合与结构元素间的相互作用,结构元素的形状就决定了这种运算所提取的信号的形状信息。基本的数学形态学运算是将结构元素在图像范围内平移,然后将结构元素与下面的二值图像进行交、并等集合运算。

基本的形态运算是腐蚀和膨胀,其定义为:

$$\text{膨胀运算: } A \oplus B = \bigcup A_{b_i}, \quad b_i \in B \quad (2-1)$$

$$\text{腐蚀运算: } A \ominus B = \bigcap A_{b_i}, \quad b_i \in \hat{B} \quad (2-2)$$

其中, A 表示被研究的图像, B 为结构元素,  $\hat{B}$  为结构元素 B 的对称集。  $A_{b_i}$  表示 A 被  $b_i$  平移。

图像 A 对结构元素 B 的形态开运算定义为:

$$A \circ B = (A \ominus B) \oplus B \quad (2-3)$$

即 A 先被 B 腐蚀,再被 B 膨胀。

图像 A 对结构元素 B 的形态闭运算定义为:

$$A \bullet B = (A \oplus B) \ominus B \quad (2-4)$$

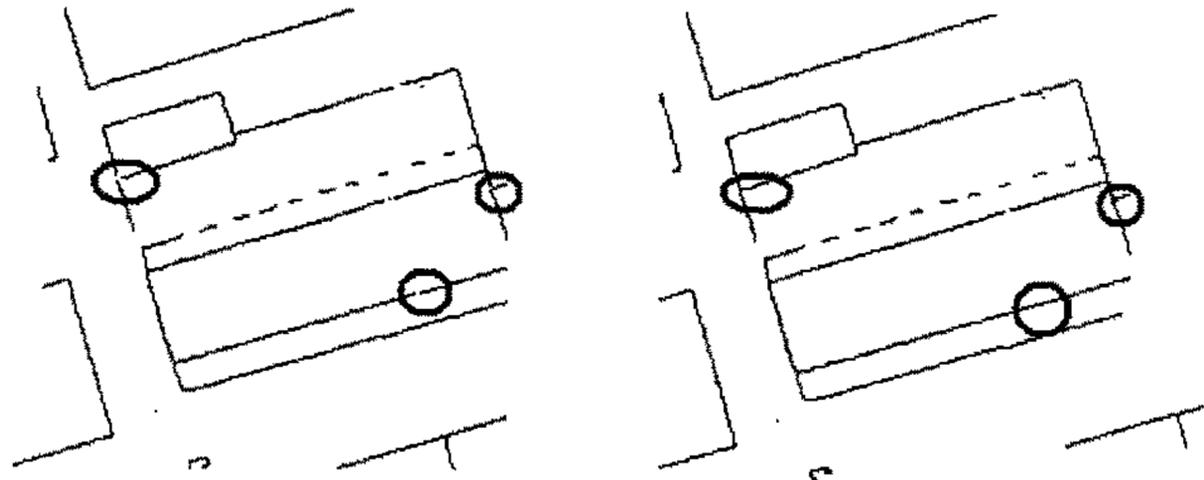
即 A 先被 B 膨胀,再被 B 腐蚀。

## 2.2.2 运用开运算保持细化结果的连通性

由上面的闭和开运算的定义可知,开运算是先做腐蚀运算,运算结构使图像中宽度小于结构元素 B 的孤立噪声和毛刺噪声得到滤除。第二步的膨胀运算则可使腐蚀后线条变窄的图像得以恢复。而闭运算则是先做膨胀运算,运算结

果可使图像中宽度小于结构元素 B 的空洞噪声和缺陷及断裂噪声得以填充。第二步的腐蚀运算可使经膨胀后线宽变宽的图像得以恢复。

可见，闭运算对于填充空洞、断裂有很好的效果。本系统对于扫描图像进行细化处理后，往往会造成一些不必要的线划断裂的情况。从而影响后续的图形识别结果。因而对细化结果进行闭运算，保证图像原有的连通性，是非常有意义的。图 2.2.1 表示了细化结果在闭运算前后的连通性比较。其中圆圈所表示部分为通过闭运算使原有连通性得以保持的部分。



(a) 闭运算前

(b) 闭运算后

(a) Before closing

(b) After closing

图 2.2.1 细化结果闭运算结果示意

Fig.2.2.1 Use closing in the thinned picture

### 3 地图扫描图像的细化处理

目前比较普及的矢量化方法大多是基于细化的方法,将多像素宽的图像处理成单像素宽的图像,这些图像信息仍是以栅格形式保存的,接下来的识别和提取运算都是以细化后的结果为基础的,所以细化处理也是矢量化过程中的一个重要步骤。

#### 3.1 细化理论及现状

图纸中的线宽通常大于一个像素,所以在矢量化的过程中会出现一些问题,如一条线段会成为两个平行的线矢量。将图线变成一个像素宽的骨架化方法可以避免这些问题。细化是图像处理中的一个基本技术,对于识别效果有着直接的联系。对细化算法,要求达到的目标是:

- (1) 细化结果应保持原图形的几何性质不变;
- (2) 各向同性,即对原图像进行旋转后,细化结果不会出现畸变;
- (3) 从骨架恢复原图形的能力;
- (4) 较高的处理能力。

自细化思想提出后,对于细化算法的研究相继出现。一般把细化算法分为两类:一类是重复地剥去边缘点的串行算法;另一类是边缘跟踪的并行算法。

##### 1. 串行算法

串行算法的主导思想是每次扫描删除图像上目标的轮廓象素(这些象素满足一定的条件),直到图像上不存在可删除的轮廓象素为止。其核心过程就是在图像上移动一个窗口,并运用一套规则来标识窗口的中心。每完成一次扫描,就删除掉所有的标识点。重复进行扫描,直到没有再多的点可被删除。例如,Naccache 和 Shinghal 算法<sup>[23]</sup>,Hilditch 经典细化算法<sup>[24]</sup>运用一个  $3 \times 3$  的窗口,采用如下标识规则,对其中心点进行标识,如果下面的规则都能够被满足,那么 P 就标识为删除:

- (1)  $P_0 + P_2 + P_4 + P_6 \leq 3$ ;
- (2)  $N_c = 1$ ;
- (3)  $P_k (0 \leq k \leq 7)$  中至少有一个目标像素为 1;
- (4)  $P_2 = 1$  或  $N_{c2} = 1$ ;  $N_{c2}$  为假定  $P_2 = 0$  时 P 的联结数;
- (5)  $P_4 = 1$  或  $N_{c4} = 1$ 。  $N_{c4}$  为假定  $P_4 = 0$  时 P 的联结数。

重复细化过程,不断收缩线体轮廓,直至剩下骨架线为止。

其中  $N_c$  表示联结数,即八邻域中互相分离的联结成分的个数。目标像素 P 的联结数可表示为:

$$N_c = (N_6 - N_6 N_7 N_0) + \sum (N_K - N_K N_{K+1} N_{K+2}) \quad K \in S \quad \text{其中 } S = \{0, 2, 4\} \quad (3-1)$$

## 2. 并行算法

并行算法往往采用两层子循环。第一个子循环中,确定目标像素  $P$  删除的条件,若条件成立,则目标像素  $P$  被标记为可删除像素。在第二个子循环中,检索整个图像像素点,将被标记为可删除的像素删除。

常用并行算法如 Deutsch 算法<sup>[25]</sup>, Zhang 快速并行细化算法<sup>[26]</sup>。

并行细化算法虽提高了处理的速度,但处理图像并不是特别有效。因为大多数处理算法的运算是靠近边缘进行,所以分配大的均匀区域中的像素的处理器多数时间空闲。并且并行算法(如 Deutsch 细化算法),对于宽度为 2 的目标曲线不适合。因为按照并行处理目标像素删除的条件,所有像素都是轮廓点,且对连通性没有必要,因此都被标记为可删除像素而造成信息丢失。

串行算法细化的结果不仅取决于前一次的迭代图像,而且与当前处理情况有关;而对并行算法当前迭代仅由前一次迭代的结果决定。串行算法的处理结果依赖于对像素处理的先后顺序,因而像素的腐蚀或保留不可预测;而并行算法对像素腐蚀时利用相同的条件同时检测所有像素,结果具有各向同性。而且串行算法虽然效果更明显,但处理速度较慢,因而并行算法更利于硬件实现。

鉴于串行和并行算法的特点,也有研究者将二者相结合,提出异步并行算法,如 Pavlidis 异步细化算法<sup>[27]</sup>。此外还将数学形态学引入细化工作中<sup>[28]</sup>。

由于细化算法简单易行,能基本保持原图像的线状结构及连通性,有利于进一步的分析和识别。但是细化方法存在着交点畸变的固有缺陷,需要在算法中或后续工作中进行畸变点的修正。在处理技术图纸以及地图学应用中,当原图包含直线,圆以及基于这些基本元素并且没有被填实物体的光栅图形时,常常使用基于细化的方法。因而基于细化的矢量化算法在现有矢量化系统中得到了广泛的应用。

### 3.2 一种 $3 \times 3$ 模板与 $4 \times 4$ 模板相结合的细化方法

无论是串行算法还是并行算法,都涉及判断一个象素点是否可删除的问题。目前判断象素点的方法主要是以  $3 \times 3$  的模板为准的,这种方法限制了判定速度,更重要的是不能保证细化骨架的连通性或者无法生成完善的 8 连通骨架。常用的措施有两种:一种措施是将每次迭代分成若干个子迭代(比如 4 次或 2 次),每次子迭代采用特定的模板按一定的方向进行,所以子迭代的顺序不同,得到的骨架也不完全一样,因此这类算法不是严格意义上的并行。另一有效途径是适当地扩大模板尺寸。这类算法<sup>[29]</sup>的优劣完全取决于所选取的细化模板。如果选用的模板不足,最终生成的骨架不是 8 连通,需要做进一步的后处理才能达到细化目的。反之,如果选用的模板太多,将会发生过腐蚀现象。

本文在选取细化模板时首先分析  $3 \times 3$  窗口如图 3.2.1(a)所示,然后针对一些在  $3 \times 3$  窗口内无法确定的模板扩大到  $4 \times 4$  窗口如图 3.2.1(b)所示。这种方法不仅避免了过度腐蚀还保持了原图良好的连通性。

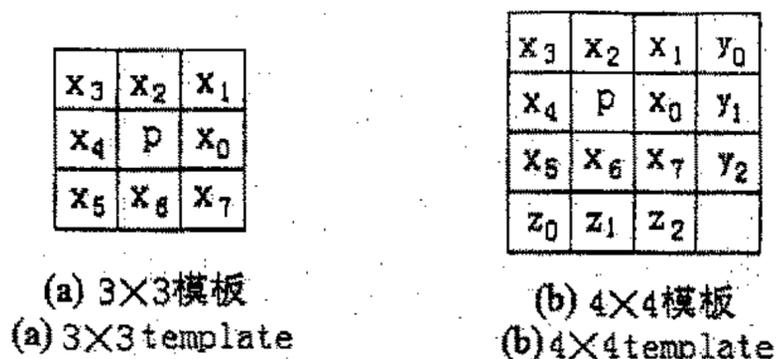


图 3.2.1 细化模板  
Fig.3.2.1 Thinning template

### 3.2.1 设计思想

采用 3×3 细化模板，存在对应  $2^8$  可能情况，而采用 4×4 的细化模板，存在对应的  $2^{16}$  种可能情况，若逐一进行判别计算量将非常大。本算法首先考虑 3×3 模板的可能情况，然后再将不确定 3×3 模板将窗口扩大为 4×4 的窗口。

### 3.2.2 模板设计

在 3×3 窗口中，象素点  $p$  的分枝数为

$$X(p) = \sum_{i=0}^7 |x_{i+1} - x_i|, \quad \text{其中 } x_8 = x_0 \quad (3-2)$$

$p$  的邻接黑象素点数为

$$b(p) = \sum_{i=0}^7 x_i \quad (3-3)$$

下面的讨论就是按照  $X(p)$  和  $b(p)$  的不同值情况对模板分类讨论的。模板下打“√”的表示选为细化模板，“×”表示不能选为细化模板，“？”表示有待扩大为 4×4 窗口后再做进一步分析。

1.  $X(p)=0$  时，只有两种情况：

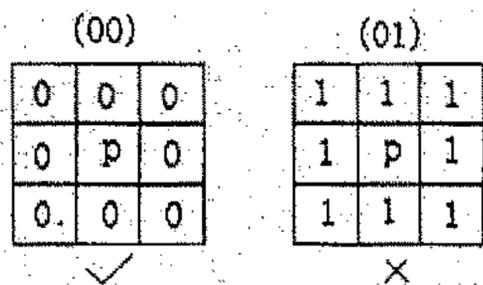


图 3.2.2  $X(p)=0$  时模板  
Fig.3.2.2 templates when  $X(p)=0$

2.  $X(p)=2$  时

(1)  $b(p)=1$  时， $p$  点处在端点位置，不存在细化模板。

(2)  $b(p)=2$  时，

(220)	(221)	(222)	(223)	(224)	(225)	(226)	(227)
0 0 1	0 1 1	1 1 0	1 0 0	0 0 0	0 0 0	0 0 0	0 0 0
0 p 1	0 p 0	0 p 0	1 p 0	1 p 0	0 p 0	0 p 0	0 p 1
0 0 0	0 0 0	0 0 0	0 0 0	1 0 0	1 1 0	0 1 1	0 0 1
✓	✓	✓	✓	✓	✓	✓	✓

图 3.2.3(a)  $X(p)=2 \& b(p)=2$  时模板

Fig.3.2.3(a) templates when  $X(p)=2 \& b(p)=2$

(3)  $b(p)=3$  时,

(230)	(231)	(232)	(233)	(234)	(235)	(236)	(237)
0 1 1	1 1 1	1 1 0	1 0 0	0 0 0	0 0 0	0 0 0	0 0 1
0 p 1	0 p 0	1 p 0	1 p 0	1 p 0	0 p 0	0 p 1	0 p 1
0 0 0	0 0 0	0 0 0	1 0 0	1 1 0	1 1 1	0 1 1	0 0 1
✓	✓	✓	✓	✓	✓	✓	✓

图 3.2.3(b)  $X(p)=2 \& b(p)=3$  时模板

Fig.3.2.3(b) templates when  $X(p)=2 \& b(p)=3$

(4)  $b(p)=4$  时

(240)	(241)	(242)	(243)	(244)	(245)	(246)	(247)
1 1 1	1 1 1	1 1 0	1 0 0	0 0 0	0 0 0	0 0 1	0 1 1
0 p 1	1 p 0	1 p 0	1 p 0	1 p 0	0 p 1	0 p 1	0 p 1
0 0 0	0 0 0	1 0 0	1 1 0	1 1 1	1 1 1	0 1 1	0 0 1
✓	✓	✓	✓	✓	✓	✓	✓

图 3.2.3(c)  $X(p)=2 \& b(p)=4$  时模板

Fig.3.2.3(c) templates when  $X(p)=2 \& b(p)=4$

(5)  $b(p)=5$  时

(250)	(251)	(252)	(253)	(254)	(255)	(256)	(257)
1 1 1	1 1 1	1 1 0	1 0 0	0 0 0	0 0 1	0 1 1	1 1 1
1 p 1	1 p 0	1 p 0	1 p 0	1 p 1	0 p 1	0 p 1	0 p 1
0 0 0	1 0 0	1 1 0	1 1 1	1 1 1	1 1 1	0 1 1	0 0 1
✓	✓	✓	✓	?	✓	?	✓

图 3.2.3(d)  $X(p)=2 \& b(p)=5$  时模板

Fig.3.2.3(d) templates when  $X(p)=2 \& b(p)=5$

(6)  $b(p)=6$  时

(260)	(261)	(262)	(263)	(264)	(265)	(266)	(267)
1 1 1	1 1 1	1 1 0	1 0 0	0 0 1	0 1 1	1 1 1	1 1 1
1 p 1	1 p 0	1 p 0	1 p 1	1 p 1	0 p 1	0 p 1	1 p 1
1 0 0	1 1 0	1 1 1	1 1 1	1 1 1	1 1 1	0 1 1	0 0 1
✓	✓	✓	?	?	?	?	✓

图 3.2.3(e)  $X(p)=2 \& b(p)=6$  时模板

Fig.3.2.3(e) templates when  $X(p)=2 \& b(p)=6$

考虑到线宽为偶数个象素点的情况，细化模板不能取完全对称的两个模板，这样会导致整个线条消失，因而在实际操作中没有将(250)与(254)，(252)与(256)同时作为细化模板。本文采用(254)，(256)作为细化模板。

3.  $X(p)=4$  时，只有  $b(p)=2, 3$  时，存在可能的细化模板，即

(1)  $b(p)=2$  时，

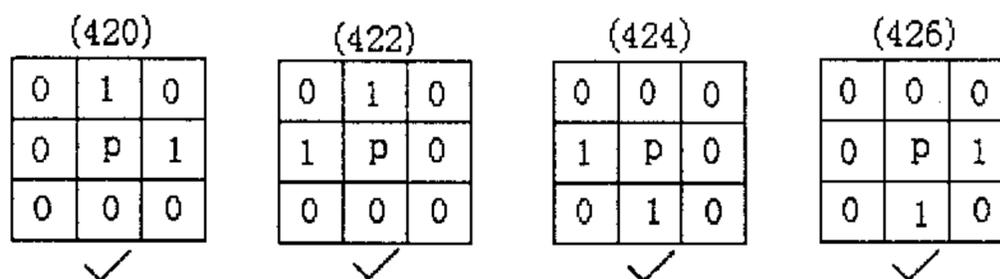


图 3.2.4(a)  $X(p)=4$  &  $b(p)=2$  时模板

Fig.3.2.4(a) templates when  $X(p)=4$  &  $b(p)=2$

(2)  $b(p)=3$  时，

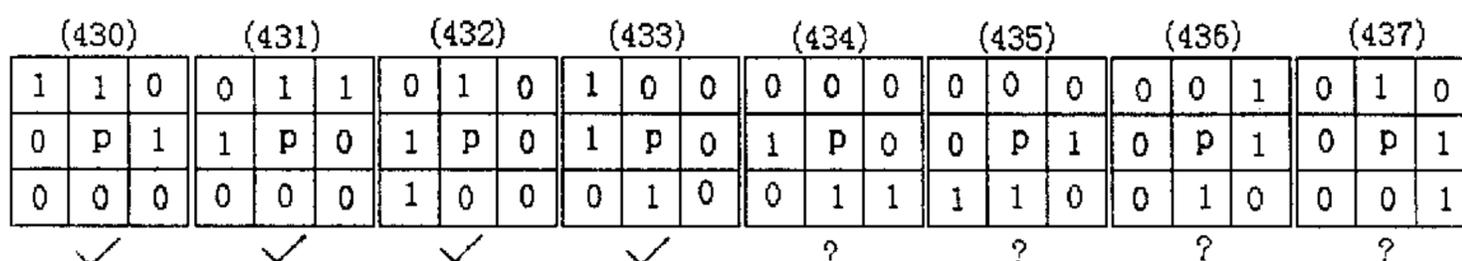


图 3.2.4(b)  $X(p)=4$  &  $b(p)=3$  时模板

Fig.3.2.4(b) templates when  $X(p)=4$  &  $b(p)=3$

4.  $X(p)=6,8$  时， $p$  点处于交叉点处，不具备细化条件。

针对  $3 \times 3$  模板中的不确定情况，本文将窗口扩大至，进行进一步判断：

(1) 对于(256)，(265)，(266)，(436)，(437)模板的处理窗口扩大为图 3.2.1(b)所示，满足如下判决条件时可对  $p$  点进行细化：

$$y_0 + y_1 + y_2 > 1 \tag{3-4}$$

(2) 对于(254)，(262)，(263)，(434)，(435)模板的处理窗口扩大为图 3.2.1(b)所示，满足如下判决条件时可对  $p$  点进行细化：

$$z_0 + z_1 + z_2 > 1 \tag{3-5}$$

### 3.2.3 算法描述与实验结果

输入图像数组元素与上文中  $3 \times 3$  模板进行匹配比较，当满足细化要求时记为  $Kind=1$ ；需要进一步分析时，按照窗口扩大的两种情况分别记为  $kind=2$  和

Kind=3, 再作下一步的判断。算法步骤为:

第一步: 输入图像数组 TwoValue[ ], i=0。

第二步: i++, 标识 Flag=FALSE。

第三步: 检测 TwoValue[ ]中每一个象素点 TwoValue[i], 如果 TwoValue[i]=1 则根据其 3×3 邻域的象素点情况与 3×3 模板进行匹配比较, 满足下面任一种情况置 TwoValue[i]的删除标志 del=1, Flag=TRUE:

a) Kind=1;

b) Kind=2, 窗口扩大为图 3.2.1(b)并满足式 (3-3) 条件;

c) Kind=3, 窗口扩大为图 3.2.1(b)并满足式 (3-4) 条件;

第四步: 如果 Falg=TRUE 表明图像已经做好细化标识, 转至第五步完成细化后, 输出结果; 否则返回第二步进行下一轮迭代。

第五步: 按照 TwoValue[ ]中象素点的删除标识完成细化, 输出细化结果。

图 3.2.2 给出了原图, 图 3.2.3(a)给出了该算法的细化结果, 与之相比较的是图 3.2.3(b)在确定此算法之前设计的基于图形整体拓扑结构的细化算法。由于这种方法只对于线性对象适用, 不具有普遍性, 最后还是放弃了。

二者相比, 本文方法良好的保持了原图的连贯性和拓扑关系, 但不可避免的存在着交点畸变的现象; 后者良好的避免了交点畸变的影响, 但是由于交点的偏差会导致图形连接关系的错误, 尤其是对于弧状图形有很大的影响。

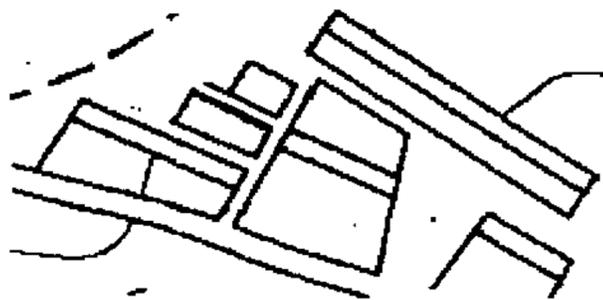
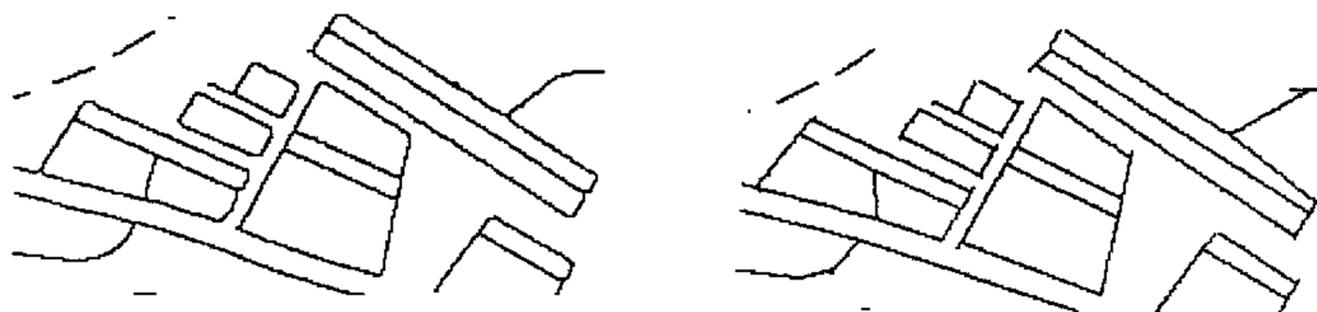


图 3.2.2 细化前原图

Fig.3.2.2 Before thinning



(a) 本文方法细化结果

(b) 基于整体拓扑结构方法细化结果

(a) The method used here

(b) A method based topology

图 3.2.3 细化后结果

Fig.3.2.3 After thinning

## 4 地图扫描图像中图形的提取与识别

### 4.1 自动矢量化简介

地图的自动输入与识别处理过程中,矢量化阶段起着十分重要的作用,其主要任务就是把经过细化处理的点阵数据文件转换为用图形数据表达的数据文件。矢量化的结果将作为进一步的图像模式的特征提取与识别提供基本的原始数据描述。

早期的识别算法需要事先人工输入左右多边形编码等辅助信息。现有的一些半自动化算法,采用数字化仪、键盘、鼠标等交互式图形设备来输入图形,也需要事先手工输入多边形内点以帮助拓扑关系的建立。这类手扶跟踪数字化采集原始数据的算法中,人工输入辅助拓扑信息的复杂性与大量的手工数字化相比减轻了,但自动化水平仍然不高,效率低,原有图纸利用率低;而且不利于寻找并消除矢量化过程中产生的一些没有地理意义的微小多边形,对于保护地理实体的完整性是不利的。

地图的扫描输入一直是计算机图形学及 GIS 技术中急待解决的难题之一,近几年来相继问世的各种扫描仪为图形的快速输入开辟了新的途径。然而,通过扫描仪获得的图形信息都是二值点阵信息。由于信息量巨大,给地图有效在计算机中存档、编辑、绘制新图提出了新的问题,采用模式识别技术,将二值点阵图形信息通过预处理、细化、图形识别等环节转换成为矢量信息存储是目前矢量化技术的主要目标<sup>[30][31]</sup>。

扫描矢量化是一种全自动的地图输入方法,它实现将扫描输入的栅格数据转化成矢量数据,即把以像素为单位的图像数据转化为,以点、线、面等图形要素为单位的图形数据,并保持相应的拓扑结构。市场上用于光栅矢量化的软件已有不少,如 RXAutoImage Pro2000, VPStudio, R2V 等<sup>[32]</sup>,它们各有特点,系统组成基本为:图像输入、图像编辑、矢量图形编辑、图纸管理、图纸管理、图纸输出等几个模块。

以矢量化为核心,系统可以分成 3 大部分:前处理,矢量化和后处理。

(1)前处理:前处理过程为矢量化准备好必要的图像信息,主要功能:

- 图纸拼接:当采用小幅面扫描仪分块扫描输入大幅面图纸时,可利用图纸拼接功能,将各个小块的图像拼接成一幅大的图像。系统提供了自动拼接与手动拼接两种方式

- 图像处理:系统提供噪声去除、平滑等图像处理工具以改善图像质量。

- 图像编辑:基本的图像编辑工具(如刷子、橡皮差等)被用来对图像进行交互修改。

- 图元提取:在表示某一图形元素(直线段、圆和圆弧等)的图像上交互地给出几个特征点,从而将该图元的图像信息提取出来,并用相应的图形来替代。

- 光栅/矢量混合编辑：在图像编辑的同时，可对用图形形式表示的线形进行复制、移动、删除等操作，必要时，可将图形光栅化成图像表示。

(2) 矢量化：接受前处理过程的结果，将图像信息自动转化成图形信息，并从中分离出各种要素。过程：

- 字符处理：将图纸中的字符提取出来另行处理。包括字符提取、字符串组合、字符识别等模块。

- 线形识别：利用整体识别方法分离出图纸中的各种线形。系统处理的线形包括水平线、一般位置直线、圆和圆弧、虚线形等

(3) 后处理：对矢量转化的结果进行调整，是最终结果与原有图纸相符合。

- 线形调整：将水平线、竖直线等特殊线形自动调整到位；重新连接误分段的线形；调整线形的端点位置。

- 图形编辑：对图形进行基本的编辑操作，包括图形的约束生成、复制、删除；图形的平行、垂线、同心等的调整。

从理论上这种操作是很容易理解的，当应用于实际大型地图时就会产生很多问题。同时，处理算法的效率还有很大的提高空间，新算法不断涌现，老算法也有很多改进和优化。扫描矢量化的发展前景还是广阔的。

## 4.2 系统实现总体算法

系统采用了一种基于细化的矢量化方法，并依照矢量化的过程——预处理、细化、矢量跟踪和图元识别，将系统实现分为几个模块，从而充分体现了面向对象、模块化的编程思想。这种结构可以使噪声滤除、细化、基本图形元素的识别提取分步进行，有利于软件的调试、维护和测试。

各个模块的功能和算法实现会在其他章节做出详尽的阐述，在这里侧重于系统实现的总体流程和对象的设计。

### 4.2.1 对象设计

系统设计采用 C++ 语言，它提供了特定的机制去支持特定的程序设计技术——面向对象的程序设计。在 C++ 语言中，面向对象的程序设计就是类的设计。类是抽象数据类型的工具，它把所处理的对象的数据和处理数据的函数封装在一起，实现了系统设计的高度结构化，提高了代码的重复利用率。

本系统对于基本图形对象直线、折线、矩形、多边形、椭圆分别用类 CLine, CPolyLine, CPolygon, CEllipse 来定义，封装了每个图形对象的属性数据和对其进行绘制、存储等操作的函数。而基于多态性它们又继承自 CFigure 类，该类封装了所有对象所共有的属性和功能函数。下面详细介绍各类的定义：

1. CFigure 类：

```
class CFigure : public CObject
```

```

{
protected:
    CFigure();
    DECLARE_SERIAL(CFigure)
    COLORREF m_PenColor;           //图形颜色
    BYTE m_Thickness;             //图形线宽
    UINT m_PenType;               //图形线型
    UINT m_Brush;                 //刷子类型
public:
    BOOL m_Deleted;               //删除标志
    virtual ~CFigure();
    CFigure(COLORREF, BYTE, UINT, UINT, BOOL);
    virtual void Serialize(CArchive& ar); //图形存储
    virtual void Draw(CDC *PDC, BYTE State) {} //图形绘制
};

```

## 2. CLine 类

```

class CLine : public CFigure
{
protected:
    CLine();
    DECLARE_SERIAL(CLine)
    float m_x1, m_y1, m_x2, m_y2; //直线起始点坐标
public:
    virtual ~CLine();
    CLine(float, float, float, float, COLORREF, BYTE, UINT);
    virtual void Serialize(CArchive& ar); //直线存储
    virtual void Draw(CDC *PDC, BYTE State); //直线绘制
    BOOL IsSelected(POINT point); //是否被选中
};

```

## 3. CPolyLine 类

```

class CPolyLine : public CFigure
{
protected:
    CPolyLine();
    DECLARE_SERIAL(CPolyLine)
    UINT m_nNodeNumber; //折线结点数
    NodeStruct* m_pNodes; //折线结点数组
};

```

```

public:
    virtual ~CPolyLine();
    CPolyLine(NodeStruct* , UINT , COLORREF, BYTE, UINT, UINT);
    virtual void Serialize(CArchive& ar);           //折线存储
    virtual void Draw(CDC *PDC,BYTE State);       //折线存储
    BOOL IsSelected(POINT point);                //是否被选中
};

```

#### 4. CPolygon 类

```

class CPolygon : public CFigure
{
protected:
    CPolygon();
    DECLARE_SERIAL(CPolygon)
    UINT m_nNodeNumber;           //多边形边数
    NodeStruct* m_pNodes;        //多边形结点数组
public:
    virtual ~CPolygon();
    CPolygon(NodeStruct* , UINT , COLORREF, BYTE, UINT, UINT);
    virtual void Serialize(CArchive& ar);       //多边形存储
    virtual void Draw(CDC *PDC,BYTE State);    //多边形绘制
    BOOL IsSelected(POINT point);             //是否被选中
};

```

#### 5. CEllipse 类

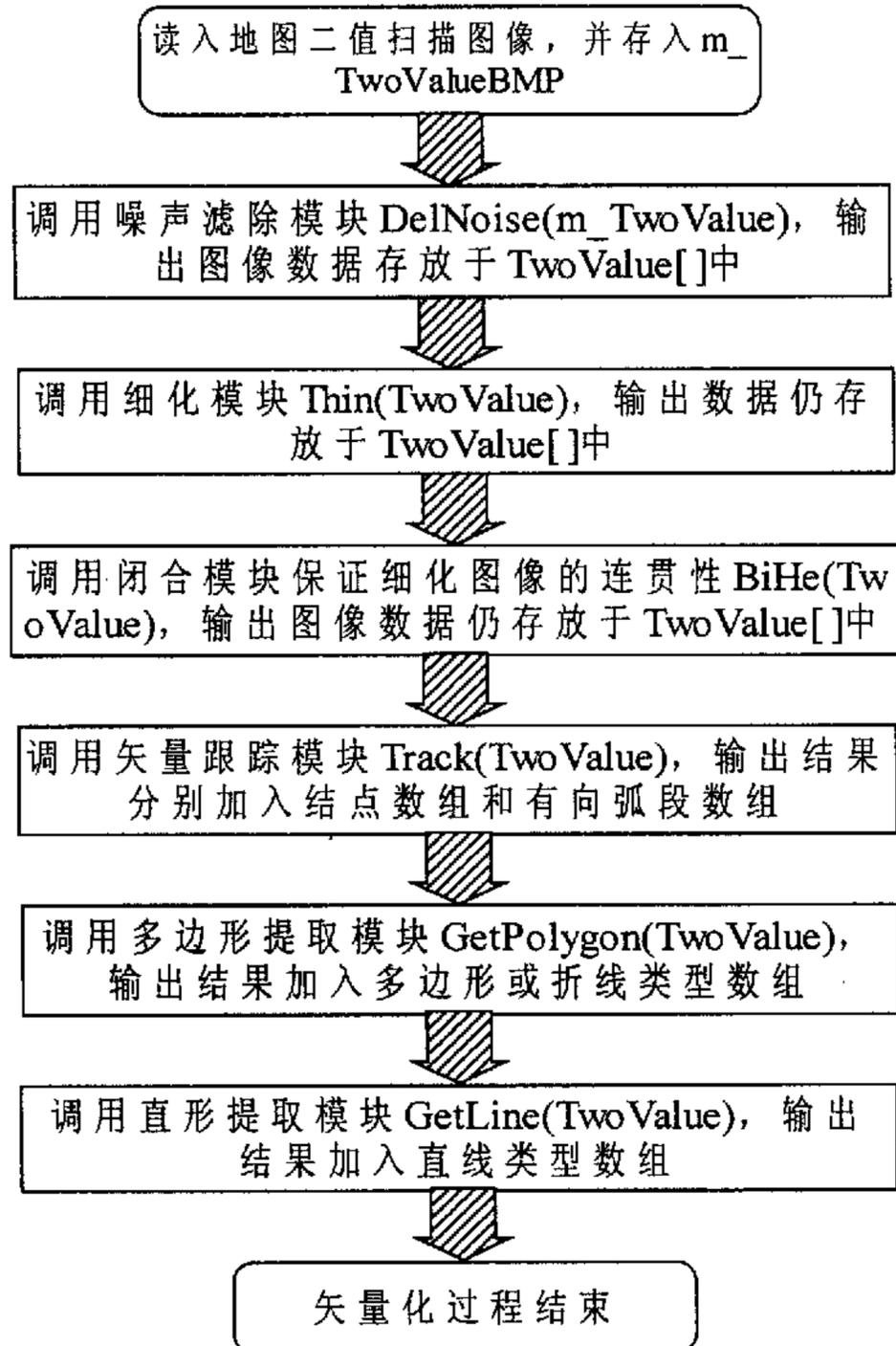
```

class CEllipse : public CFigure
{
protected:
    CEllipse();
    DECLARE_SERIAL(CEllipse)
    Float m_x1, m_y1, m_x2, m_y2; //椭圆外接矩形左上, 右下点坐标
public:
    virtual ~CEllipse();
    CEllipse(float, float, float, float, COLORREF, BYTE, UINT, UINT);
    virtual void Serialize(CArchive& ar);       //椭圆存储
    virtual void Draw(CDC *PDC,BYTE State);    //椭圆绘制
    BOOL IsSelected(POINT point);             //是否被选中
};

```

### 4.2.2 系统实现总体流程

模块化的编程思想贯穿系统实现的全过程，每个模块的输出在下一个模块使用过后就可以释放其所占空间，到最后图形识别阶段只需要保留结点数组和弧段数组，从而使图像的数据量逐步减少，大大的节省了内存空间。而且便于系统功能的调试、维护、测试和移植。按照设计的矢量化步骤，可将整体流程分为 6 大块，如流程图 4.2.1 所示：



流程图 4.2.1 矢量化总体流程

Fig.4.2.1 Whole vectorization process

### 4.3 矢量跟踪算法

在地图扫描输入计算机后，经过细化处理的图形已变成线宽只有 1bit 的单

线骨架图形，但仍是一个简化了的二值图像，它并未包含图形的拓扑信息，因此，我们还不能直接进行矢量化处理。矢量自动跟踪则是实现这一过程的桥梁，它的作用就是在细化图像的基础上按一定规律对骨架点进行整理分类，把整幅图像分割成一段段点列，并记录下有关拓扑信息，以便以后作进一步识别处理，因此，矢量跟踪在矢量化过程中起着十分重要的作用。

我们知道图的最基本单元是用两点表示线段。因此，对于点列分段，也可采用同样的方法，即在端点或交点处将图像断开，从而将整幅图分成若干段点列，并按一定的规律进行存储，形成矢量数据。这也就是矢量自动跟踪算法的基本思想。

矢量跟踪转换较为通用的方法是采用 Freeman 链码<sup>[33]</sup>。这种方法首先通过规定步进的方向码，然后做线段跟踪，得到一组图像模式骨架线的链码表示，对链码数据进行处理，即可得到其相应的矢量数据。这种方法处理起来比较繁琐，不太适应地图处理的需要。另一种矢量化转换方法是采用扫描线轮廓跟踪，即规定优先跟踪方向，逐行扫描图像文件，得到一起始点后做线段追踪，记录终止点前全部特征位置矢量。这种方法容易破坏图像中的网络拓扑关系，对后续的基本图素及其属性的提取、识别产生困难。

本文采用了一种基于网络拓扑结构结点提取并通过查表进行矢量跟踪的矢量转换方法。不仅保持了原图的拓扑完整性，而且构造简单，复杂性只与被跟踪的象素点个数有关，有利于软件实现，适用于大数据量的地图图形。

### 4.3.1 基本概念



图 4.3.1 象素 p 及其近邻表示与编码位置

Fig.4.3.1 Pixel p and its neighbor codes responding to their position

图 4.3.1 是 3×3 模板窗口。其中，(a)表示当前象素 p 与其近邻的关系；(b)是矢量跟踪中使用的当前象素 p 的近邻位置编码。

定义 1: 像素 p 的分枝数

经过细化得到的 8-连通骨架上象素 p 的近邻 N(p)中值为 1 的像素的个数，称作像素 p 的分枝数。

p 的分枝数实际上是 N(p)中的元素构成的一个序列 x<sub>0</sub>, x<sub>1</sub>, x<sub>2</sub>, x<sub>3</sub>, x<sub>4</sub>, x<sub>5</sub>, x<sub>6</sub>, x<sub>7</sub>, x<sub>0</sub> 中 01 模式的个数为

$$B_r(p) = \frac{1}{2} \sum_{i=0}^7 |x_{i+1} - x_i| \quad (4-1)$$

定义 2 分枝数为 1、3、4 的像素  $p$  为分枝点；分枝个数为 2 的像素  $p$  为连接点；跟踪中步进方向改变的连接点称为拐点。分枝点又可分为端点、交叉点及多分枝点，如图 4.2.2 所示

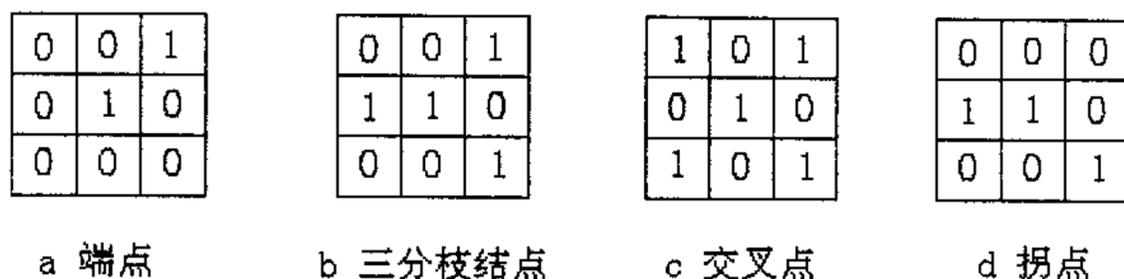


图 4.2.2 3×3 模板中心像素分枝类型

Fig.4.2.2 4 kinds of 3×3 template's central pixel

### 4.3.2 基于查表的矢量跟踪算法

#### 1. 算法描述

跟踪过程分为两步，第一步先提取网络装结构图形中的所有分枝点，并存放于一个结点的结构数组  $Nod[]$  中，数组中每个元素都是一个结构元素，记录了该结点在像素数组中的下标、分枝个数和该结点所连接弧段在弧段数组中的位置；第二步，顺序从结点数组中取出某一结点  $p_n$ ，再依次按照  $p_n$  的分枝方向进行矢量跟踪，每一分枝在下一分枝点处结束。对于闭合曲线则通过确定一起始点，从该点出发作轮廓跟踪，直至回到初始位置。

在实际的线图结构中，分枝点和交点的情况是十分复杂的，很难找到一个完全正确的判定通式、我们根据地图的实际情况以及细化算法的特点，以某像素  $p$  的  $B_r(p)$  值 ( $p$  的 8 邻域中象素的数目) 和连接数为基础，提出了一个比较完善的判定条件，用这种判定条件对地图扫描图像进行处理，取得了比较好的效果。判定条件描述如下：

经过细化后，骨架像素  $p$  的分枝构成如图 所示，其中，标记为  $x$  的像素群位置上至少有一个为 1。这里包括(a), (b), (c)三种情况的连续  $90^\circ$  旋转。上述方法中的分枝结点提取就是从骨架上抽出所有形如图 所示的分枝点。分枝点的提取及线段的矢量跟踪在算法实现中，均通过查表加以实现。

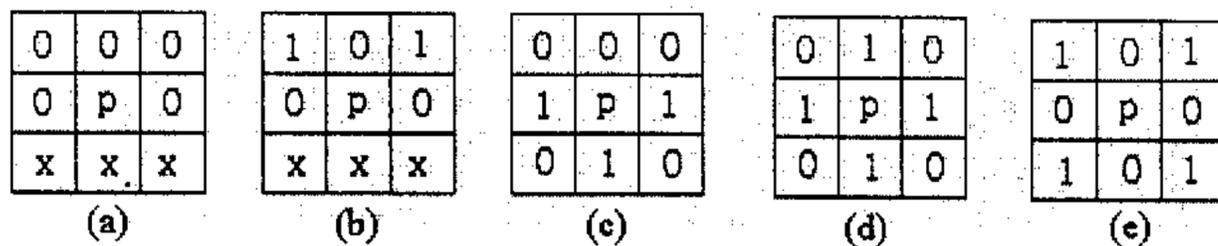


图 4.2.3 分枝点提取模板

Fig.4.2.3 Template to get ramose node

对于图 4.2.4 中想找出分枝点  $p$  是非常困难的，针对这个问题，我们从前面地图的交点情况分析以及所采用细化算法的特点出发，进行了观察和实验，

发现细化后的地图中的折线相交的情况基本不超过二线，三折线以上相交的情况非常少，因而总结的判定条件中对这种情况没有包括在内。

1	0	0	1	1
0	1	0	1	0
1	1	p	1	1
0	0	1	1	0
0	0	1	0	1

图 4.2.4 特例

Fig.4.2.4 Exception

## 2. 算法构造与实现

本算法采用  $3 \times 3$  模板窗口对细化后的骨架进行处理。这样，每一步操作只与当前像素  $p$  的 8 个近邻的形态有关，而像素  $p$  的 8 个近邻的形态组合共有  $2^8 = 256$  种，使得整个算法可以通过构造一个表并依赖于该表来完成。在内存中开一段 256 字节的缓冲区，缓冲区数组的下标对应像素  $p$  的 8 个近邻的某一组合方式，相应地址位置存放改形态下进行某种操作的标志。如图 4.2.5 所示，当给定一种近邻的组合方式时，其在缓冲区中相应地址为

$$A(p) = \sum_{i=0}^7 2^i \cdot a_i \quad (4-2)$$

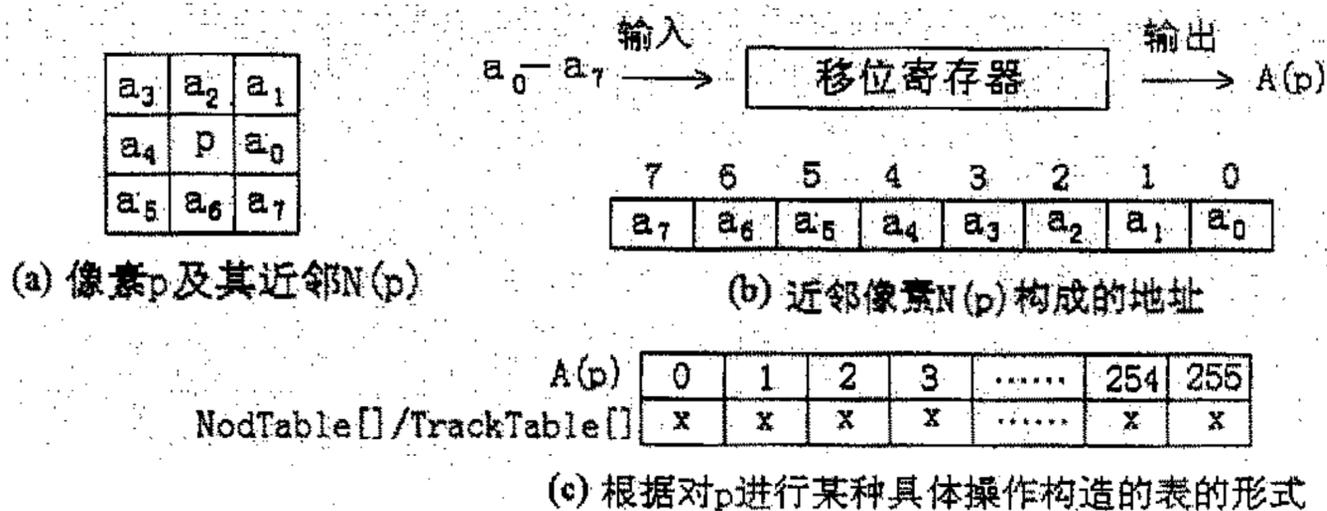


图 4.2.5 表的构造

Fig.4.2.5 Build the list

这里， $a_i(0 < i < 7)$  为 0 或 1，是图中  $3 \times 3$  模板相应位置的像素值， $A(p)$  为该种组合映射到缓冲区中的相应地址位置，对该种组合进行何种操作的标志放在  $NodTable[A(p)]/TrackTable[A(p)]$  的相应单元。

本方法根据一定的规则构造两个表。一个是  $NodTable[]$  用于结点提取；另一个是  $TrackTable[]$  用于跟踪时确定操作方式；设计一个循环过程，对当前像素  $p$  取近邻  $N(p)$ ，构造地址  $A(p)$ ，按  $A(p)$  从  $NodTable[A(p)]/TrackTable[A(p)]$

中得到操作标志，依据该标志进行操作，直到操作结束。

#### (1). 网络拓扑结点的提取

确定 0—255 所对应形态分枝的个数，并在分枝个数为 1, 3, 4 的分枝点，分枝数为 2 的拐点时，将 NodTable[ ] 中相应地址位置上置 1，其他情况置 0。提取过程如下：

开始

For(像素个数=0; 像素个数<全体像素个数; 像素个数+1)

IF(像素不为 0)

取当先像素 p 的近邻 N(p);

构造地址 A(p);

IF(NodTable[A(p)]不为 0)

把像素的位置写进结点数组 Nod[ ]中;

结束

#### (2). 矢量跟踪算法

构造矢量跟踪表 TrackTable[ ]的关键点是步进规则及终止条件的确定。步进规则是指在矢量跟踪过程中下一个搜索方向的确定，终止条件是指线段被跟踪过程的结束条件。

根据分枝点和拐点的各种形式将终止条件划分为 3 类：

1) 当前点为分枝结点，根据另外的准则判定应删除时，TrackTable[ ]表中相应位置的元素记为 ox10;

2) 当前点为分枝结点，根据另外的准则判定不可删除时，TrackTable[ ]表中相应位置的元素记为 ox20;

3) 当前点为端点，TrackTable[ ]表中相应位置的元素记为 ox30;

4) 当前点为拐点时，即在跟踪某一分枝过程终止前，步进方向发生改变时。步进方向依据图 4.2.1 中编码位置 0, 1, 2, 3, 4, 5, 6, 7 上何处像素为 1 来确定，并把表 TrackTable[ ]相应位置上的元素值分别用 ox00, ox01, ox02, ox03, ox04, ox05, ox06, ox07 来表示。

上面提到的另外的准则指的是每个像素检索一遍后，分枝数减 1，若当前点检索时的分枝数等于 1，则从结点数组中删除该点。

按照上述描述，能够依据 0—255 所对应的近邻近邻形态完成 TrackTable[ ]表的构造。流程图 4.2.6 表示了跟踪的过程。

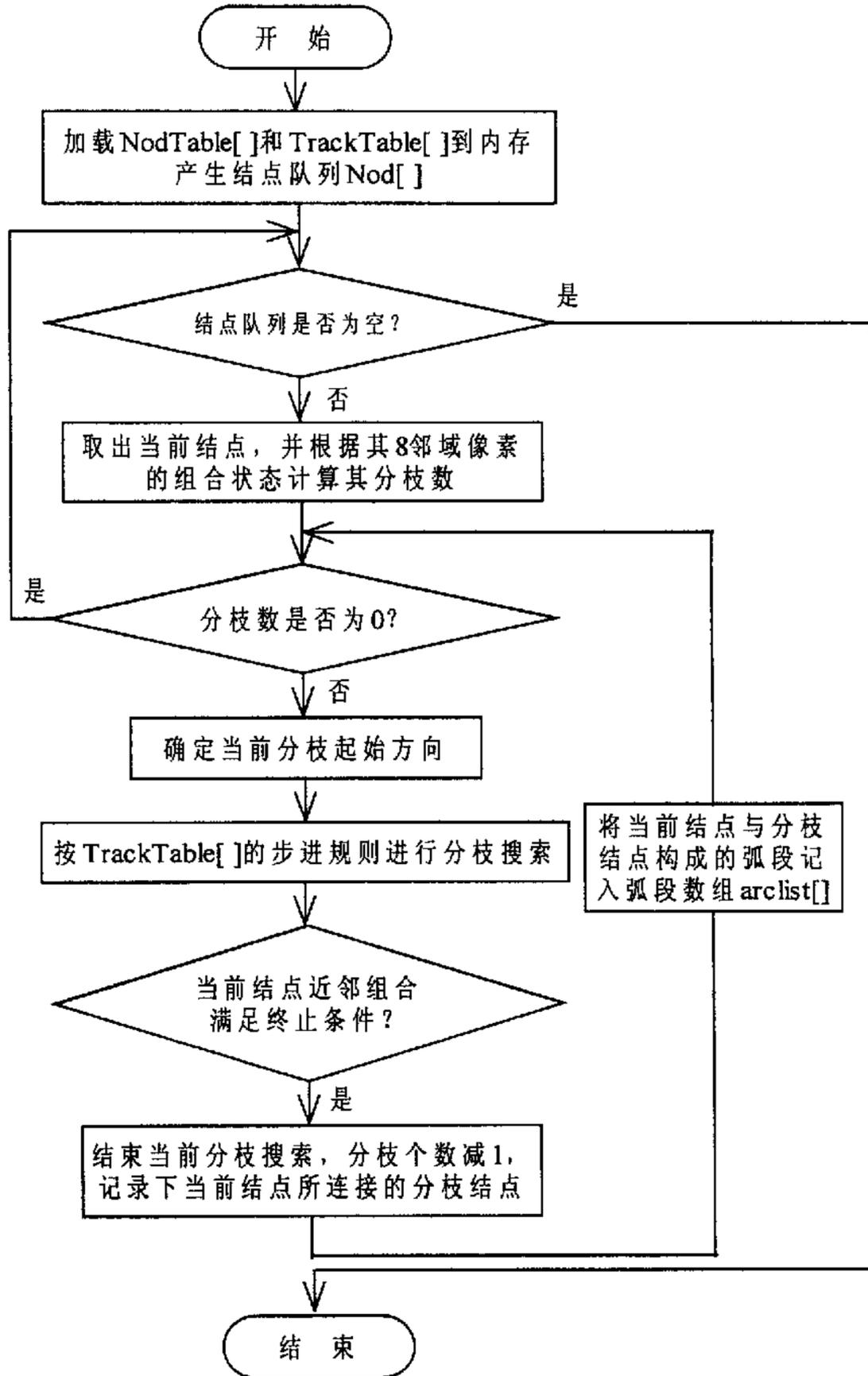


图 4.2.6 表的建立过程

Fig.4.2.6 Process to build the list

#### 4.4 多边形的提取与识别

在地理信息系统中，“原始数据到多边形拓扑关系的转换”是一个非常重要的问题。这个问题被国际摄影测量与遥感协会 (ISPRS) 列入第 III 委员会的研究重点。其中地理实体多边形拓扑关系的自动建立是该问题中的一个重要过程。

#### 4.4.1 传统算法比较

目前,对拓扑多边形的自动构建算法,学者们的研究侧重点各异,但基本都是从自动化程度、时间效率和算法的复杂性出发来进行优化与改进,其早期的算法一般都离不开人工干预(如输入内点、多边形编码等)<sup>[34]</sup>。目前一些算法也要事先手工输入多边形内点以帮助拓扑关系的建立<sup>[35-38]</sup>。在这类基于手扶数字化仪采集原始数据的算法中,过多的人工操作降低了过程的自动化水平,在拓扑关系建立之前,不能借助多边形自动寻址的方式输入内点,使得交互输入内点的过程不方便,也不利于寻找并消除矢量化过程中产生的一些没有地理意义的微小多边形,对于保证多边形数据的完整性不利。

自动建立结点上弧段—弧段之间拓扑关系时,传统算法是以角度参数进行弧段之间的排序,而这需要通过计算反三角数  $\tan^{-1}(x)$  来求出角度,算法的低效性显而易见,也有人提出利用  $\tan^{-1}(x)$  的泰勒级数展开式来计算角度的近似值,算法的效率得到提高,但泰勒级数展开式的复杂性不尽如人意,其精确度也不是很高。文献<sup>[39][40]</sup>提出 Qi 算法,把  $\tan^{-1}(x)$  的计算置换为 Qi 函数值的计算,在时间效率上有了较大的改进,自动化程度也较高,然而在计算 Qi 值时需要进行比值运算和弧段所在象限的判断,因此也不是很理想。文献<sup>[41]</sup>出了基于方位角计算的多边形自动构造方法,结构清晰,简单易懂,程序设计易于实现,但仍没有脱离传统基于角度参数识别的范围。

本文提出了一种基于矢量外积的多边形自动构建方案,运用计算几何学原理,通过各线段的矢量外积直接进行比较排序,确定结点上弧段间的连接关系。其基本思路是:(1) 结点匹配,建立结点和弧段的连接关系;(2) 建立结点上弧段之间的相邻关系;(3) 多边形搜索;(4) 拓扑关系确定。该算法避免了以往算法中参数值计算和其它烦琐的数学运算,生成算法明显优化了代码并提高了算法的执行效率,对于整个多边形自动生成系统来说是一大改进。

#### 4.4.2 数学背景

##### 1. 多边形拓扑信息生成的数学基础

###### (1). 多边形图的性质

多边形拓扑信息是指一个多边形由哪些弧段组成及其组成弧段在多边形上的邻接关系。多边形图的拓扑信息是指图上全部多边形的拓扑信息及这些信息间的关系。

GIS 中的多边形图,输入的是弧段,所要描述的是连通区域,拓扑信息是用弧段表示连同区域必不可少的数据。

封闭多边形拓扑信息的形成可简单描述为:沿任意一条弧段前进,到达终点时向左转,顺次遍历所有与该终点相连接的弧段。沿着最后遇到的一条弧段继续前进,如此重复直至回到出发点,依次经过的弧段即形成的弧段形成一个

封闭的多边形。这一算法思想可以形成单个封闭多边形，对于整个一幅多边形图，含有的多边形常常不止一个，可重复形成一些多边形，也可能遗漏一些多边形。要是每一个多边形的拓扑信息形成一次且仅形成一次可根据下述定理公理设计算法。

定义 1: 平面上由一条不自相交的封闭曲线构成的图形叫最简多边形图。

公理 1: 最简多边形图含有两个多边形。

定义 2: 从最简多边形图上引出有限条互不相交且不与任何曲线相交的自封闭曲线所形成的图形叫多边形图；在多边形图上起止于多边形的边，引有限条互不相交曲线，且不与任何已有的曲线相交，形成的图形也叫多边形图，所引的曲线叫弧段。

公理 2: 多边形图的每条弧段由两个且仅由两个多边形所共有。

可见，多边形的弧段具有方向属性，定义了方向的弧段称之为有向弧段。一幅完整的多边形图上，有向弧段的总数是弧段总数的 2 倍。

(2). 理论在实际应用中的有用结论

结论 1 多边形图中的每个多边形(无论是简单多边形还是复合多边形)，表示一个连通区域，按左转方法定义的连通区域的边界线具有方向性；每条有向弧段是唯一的连通区域的边。

结论 2 顺次扫描有向弧段文件，每遇到一条有向弧段，将其编号写入多边形拓扑信息文件中，文件扫描完后，拓扑信息形成完毕。由此得到的拓扑信息，对于整幅多边形图协调一致。

结论 3 每条弧段有两个方向，代表两条有向弧段。可对弧段定义方向标志数据。可用带方向标志数据的弧段文件代替有向弧段文件，以减少数据的重复记录。

2. 计算几何学基础

在计算几何学中，矢量外积是各种线段算法的基础，下面是矢量外积的有关知识。如图 4.3.1 所示，二维平面上—矢量  $p$ ，则矢量  $p$  有其顺时针区域和逆时针区域。如图 4.3.2 所示，二维直角坐标平面  $x \ o \ y$  中，从坐标原点出发的两条矢量  $P_1(x_1, y_1), P_2(x_2, y_2)$ ，则有矢量  $P_1, P_2$  的矢量外积为  $P_1 \times P_2$ 。

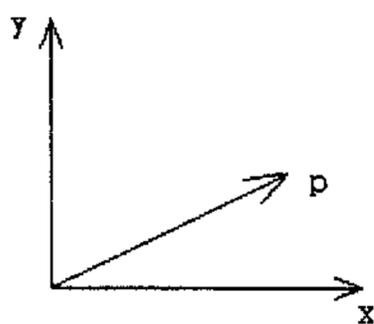


图 4.3.1 单一矢量

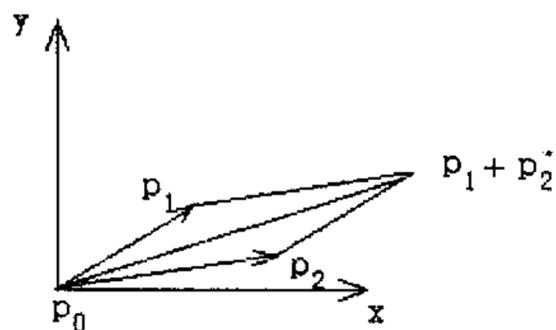


图 4.3.2  $p_1$  与  $p_2$  的矢量外积

Fig.4.3.1 Single vector Fig.4.3.2 The exterior product of  $p_1$  and  $p_2$

$$P_1 \times P_2 = \begin{bmatrix} x_1 & x_2 \\ y_1 & y_2 \end{bmatrix} = x_1 \cdot y_2 - x_2 \cdot y_1 \quad (4-3)$$

$p_1$  和  $p_2$  的矢量外积表示是由点  $p_0, p_1, p_2, (p_1+p_2)$  所围成的平行四边形的面积  $S$ 。已知  $S = p_1 \times p_2 = -p_2 \times p_1$ , 如果  $p_1 \times p_2 > 0$ , 则相对于  $p_0$  来说, 矢量  $p_1$  位于  $p_2$  的顺时针方向上; 如果  $p_1 \times p_2 < 0$ , 则相对于  $p_0$  来说, 矢量  $p_1$  位于  $p_2$  的逆时针方向上; 如果  $p_1 \times p_2 = 0$ , 即边界条件情况, 矢量  $P_1$  与  $P_2$  共线, 方向相同或相反。

### 3. 多边形搜索的最小角法则

一条弧段可作为一个或两个多边形的组成边而存在, 亦即一条弧段最多可以搜索出两个正确的多边形。如图 4.3.3 所示, 若从弧段  $A_1$  的一端  $O$  出发, 并把它作为起始弧段, 把与  $A_1$  的  $O$  端拓扑关联的其它弧段作为终止弧段, 然后比较并找出与  $A_1$  顺时针方向夹角最小的终止弧段  $A_2$ , 并把  $A_2$  作为新的起始弧段, 再从它的另一端出发重复以上过程继续搜索, 直到回到出发弧段  $A_1$  的另一端为止, 则所搜索出的弧段就构成了一个多边形。同样, 从  $A_1$  的  $O$  端开始, 并把它作为终止弧段, 把与它拓扑关联的其他弧段作为起始弧段, 然后比较并找出与该弧段夹角最小的弧段, 并把找出的弧段作为新的弧段作为终止弧段, 再从新弧段的另一端点出发重复以上搜索过程, 直到回到  $A_1$  的另一端为止, 则所有搜索的弧段就构成了另一个多边形。这样, 从一条弧段出发可以跟踪出两个多边形, 此方法可称为多边形搜索的最小角法则。

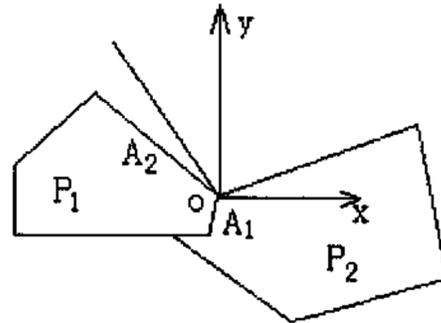


图 4.3.3 运用最小角法则搜索多边形

Fig.4.3.3 Search polygon by min-angle principle

### 4.4.3 结点的匹配、弧段间邻接关系的确定

#### 1. 结点匹配

在矢量跟踪过程中, 由于同一结点被多次采样, 使它对应着不同的坐标值。这种差异需要通过结点匹配来消除。而且结点匹配还可以对由细化造成畸变的结点进行修正。

弧段结点匹配的算法思想是: 根据同一结点对应的坐标唯一的原则, 结合实际情况给定阈值  $\epsilon$  以  $2\epsilon$  为边构成最小间隙盒。若几个采样结点落在最小间隙盒中, 则认为它们为同一逻辑结点, 多个结点不匹配情况如图(a)所示, 结点

畸变情况如图(c)所示:在不考虑算法优化的情况下,每一个结点匹配需要对结点文件遍历一次,可取采样结点坐标的平均值作为该结点的最后坐标,如图(b),图(d)所示,记录该结点所连接弧的标识。重复以上过程,直到所有的结点匹配完成为止。按结点匹配成功的先后顺序从1开始编码。结点标识在结点匹配过程中完成。

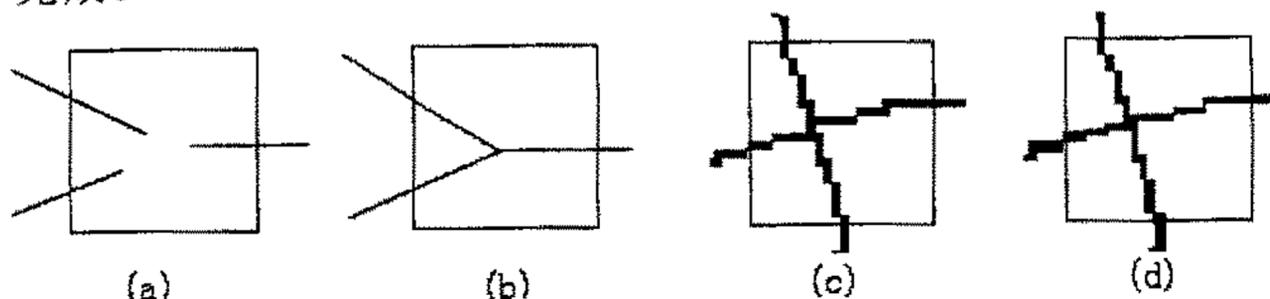


图 4.3.2 结点匹配

Fig.4.3.2 Match the nodes

## 2. 弧段间邻接关系的确定

多边形区域的自动生成是利用结点和弧段的信息,根据指定的方向(顺时针或逆时针),由系统通过弧段文件自动生成组成多边形的弧段信息,按照左转算法(或右转算法)的思想,对于当前有向弧段,需对从汇集于同一结点的所有弧段进行排序。

通常的算法是先计算出结点上各条弧段的参数值,再通过对参数值的比较完成弧段的排序。这种算法思想过于麻烦,实质上这些弧段进行排序的目的是确定各条弧段之间的相对位置关系,没有必要计算出每条弧段的参数值,得出每条弧段的绝对位置,再通过比较来确定弧段之间的相对位置关系。利用弧段之间矢量外积的比较各条弧段之间的相对位置关系可以直接得出。

### (1) 同一结点上任意两条弧段位置关系判断

设该结点为  $p_0(x_0, y_0)$ , 从结点  $p_0$  出发的  $n$  条弧段的另一端点序列为  $V$ ,  $V = \{p_1, p_2, \dots, p_n\}$ , 其中  $p_i = (x_i, y_i)$ ,  $i \in [1, \dots, n]$ 。对于弧段序列  $V$  中任意两条弧段  $s, t$ , 端点为  $p_s(x_s, y_s), p_t(x_t, y_t)$ , 则有弧段  $s, t$ , 相对于点  $p_0(x_0, y_0)$  的矢量外积为  $p_s \times p_t$ :

$$p_s \times p_t = \begin{bmatrix} x_s - x_0 & x_t - x_0 \\ y_s - y_0 & y_t - y_0 \end{bmatrix} = (x_s - x_0) \cdot (y_t - y_0) - (x_t - x_0) \cdot (y_s - y_0) \quad (4-4)$$

由  $p_s \times p_t$  的正负号可得出弧段  $s, t$  的相对位置关系。

### (2) 同一结点上任意弧段的排序

利用矢量外积对弧段进行排序,首先从原始序列中任意选取一弧段作为基准弧段,将位于基准弧段逆时针方向上的弧段置于序列中的该弧段之前,而位于基准弧段顺时针方向上的弧段置于序列中基准弧段之后。对于边界条件,将该弧段置于基准弧段之前处理;最后利用排序方法通过矢量外积的比较完成弧段的排序。

程序采用二叉树实现利用矢量外积对弧段的排序过程,使用逐步插入结点的方法来构造一棵二叉树  $Q$ :

设原始弧段序列  $V=\{p_1, p_2, \dots, p_n\}$ , 从  $p_1$  开始, 每次选取一个弧段元素  $p_i$ , 按下列原则建立二叉树  $Q$  的一个新结点: ①若二叉树为空, 则将新的弧段元素作为该二叉树的根结点; ②若二叉树  $Q$  非空, 则将新的弧段元素  $p_i$  与二叉树  $Q$  的根结点弧段(基准弧段)  $p_0$  进行矢量外积比较, 设  $S=p_i \times p_0$ , 如果  $S \leq 0$ , 则将新的弧段元素  $p_i$  插入到根结点的左子树中; 如果  $S < 0$ , 则将新的弧段元素  $p_i$  插入到根结点  $p_0$  的右子树中。

该过程为一递归过程, 每一新的弧段元素  $p_i$  插入到左子树或右子树也同样按照这个原则递归执行。

如图 4.3.3 所示, 从一结点出发的六条弧段 0—5, 设原始的弧段序列  $V=\{0, 2, 1, 4, 3, 5\}$ , 按照上述步骤执行排列过程, 所生成二叉树结构如图 4.3.4 所示。

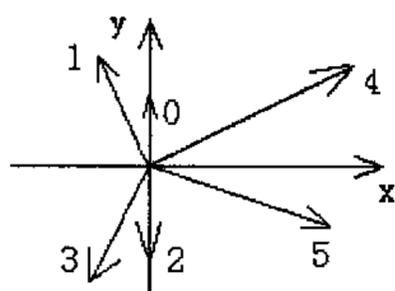


图 4.3.3 弧段位置关系图

Fig.4.3.3 Relation between segments

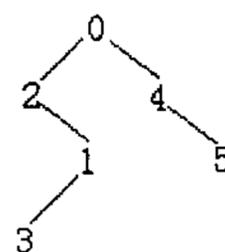


图 4.3.4 二叉树结构图

Fig.4.3.3 Binary tree structure

建立好二叉树后, 只需对此二叉树进行中序遍历即可得到排列好的弧段序列。对于上面的二叉树进行中序遍历后的弧段序列为  $\{2, 3, 1, 0, 4, 5\}$ 。从而建立起同一结点上弧段之间的拓扑关系。

如图 4.3.3 所示, 与弧段 0 顺时针夹角最小的弧段为弧段 4, 即在进行二叉树排序后的弧段序列中位于弧段 0 右侧的第一条弧段。因而, 本系统在多边形搜索过程中与前面介绍的多边形搜索的最小角法则相结合, 在按照外积关系组成的弧段序列中, 取当前弧段右侧弧段中第一条弧段为与其夹角最小的弧段。

#### 4.4.4 拓扑信息的生成

1. 从弧段文件中读取一条有向弧段作为多边形的第一条有向弧段及当前有向弧段, 并从弧段数组中删除它。

2. 找出以当前有向弧段的终结点  $p$  为始结点且不与当前有向弧段重合的全部有向弧段。按照以该弧段为基准弧段所建立的二叉树, 在对二叉树进行中序排列后组成的弧段序列中, 选出该弧段右侧第一条有向弧段作为多边形的下一条有向弧段, 并从弧段数组中删除该基准弧段。

3. 若选出的有向弧段的终结点与当前多边形的第一条有向弧段的始结点不重和则转步骤(4), 否则, 已形成一个闭合多边形, 继续判断: 若弧段数组已空, 则拓扑信息生成完毕, 结束搜索; 若非空转步骤(1), 继续形成下一个多边形。

4. 将选出的有向弧段作为当前有向弧段转步骤(2)。

重复以上过程,直到多边形图上所有的有向弧段被采样一次为止。

## 4.5 直线的提取与识别

在地图中,直线可以分成一般直线和任意角度直线。一般直线是指  $0^\circ$ ,  $45^\circ$ ,  $90^\circ$  和  $135^\circ$  的直线;任意角度直线是指除一般直线以外的所有直线,然而这些直线实际上都是一般直线组合而成的。本节分两部分介绍采用的针对两种直线进行的矢量化算法。多边形和折线提取出来后,在结点数组  $Nod[]$  中未被检索的就是直线的端点和任意角度直线上被误认为折线拐点的象素点了。

### 4.5.1 一般直线的矢量化算法

一般直线是指  $0^\circ$ ,  $45^\circ$ ,  $90^\circ$  和  $135^\circ$  的直线,也就是说对于直线上每一个像素的下一个跟踪点与其上一个像素进行跟踪时的跟踪方向相同。如图 4.5.1 所示

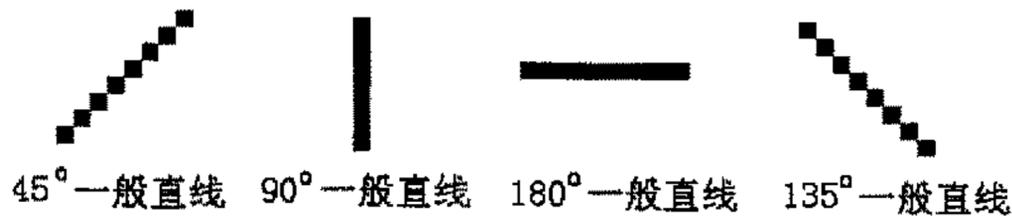


图 4.5.1 一般直线类型

Fig.4.5.1 Normal lines

这样的直线跟踪方向固定,易于跟踪,几何特性便于提取。所参阅的直线矢量化文献,基本上都是针对一般直线进行的处理。

#### 1. 算法描述

输入结点数组  $Nod[]$ ,从端点象素点进行跟踪,如果能够保持跟踪方向不发生改变达到另一个端点象素点,则该直线为一般直线。

第一步:读入结点数组  $Nod[]$ ,寻找端点象素点  $Nod[i]$ 作为跟踪起始点,将其记录为直线起点。

第二步:对跟踪起始点进行跟踪,将其邻域象素点  $TwoValue[j]$ 作为跟踪起始点,并记录跟踪方向  $s$ ;

第三步:对  $TwoValue[j]$ 的邻域进行跟踪,如果邻域中象素点  $TwoValue[k]$ 是端点,则一条一般直线跟踪结束,将其记录为直线终点,从  $Nod[]$ 中删除两个端点,并转到第一步;否则将  $TwoValue[k]$ 作为跟踪起始点,若跟踪方向与  $s$ 相同转到第二步,若与  $s$ 不同详见任意角度直线处理方法。

2. 流程图

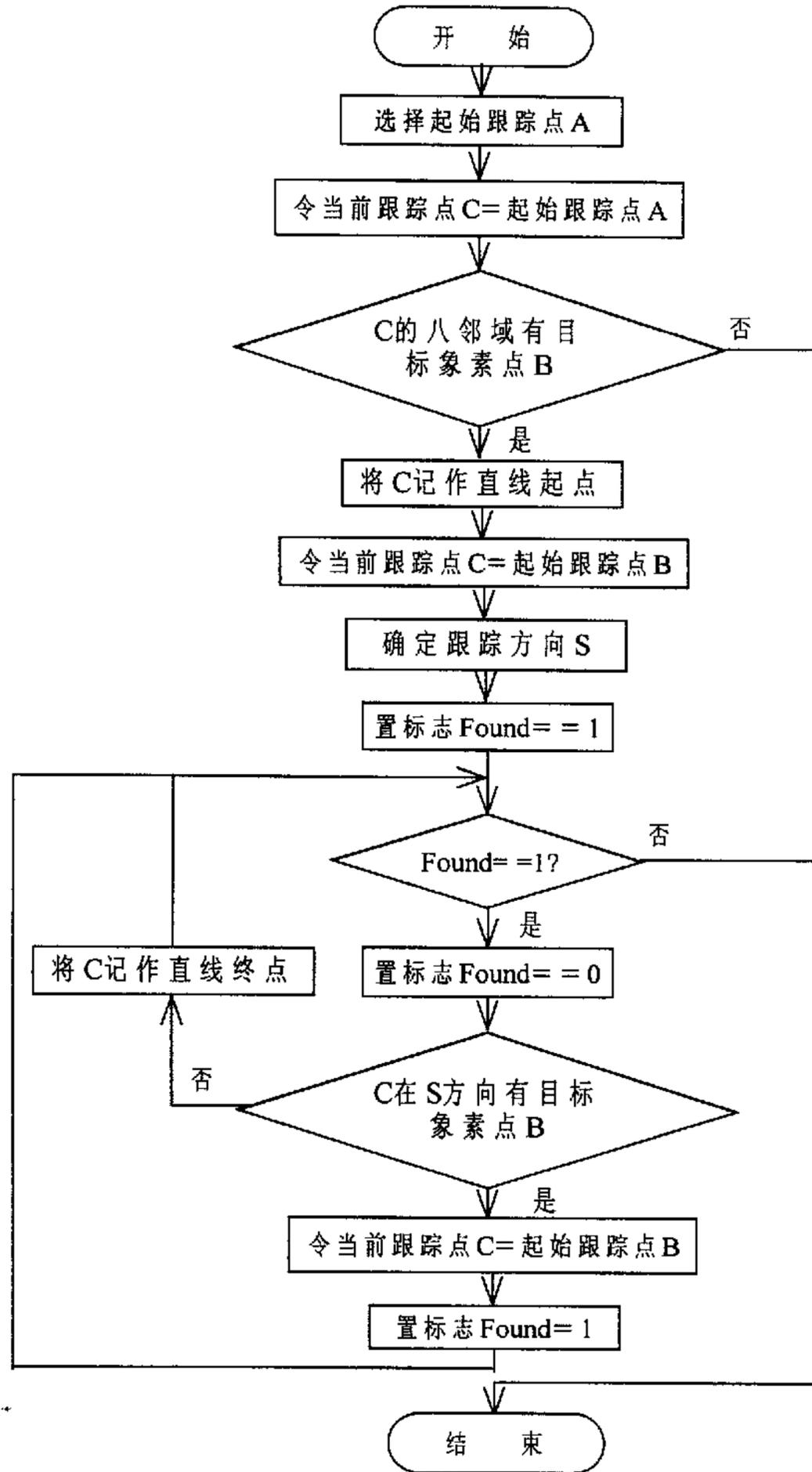


图 4.5.2 一般直线形成过程

Fig.4.5.2 Form normal lines

## 4.5.2 任意角度直线的矢量化

### 1. 算法描述

输入结点数组  $Nod[]$ ，从端点像素点进行跟踪，对组成任意直线的一般直线进行跟踪，直至达到另一个端点像素点，则该直线为任意直线。

第一步：读入结点数组  $Nod[]$ ，寻找端点像素点  $Nod[i]$  作为跟踪起始点。

第二步：用结构数组  $K[100]$  记录任意直线的起始点坐标。

第三步：从  $Nod[i]$  调用一般直线算法跟踪一条直线段  $L[j]$ 。将直线段  $L[j]$  的中点记录为该任意直线的起点。

第四步：并记录跟踪方向  $s$ ， $j++$ 。

第五步：对  $L[j]$  的终点邻域像素点进行判断，在  $s+1$  或  $s-1$  方向上如果有邻接像素点  $TwoValue[k]$ ，将其作为跟踪起始点，调用一般直线算法跟踪一条直线段  $L[j]$ ，并转到第四步。否则，将  $L[j]$  的中点记录为该任意直线的终点。从  $Nod[]$  中删除该直线涉及的端点。

第六步：判断  $Nod[]$  中是否为空，为空则算法结束，否则转至第一步。

### 2. 流程图

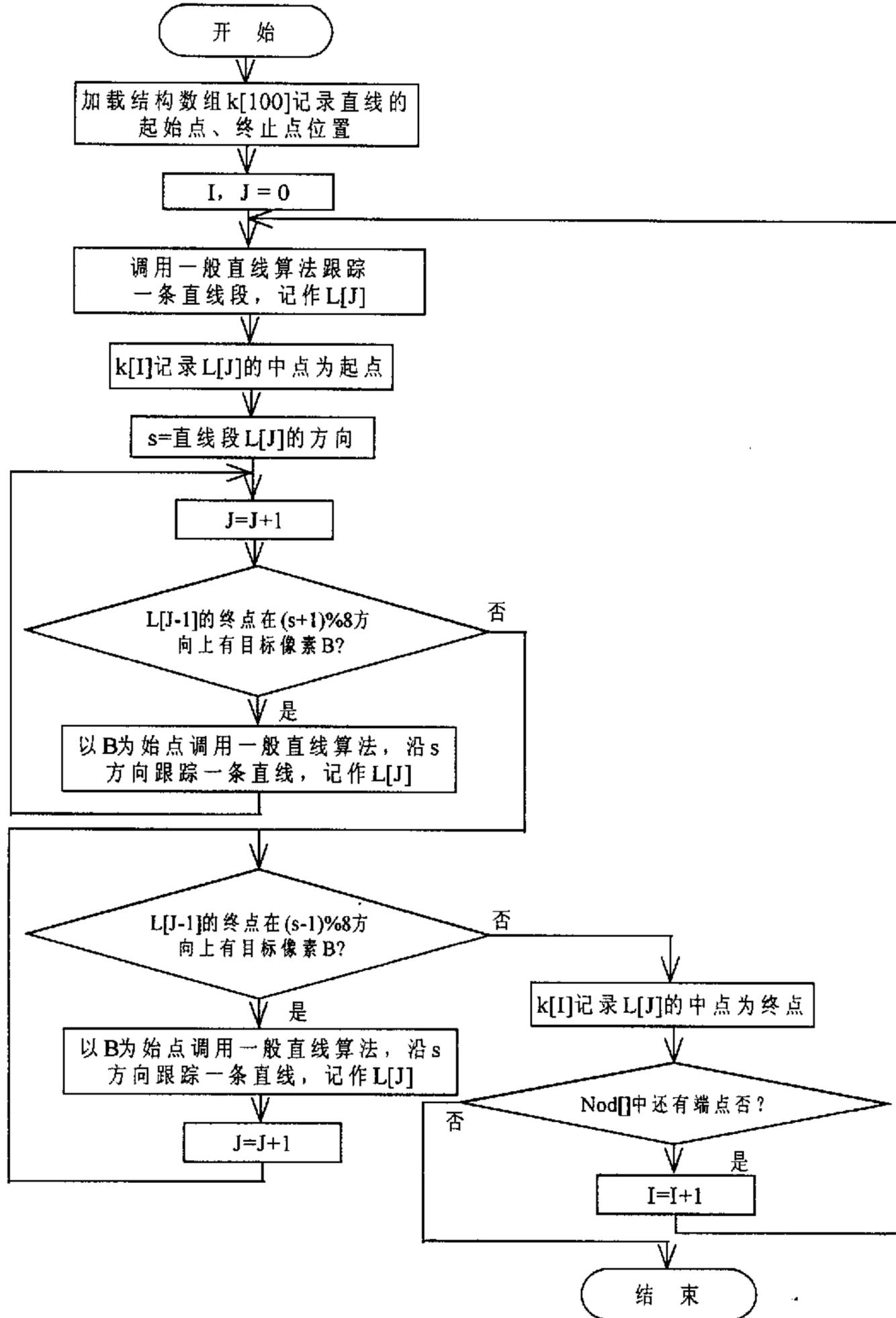


图 4.5.2 任意角度直线形成过程  
Fig.4.5.2 Form random-angle lines

## 5 实验系统功能

本章是在前面提出的矢量化算法基础上,在 Windows 2000 平台下, Visual C++ 编程环境中<sup>[42][43]</sup>,使用 C++ 语言设计并实现了一个光栅图矢量化软件实验系统。该软件系统将智能自动数字化技术与方便易用的菜单驱动图形用户界面有机地结合到 Windows & NT 环境中。是一个可以用扫描光栅图像为背景的矢量编辑工具。下面就对系统的基本的转换和编辑的功能和使用做以介绍。

### 5.1 系统功能介绍

本实验系统具有如下基本功能:

- (1) 栅格图形的读取、显示和存储。
- (2) 基本图形元素的绘制、选取和删除; 栅格图像及矢量图形的移动和缩放。从而成功实现对图形的修正。
- (3) 二值图像的处理: 噪声去除、细化、数学形态学的闭合运算。
- (4) 图形的识别提取: 本文能够较成功的提取出直线, 折线和多边形。

#### 1 栅格图形的读取、显示和存储

扫描图像均为栅格图像(BMP, TIFF 格式等), BMP 在表达位图和像素点处理方面更方便, 更高效。因而本实验系统在做栅格格式向矢量格式转化时, 选取 BMP 文件格式作为栅格格式的代表。本系统对 BMP 文件的操作是通过加载动态连接库 ImageObject 实现的。

加载了 ImageObject 后, 可以从中导出 CimageObject 类, 从而实现对图象文件的读取、显示、存储等操作。例如显示图像时, 只需在视图类的 OnDraw() 函数中加入如下语句, 其中 m\_ImageObject 为 CimageObject 类的实例:

```
m_ImageObject.SetPalette(pDC);  
m_ImageObject.Draw(pDC);
```

#### 2 基本图形元素的操作

实验系统提供了图形系统的一些基本功能, 如绘制、选取和删除; 栅格图像及矢量图形的移动和缩放。能够实现包括直线、折线、矩形、多边形、圆和椭圆在内的多种图形的操作。

#### 3 二值图像处理

对于输入的扫描地图图像要对其进行一些前处理, 如噪声滤除、细化、数学形态学运算具体的算法和技术实现在前面的章节已经做了具体的说明。

#### 4 识别提取

前处理过的二值图像要经图形识别提取，转化成矢量元素。实验系统提供了两种方式，一是全自动识别；另一种可以使直线、多边形和折线分步实现，满足 GIS 软件的分层空间模型的要求。

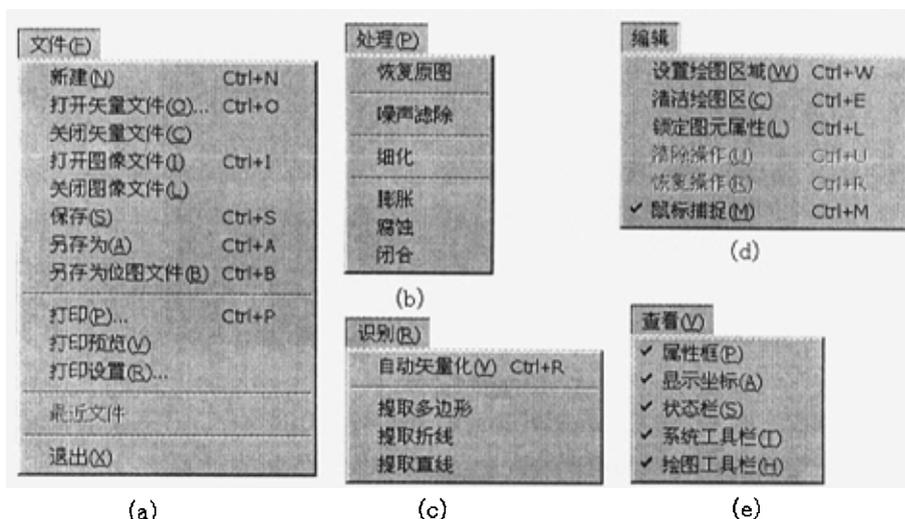
### 5.2 实验系统界面

界面的菜单共有 7 项：文件、编辑、处理、识别、查看、窗口和帮助。此外，还设有可停泊的文件工具条、绘图工具条和图元属性框。

文件菜单包含矢量文件操作功能、图像文件操作功能、打印功能，如图 5.2.1(a)所示。

处理菜单包含噪声消除、数学形态学运算、细化等光栅图像操作，如图 5.2.1(b)所示。

识别菜单包含矢量化分步实现和全自动实现功能，如图 5.2.1(c)所示。



(a)文件菜单项 (b)处理菜单 (c)识别菜单 (d)编辑菜单项 (e)查看菜单项  
 (a) File menu (b) Process menu (c) Recognize menu (d) Edit menu (e) View menu

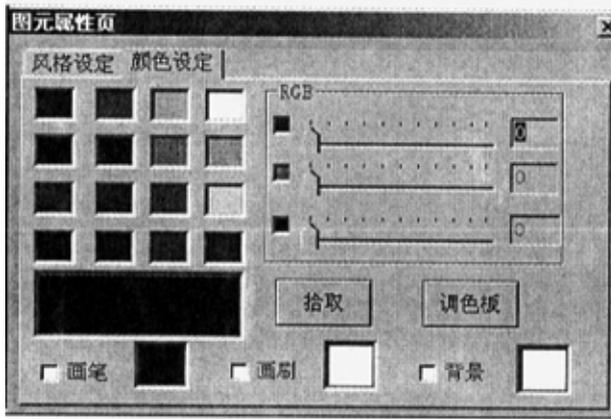
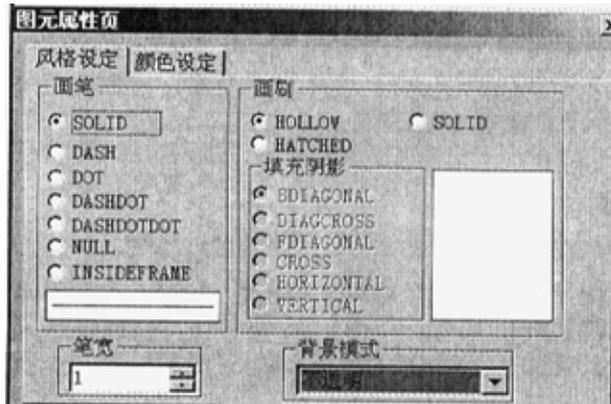
图 5.2.1 菜单项

Fig.5.2.1 Menus

文件工具条包括矢量文件和图像文件的新建、打开、和保存；设置绘图区域；绘图工具条、图元属性框的隐藏/显示，如图 5.2.2(b)所示。

绘图工具条包括直线、四边形、多边形、圆、椭圆和弧的绘制。绘制过程可双击鼠标右键结束，如图 5.2.2(c)所示。

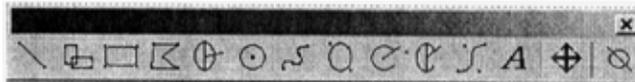
图元属性框包括画笔风格设定和画笔颜色设定两部分，如图 5.2.2(a)所示。



(a) 图元属性对话框  
(a) Elements' properties Dialog



(b) 文件工具条  
(b) File tools bar



(c) 绘图工具条  
(c) Drawing tools bar

图 5.2.2 工具条

Fig.5.2.2 Tools bars

### 5.3 矢量化操作过程

利用本系统对输入图像进行矢量化操作的步骤如下：

- (1) 打开光栅格式的图像文件，输入二值图像(单色、黑白)。

点击菜单：文件(F)\打开图像文件(I)

- (2) 去除图像噪声点。 点击菜单：处理(P)\噪声滤除

- (3) 对图像进行细化处理。 点击菜单：处理(P)\细化

- (4) 应用数学形态学闭运算对细化后图像处理，确保图像的连通性。

点击菜单：处理(P)\闭合

- (5) 识别。

识别可以采用分步完成的方式，即可以单步显示处多边形(点击菜单：识别(R)\提取多边形)、折线(点击菜单：识别(R)\提取折线)和直线(点击菜单：识别(R)\提取直线)的提取；也可以用自动识别方式实现多边形、折线和直线的同时提取(点击菜单：识别(P)\自动矢量化)。

矢量化的过程就是连续调用程序各个功能模块。在识别菜单项中自动矢量化功能，能够自动顺次实现上述所有功能，达到全自动化的效果。图 5.2.3 给出了实验系统界面。

另外有关二值图像处理的操作：

1. 腐蚀：点击菜单：处理(P)\腐蚀
2. 膨胀：点击菜单：处理(P)\膨胀

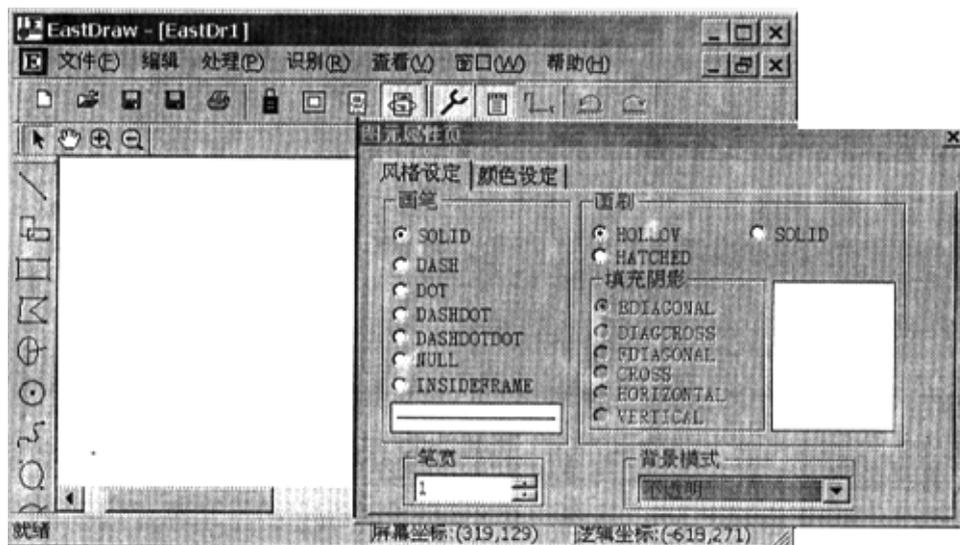


图 5.2.3 实验系统界面

Fig.5.2.3 System interface

## 6 结论

### 6.1 实验系统特点

依照本文算法设计的实验系统已在 PC 机上 Visual C++ 编程环境下实现。整个系统框架具有基本的栅格图像处理功能、矢量化功能以及图形系统所应有的一般功能(如, 绘制、选取和删除; 栅格图像及矢量图形的移动和缩放)。总体来说, 本系统具有以下特点:

#### 1. 可实现性

系统采用了现在比较常见的基于细化的矢量化方法。按照常规矢量化软件功能设计的组成, 完成实验系统的三大组成部分: 前处理(噪声滤除)、细化和连通性保持、矢量提取。并经过试验, 对  $662 \times 545$  大小的地图图像有良好的识别的效果。附录中图(a)给出地图扫描二值图像, 图(b)给出去噪、细化、闭合处理后的结果, 图(c)给出本实验系统的识别结果, 图(d)给出原有实验系统的识别结果。通过图(c)与图(d)的比较能够看出本实验系统较以前的识别结果有了很大的进步, 在图形提取中避免了误延伸造成的错误。

#### 2. 创新性

算法设计上, 本文提出了一种  $3 \times 3$  模板与  $4 \times 4$  模板相结合的细化方法, 不仅避免了过度腐蚀还保持了原图良好的连通性; 并在多边形提取算法中, 将计算机几何学的矢量外积概念与多边形搜索的最小角法则相结合, 避免了以往算法中参数值计算和其它烦琐的数学运算, 生成算法明显优化了代码并提高了算法的执行效率。

#### 3. 可移植性

系统采用模块化的设计思想, 便于功能的调试、维护、测试和移植。而且每个模块的输出在下一个模块使用过后就可以释放其所占空间, 到最后图形识别阶段只需要保留结点数组和弧段数组, 从而使图像的数据量逐步减少, 有利于节省内存空间。

#### 4. 局限性

虽然本系统以能较准确重建地图中线性对象, 但由于时间和精力有限, 尚未实现圆和曲线的识别与提取。而且细化算法的过度腐蚀在局部仍不能完全避免, 也对最后的识别效果产生了不良的影响。这些问题都是今后进一步研究中有待于解决和完善的。

## 6.2 地图信息识别与提取的展望

地图数字化是计算机技术、信息科学、模式识别、决策科学等学科发展的必然结果。地理信息获取技术正朝着自动化、智能化的方向发展。扫描输入是解决图样资料数字化的最好途径,扫描后的结果是以栅格形式存储。在GIS中矢量数据是不可缺少的,因而将扫描输入的栅格数据转化成矢量化数据(即矢量化)已成为GIS技术中的一个研究重点。

### 1 发展方向

有关学者已从不同侧面利用各种方法和策略对地图信息识别和智能化获取进行了广泛而深入的研究。但是,地图信息识别与提取技术目前仍处于边研究、边开发、边应用阶段,在理论和应用领域还存在不少问题有待解决:

- 1 针对影响地图数字化作业效率和自动化程度的主要环节(地图上大量出现的地图要素)的识别提取进行研究,以提高作业效率和自动化程度。
2. 由组合线划构成的复杂地图符号的自动跟踪提取;
3. 在对地图符号进行自动跟踪矢量化的同时,获取其相应的属性信息,做到真正的自动识别,目前这方面的研究还很薄弱;
4. 根据地图信息的结构特征,研究完整、统一的地图信息识别与提取理论、方法和策略;
5. 将业已成熟的文字识别(OCR)技术引入地图信息识别与提取领域,并针对地图特点加以扩展和改进,以解决地图上文字和数字注记信息的识别问题;
6. 建立高效、实用的全自动地图信息识别和提取软件系统。

### 2 广泛的应用领域

伴随着空间技术的发展,地图自动矢量化技术已在地理信息系统(GIS)领域外拥有了更为广阔的发展延伸。例如,利用航拍照片、遥感图片等图像资料,进行地质勘探、环境评估、农业产量预测、森林火灾预报、洪水灾害统计等工作。这些工作都正在或即将利用地图自动矢量化技术。

综上所述,地图自动矢量化技术将随着科学技术的发展,获得更多的应用领域,获得更大发展空间,更加成熟的理论体系。

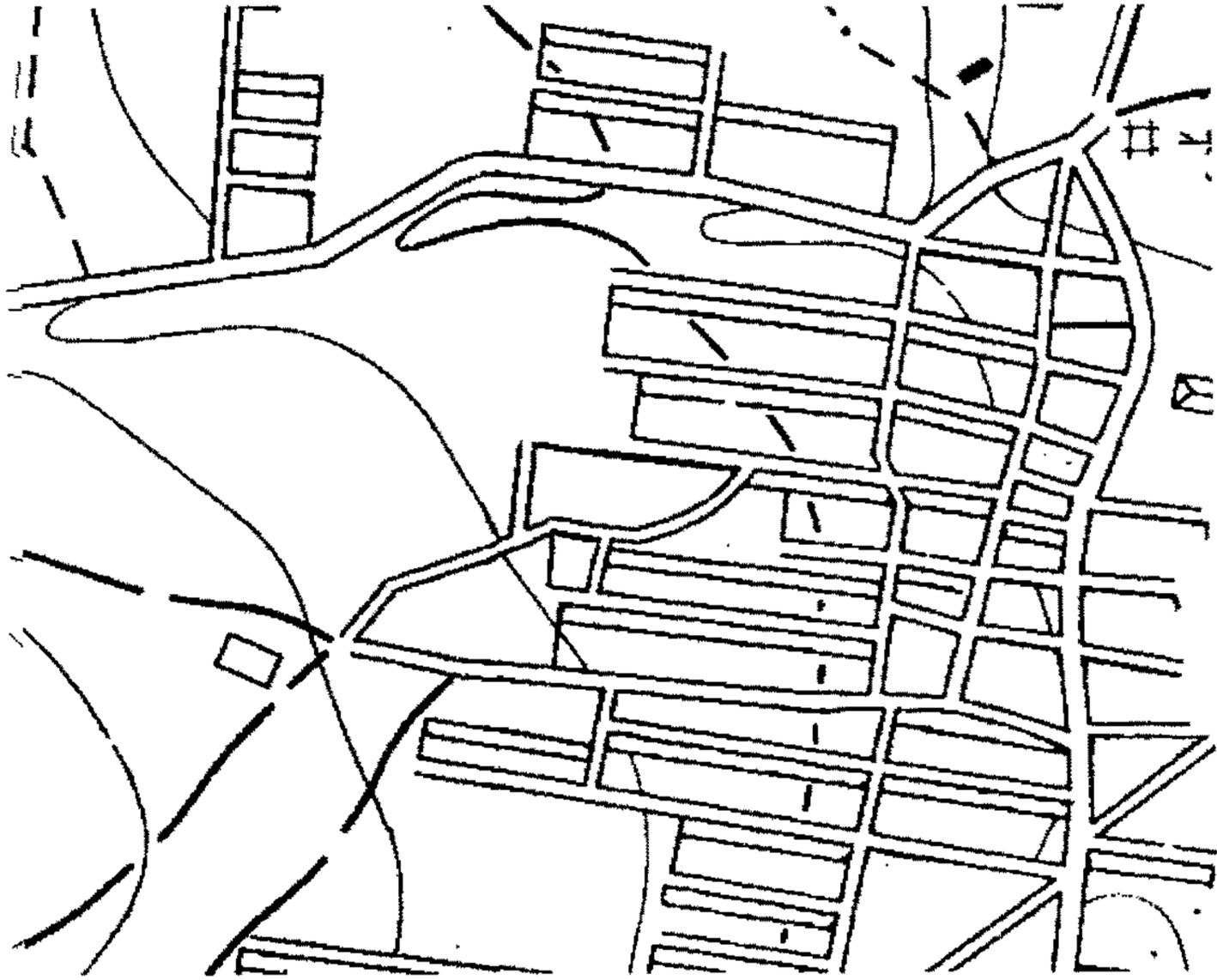
## 参考文献

- [1]. 郝向阳. 地图信息识别与提取技术. 北京:测绘出版社, 2001年6月. 1—105
- [2]. Rangachar Kasturi, Rodney Fernandez, Mukesh L. Amlani, \* Wu-chunFeng. Map Data Processing in Geographic Information Systems. IEEE Computer, 1989, Dec: 10-21
- [3]. Kgong-HO Lee\*, Sung-Bao cho, Yoon\_Chul Choy. Automated vectorization of Cartographic maps by a knowledge-based system. Engineering Application of Artificial Intelligence, 2000, 13(3):166-174.
- [4]. Jean Marc Ogier, Remy Mullot, Jacques Labiche, Yves Lecourtier. Semantic Coherency: The Basic of an Image Interpretation Device—Application to the Cadastral Map Interpretation. IEEE Trans. Sys, Man, Crybn, 2000, 30(2):322-337
- [5]. 谭柏珠, 叶邦彦, 陈澄洲. 基于知识的窗口矢量化技术. 计算机辅助设计与图形学学报, 2000, 2(2):105-109
- [6]. Youyi Zhang. Redundancy of parallel thinning. Pattern Recognition Letters. 1997, 18:27-35
- [7]. 彭京亮. 一种快速的扫描铅笔稿图细化算法. 中国图象图形学报, 2000, 5(A)(5): 434-439
- [8]. 崔凤奎, 王晓强, 张丰收, 王永森. 二值图像细化算法的比较与改进. 洛阳工学院学报, 1997, 18(4):48-51
- [9]. Yuncai Liu. An automation system: generation of digital map data from pictorial map resources. Pattern Recognition, 2002, 35:1973-1987
- [10]. Jiqiang Song, Feng Su, Heng Li, Shijie Cai. Raster to vector conversion of construction engineering drawings. Automation in construction, 2002, 11: 597-805
- [11]. 郭丙炎, 朱建新, 周济. 工程图形扫描图像的智能识别. 计算机应用研究, 1998, 3:13-14
- [12]. 文贡坚, 王润生. 一种稳健的直线提取算法. 软件学报, 2001, 12(11):1660-1663
- [13]. Xinggang Lin, Sgigeyoshi Shimoteuji, Michihiko Minoh. Efficient Diagram Understanding with Characteristic Pattern Detection, 1985, 30:84-106
- [14]. Javie Jimenez, Jose L. Navalon. Some Experiments in Image Vectorization. IBM J. RES. DEVELOP, 1982, 26(6):724-734
- [15]. 谭建荣, 彭群生. 基于图形约束的工程图扫描图像直线整体识别方法的研究. 计算机学报, 1994, 17(8):561-569
- [16] 李伟青, 谭建荣, 彭群生. 基于图段结构的整体识别方法的研究. 计算机学报, 1998, 21(8):753-758
- [17]. 王金鹤, 彭群生. 基于图段结构的整体识别方法的研究. 计算机学报, 1998, 28(4): 476-481
- [18]. 李岩影, 过静珺. 大比例尺地图扫描图像的整体矢量化方法研究. 测绘通报, 2000, 8:

20-22

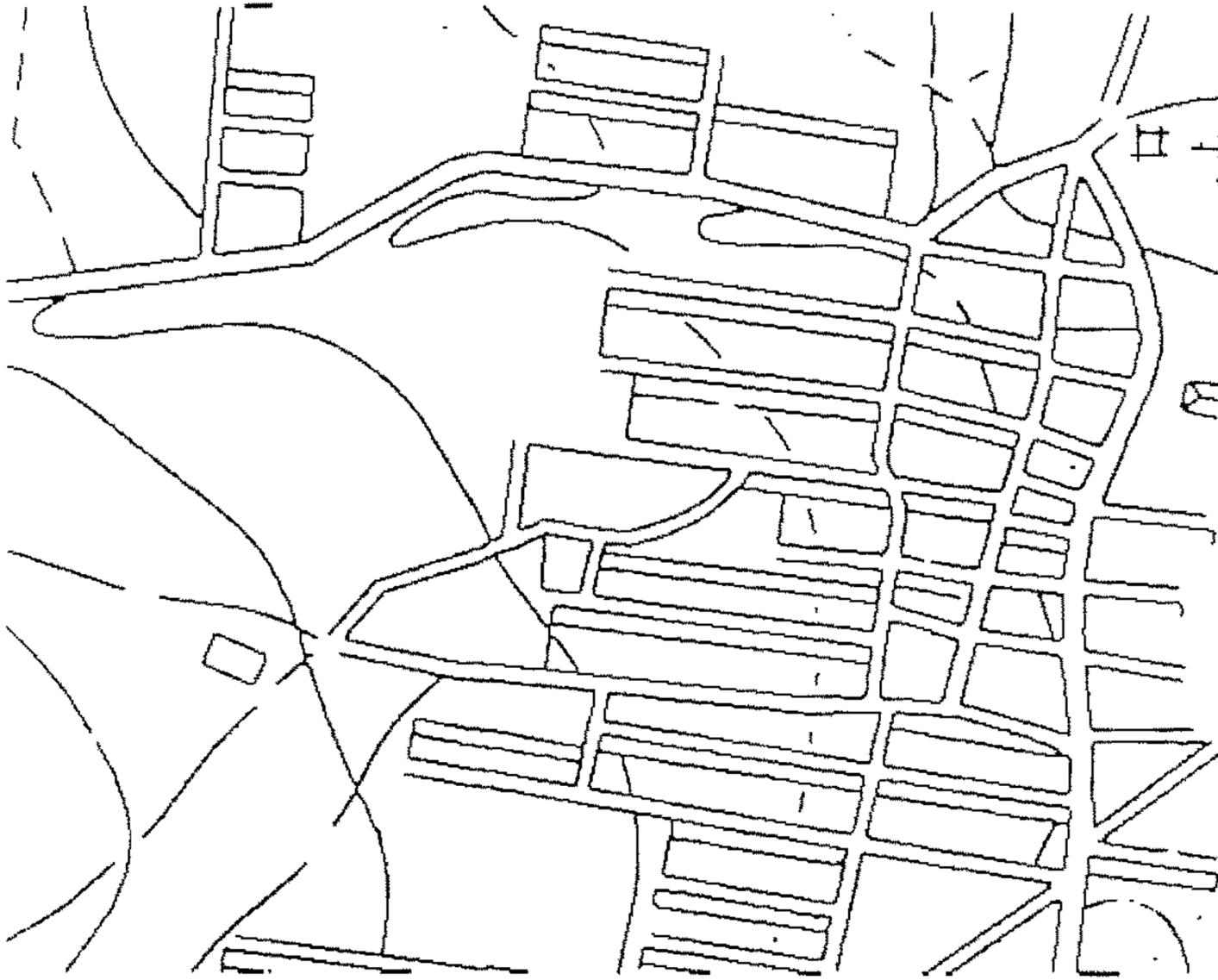
- [19]. 郭子华, 庄镇泉. 基于遗传算法的地图矢量化算法. 中国科学技术大学学报, 1998, 28(4):476-481
- [20]. 高隽, 张维勇, 蒋建国. 基于神经网络的形状识别系统及优化. 系统工程与电子技术, 1999, 21(3):52-55
- [21]. 曹爱军, 史杏荣, 孙贞寿, 梅建华. 基于多层神经网络的交叉线自动识别. 2000, 5(A)(2):149-152
- [22]. 唐常青等. 数学形态学及其应用. 北京:科学出版社, 1990年8月. 1-20
- [23]. 林春蔚等. C 环境下地图图像矢量化及图形编辑技术与实例. 北京:海洋出版社, 1993年
- [24]. N. J. Naccache, Shinghal. An Investigation into the Skeletonization Approach of Hilditch. [J] Pattern Recognition, 1984, 17:279-284
- [25]. 马建波. C 语言图像处理程序集. 北京:海洋出版社, 1992
- [26]. Zhang T Y and Suen C Y. A Fast Parallel Algorithm for Thinning Digital Patterns [J]. Comm. of ACM, 1984, 27(3):236-239
- [27]. Pavldis T. A Thinning Algorithm for Discrete Binary Images [J]. Comput. Graphics Images Processing, 1980, 13:142-157
- [28]. 盛业华, 唐宏, 杜培军, 郭达志. 一种保形的快速图象形态细化算法. 中国图象图形学报, 2000, 5(A)(2):89-93
- [29]. 李庆华, 陆枫, 杨叔子. 图象处理中一种  $4 \times 4$  细化算法. 华中理工大学学报, 1994, (22), 7:87-91
- [30]. 宋晓宇, 王永会. 工程图自动矢量化算法的研究. 计算机工程与应用, 2000, 3:41-43
- [31]. 张显全, 程雪梅. 一种识别工程图的新方法. 渝州大学学报, 2001, 18(1):25-28
- [32]. 栅格矢量化软件巡礼. 计算机辅助设计与制造, 2000, 8:29-32
- [33]. Freeman H. Computer processing of line-drawing images. Comput Surveys, 1974, 6(1):57-97
- [34]. Mark Monmonier S. Computer Assisted Cartography Principles and Prospects. Prentice-Hall, Inc, 1982:110-117
- [35]. 王世新, 詹慈祥. 多边形所属弧的搜索及图幅的拓扑关系的自动建立. 资源与环境信息系统软件研究. 北京:测绘出版社, 1990年. 22-29
- [36]. 杜清运. 地图数据库中多边形数据的自动组织. 测绘学报, 1989, 18(3):202-212
- [37]. 熊德敏. 地理信息系统中多边形拓扑关系的建立与应用. 黄土高原(重点产沙区)信息系统研究. 北京:测绘出版社, 1988年. 155-164
- [38]. 孙文惠. 地理信息系统的表示及其处理算法. 再生资源环境遥感研究——“三北”防护林地区平泉遥感综合调查的方法与实践. 北京:科学出版社, 1988年. 274-278
- [39]. 齐华, 刘文熙. 建立结点上弧—弧拓扑关系的 Qi 算法. 测绘学报, 1996, 25(3):233-235
- [40]. 齐华. 自动建立多边形拓扑关系算法步骤的优化与改进. 测绘学报, 1997, 26(3):254-260

- [41]. 闫浩文, 杨维芳, 陈全功, 梁天刚. 基于方位角计算的拓扑多边形自动构建快速算法. 中国图象图形学报, 2000, 7:563-567
- [42]. David J. Kruglinski. Visual C++ 技术内幕 (第四版). 北京: 清华大学出版社, 2001 年 9 月
- [43]. 陈建春. Visual C++ 开发 GIS 系统. 北京: 电子工业出版社, 2000 年 9 月. 1-251



(a) 地图扫描二值图像

(a) binary scanned maps images

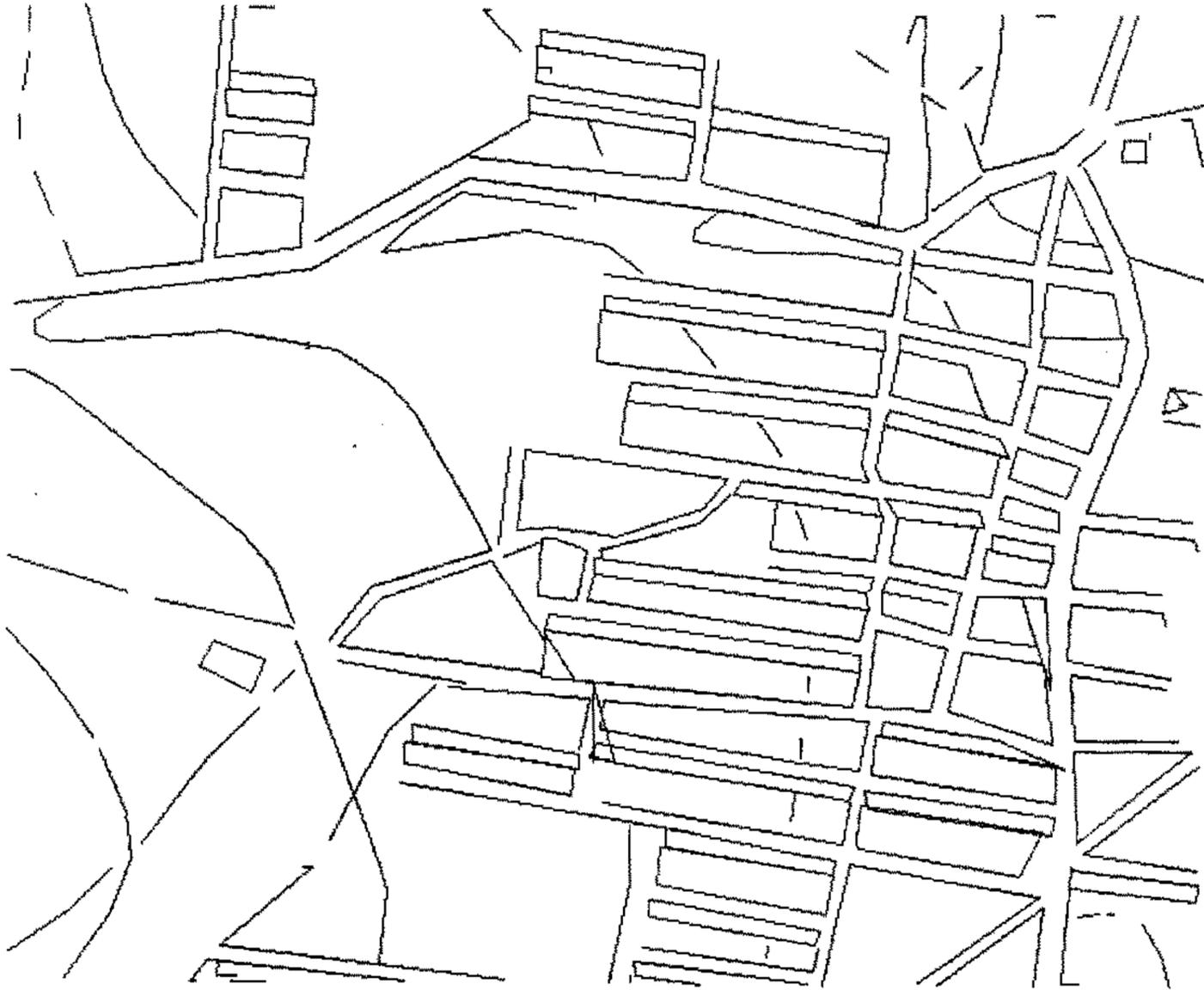


(b) 去噪、细化、闭合处理后的结果

(b) the result after deleting noise, thinning, and closing



(c) 本系统识别结果  
(c) the result of this system



(d) 原有实验系统识别结果  
(d) the result of former system

## 致 谢

研究生生活转瞬即逝，我即将离开培养我的母校和教导我的老师。在此论文完成之际，我需要感谢的人很多。首先向我的导师江崇礼教授致以最诚挚的感谢。江老师学识渊博，治学严谨，待人诚挚和蔼，在治学和做人方面都是我学习的榜样。本论文是在江老师的悉心指导下完成的。从理论到实践，从研究方案的制定到论文的撰写，都和江老师的辛勤指导是分不开的。

在这三年的学习和生活中，我还要特别感谢董明老师，感谢董老师中肯的意见和无私的帮助。

感谢教研室的马振涛、任昌荣、丁柏玲、尹斌、李继勇等同学，以及研究生班的刘宇新、林艳、李天昊、贾小勤、王刚等同学，谢谢他们给予我的帮助和支持。

最后我要感谢亲爱的家人。今天的成绩离不开他们对我的支持和鼓励，他们永远是我奋斗的动力。

樊琨

2003年3月10日