MS THESIS

Design and Optimization of Graghical Configuration Experimental System of Digital Image Processing

Specialty: Computer Science and Technology

Master Degree Candidate: Chen Jiangting

Supervisor: Prof. Luo Sanding

School of Information Science & Engineering

Central South University

ChangSha Hunan P.R.C

原创性声明

本人声明，所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了论文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得中南大学或其他单位的学位或证书而使用过的材料。与我共同工作的同志对本研究所作的贡献均已在论文中作了明确的说明。

作者签名：张江婷　　日期：2010 年 5 月 20 日

学位论文版权使用授权书

本人了解中南大学有关保留、使用学位论文的规定，即：学校有权保留学位论文并根据国家或湖南省有关部门规定送交学位论文，允许学位论文被查阅和借阅；学校可以公布学位论文的全部或部分内容，可以采用复印、缩印或其它手段保存学位论文。同时授权中国科学技术信息研究所将本学位论文收录到《中国学位论文全文数据库》，并通过网络向社会公众提供信息服务。

作者签名：张江婷　导师签名　　日期：2010 年 5 月 20 日

摘 要

随着计算机技术和成像技术的发展，数字图像处理技术在日常生活、军事、工业和医疗等许多领域得到了广泛的应用。提高图像处理的速度，使图像处理过程更加可视化，是设计基于图形化组态的图像处理实验系统的目的。

首先，在分析了国内外现有图像处理软件的优缺点和现有图形化组态软件的基础上，提出了系统的需求。本系统是一个图形化组态的图像处理实验系统，它将各种图像处理算法封装成图形控件，用户可以通过拖拽的方式组成组装图，进行图像处理实验。它是一款不仅可以适用于专业的图像处理研究人员，也可以适用于图像处理初学者的软件。此系统采用VB和VC程序设计语言实现，为了提高系统运行速度，增加系统的灵活性，将大量的图像处理算法编写成动态链接库。

其次，围绕系统的需求分析和总体设计，分析了系统的详细设计过程。介绍了组成组装图的基本元素图形控件和图形控件之间连线的结构以及它们的绘制方法，并且也介绍了系统主要功能模块的设计。在分析组装图中的顺序、分支、循环、子程序四种结构的特点和图形化组态软件常用运行机制的基础上，提出了本系统用户所绘制的组装图的运行机制。

再次，分析了数字图像实验的特点以及优化的必要性，在组装图的编译过程、组装图结构、图像质量、内存管理等方面进行了优化。其中，编译过程优化可以减少组装图的运行时间，组装图结构优化不仅可以减少组装图的运行时间也可以减少内存的使用，图像质量优化结合图像质量的评价指标实现。

最后，以车牌的初定位实验在系统中的实现过程为例，分析了实验在系统上的实现过程和优化过程。

关键词 图形化组态，图形控件，动态链接库，优化

ABSTRACT

With the development of computer technology and imaging technology, the digital image processing technology has been widely used in the daily life, military, industry, medical treatment and many other areas. Improving image processing speed and making more visual image processing is the purpose of designing this graphical configuration experimental system.

Firstly, based on analysis of the advantages and disadvantages of the existing image processing software home and abroad, and analysis of the existing graphical configuration software, system requirements have been proposed. This system is a graphical configuration experimental system. All kinds of image processing algorithms are encapsulated into graphics controls. Therefore, the users can form an assembly drawing by dragging and dropping the graphics controls. This system is not only for professional image processing researchers but also for the beginners of image processing. And meanwhile, it uses VB and VC programming languages. In order to increase system speed and flexibility,a large number of image processing algorithms have written into Dynamic Link Library.

Secondly, around the system requirements analysis and overall design, analysis of the detailed design procedure. Introduced the basic elements of the composition of assembly drawing graphic control and the connection between the graphic control, and their method of drawing. And also introduced the design of main function modules. Based on analysis of the feature of the order, branch, loop, subroutine four flow structures and operation mechanism of graphical configuration software, operation mechanism of assembly drawing which users drawing is proposed.

Again, analysis of the characteristics of digital image experiments and the need for optimization. And optimized the compiling process of assembly drawing, process structure, image quality, memory management, and so on. Among them, the optimization of compiling process can reduce the running time of assembly drawing, and the optimization of assembly drawing structure can not only reduce the assembly drawing's running

time but also reduce usage of memory, and optimization of image quality is realized combining image quality evaluation indexes.

Finally, realization process of location of license plate is used to illustrate the realization and optimization of the experiment on the system.

KEY WORDS graphical configuration software, graphic control , Dynamic Link Library, optimization

目 录

第一章 绪论

1.1 研究背景

数字图像处理技术起源于20世纪20年代，早期的图像处理的是通过人的视觉感知来改善图像的质量，达到改善人的视觉效果的目的。随着计算机技术、信号处理技术以及成像技术的发展，数字图像处理的应用范围也越来越广泛[1]。数字图像处理技术已经广泛应用到工业生产、农业、科学研究[2]、军事、安全、政府部门、医疗卫生[3]等许多领域。70年代中期，随着思维科学、人工智能研究的迅速发展，数字图像处理也随之向着更高、更深层次的方向发展，图像理解技术以及计算机视觉等也成为了研究的一个热点，数字图像处理技术在科学研究中所起的作用也越来越大。

图像中含有大量边缘、颜色、形状、纹理等有用的信息，这些有用信息可以通过数字图像处理技术中相应的处理算法得到。然而，一个数字图像处理过程常常要用到大量复杂的图像处理算法，如果在进行实验的时候临时编写这些图像处理算法，不仅算法的正确性难以保证[4]，需要浪费很多时间去调试程序，验证算法的正确性。而且通过临时编写图像处理算法进行图像处理这种方式也存在着许多不足之处。首先，临时编写算法这种方式比较适合于那些受过计算机图像编程专业培训的人员，而对于那些不熟悉编程的图像处理爱好者或者图像处理初学人员用起来则比较困难。其次，维护起来比较困难，效率比较低，数字图像处理过程是多个算法模块相互协调共同完成的过程，当对处理流程进行改动的时候，需要改变源程序，会浪费大量的时间。再次，存在重复编码问题，不同的图像处理过程如果用到了相同的处理算法，则还需要重新编写算法，这样降低了开发效率。

因为图形具有易于理解，一目了然的特点，所以为了提高编程的效率，程序质量与可靠性，同时降低编程难度，软件开发者开始探索基于图形化组态这种软件开发的方法。图形化组态软件中的图形控件封装了一定的算法，图形化组态的方式不仅用起来比较方便，而且它易于理解，在使用的时候不需要编写程序。目前，图形化组态的软件设计方法已经成为了一种重要的软件设计方法，比如说在虚拟实验室的构建中就经常用到这种方法。在虚拟实验的构建过程中，有的时候需要利用相同的实验仪器或者实验步骤灵活地组建不同的测试实验，这就需要快速的软件编程。很多熟悉的软件也采用了基于图形化组态的软件设计方法，比如说，NI公司开发的Labview[5][6]、用于系统仿真的Matlab[7]，HP公司的VEE[8][9]，

还有 Authorware[10]、Visio[11]等等。

本文研究的具有优化功能的可扩展的数字图像处理实验系统运用了基于图形化组态的软件设计方法，在进行图像处理的时候不需要再临时编写图像处理算法的代码。不仅专业从事图像处理的专业人员可以用它来进行图像处理方面的研究，而且不熟悉图像编程的初学者或对图像处理感兴趣的用户也可以用它来熟悉图像处理过程，进行简单的实验。用户可以利用系统已有的图标进行灵活地组态，搭建出不同的组装图，方便算法的比较以及图像处理结果的对比。实验系统的可扩充性可以满足数字图像处理技术不断发展的需要，同时，系统的优化功能可以提高图像处理的速度。

1.2 研究现状

1.2.1 数字图像处理系统的发展现状

数字图像处理系统有很多实现方法，比如说文献[12]中提到用软件实现的图像处理和分析系统，文献[13]中提到的用硬件方法开发的图像处理系统，文献[14]中提到的用软件和硬件相结合的方式进行图像处理系统的开发，或者是利用特殊的图像信号处理芯片来实现等。数字图像处理系统的发展过程可分为以下几个阶段[15]：

第一阶段：从 20 世纪 60 年代到 80 年代中期，在这个阶段中的图像处理系统主要采用机箱式结构和双屏操作方式实现，而所采用的计算机为小型机。

第二阶段：从 20 世纪 80 年代中期到 90 年代初期，这个阶段中的图像处理系统更加小型化，采用由大规模集成电路或者是专用集成电路组成的图像卡组成，并且采用插卡式的外形，操作方式为双屏式。

第三阶段：从 20 世纪 90 年代初开始，这一阶段的图像处理系统采用图像通信方式为主要的方式，这一方式以单屏和图像压缩为特点，仍然采用微机来进行数字图像处理。

第四阶段：目前正处理数字图像处理系统发展的第四阶段，这一阶段主要采用 DSP 和 SOPC[16][17]来实现数字图像处理系统。

目前，有很多常用的数字图像处理软件，比如说 Photoshop 以及 Linux 操作系统下的图像处理软件 GIMP[18]等，而某些软件中也具有图像处理功能，比如说 Matlab、Labview 等软件中也具有图像处理模块。Photoshop 是一种常用的图像处理软件，它提供了图层、光影效果、蒙板、艺术处理等功能，常用于图形图像设计和数码照片的处理。但是，它对扫描输入的图像不能实时显示，也不具备某些专业领域所需要的功能，比如说，印刷工业上的去划痕、去污点、去网点、高质

量锐化、高质量差值等；医学上的无损高效图像压缩、加密、加信息、高速解压缩等；勘测技术上的几何校正、高质量二值化、分色、特征参数提取等，不适合于用它来从事某些专业领域的研究。GIMP 是运用于 Linux 平台的一个通用图形图像处理的软件，它具有 Photoshop 的大部分功能，但是它在界面、易用性方面没有 Photoshop 那么强大。Matlab 的数字图像处理工具(Image Processing Toolbox)提供了将近 200 种最基本的图像处理函数，利用这个图像处理工具箱，结合其强大的数据处理能力，大大提高了工作效率，但是它有个缺点就是需要用户熟悉 Matlab 的语言特点。Labview 的视觉开发模块将 400 多种图像处理函数集成到 Labview 的开发环境中，为图像处理提供了强大的开发功能，但是用它进行图像处理需要用户花费相当大部分时间去熟悉 Labview 开发环境。

目前，世界上也出现了很多针对于某个专业领域的图像处理系统。比如说，文献[19]从数字图像处理的基本理论出发，设计并初步实现了一个医学专用图像 DICOM 的处理软件；文献[20]提出了用来修正红外图像的基于 DSP 的红外信号处理系统。

1.2.2 图形化组态软件的发展现状

图形化组态的软件设计方式代表了软件开发的一种新方法，它将算法封装到图形控件中，通过图形控件和图形控件之间的连线来构成处理流程，具有直观易于理解的特点。图形控件具有可复用性、独立性等特点。并且图形控件之间的连接是在用户编辑流程的时候确定，而不是事先确定的，被连接的图形控件可以随时被撤销和改变，它们之间的连线也可以随时改变。因为这种方式不仅减少了开发时间，而且直观易于修改，所以基于图形化组态的软件设计方法成了近几年国内外研究的热点，出现了许多优秀的基于图形化组态的软件。比如说，用于图像处理的 Khoros、用于科学数据可视化系统的 AVS、HP VEE、用于组建虚拟仪器系统的 Labview 等等 。

德国 Darmstadt 大学计算机系集成电路和系统实验室 Konstantions Konstantinides 等人开发的 Khoros[21]，具有丰富的图像和数字信号处理工具，常常被用于数字图像处理中。它的主要特点有：支持分布式计算，程序和数据共享；元件可以选择用于执行的目标机；应用大数据粒度；主要的数据类型为图像。

Labview 是美国国家仪器公司的创新软件产品，它较完整的结合了图形的美观易用和文本语言的强大灵活，并在此基础上提供面向广泛测试领域的虚拟仪器解决方案，在全球虚拟仪器领域占有近乎垄断的地位。它的特点是：具有良好的用户接口，类似于传统仪器的面板；编辑方式简单，直接采用图形语言(G[22][23]语言)，图标和连线代替编写的程序；提供程序调试工具，包括在源代码中可以设

置断点，可以单步执行，也可以启动。Labview 支持不仅支持多种硬件设备，比如说：RS-232、VXI、GPIB、PXI、内插式的数据采集卡等，而且它还支持 Windows 操作系统中的 ActiveX 以及 TCP/IP 的网络通讯。Labview 可以通过一个运行时引擎来执行用 Labview 设计生成的程序编译而成的一个可以独立运行的 32 位的代码。

美国 HP 公司开发的产品 HP VEE[24] 与 Labview 相似，也是一款图形化组态的软件。但是它与 Labview 相比，比 Labview 更容易使用。HP VEE 有 X-Window 和 Windows 两个版本，其程序也包括数据流程图和软件面板两个部分。HP VEE 不直接提供对硬件端口和底层操作系统的访问能力，它为了使用户可以更加方便地搭建虚拟仪器系统，提供许多高层次的控件。HP VEE 是一种非常典型的数据流语言，它定义了一些特殊的和元件，并且对它们进行了特殊的处理，目的是为了实现一些复杂的流程控制功能。在运行上，HP VEE 和 Labview 不同的是它生成的程序必须由 HP VEE 解释执行，而不能像 Labview 一样编译生成独立运行的程序代码。

Softwire Technology 公司的 SoftWIRE[25][26]提供了与微软 Visual Studio 结合的图形化编程环境，可以结合文本和图形两方面的优势，使用非传统的方式进行虚拟仪器系统的开发。SoftWIRE 的新版本扩展了数据库、网络传输、财务和统计等功能。

TestPoint[27][28]是 CEC 公司的一个产品，是一个以事件驱动运行的图形编程环境，适用于多种数据采集系统，它还提供了一系列的公开接口供开发者调用。

另外，Visual Designer[29]、DIAdem[30]也是基于图形化组态的软件。

目前，国内市场上也出现了许多基于图形化组态的软件。浙江大学开发的图形化编程软件平台 VPP(Visual Programming Platform)[31][32]。该平台是一种基于扩展的数据流语言的平台，这种扩展的数据流语言具有以下几个特点：在进行图形控件设计的时候定义了图形控件的可激活函数，不同类型的图形控件的可激活函数不一样，通过可激活函数来控制流程的运行；为了解决不同图形控件之间各种数据的传递问题，在进行图形控件之间连线的定义中引入了类型转换函数，在输入端口和输出端口的设计中引入了可连接函数。北京亚控科技公司的“组态王”提供了资源管理器式的操作主界面，并且提供了以汉字作为关键字的脚本语言支持，它也提供了多种硬件驱动程序。但是它的网络功能比较薄弱，支持不了真正意义上的分布式系统。

为了降低教学和科研成本，现在许多学校和研究机构采用虚拟实验室来进行相关学科和研究的实验，而基于图形化组态的软件设计方法也成为构建虚拟实验室的一种重要的方法[33]。目前，出现了各种各样类型的虚拟实验室，比如说，意

大利帕瓦多大学建立的远程虚拟教育实验室；西班牙大学电子系开发的电子仪器虚拟工作平台；美国亚利桑那州大学利用 Java 语言和数字信号处理技术相结合开发的数字信号处理的仿真系统[34]；文献[35]中提到的一种虚拟网络实验室，学生可以利用此虚拟实验室进行网络实验，从而熟悉网络的各种配置；文献[36]中提到的一种方便进行电路实验的虚拟实验室系统；中南大学研究开发的数字通信原理虚拟实验室[37]。

从目前的研究现状来看，基于图形化组态的图像系统仍然具有重要的研究价值，这是因为：

首先，从数字图像处理的应用领域来看，数字图像处理技术涉及到了医学、农业、工业、军事、航天等诸多领域。在医学领域对 CT 图像、X 光图像的分析，超声波图像的处理等方面都用到了图像处理技术；在农业领域不仅可以利用图像处理技术来分析作物的成熟状况，也可以对作物的各个阶段的生长情况进行分析；在工业方面可以利用图像处理技术来进行产品质量检测、生产过程的自动控制等等，比如说文献[38]中提到的利用图像处理技术进行印刷电路板的瑕疵检查；文献[39]中提到的利用图像处理技术对啤酒瓶的质量进行检查；在军事方面和公安部门利用图像处理技术进行军事目标的跟踪和侦查以及犯罪案件的侦破[40]。同时，随着机器视觉技术的发展，数字图像处理技术也运用到了机器视觉的研究中。为了方便数字图像处理的研究，使图像处理实验过程更加简便，需要有一个图像处理系统，大量的图像处理过程都可以通过这一系统来实现。

其次，从现有的图像处理软件的发展来看，运用比较广泛的图像处理软件 Photoshop 虽然可以对图像进行一些基本的处理，但是它对于某些专业领域不适用，同时，它在使用上也具有某些缺陷。比如说，不能查看算法的具体参数值，不能对历史流程进行保存和修改，虽然可以通过“历史记录”查看图像的处理步骤，但是处理步骤之间的相互联系并不是很清楚。相比之下，例如 Matlab、Labview 这些软件中的图像处理模块虽然弥补了 Photoshop 的上述缺点，但是它们只是某些软件中的一个部分，如果仅仅是用这些软件进行图像处理，不仅用户需要花费大量的时间去熟悉这些软件的开发环境，而且安装这些软件需要占用大量的空间。目前的大部分的图像处理的软件只是具有简单的图像到图像的图像处理功能，并不具备像匹配、分割、追踪等算法，并且它也不具备提取图像特征的算法，不具有评价和优化功能，不能根据用户的需求以及系统的评价结果，对图像处理的处理时间、流程的步骤、算法中的参数取值等内容进行优化。

再次，从图形化组态软件的优点来看，利用图形化组态的设计方法可以将图像处理流程用简单、直观的处理流程的形式来表现，使处理流程更加一目了然，方便用户对组装图做适当的修改，从而达到自己想要的结构，也方便进行对比实

验。并且，用图形化组态的方法进行图像处理软件的设计，使用户在进行图像处理的时候不需要再编写复杂的图像处理程序，只要通过拖拽的方式来选择处理方法，搭建处理流程，这样可以避免重复操作。

1.3 研究内容

基于现有的图像处理软件的一些缺陷和基于图形化组态的软件设计方式的优点，本文主要介绍具有扩展和优化功能的基于图形化组态的数字图像实验系统的设计和实现。本论文由系统的基本需求展开分析，分析了系统的基本功能。本系统采用 VB 和 VC 程序语言实现。借助 VB 和 VC 编程语言的特点，对于界面以及不涉及大量计算的内容用 VB 实现，而需要大量计算的各种图像处理算法用 VC 中的动态链接库实现。

根据系统的需求分析，设计系统的总体结构和功能模块结构。本系统是一个基于图形化组态的数字图像实验系统，根据图形化组态软件的特点，研究组装图的绘制和组成组装图的基本元素结构的设计。在分析图形化组态的软件常用的运行机制的基础上，确定本系统的运行机制，它适用于顺序、选择、循环、子过程调用四种结构。

为了提高图像处理的速度，提高图像处理结果的质量，本实验系统具有优化功能。优化分为以下几方面：运用循环或者分叉结构对组装图的结构进行优化；为了减少通过流程进行图像处理的时间，对组装图的编译过程进行优化；结合图像某些特征的特征值，选用合适的算法对图像的质量进行优化；通过对内存进行合理的组织和分配来优化内存的使用，减少内存的消耗。

1.4 研究难点

本文研究的基于图形化组态的数字图像处理试验系统，需要满足从事专业数字图像处理的人员对图像处理的研究需求，也要满足数字图像处理初学者以及对数字图像处理感兴趣的人员熟悉图像处理算法，进行简单实验的需求，要根据需求对系统进行分析和设计，这一过程研究的主要难点有：

(1) 围绕系统的需求分析，设计系统的总体框架和各个功能结构。

(2) 本系统是一个基于图形化组态的图像处理系统，用户绘制的组装图代表了对图像的处理步骤，流程中可能会具有顺序、分支、循环、子过程调用等结构。研究一种对解析具有顺序、分支、循环、子过程调用几种结构的组装图都适用的运行机制。对组装图的解析包括对流程中图形控件的排序，以及组装图中图形控

件之间的信息传递。

(3) 为了提高图像处理的速度，优化组装图的结构，如何利用图像评价的结论，对图像的质量、算法流程的结构、图像的处理速度、内存的使用等方面进行优化。

1.5 论文章节安排

本论文主要是针对基于图形化组态的数字图像实验系统的设计和优化进行研究，分为以下几部分：

第一章 绪论，阐述了随着计算机技术的发展，数字图像处理系统和基于图形化组态软件的研究现状，提出了图像处理实验系统的设计需要满足方便性、快速性、可灵活组态、可扩展性的特点，最后阐述了课题研究的主要内容和难点。

第二章 系统的需求分析与总体设计，分析了系统的基本需求，介绍了基于图形化组态的软件设计技术以及系统的总体设计，在系统需求分析的基础上分析了系统的基本功能，提出了在系统设计时所用到的关键技术，以及系统主要的开发工具。

第三章 系统的详细设计与实现，阐明了系统关键类的设计以及用户操作界面、实验结果显示、组装图绘制等功能模块的设计。为了满足系统可扩展性的需求，详细介绍了系统可扩展性接口的设计和使用方法。在充分分析基于图形化组态软件常用运行机制的基础上提出了本系统所采用的运行机制。最后，通过示例演示了顺序、选择、循环、子过程调用四种结构的组装图的使用和实验结果。

第四章 系统优化的设计，为了进一步提高系统进行图像处理的速度，本章重点研究系统的优化。首先，分析了图像处理的特点和优化的必要性。本论文主要在组装图的结构、编译过程、图像的质量、内存的消耗上对系统进行优化，对优化前后的结果进行了比较。

第五章 系统的运行实例。为了进一步说明系统的使用，用车牌的定位实验来说明系统的使用过程和优化过程。

第六章 工作总结与展望，对本论文对数字图像处理试验系统的设计和研究工作进行总结，并阐述了进一步的研究方向。

第二章 系统的需求分析与总体设计

基于图形化组态的图像处理实验软件不仅是一款针对于从事专业图像处理的人员设计的软件，也是一款针对于图像处理初学者或者是对图像处理感兴趣的人员设计的软件。这就要求它在使用上具有易用性和易于交互性的特点，不仅能够提高图像处理的速度，而且能够满足不同层次的人对图像处理的需求。因此，分析本系统的功能需求，同时围绕系统的功能需求，对系统进行总体设计，是本章需要研究的问题。

2.1 基于图形化组态的软件设计

因为图形图像信息具有简单直观、易于理解的特点，所以在科学技术尚未发达的远古时代，人们就发明了象形文字，他们通过象形文字来进行信息交流，而在科学技术发展的今天，在某些方面人们仍然用图像来表达信息，比如说用地图来表示交通路线的分布或者建筑物的位置等信息，用电路图来表示某个电路的结构，等等。基于图形化的软件设计方法正是借助于图形图像简单直观、易于理解的特点而发展起来的。

组态的概念最早来自于英文 Configuration，意思是为了满足使用者的要求，使计算机或者软件能够按照预先的设置自动执行任务，而需用使用软件工具对计算机及软件的各种资源进行配置[41]。随着 DSC(Distributed Control System)集散型控制系统的出现，组态的概念被广大的生产过程自动化技术人员所熟知。因为每一个 DCS 都是比较通用的控制系统，可以应用到很多领域中，所以为了使用户不需要重复编写代码程序，避免重复操作，每个 DCS 厂商都在自己的 DCS 中预先装上了一套组态软件[42]。目前，很多应用软件的设计都采用了基于图形化组态设计方法。比如说，构建虚拟实验室的构建，工控仿真，教育类软件，等等。虚拟仪器是以计算机为中心，把测量技术和计算机结合起来，将硬件电路和仪器方面实验的分析和结果的现实过程用计算机来实现。用计算机搭建起硬件或者仪器实验的流程，并且配置好各种参数，通过计算机的仿真来得到相应的结果。在实验教学的过程中，一些与硬件相关的实验，因为实验仪器的成本和费用很高，而且实验仪器的元件也容易老化和损坏，为了清楚地了解硬件的执行流程和元件之间的数据流动，提高实验的透明度，减低实验的成本，许多硬件实验也在基于图形化组态的软件上实现。

基于图形化组态的软件开发代表了一种新的软件开发方式。基于图形化组态的软件用有向图来表示程序的运行过程，此有向图是由节点和节点间的有向连线构成的。节点可以看成是具有输入和输出端口，并且封装了一定算法的功能模块，输入端口用于接收数据，输出端口用于发送数据。节点间的有向连线具有连接节点，进行数据传递的功能。代表程序运行过程的有向图必须具备两个条件：一是这个有向图必须是可以运行的，它与那些静态的流程图不同，代表的是程序的运行过程，而且组成有向图的节点在运行过程中会执行一定的操作；二是这个有向图可以被可视化的修改，这里所说的修改不仅是改变节点的某些参数和变量，有向图的结构也可以被改变。

采用基于图形化组态的图像处理软件与传统的基于本文代码输入的图像处理软件相比，主要有以下的几个优势：

(1) 实验系统的最大特点就是它采用图形化的方式来实现，结合了图形图像易于理解，简单直接等特点，使用户能够脱离文本语言的复杂与繁琐，专注于图像处理的实验过程。

(2) 用户在进行图像处理实验的时候，只需要对图形控件进行拖拽操作，设置控件的参数、属性等，这样降低了开发的难度，使得一些非专业的人员也可以用系统来进行图像处理实验。

(3) 图形控件封装了一定的算法，当设置好一个图形控件时，控件中的代码就已经固定了，这样就避免了因为少输入一个括号或者单词输入错误等语法上的错误而出现编译不能通过的情况。由于基本上不存在了编译错误，实验人员就可以避免进行长时间的语法调试等工作，可以从大量费时的调试工作中解脱出来，缩短了实验的时间。

2.2 系统的需求分析

随着数字图像处理技术的发展，研制开发的图像处理系统有很多种，但是它们也存在着这样或那样的缺点。比如说，无法完成某些专业领域的一些基本功能；开发软件的程序员一旦离开，由于软件的可重用性差，新来的开发人员就不能修改软件，只能重新开发；由于软件在设计的时候没有考虑到软件功能的扩展和性能的优化，而没有设计可扩展性接口，致使软件很难升级，等等。Photoshop 是目前使用最广泛的一款图像处理软件，它在图形图像设计上和数码照片的处理上做的非常出色，但是这款软件很难满足某些专业领域进行图像处理实验的需求。它在进行图像处理时不能查看处理步骤所设置的参数值，而且不能保存处理流程，使用户每次都要重新设计流程，不利于实验的分析比较。Matlab 和 Labview 软件

中虽然具也有图像处理的功能，但是它们都是功能强大的软件，所占用的磁盘空间比较大，而图像处理只是 Matlab 和 Labview 软件中的一个小小的模块，如果只用它来进行图像处理不仅浪费资源，而且用起来比较麻烦。因此，本系统在开发的过程中应该尽量避开上述缺点。

2.2.1 系统的特点

本文研究的图形化组态的图像处理系统的目的是能够提供各种各样的图像处理算法，用户能够用系统所提供的图像处理算法进行各种各样的图像处理实验，进行方法的探讨和实验的对比。根据以上目的，本文所研究的系统应该具有以下几个特点：

(1) 具有完备的图像处理算法库，能够进行各种各样的图像处理实验

图像处理算法多种多样，表 2-1 为常用图像处理算法分类表。

表2-1 常用图像处理算法分类表

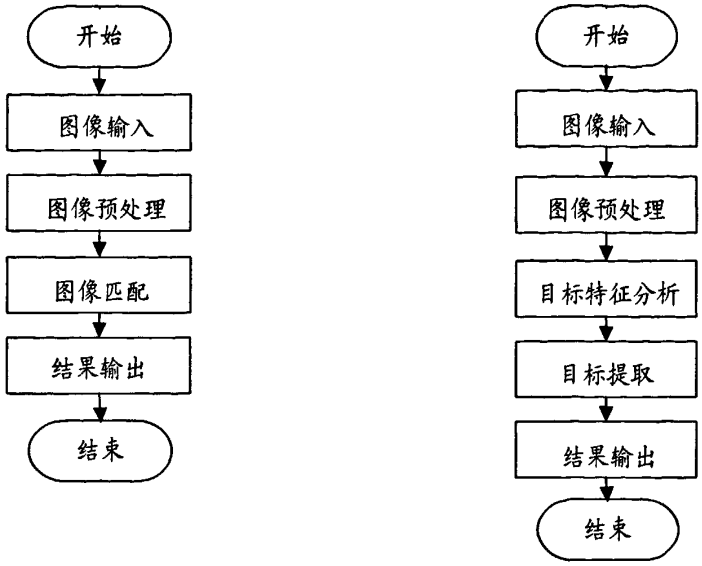
算法类别	方法
图像增强	灰度变换、直方图处理、算术\逻辑操作、滤波、锐化……
几何变换	图像比例缩放、图像平移、图像镜像、图像旋转……
图像分割与边缘检测	二值化、边缘检测、基于区域的分割、分水岭分割……
频域处理	傅立叶变换、离散余弦变换（DCT）、离散沃尔什-哈达玛变换（WHT）、小波变换……
形态学	腐蚀、膨胀、开运算、闭运算、击中、不击中、骨架抽取……
特征提取与对象识别	几何特征、形状特征、纹理特征、图像匹配……
图像复原	非约束复原、非线性复原、几何畸变校正……

系统中的算法库应该提供这些常用的图像处理算法，为了便于查找和使用，应该对它们分门别类。

(2) 实验过程模块化、流程化，能够灵活地搭建组装图

根据用户实验目的的不同，可以将图像处理任务分为四种类型：a 简单的图像到图像的过程，比如，图像的灰度变换、灰度拉升、滤波、锐化等都属于图像到图像的过程；b 图像分析，比如，分析图像的颜色特征、分析图像的噪声类型、分析图像的几何特征、分析图像的纹理特征等等；c 图像量测，比如，测量图像中特征点之间的距离、测量目标的几何量等等；d 对象的识别，比如，识别车牌信息、识别交通标志等等。这些实验过程的实现都需要多种图像处理算法相互协

调完成，而完成一个实验过程不仅可以选用不同的方法实现，同时还可以在实现方法中的每一个步骤时选用不同的算法，进行实验的对比。比如，图 2-1 为实验目标检测的两种不同的方法，而每种方法的每一步又可以采用不同的方法来实现。



(a) 目标检测实验方法一　　(b) 目标检测实验方法二

图2-1 目标检测实验过程

图像处理任务的类型多种多样，对于不同的处理任务，它们之间可能采用某些相同的算法实现，而对于同一处理任务的实现方法并不是固定不变的，可采用不同的处理流程来实现，呈现多态性。所以，为了避免在实验过程中临时编写程序，并且快速地搭建处理流程，进行实验方法的比较，本系统采用图形化组态的软件设计方法，将上述大量的图像处理算法封装成独立的图像处理模块，在外观上用图形控件来表示，并且将这些图形控件分类，用可视化的实验处理流程来代表图像处理过程。为了实现流程的多态性，方便方法的对比，可视化的组装图在绘制的时候应该可以动态地修改，并且可以修改每个算法的相关参数。

为了实现组装图的流程化，相互连接的图形控件必须能相互连通，而图形控件能相互连通的前提是，相连接的图形控件的相应端口的数据类型必须一致。按照图像处理过程输入和输出的输入数据类型可分为：输入是图像输出也是图像，它又可以分为输入为彩色图像输出为灰度图像和输入为灰度图像输出也为灰度图像两种情况；输入是图像输出是数据，比如图像面积求取、图像对比度求取，等等；输入为图像输出为一维图，比如求取图像的直方图；等等。按照图像处理算法所需要的输入图像数目和输出图像数目，图像处理算法可分为一输入一输出、一输入多输出、多输出一输出、多输入多输出四种情况。封装了图像处理算法的图形控件应该具有输入接口、输出接口、所封装的算法、参数四个基本属性。图

形控件的输入输出端口应该包括的属性为：区分输入接口和输出接口的标志、输入图像的数目、端口名、端口传递的数据类型。

(3) 用户操作、实验过程及结果的可视化

可视化有利于用户直观、便捷地进行算法流程组装操作，同时也是图像处理算法的必然要求。用户可以通过拖拽的方法来连接组装图，可以通过结果显示面板来查看实验结果，也可以通过单步执行来设置断点，对实验的中间结果进行查看。

(4) 提供强大的帮助功能

为了使用户更为快速、便捷地搭建组装图，帮助功能需包含以下几项内容：系统操作指南，指导用户如何进行系统的界面操作；图像处理算法的介绍，包括其算法思想、功能、适用范围、性能等，帮助用户搭建各类算法流程。

(5) 具有检错和优化功能

所搭建的流程难免会存在各类问题或错误，例如数据类型的不匹配、无效操作等，对于这类问题平台应能够自动检测，确保算法流程的有效执行。不仅可以根据检错结果对流程结果进行优化，而且还可以在时间、内存消耗、图像质量等方面对处理过程进行优化。

(6) 能够实现图形控件和典型流程的动态扩展

为了使系统的算法库更加完备，用户在使用过程中不仅可以往算法库中添加图形控件，也可以将一些常用的处理流程添加到典型流程库中。

2.2.2 系统的功能需求

基于图形化组态的图像处理系统是为了加快图像处理的速度而设计的一款系统，它可以满足不同层次的用户进行图像处理实验时的需求。基于图形化组态的图像处理实验系统应具有的功能需求如下：

(1) 处理的图像来源多样化

实验将要处理的图像即可以来源于指定的文件夹，也可以通过摄像头实时拍摄。

(2) 组装图的绘制

通过拖拽控件面板上的控件来绘制实验处理流程，给控件设置参数。可以根据用户的需求对组装图进行修改，修改包括删除连线、添加连线、删除控件、添加控件、控件参数的更改，等等。

(3) 组装图的保存

将绘制好的组装图保存下来，同时，若流程中控件的参数已经设置，则将控件的参数也保存下来，方便在以后的实验中使用。

(4) 已保存组装图的打开

打开已经保存的组装图，将组装图显示在流程绘制面板上，用户可以根据自己的需要修改流程。如果组装图符合了运行条件，那么可以直接运行组装图。

(5) 系统的仿真运行

根据用户绘制的组装图进行图像处理。用户绘制的组装图可能出现结构上的错误，或者控件之间的组合不满足数字图像处理的要求，系统可以根据错误的特点提醒用户发生了错误，然后停止运行。系统可以运行包含顺序结构、循环结构、判断结构和子过程调用结构的组装图。

(6) 联机帮助文档

联机帮助文件的作用是告诉用户控件面板上的每一个控件的作用和用法一级系统的使用，帮助用户很快熟悉每个控件和系统的使用。

(7) 添加新的图像处理功能控件

提供一个开放性的外部接口，用户可以通过此接口加入新的算法形成新的控件，供以后使用。

(8) 系统的优化

为了进一步提高图像处理的速度，系统必须具备一些优化功能，比如说，在进行仿真实验时对流程的结构、流程的运行时间、处理结果的质量、内存的使用等方面对系统进行优化。

用户与系统交互的用例图如图 2-2 所示：

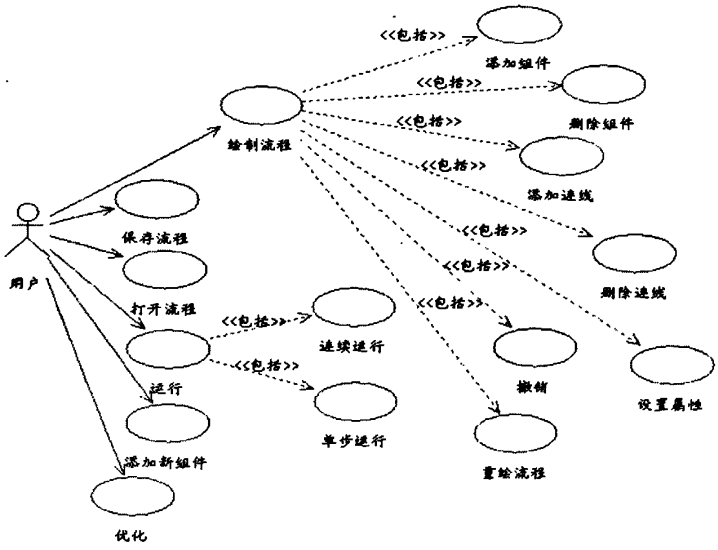


图2-2 用户与系统交互用例图

2.2.3 系统的其它需求

系统在设计的过程中除了需要满足上述的功能需求以外，还应该满足界面友

好性、可扩充性等其它的需求。

(1) 界面友好性

界面设计应该满足使用户易于操作、界面友好、交互性强等要求。

(2) 可扩充性

在软件设计时，要考虑到软件的功能扩充和软件的升级，应该设计一个开放性的外部接口用于软件的扩充。

(3) 完整的文档支持

必须设计完整的文档支持，用于向用户展示每个控件的作用和使用方法，使系统在使用过程中变得简单易用。

2.3 系统的总体设计

2.3.1 系统的功能分析

根据实验系统的需求分析，将基于图形化组态的图像处理实验系统的功能划分为五个相对独立的模块，系统的总体设计模块图如图 2-3。这五个相对独立的模块包括实验控件管理模块、组装图绘制模块、组装图运行模块、实验结果显示保存模块、系统优化模块。

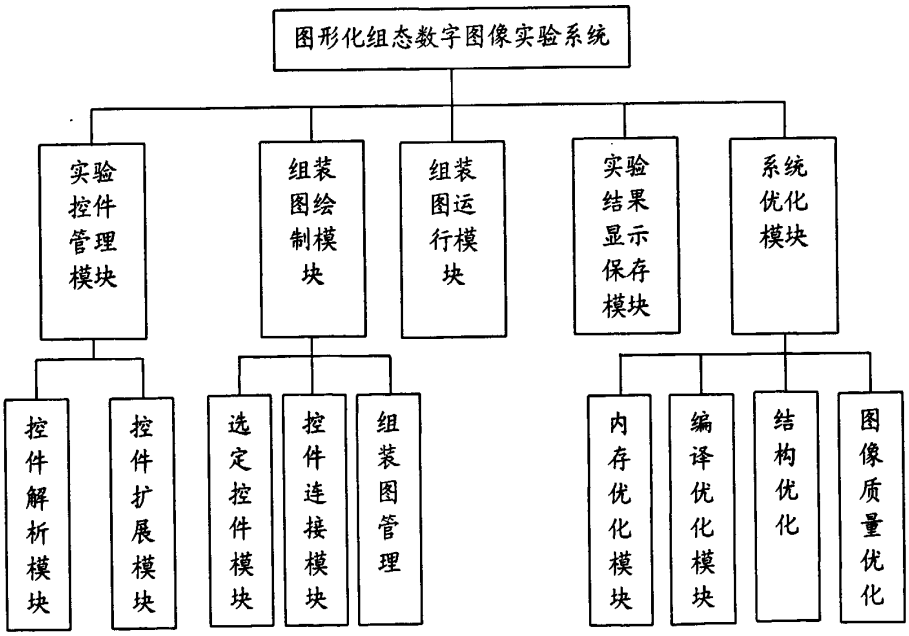


图2-3 系统总体设计模块图

实验控件管理模块：这个模块主要实现对系统控件面板中的控件进行管理，其中包括控件解析模块和控件扩展模块。控件解析模块主要实现在运行过程中，根据控件的参数对组装图中的控件进行解析，调用控件中封装的算法。控件扩展

模块主要实现控件的添加功能，将新的算法封装成控件并且添加到控件面板上，方便用户使用。

组装图绘制模块：这个模块主要实现用户通过鼠标的操作或者是打开已保存的组装图来绘制图像处理组装图，其中包括选定控件模块、控件连接模块和组装图管理模块。选定控件模块主要实现用户通过在控件面板上的鼠标单击事件来对控件在组装图绘制面板上进行实例化，并且实现对控件属性的设置。控件连接模块主要实现需要相连的两个控件的连接。组装图管理模块主要是管理已经存在的组装图，用户可以打开已经保存的组装图，或者是将绘制好的组装图保存到组装图管理库中，方便下次直接打开使用。

组装图运行模块：这个模块主要根据系统的仿真运行机制对组装图中的控件进行排序，执行控件中封装的算法。

实验结果显示保存模块：这个模块主要实现对实验结果的可视化显示，并且将每步的结果保存下来，方便中间结果的查看。

系统优化模块：这个模块只要是为了进一步加快系统对图像处理的速度，其中包括编译优化模块、组装图结构优化模块、图像质量优化模块、内存优化模块。编译优化模块主要是实现在组装图的编译过程中进行优化；组装图结构优化模块主要是对组装图的结果进行优化，合并重复的结构，删除不必要的步骤；图像质量优化主要是选择合适的方法和参数对处理结果进行优化；内存优化主要是减少内存的使用量，尽量避免内存碎片的出现。

系统的模块结构关系图如图 2-4 所示：

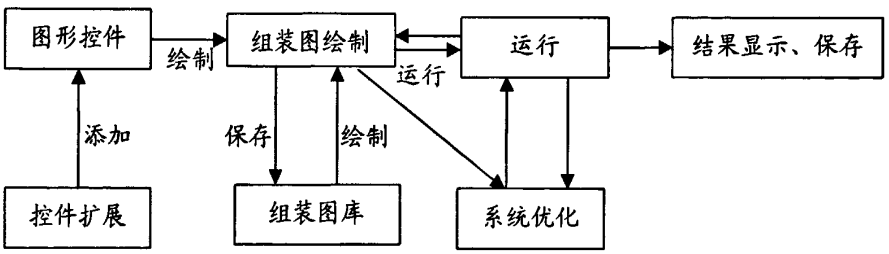


图2-4 系统模块关系图

2.3.2 系统的主要开发工具

实验系统的开发主要使用 Visual Basic 6.0 和 Visual C++ 6.0 开发实现的。Visual Basic 6.0 主要实现软件系统的界面设计，而大量的图像处理算法则由 Visual C++ 6.0 实现。下面将对实验系统所使用的开发工具做一个具体介绍。

(1) Visual Basic 6.0

Visual Basic 6.0 是一种面向对象的程序设计语言，它提供了大量的应用程序

接口，同时也具有对象的链接与嵌入、开放式的数据连接、动态链接库等技术，可以运用这种语言在 Windows 环境下快速地开发各种各样的应用程序。

运用 Visual Basic 6.0 不仅可以建立简单的用户图形界面，还可以建立相对复杂的系统。Visual Basic 6.0 在绘制界面的时候非常方便，可以通过拖放的方式来选择所绘制界面中的控件，即可以通过编写代码的方式设置控件的属性，也可以通过控件的属性面板直接设置控件的属性。程序在运行的时候，控件的属性会随着用户对控件的操作动作而动态地改变。例如：在窗口大小改变的时候，窗口中控件的位置和大小也会相应地发生改变，所以，其相应的属性也会随之发生变化。

Visual Basic 的中心思想就是便于程序员使用，它基于图形界面的开发环境使开发者对各种功能一目了然、容易理解。用户仅仅通过鼠标的简单操作就可以构建出一个软件图形界面。

(2) Visual C++ 6.0

Visual C++ 6.0 是一个功能非常强大的可视化应用程序开发工具，它是微软公司 1998 年推出的产品，它集成了编译器、链接器、调试器、APP Wizard、Class Wizard、AppStudio 等多种多样的可视化编程工具，利用它可以完成各种各样的应用程序的开发。它能够对网络、数据库等方面的编程提供相应的环境支持，同时也能够对多种操作系统下的 C++程序设计提供完善的编程环境。Microsoft 的基本类库 MFC 提供了大量的类，例如，进行文件操作、数据库操作、网络连接、绘图的类，等等，这使得开发 Windows 应用程序变得比较简单，使得开发人员能够快速得建立各种应用程序，减少了开发人员的工作量。C++语言相对于其它高级语言有很高的代码效率，算法能够快速有效地执行，具有较好的算法可移植性。提供的各种函数、指针操作和直接对硬件的操作使得它对数字图像处理的速度加快[43]。

Visual Basic 6.0 在界面编程中具有所见即所得的优势，且简单，快速，但是代码的运行速度比较慢。Visual C++ 6.0 编写的程序编译后代码的执行速度比 Visual Basic 6.0 快，但是编程较为复杂。因此，对于本文所介绍的图像处理实验系统的开发结合 Visual Basic 6.0 和 Visual C++ 6.0 的优点，对于系统界面等不涉及大量数值计算的程序可以用 Visual Basic 6.0 来实现，而对于图像处理等涉及大量数值计算的程序可以用 Visual C++ 6.0 来实现。在实现的过程中，主要采用了 Visual C++ 6.0 中的动态链接库(DLL)技术，实现 Visual Basic 6.0 和 Visual C++ 6.0 的优化组合，这样开发出来的图像处理系统不仅能够提高编程的效率，同时也能提高图像处理的速度。

2.3.3 系统采用的关键技术

在系统的开发过程中，主要使用了 Visual C++ 6.0 中的动态链接库(DLL)技术，实现 Visual Basic 6.0 和 Visual C++ 6.0 的优化组合。

动态链接库，即 DLL(Dynamic Link Library)，是一个可以被其它程序共享的程序模块，其中封装了一些可以被共享的资源，但是它不能单独运行，必须由其它应用程序直接或者间接调用。动态库实际上是一个单独的程序模块，将程序中的某一部分独立出来以动态链接库的形式存在，可以使软件更加地模块化。一方面，在开发阶段，编译时只需要重新编译部分更改了的模块；另一方面，在程序的使用阶段，如果有更新，也只需要更新有问题的模块。在软件的开发过程中使用动态链接库有以下几个优点：

首先，使用动态链接库可以节省内存空间。当动态链接库被可执行文件调用时，动态库中的函数和数据并不需要复制到可执行文件中，因此在应用程序的可执行文件中，存放的不是被调用函数的代码，而是动态链接库中所要调用的函数的内存地址。这样，动态链接库只需要在内存中加载一次，所有要使用此动态链接库的程序就会共享这部分内存，从而可以节省内存空间。

其次，可以隐藏软件实现的细节。软件的某些关键实现代码如果不想被其他人知道，就可以将需要隐藏的那部分用动态链接库实现。

再次，使用动态链接库便于软件的升级。当一个软件需要升级的时候，如果在软件的开发过程中使用了动态链接库技术，系统升级只需要更换系统中原有的动态链接库，而不需要重新编译整个系统。

除此之外，动态链接库具有与语言无关的特点。因此，可以创建一个动态链接库，这个动态链接库可以被 C++、VB 或者是任何支持动态链接库的语言调用。这样，如果一种语言在某方面存在不足，就可以通过访问另一种语言创建的动态链接库来弥补。因为 Visual Basic 6.0 在设计界面的时候具有可见即可得的优点，但是它的代码运行速度比较慢，如果用它来编写需要有大量计算的图像处理程序，就会大大降低图像处理的速度。所以，结合 Visual Basic 6.0、C++以及动态链接库他们各自的优势，采用动态链接库使 Visual Basic 6.0 和 Visual C++ 6.0 优化组合，在此基础上共同开发图像处理系统。

2.4 小结

本章介绍了基于图形化组态的软件设计，通过对系统的需求分析，确定了系统的基本功能和总体设计方案，以及主要使用的开发工具和系统开发所采用的关键技术。下一章将根据本章的分析，具体介绍系统的详细设计与实现。

第三章 系统的详细设计与实现

在围绕系统的需求分析的基础上，本章将分析和论述系统的详细设计与实现过程。其中，系统关键类的设计，系统主要功能模块的设计，通用性扩展接口的设计，以及在分析已有的图形化组态软件运行机制的基础上提出本系统的图形化组态的运行机制将是本文研究的重点和难点。

3.1 关键结构的设计

由第二章的分析可知，基于图形化组态的图像处理实验系统用组装图来表示图像处理实验步骤。因此，用户所绘制的组装图是本系统的一个关键结构。这个有向组装图是由图形控件和图形控件之间的连线组成。若用 P 表示有向组装图，GS 表示有向组装图中的所有图形控件，ES 表示组装图中所有图形控件之间的连线，则可以用 P=(GS，ES)来表示有向组装图。下面将详细介绍图标控件和图标控件之间的连线的结构。

3.1.1 图形控件结构

根据组装图的要求，图形控件可分为操作类图形控件、控制类图形控件和复合类图形控件三大类。操作类控件包括算法类控件和人机界面类控件，算法类控件中封装了一种图像处理算法，人机界面类控件封装了用来打开原图像和显示功能的算法；控制类图形控件主要是在包含选择结构或者循环结构的组装图中用到，用来控制组装图的走向；复合类图形控件中封装了两个或以上的图像处理算法。图形控件 V 可以表示为 V=(Index，Type，In_num，Out_num，Para，Fun，P_{in}，P_{out})，其中：

Index 表示图形控件的索引值；

Type 表示图形控件的类型，当它的值为 0 时表示操作类图形控件，值为 1 时表示控制类图形控件，值为 2 时表示复合类图形控件；

In_num 表示组装图中以此图形控件为终点的连线的数目，即图形控件的入度，它是在对组装图中的图形控件进行仿真运行的时候将要用到的一个重要的量；

Out_num 与 In_num 相反，表示组装图中以此图形控件为起点的连线的数目，即图形控件的出度，它也是在对组装图中的图形控件进行仿真运行时将要用到的一个重要变量；

Para 表示图形控件中所封装函数所需要的参数；

Fun 表示图形控件中所封装的函数；

P_{in} 表示图形控件的输入端口的集合，它可以为空；

P_{out} 表示图形控件的输出端口的集合，它可以为空。

这三类图形控件都是由图形部分，功能部分和属性面板三部分组成，图 3-1 为图形控件结构图。其中，图形部分是图形控件的外部表现形式，功能部分是图形控件的核心部分，里面封装了函数的具体实现，外部是不可见的，属性面板则可看成是为控件的功能部分传递参数的外部接口。

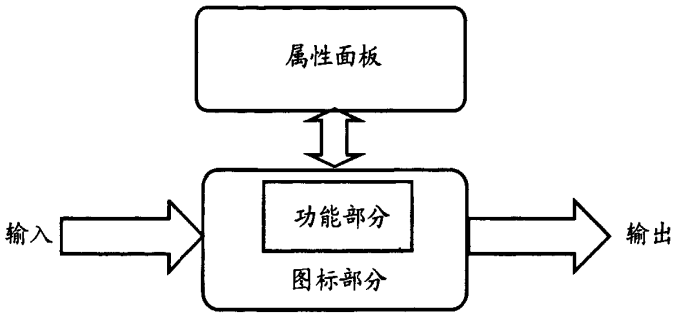
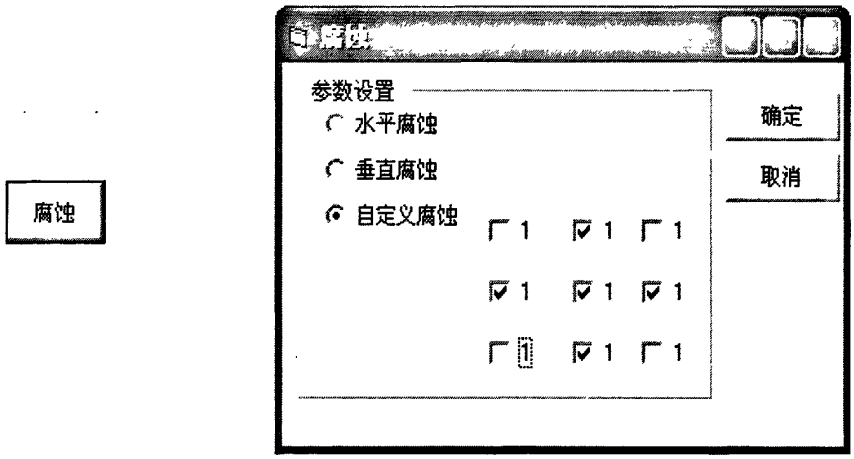


图3-1 图形控件结构图

以“腐蚀”图形控件为例，图 3-2 表示“腐蚀”图标控件的图形部分和属性面板。



(a) 图形部分 (b) 属性面板

图3-2 图形控件的图形部分和属性面板

3.1.2 连线结构

组装图中图形控件之间的连线主要有两方面的作用：一是作为图形控件之间的可视化连接，二是表示数据传输的方向，两个相互连接的图形控件之间有数据从连线的起点图形控件传输到连线的终点图形控件，从而完成数据的传输过程。图标控件之间的连线 E 可以用 E=(Index, Type, Start_index, End_index, Left_Right),

其中：

Index 表示连线的索引值；

Type 表示连线的类型。有两种类型的连线：一种是带箭头的直线，它的 Type 为 0；另一种是带箭头的折线，它的 Type 为 1，当组装图中包含循环结构时，会绘制出带箭头的折线；

Start_index 表示与连线起点相连接的图形控件的索引值；

End_index 表示与连线终点相连接的图形控件的索引值；

Left_Right 用来区分与控制类图形控件的左端口还是右端口相连的标志。为 0 表示与控制类图形控件的左端口相连；为 1 表示与控制类图形控件的右端口相连。

3.2 主要功能模块的设计

3.2.1 用户操作界面

图 3-3 为用户操作界面。用户操作界面包括菜单栏、方法选择列表、组装图设计面板。菜单中包含了“文件”、“运行”等操作。

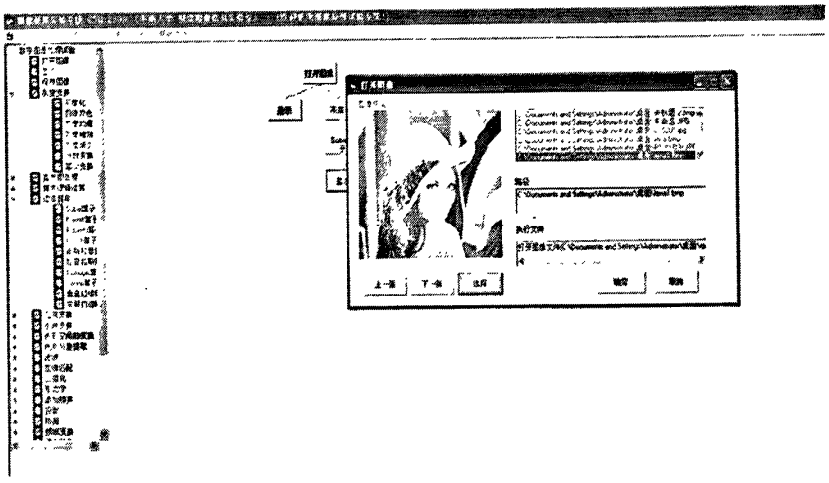


图3-3 用户操作界面

用户操作界面最主要的作用就是用户可以根据自己的需要绘制组装图，因此，用户操作界面中也包含一系列的在组装图绘制过程中对组成组装图的图形控件和连线的操作。图 3-4 是用户操作界面中的组装图绘制的状态转换图，它以绘制组装图为中心，主要包括四部分的内容：从方法选择列表中选择所需的方法，生成新的图形控件并且显示出来；分别选定连线的起点控件和终点控件，绘制两个控件之间的连线；选中已经存在的需要移动、删除或者是要进行属性设置的图形控件，对它们进行移动、删除或者属性设置的操作；选中将要删除的连线并且删除此连线。

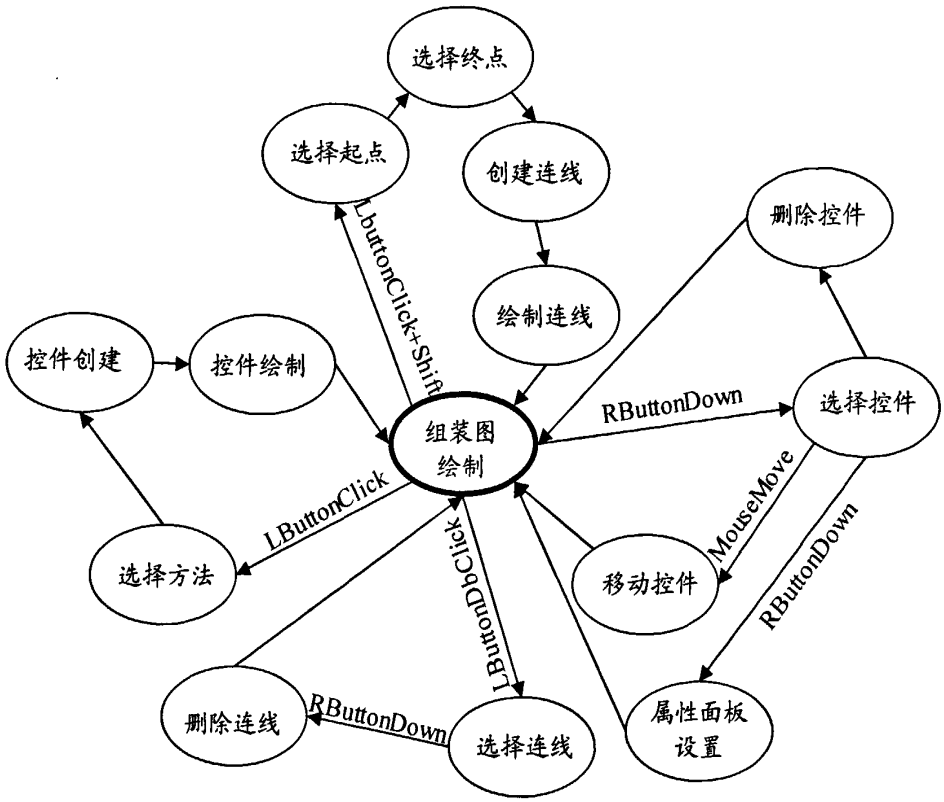


图3-4 组装图绘制状态转换图

从图 3-4 可以看到，所有的操作都是以组装图绘制为中心的。在方法选择列表的待选方法的相应位置按下鼠标左键就可以完成方法的选定，并且在组装图绘制面板上绘制出相应的图形控件；按下鼠标左键加上“shift”键就可以在选定的两个图形控件上画线；在要进行处理的图形控件上按下鼠标右键，通过弹出菜单就可以对相应的图形控件进行一系列的操作，等等。所有的这些过程都可以通过鼠标的点击，同时借助少量的键盘操作就可以完成，通过这种方式进行图像处理不仅可以使图像处理过程更加可视化，而且还可以使图像处理的速度大大提高。

3.2.2 组装图的绘制

本系统可以通过两种方式进行组装图的绘制：一种方式是通过打开已经保存的组装图，根据已保存组装图的配置文件里面的相关信息进行组装图的绘制，用户可以对打开的已保存的组装图进行修改；另一种方式是通过鼠标的操作来完成组装图的绘制。这里将重点介绍第二种组装图绘制方法。

因为，动态数组不需要预先定义数组的大小，数组的大小在任何时候都可以改变，所以，组装图中的图形控件和连线分别用类型为自定义的控件结构的动态数组 *VList* 和自定义的连线结构动态数组 *EList* 来保存。当需要生成新的图形控件或者是连线时，只需新建一个自定义控件类型的变量或者自定义连线类型的变量，

将他们分别添加到动态数组中，同时改变动态数组的大小即可。

设*ST*为连线的起点图形控件，*TT*为连线的终点图形控件，*ST.top*、*TT.top*分别为起点图形控件和终点图形控件的上边界的纵坐标，*ST.left*、*TT.left*分别表示起点图形控件和终点图形控件在左边界的横坐标，*width*表示图形控件的宽度，*height*表示图形控件的高度，*line*为将要绘制的连线，*line.x1*为*line*的起点的横坐标，*line.y1*为*line*的起点的纵坐标，*line.x2*为*line*的终点的横坐标，*line.y2*为*line*的终点的纵坐标，*pos[]*记录折线的折点。按照连线的起点图形控件和终点图形控件的类型和相对位置，可以将连线的绘制分为以下几种情况：

*ST.top*小于或等于*TT.top*，并且*ST*非控制类图形控件；

*ST.top*小于或等于*TT.top*，并且*ST*为控制类图形控件；

*ST.top*大于*TT.top*，并且*TT*非控制循环结构的控制类图形控件；

*ST.top*大于*TT.top*，并且*TT*为控制循环结构的控制类图形控件。

根据这四种情况，画线的算法为：

(1) 当*ST.top*小于或等于*TT.top*，并且*ST*非控制类图形控件，这种情况又可根据*TT*是否为控制循环结构的控制类图形控件分为两种情况。

当*TT*为非控制循环结构的控制类控件时，连线*line*的起点和终点坐标为：

$line.x1 = ST.left + width \backslash 2;$

$line.y1 = ST.top + height;$

$line.x2 = TT.left + width \backslash 2;$

$line.y2 = TT.top$

当*TT*为控制循环结构的控制类控件时，连线*line*的起点和终点坐标为：

$line.x1 = ST.left + width \backslash 2;$

$line.y1 = ST.top + height;$

$line.x2 = TT.left + 2*(width \backslash 3);$

$line.y2 = TT.top$

(2) 当*ST.top*小于或等于*TT.top*，并且*ST*为控制类图形控件，以控制类图形控件为起点时，连线分为左连线和右连线，根据鼠标在控制类控件上的点击位置来确定。

当鼠标点击控制类控件的左半部分时，*line*的起点和终点坐标为：

$line.x1 = ST.left + width \backslash 3;$

$line.y1 = ST.top + height;$

$line.x2 = TT.left + width \backslash 2;$

$line.y2 = TT.top$

当鼠标点击控制类控件的右半部分时，*line*的起点和终点的坐标为：

$line.x1 = ST.left + 2*(width \backslash 3)$;
$line.y1 = ST.top + height$;
$line.x2 = TT.left + width \backslash 2$;
$line.y2 = TT.top$

(3) 当*ST.top*大于*TT.top*，并且*TT*非循环类控制图形控件时，*line*的起点和终点的坐标为：

$line.x1 = ST.left + width \backslash 2$;
$line.y1 = ST.top + height$;
$line.x2 = TT.left + width \backslash 2$;
$line.y2 = TT.top$

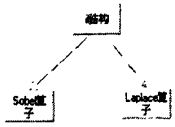
(4) 当*ST.top*大于*TT.top*，并且*TT*非控制循环结构的控制类图形控件时，*line*的绘制方法为：

$pos[0].x = ST.left + width \backslash 2$;
$pos[0].y = ST.top + height$;
$pos[1].x = pos[0].x$;
$pos[1].y = pos[0].y + 150$;
$pos[2].x = ST.left - 200$;
$pos[2].y = pos[1].y$;
$pos[3].x = pos[2].x$;
$pos[3].y = TT.top - 150$;
$pos[4].x = TT.left + width \backslash 3$;
$pos[4].y = pos[3].y$;
$pos[5].x = pos[4].x$;
$pos[5].y = TT.top$

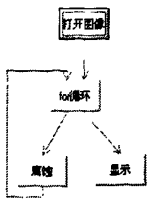
将求得的折线折点数组*pos[]*中的折点两两相连，得到从起点*ST*指向控制循环结构的控制类图形控件终点*TT*的折线。

图 3-5 为组装图几种结构的连线示例。(a)为组装图中不存在控制类图形控件的情况；(b)为组装图中存在控制分支结构的控制类图形控件的情况；(c)为组装图中存在控制循环结构的控制类图形控件的情况。

(a) 无控制控件



(b) 有分支控制控件



(c) 有循环控制控件

图3-5 连线示例

3.2.3 实验结果显示

系统的界面主要可分为前面板和后面板，前面板也就是用户操作界面，而后面板主要用来显示实验的结果。从组装图到输出结果这一过程实际上是前面板和后面板相互交互的过程。在后面板上都有一个虚拟的图形控件与前面板上组成组装图的每一个图形控件相对应。如图 3-6 所示，节点 A、B、C 为前面板上组成组装图的图形控件，它们在后面板上对应的节点分别为 A1、B1、C1，连线 e1、e2、e3、e4、e5 表示前面板和后面板之间的数据流动方向。

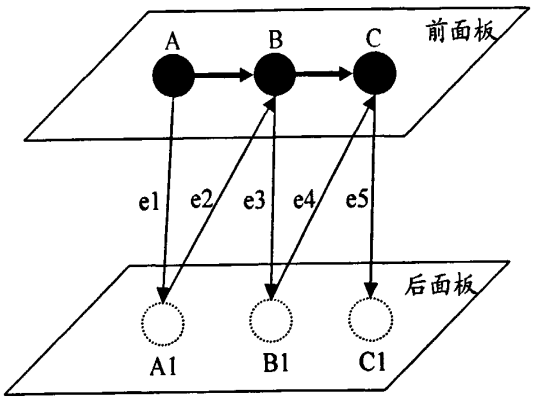


图3-6 前面板和后面板交互图

前面板中组成组装图的每一个图形控件都代表了一个功能的具体实现，而正是在前面板和后面板的交互过程中实现图形控件的解析。每个图形控件封装的函数模型如式(3-1)：

$$fun(p_{in0}, p_{in1}...p_{inn}, p_{out0}, p_{out1}...p_{outn}) \tag{3-1}$$

其中，fun 为函数名，$p_{in0}, p_{in1}...p_{inn}$ 为函数的输入参数，$p_{out0}, p_{out1}...p_{outn}$ 为函数的输出参数。

3.3 组装图分析与设计

3.3.1 组装图模型

组装图是由节点和节点之间的连线组成的，可出有向图来表示它们相互之间的关系。节点可表示为封装了图像处理算法的图形控件，节点间的连线表示为图形控件之间的数据流向。通过拖拽算法控件的方式组装的结构，必须是可以运行的，所以用户所绘制的组装图必须具有以下的几点特点：

(1) 有向性：从一个节点到另一个节点存在一种有向的关系，前者是后者的前驱，而后者是前者的后继，有向性决定了整个组装图中数据的流动是有方向的。

(2) 有限性：组装图的运行可以在有限的时间内完成，不存在死循环。

(3) 条件性：某个节点可能存在有多个前驱，一个节点若想被激活，必须是它的所有前驱运行完毕并且有输出，也就是说每个节点的所有输入接口都有输入。

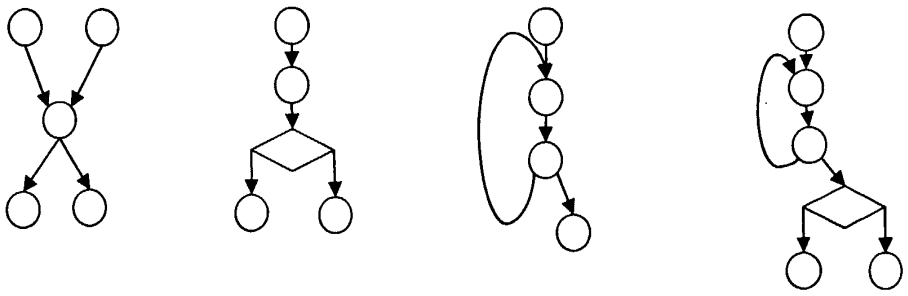
根据上述特点，组装图可用有向图来表示，合格的有向图应该满足的条件为：

(1) 组装图中至少有一个没有前驱节点的节点，并且至少有一个没有后继节点的节点。

(2) 为了避免死循环，不存在节点自身的回路。

(3) 当有控制循环结构和分支结构的节点存在时，控制和分支类节点的两个端口分别指向不同的支路。

图 3-7 为同时满足上述几种条件的几种特殊结构的有向图，其中，(a)为不包含分支和循环结构的组装图，(b)为包含分支结构的组装图，(c)为包含循环结构的组装图，(d)为既包含分支结构也包含循环结构的组装图：



(a) 顺序结构 (b) 分支结构 (c) 循环结构 (d) 分支循环结构

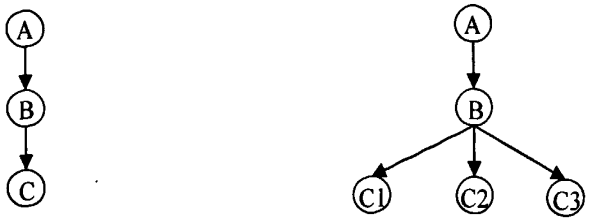
图3-7 有向图示例

从图 3-7 可以看到，当有向图中不存在分支和循环结构时，可以将它看成是一个有向无环图，当它满足这一条件时，可用顶点表示活动的 AOV 网来表示组装图。AOV 网是一种特殊的有向图，AOV 网具有的特点是：从一个节点到另一个节点存在有向的关系，这种有向的关系可以用来描述时间或者位置之间的优先关系；某个顶点可能有多个前驱节点，即它的入度大于 1，将所有前驱节点都全部满足运行条件来作为某个节点的开始条件；图中不存在环。

3.3.2 顺序结构

这种结构相对于比较简单，就是让处理过程按照连线的顺序执行。但是顺序结构又存在两种形式，即存在分叉和不存在分叉，如图 3-8 所示。从图 3-8 可以看到，(a)的结构相对于(b)简单，(b)中在节点 B 处出现了分叉，节点 C1、C2、C3 的直接前驱都是节点 B，它们之间是一种平等的关系，理论上来说，只要它们的运行条件满足，是同时运行的。但是，因为一个时间只能处理一个节点，所以它们之间的运行存在一个先后顺序。分叉并行结构中分叉节点的运行顺序是由在连

线动态数组中分叉节点与它们的直接前驱之间连线的先后顺序决定的。



(a) 不存在分叉 (b) 存在分叉

图3-8 顺序结构示例

3.3.3 分支结构

分支结构是组装的基本结构，也是一个比较复杂的结构，相当于 if 语句：

if(*condition*)

{.....}

else

{.....}

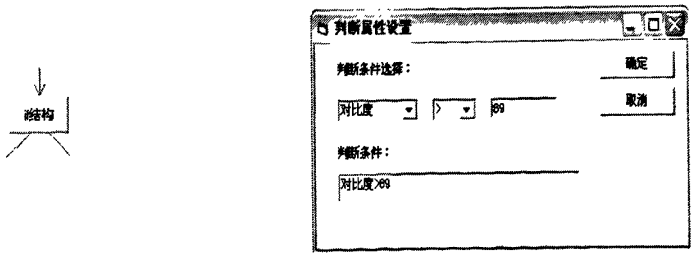
它由表达式 *condition* 计算出来的结果来决定哪些语句块，*condition* 的计算结构只可能是两种情况，真值“*true*”和假值“*false*”。组装图中的控制分支结构的图形控件就相当于 if 语句，它可以根据表达式的结果来确定下一步选择哪条路径来运行。图 3-9 为控制分支结构的图形控件和它的属性面板，(a)为分支控制图形控件的图形部分，上端为输入端口，左下端为“真”值端口，右下端为“假”值端口，当表达式的值为“真”时，则选择与分支控件相连的左支路部分继续运行；相反，当表达式的值为“假”时，则选择与分支控件相连的右支路继续运行。(b)为分支控制图形控件的属性面板，此表达式等价的 if 语句为：

if(对比度>89)

{.....}

else

{.....}



(a) 分支控件 (b) 属性面板

图3-9 分支控件及其属性面板

3.3.4 循环结构

循环结构和分支结构一样，也是组装图中的一个重要结构，正因为分支和循环结构的存在，使实验系统可以实现相对于比较复杂的递归结构。组装图中的循环结构可分为两种：一种是直接设定循环的次数，它相当于程序语言中的 for 语句；另一种是将某一变量作为循环的判定条件，当判定条件为“真”时，继续执行循环结构中的循环体，当条件为“假”时，则跳出循环。图 3-10 为循环结构的图形控件的结构设计，循环类控件一共有四个端口，A、A0、T、F，其中：

端口 A0：循环控件的直接前驱控件的输出数据，若为控制 while 循环结构的控制类图形控件还包括判断表达式中判断变量的初始值；

端口 A：若为控制 for 结构的控制类图形控件，此端口表示输入的循环次数；若为控制 while 循环结构的控制类控件，此端口表示输入的判断变量的值；

端口 T：表达式的值为“真”时选择的输出端口；

端口 F：表达式的值为“假”时选择的输出端口。

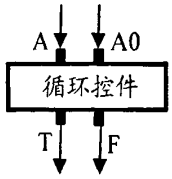


图3-10 循环控件结构

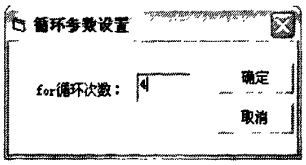
(1) for 循环结构

程序语言中的 for 循环语句为：

```
for(i = 0;i < N;i++)          或          for(i = N;i > 0;i--)
  {                                         {
    循环体语句;                               循环体语句;
  }                                         }
```

本系统在 for 循环的设计上采取第二种 for 循环形式，当循环次数大于 0 时，继续执行循环体中的内容，并且将循环次数减 1，当循环次数不大于 0 时，则跳出循环，循环结束。图 3-11 为 for 循环控件的图形部分以及属性面板。

(a) for 循环控件



(b) 属性面板

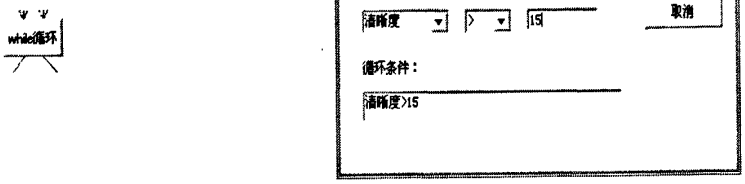
图3-11 for 循环控件及其属性面板

(2) while 循环结构

程序语言中的 while 循环语句为：

```
while(逻辑表达式)
  {
    循环体语句;
  }
P;
```

while 循环结构比 for 循环结构要稍微复杂一点，它需要在执行 while 语句时计算逻辑表达式的值，并且根据表达式的值来确定是否继续执行循环体。当逻辑表达式的值为“真”时则继续执行循环体中的内容，当逻辑表达式的值为“假”时，则跳出循环，执行循环语句后的“P”语句。图 3-12 为 while 循环控件的图形部分以及属性面板。



(a) while 循环控件 (b) 属性面板

图3-12 while 循环控件及其属性面板

3.3.5 子程序结构

有的实验涉及到的实验步骤很多，在组装图的搭建过程中必定会涉及到很多图形控件，为了简化组装图，让组装图看起来更直观，利用模块化设计的思想，将经常用到的组装图封装到一个图形控件中，形成复合图形控件。当实验运行到复合图形控件这步时，会自动按次序去执行符合图形控件中所封装的简单图形控件相关的处理函数，当处理函数都执行完以后又返回执行复合图形控件后面的步骤。

3.4 系统运行机制的设计

3.4.1 图形化组态软件常用运行机制

图形化组态软件的运行机制主要是根据用户所绘制的组装图得到输出结果。在图形化组态软件中，组装图可分为静态结构和动态结构。静态结构中节点的运行顺序在编译阶段就可以被确定，也就是说在静态结构中，节点的执行顺序是根据它们的连接情况是可以预先确定的；动态结构要比静态结构复杂得多，在动态结构中，每个节点可能会有不同的激活条件，根据条件的不同，下一步执行的顺

序也可能不同，在这种情况下，当一个组装图绘制好后，其节点的顺序是无法预先确定的，而且节点的运行情况还依赖于运行时的激活条件。

文献[44]中用到拓扑排序的思想对组装图中的节点进行排序。拓扑排序的原理是静态结构运行机制中常用到的原理，运用拓扑排序的原则，首先将组装图中的节点按照它们相互之间的连接顺序排序，然后按照排好的顺序逐个运行，这种方法的运行步骤为：

(1) 定义两个节点队列，分别保存的初始节点和运行节点，再定义一个连线队列，保存节点之间的连线；

(2) 节点队列中是否存在入度为 0 的节点，如果存在执行步骤(3)，否则跳到步骤(5)；

(3) 将入度为 0 的节点放入运行节点队列，同时将此节点从初始节点队列中删除；

(4) 将所有依赖于该节点的连线从连线队列中删除，修改初始节点队列中节点的入度，转到步骤(2)；

(5) 运行节点队列是否为空，如果不为空则执行步骤(6)，否则执行步骤(7)；

(6) 取出运行节点队列的第一个节点运行，删除该元素，转到执行步骤(5)；

(7) 结束。

这种方法是完全依赖于节点的入度信息来进行排序的，它不适合于有环的组装图；它首先对组装图中的控件进行排序，根据排序的结果再运行节点，所以也不适合于动态结构，同时，这种方法也会增加一个用于保存运行节点的内存空间。

文献[45]提出了一种基于数据驱动的图形化组态软件的运行机制。这种运行机制在节点的结构上增加了数据输入端口、数据输出端口以及控制流端口，数据输入和数据输出端口用于描述数据的状态、类型以及传递状态情况，控制流端口描述组装图的控制情况。同时，这种机制也引入了节点的可激活条件以及节点的优先级，并在运行之前对节点列表进行排序。这种方法也是在运行之前对节点列表进行排序，不能满足一边运行一边排序的要求。

3.4.2 系统的运行机制

因为组装图可能包含分支和循环结构，所以组装图为动态组装图。本系统所采用的运行机制是一种基于数据驱动的方法，它是一种边排序边运行的方法，只要组装图中的图形控件满足激活条件，它就可以运行。控件的激活条件也是指控件的所有输入端口都有有效的数据，如果控件不满足激活条件，则将继续等待，转向处理它后面的控件。在运行机制的实现上借助栈这种数据结构先进后出的特点实现，当前的栈顶元素一定是满足激活条件，等待运行的控件元素。步骤如下：

(1) 计算组装图中控件的入度，将入度为 0 的控件入栈；

(2) 判断栈是否为空，如果为空转到执行步骤(8)，否则执行步骤(3)；

(3) 栈顶控件出栈，记为 p；

(4) 运行控件 p 相映的函数体；

(5) 查找控件 p 的直接后继控件 q。若控件 p 为简单控件则查找它的所有直接后继控件；若控件为分支类或者循环类图形控件，则根据判断表达式或者是循环逻辑表达式计算的结果来决定所要选择的直接后继；

(6) 判断直接后继控件 q 是否满足激活条件，如果满足则执行步骤(7)，否则转到执行步骤(2)；

(7) 满足条件的直接后继控件入栈，转到执行步骤(2)；

(8) 结束。

系统的仿真运行组装图如图 3-13 所示。

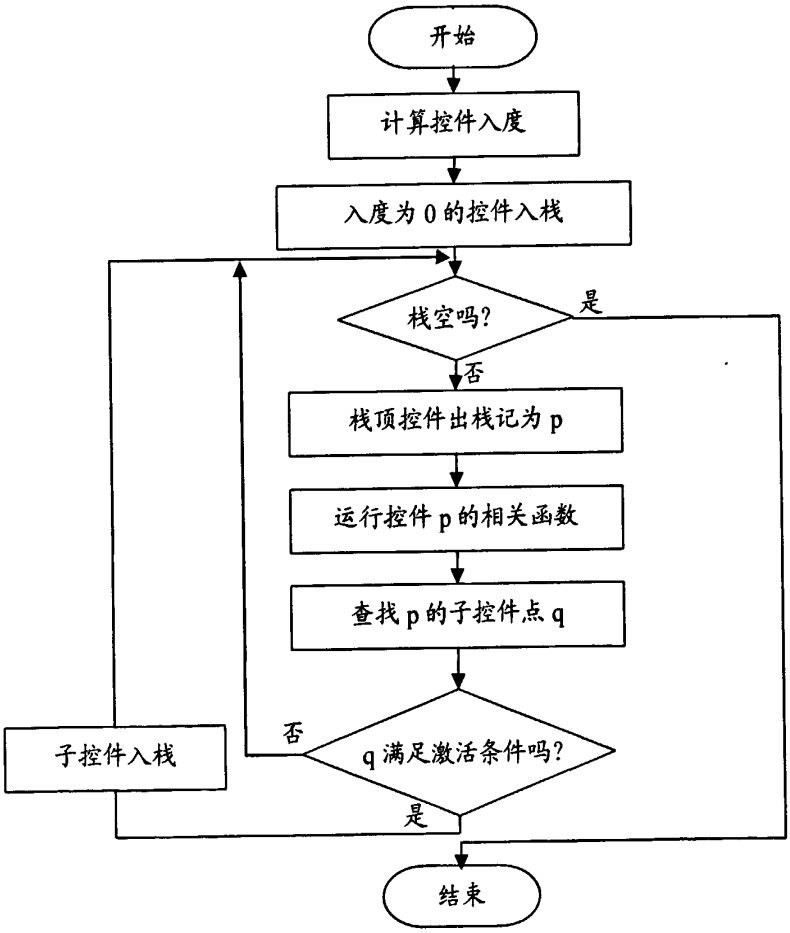


图3-13 系统仿真运行流程图

这种运行机制的特点是：它相当于一个深度优先过程，可以依次连续运行分叉结构中同一条支路上的串联节点，当运行到汇聚点的时候则停下来，再继续运

行其它支路上的节点，这样同一条支路上的串联节点可以看成一个复合节点。有利于组装图的优化。此方法和基于拓扑排序的运行机制相比，不仅可以适用于动态组装图，而且因为它采取的是边排序边运行的方法，不需要连续计算图形控件的入度，所以不需要额外定义一个数组来保存运行的控件，可以减少内存的开销。同时，这种运行机制还有一个优点就是对于具有分叉结构的组装图来说，它可以将一条支路上的所有满足激活条件的图形控件运行完成以后，再转向去执行其它支路上的图形控件。如图 3-14 所示，是一个包含分叉结构的有向图，控件 a 具有三条分支，它们之间是并行的关系。用拓扑排序的方法，此组装图的控件运行顺序为 ABCDEFGHI，而运用本系统的运行机制，此组装图的控件运行顺序为 ABEGCFDHI。可以看出后一种运行顺序将处于同一条支路上的控件放在一起运行，这样可以使组装图的结构看起来更加清晰。

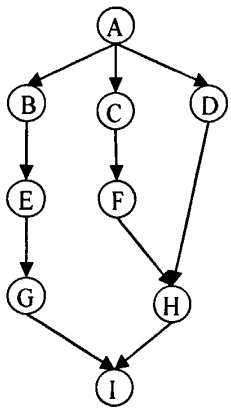


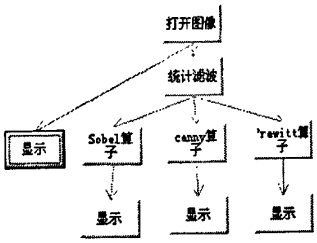
图3-14 具有分叉结构的有向图示例

3.5 扩展接口的设计

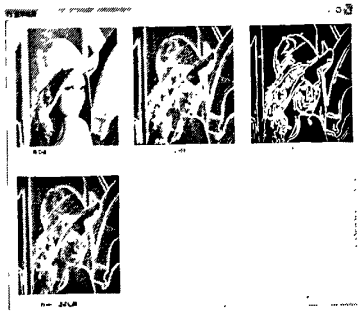
扩展接口可以将用户自己编写的图像处理程序自动封装成图形对象，添加到系统中去，供以后使用。此开放性的外部接口主要包括添加函数的函数名，添加函数的名字，函数的参数的个数以及参数。添加函数的函数名和添加函数的名字是不可缺少的，但是输入的参数信息可以为空。当用户要往系统中添加一个新算法时，只要提供外部接口中的相关信息，系统就可根据用户提供的信息，生成与添加函数的名字相同的图标对象，并且将生成的图标对象添加到已有的算法工具栏中保存下来。同时，系统将对应添加函数的函数名和添加函数的名字对应的保存下来，使组装图解析的时候能够正确的自动调用相应的函数。

3.6 运行示例

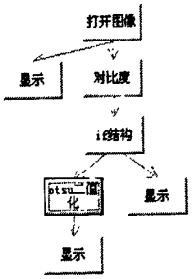
图 3-15 为系统的运行示例图。其中，(a)为具有分叉结构的顺序组装图；(b)为(a)中所绘制组装图的运行结果，第一幅为原图像，后面几幅分别为原图像经过滤波处理后分别进行用 sobel 算子、canny 算子、prewitt 算子进行边缘检测后的结果；(c)为具有分支结构的组装图；(d)为(c)中所绘制组装图的运行结果，其中“if 结构”控件中设置的判断条件为“对比度>56”，经过计算后图像的对比度大于 56，选择左边的值为“真”的支路继续运行，第一幅图为原图像，第二幅为运行的结果；(e)为具有循环结构的组装图；(f)为(e)中所绘制组装图的运行结果，其中循环的次数设置为 5，第一幅图为原图像，第二幅图为经过五次腐蚀后的结果。



(a) 组装图示例一

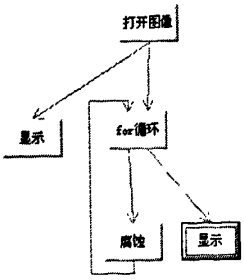


(b) (a)运行结果

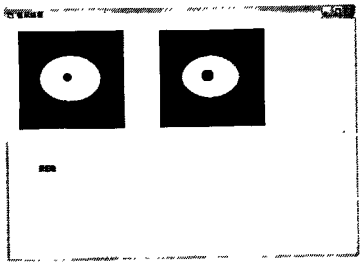


(c) 组装图示例二

(d) (c)运行结果



(e) 组装图示例三



(f) (e)运行结果

图3-15 运行示例图

3.7 小结

本章介绍了系统的详细设计过程，其中重点介绍了组成组装图的基本单位图形控件和控件间的连线结构，以及组装图的绘制方法和主要功能模块的设计。在分析组装图的基础上提出了一种适用于顺序、分支、循环、子过程调用四大组装图结构的运行机制。最后，介绍了系统扩展接口的设计方法。

第四章 系统的优化

为了进一步提高图像处理的速度，使组装图更简化，在图像处理过程中占用的内存空间更少，并且提高图像处理的质量，本章主要研究系统的优化。在分析了图像处理特点和优化必要性的基础上，从组装图的编译过程、组装图、图像的质量、内存等方面对系统进行优化。

4.1 优化概述

4.1.1 图像处理特点的分析

数字图像处理过程是将图像信号转化为数字信号，并且交给计算机进行处理的一个过程。数字图像处理具有以下特点：

(1) 图像处理涉及的领域比较广泛，它涉及到军事、工业、医学、航空、教育等诸多领域。同时，它涉及到的技术领域也相当广泛，如通信技术、计算机技术、电子技术、电视技术等。

(2) 图像中含有丰富的信息，图像的数据量非常庞大，可以通过图像处理技术来获取图像中的有用信息。图像处理的信息大多是二维信息，图像是由图像矩阵中的像素组成，若每个像素用红、绿、蓝三种颜色表示，每种颜色用 8bit 表示，则一幅 1024×1024 的彩色图像，数据量达到 24Mb(即 1024×1024×8bit×3=24Mb)，如此庞大的数据量给图像的存储和处理带来了巨大的困难，必定要耗费相当多的处理时间。可见，随着图像的增大或者是图像分辨率的提高，所需的处理时间也会大幅度的增加。

(3) 不管要对图像进行何种处理，都需要经过图像输入、图像加工、图像输出等几个步骤。数字图像处理研究的主要内容包括图像的获取、图像增强、图像特征提取、图像分析、图像复原等。而针对不同的内容通常又有不同的算法，而不同算法之间的时间复杂度和空间复杂度又是不同的，并且不同的方法适用于不同的图像。

(4) 在进行图像处理的过程中，可以设定或者改变算法的各种参数，可以有效地控制处理过程，达到不同的处理效果。这一特点能突出表现在改善图像质量的处理上。

(5) 图像处理结构的好坏通常是由人进行判断的。因为人的视觉系统很复杂，受环境条件、视觉性能、情绪以及知识结构的影响很大，所以图像处理效果的好

坏受人主观感觉的影响很大。

4.1.2 优化的必要性分析

用本系统进行图像处理实验也存在如下一些问题：

(1) 从以上对图像处理的特点分析中可以看到，一幅图像的信息量是非常大的，而且随着图像大小的增大，所拥有的信息量以及图像的处理时间和占有的内存空间也随之增大，所以图像的大小直接影响到图像处理的速度和空间内存的消耗，而许多图像处理算法都是在基于像素的基础上进行操作的，图像的大小直接影响到图像处理的速度。因此，在进行实验的时候，需考虑怎样节省额外的开销。

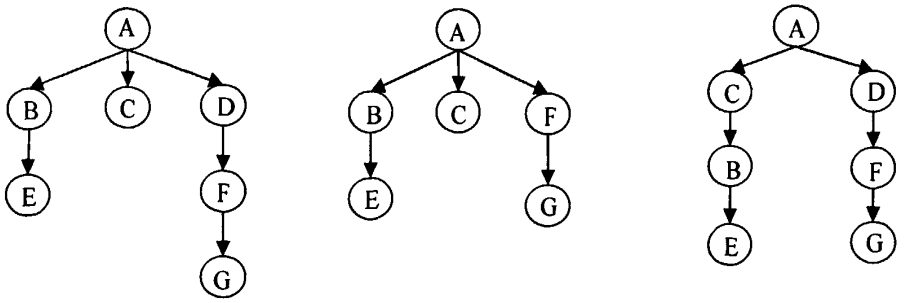
(2) 本实验系统是一个基于图形化组态的系统，组装图结构是否合理会直接影响到图像处理实验的速度，将组装图中不必要的操作删除，同时合并必要的操作，所以怎样使组装图的结构最优也是需要解决的一个问题。

(3) 图像质量的好坏是评价实验效果好坏的一个重要因素，而图像质量的好坏受到多种因素的影响。图像的质量可以由对比度、亮度、饱和度、信噪比等特征的值来评价，不同特征值对图像的评价标准和结果也是不同的，怎样使人们感兴趣的图像特征都达到一个相对的最好值同样也是一个需要研究的问题。

为了解决上述的诸多问题，因此，需要对系统进行优化设计。

4.2 编译过程优化

所谓编译过程的优化是指组装图运行一次以后，如果组装图发生改变，则下次运行组装图的时候从发生改变的地方开始运行，而发生改变的地方以上的那部分图形控件则不需要再次编译运行。组装图的改变主要分为两种情况：1)组装图中图形控件参数的改变；2)组装图结构的改变，其中包括连线的改变、图形控件的添加和删除。如图 4-1，为组装图对应的有向图的变化过程。



(a) 初始有向图 (b) 删除节点后有向图 (c) 删除连线后有向图

图4-1 有向图变化示例

其中，(a)为一个组装图对应的有向图，若改变组装图中节点 D 的参数值，则

只需运行 D、F、G；(b)为(a)删除节点 D 后的组装图，则运行的时候只需要运行节点 F 和 G，若删除的是叶子节点，则不需要在运行组装图中的节点；(c)为(a)中的组装图删除节点 A 到 B 之间的连线，然后再连接 B 和 C 的组装图，在这种情况下只需要运行节点 F 和 G。

组装图执行的时间有两部分组成：一个是组装图中算法运行时间的总和，另一个是解析组装图所用的时间，解析组装图所用的时间包括组装图中图形控件排序所花费的时间，查找前驱图形控件所花费的时间以及查找后继图形控件所需要的时间。为了减少查找前驱图形控件和后继图形控件所用的时间，用户所绘制的组装图可采用邻接表和逆邻接表来保存。

图 4-2 为图像滤波、边缘检测、二值化等一系列操作的组装图，表 4-1 为按照图 4-2 中的组装图分别对大小不同的三幅图像进行操作，并且改变组装图中某些步骤的属性，改变属性前后组装图运行时间的比较。从表 4-1 可以看到，随着图像大小的增大，组装图的处理时间也随之增大。如果改变组装图中某一处理过程的参数值，且该参数值是组装图中第一个运行的处理过程的参数值，则组装图的运行时间不会发生变化。组装图的运行时间随着被改变参数值的处理过程层次的增大而减小。

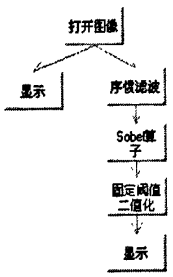


图4-2 组装图示例

表4-1 编译优化前后运行时间比较

图像大小(像素)	原运行时间(s)	改变打开图像运行时间(s)	改变滤波后运行时间(s)	改变二值化后运行时间(s)
110×111	0.7	0.7	0.4	0.1
320 × 240	1.9	1.9	1.3	0.7
800 × 600	4.6	4.6	3.7	2.9

4.3 组装图结构优化

组装图可以看成是静态和动态的统一体，对组装图结构的优化和组装图中相关算法参数的优化属于静态优化，而对组装图的执行过程的优化属于动态优化。为了使组装图更加清晰明了，减少内存的消耗和组装图的运行时间，将利用循环

结构，分叉结构以及并列复合图形控件对用户所绘制的组装图结构进行优化。

4.3.1 利用循环结构进行优化

若组装图中有串联的连续相同的处理操作，且这些操作的输入参数也相同，根据本系统的运行机制，可以将组装图中串联的结果放在一起连续运行，直到结束或者遇到汇聚点，所以系统可以根据组装图的操作序列，自动识别出这些串联的连续相同步骤，并且将这些串联的连续相同步骤用循环结构代替，过程如下：

(1) 将操作序列的第一个节点记为 p，第二个节点记为 q，n 为相同控件的数目，设为 1；

(2) p、q 是否为相同的图形控件，如果相同，则执行(3)，如果不同则执行(2)，若 q 为空则执行(7)；

(3) q 指向它的下一个节点，p、q 是否为相同的图形控件，如果 q 与 p 相同，则执行步骤(3)，n 加 1，如果不同则执行(4)；

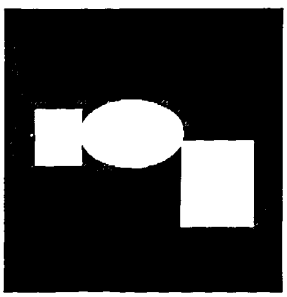
(4) 将 p 的前驱节点记为 s1，q 的后继节点记为 s2；

(5) 添加一个控制循环结构的控制类图形控件，将此控件的循环次数设为 n，并将删除 n-1 个相同的控件；

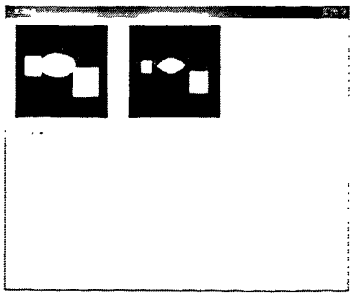
(6) 将添加的控制类控件连接到组装图中；

(7) 结束。

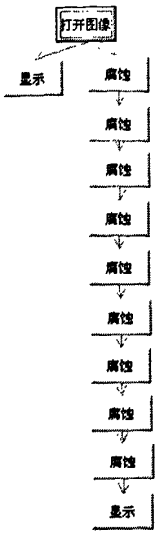
图 4-3 为利用循环结构将连接的三个图形分开的例子。(a)中三个连接图形分开，先采用腐蚀的方法先将它们分离，然后再通过标号化将它们分开，最后再分别对它们进行膨胀操作即可将分开后的图片恢复到原来的大小。图 4-3 为这一过程中的腐蚀阶段，(c)为未经结构优化的组装图，经过实验可得到，需要对原图像进行九次腐蚀后才能将原图中的三个图形分开，得到(b)中右图的结果。由实验可知，需要经过九次腐蚀操作才能将三个图形分开。可以从(c)的处理组装图看到此组装图连续包含了九个“腐蚀”图标控件，这些“腐蚀”图形控件输入的参数一样，在进行结构优化时候，系统可以根据组装图的操作序列识别这一结构，从而将组装图优化成(d)所示的结构，用循环结构进行实现。



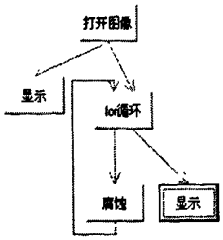
(a) 原图



(b) 处理后结果



(c) 优化前组装图



(d) 优化后组装图

图4-3 利用循环结构进行结构优化示例

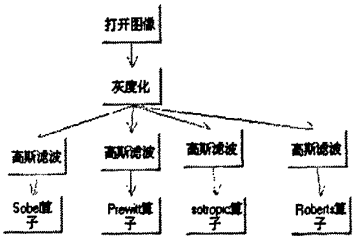
表 4-2 为上述示例利用循环结构进行结构优化前后的组装图所占的内存量。优化后比优化前所占用的内存减少。运用循环结构来进行结构优化，不仅可以避免频繁的删除、增加连线、添加控件等操作，而且不需要重复设置控件的属性。

表4-2 利用循环结构优化前后组装图所占内存

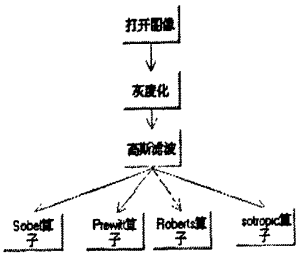
	图形控件数目	图形控件所占内存大小（B）	连线数目	连线所占内存大小（B）	共占内存（B）
优化前	12	384	11	220	604
优化后	5	160	5	55	215

4.3.2 利用分叉结构进行优化

用户绘制的组装图可能存在相同操作具有同一前驱节点的情况，如图 4-4。



(a) 优化前组装图



(b) 优化后组装图

图4-4 利用分叉结构优化前后示例

图 4-4 中，(a)四个完全相同的高斯滤波控件的前驱节点都为灰度化控件，即它们都是对灰度化后图像进行高斯滤波处理，因为他们的输入一样，而且经过的处理步骤一样，所以处理结果必然一样。系统可以通过组装图的逆邻接表自动识

别这种结构，将后面几个高斯滤波控件的后继节点连接到第一个高斯滤波控件的后面，优化后的组装图如(b)。当组装图中出现完全相同操作的前驱节点一样的情况时，系统可以将其用分叉结构进行优化。表 4-3 为图 4-4 所示组装图优化前后组装图所占用的内存和时间的比较。采用大小为 640×480 的图像来测试组装图的运行时间。从表 4-3 可以看出利用分叉结构来进行组装图结构的优化，优化后不仅组装图占用的内存减少了而且组装图的解析运行时间也减少。

表4-3 利用分叉结构优化前后内存和时间比较

	图形控件数目	图形控件所占内存大小（B）	连线数目	连线所占内存大小（B）	共占内存（B）	运行时间（s）
优化前	12	384	11	220	604	9.9
优化后	5	160	5	55	215	5.6

4.3.3 利用并列复合控件进行优化

并列复合控件是复合控件的一种特殊形式，它是将功能相似的算法全部封装到一个图形控件里，用于比较同类算法对图像处理结果的好坏。如图 4-5，模块 1、模块 2、模块 3 组成一个试验组装图，其中模块 2 的图形控件为一个并列复合图形控件，此图形控件中封装了四个算法，算法 A、算法 B、算法 C 和算法 D，算法 A、算法 B、算法 C 和算法 D 之间是一个并行的关系，它们之间不存在相互联系，也不存在先后关系。当运行到模块 2 的时候，将模块 2 图形控件中封装的所有算法运行一次，或者运行复选的算法。

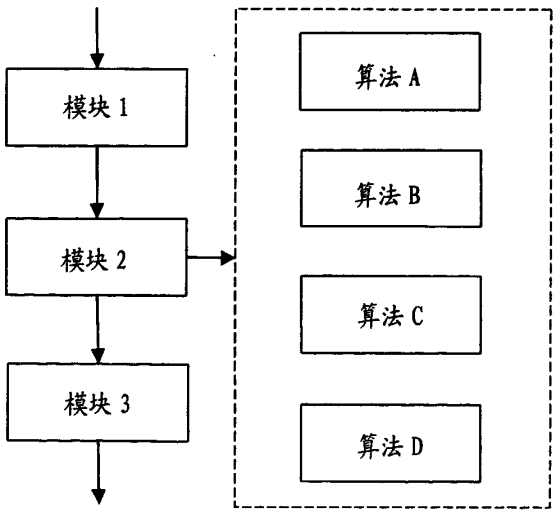
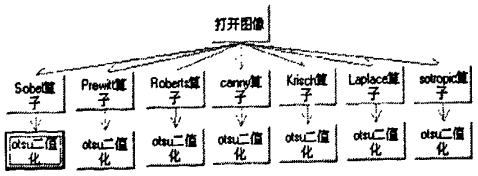


图4-5 包含并列复合图形控件示例

当组装图中有一类功能相似的算法，且它们的前驱节点相同时，可以用它们相应的并列复合控件代替，并且可以利用这种优化方式来优化图像的质量，经过并列复合控件的处理后，总是选择最优的结果进行后续的操作。例如，图 4-6 为

利用并列复合图形控件进行组装图结构优化的示例，(a)中的组装图表示对打开的原图像用不同的边缘检测算子进行边缘检测，然后对边缘检测后的图像进行 otsu 二值化，(b)中的组装图为对(a)中的组装图用并列复合控件优化后的结果。表 4-4 为利用并列复合控件进行组装图结构优化前后组装图所占的内存对比，可以看出，组装图中利用并列复合控件不仅可以使组装图的结构变得简单，而且组装图所消耗的内存数量也可以大大减少。



(a) 优化前组装图

(b) 优化后组装图

图4-6 利用并列复合控件进行结构优化示例

表4-4 利用并列复合控件组装图结构优化前后组装图所占内存

	图形控件数目	图形控件所占内存大小（B）	连线数目	连线所占内存大小（B）	共占内存（B）
优化前	15	480	14	280	760
优化后	3	96	2	40	136

4.4 图像质量优化

图像质量是对图像内容的有用度和图像内容的可辨识程度的满意程度[46]，人们对图像质量的好坏的评价受人的视觉性能、情绪、知识结构等主观因素的影响。人们评价一幅图像质量的好坏通常有两种方法：一种是与标准图像进行对比，如果认为和标准图像相差很小或者几乎没有差别的话，就认为图像的质量好，反之就认为图像质量不好；另一种是在没有标准图像的情况下，如果图像的某些指标达到了预期的期望标准，则认为图像的质量好，反之则认为图像的质量不好。而对图像质量的优化通常是以图像质量的评价为基础的，是在评价的基础上进行的优化。

4.4.1 图像质量评价方法

(1) 图像质量的客观评价

图像质量的客观评价方法是在不考虑人类视觉系统特点的情况下，用客观的数学的计算方法来代替人的观察，通过计算得到图像的某些特征值，并将这些特征值与相应特征的标准值进行比较，得到评价结果。常用的客观评价方法最常用的方法有均方误差 MSE 方法、峰值信噪比 PSNR 方法。

图像均方差 MSE 如式(4-1)，M、N 分别表示图像的高和宽，$f(x,y)$表示待评价的图像，$f'(x,y)$表示标准图像，利用计算得到的 MSE 可以得到图像的信噪比 SNR，如式(4-2)，以及图像的峰值信噪比 $PSNR$，如式(4-3)：

$$MSE=\frac{1}{MN}\sum_{i=1}^{N}\sum_{j=1}^{M}\left(f(i,j)-f'(i,j)\right)^2 \tag{4-1}$$

$$SNR=-\log(MSE) \tag{4-2}$$

$$PSNR=10\times\log_{10}\frac{255\times255}{MSE} \tag{4-3}$$

用 MSE 和 $PSNR$ 来评价图像的质量虽然计算简单，物理意义明确，但是它们把图像看成是一些单一点的结合，而没有考虑图像像素之间的相关性，所以其准确性较差。

除了通过求图像的信噪比来对图像的质量进行评价，还可以根据图像的对比度、清晰度、亮度等特征的值来对图像的质量进行评价。

清晰度[47][48]是指图像的清晰程度，一般来说图像的高低频成分的比例对图像的清晰度有一定的影响，如果高低频成分比例适当的话图像会给人一种清晰的感觉。图像的清晰度可分为绝对清晰度和相对清晰度，绝对清晰度是不利用参考图像的清晰度而求取清晰度的一种方式，相对清晰度是和参考图像的清晰度进行比较，从而得出图像的清晰度的一种方法。本系统主要用基于标准偏差原理和基于熵函数原理两种方法来求取图像的绝对清晰度，而基于熵函数的方法主要通过能量 E 和熵 H 两个标准来评价图像的清晰度。图像能量 E 和熵 H 的定义见式(4-4)和式(4-5)：

$$E=\sum_{x=1}^{w}\sum_{y=1}^{h}f(x,y) \tag{4-4}$$

$$H=\sum_{x=1}^{w}\sum_{y=1}^{h}f(x,y)Inf(x,y) \tag{4-5}$$

根据香农信息理论，熵最大时信息量最多，当 E 一定时，H 越大，图像越清晰。

图像的对比度反映了一幅图像的灰度分布情况，当图像的对比度比较大时，表示图像的灰度层次比较丰富，当图像的对比度较小时，表示图像的灰度层次比较小。本系统采用的对比度评价方法是基于图像的灰度直方图面积的一种方法，取图像灰度直方图面积的 95%处的灰度值和灰度直方图 5%处灰度值的差值作为图像的对比度。对比度的计算公式如式(4-6)，其中，$BrightD$ 表示图像的对比度，A 为图像灰度直方图面积的 5%处的灰度值，B 为图像的灰度直方图面积的 95%处的灰度值。

$$BrightD = B - A \tag{4-6}$$

(2) 图像质量的主观评价

人们对图像质量的主观评价主要采用绝对主观评价方法和相对主观评价方法两种方法。绝对主观评价方法是人们根据设定的评价标准或者是自己的经验，对图像的质量进行评级；相对主观评价方法是人们先对将要评价的图像按照图像质量的好坏先对图像进行分类，然后再对图像质量给定相应的评分。现在，在对图像质量的评价中常常考虑到人的视觉系统的特征，利用人类视觉的特征来进行图像的评价。而目前最常用到的方法是基于 HVS 的评价方法和基于感兴趣部分的评价方法。

因为人的视觉系统具有掩蔽性等特点，所以根据人的视觉系统的特点，人们建立了各种 HVS 模型。图 4-7 是一个典型的 HVS 模型。这个模型通过视觉的非线性、CSF[49]滤波、视觉的多通道等过程来对 HVS 进行建模。

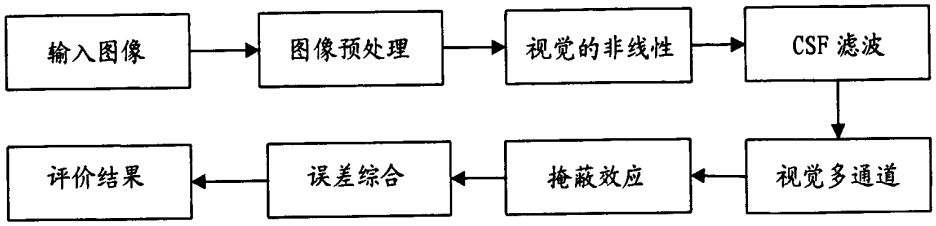


图4-7 HVS 模型

图像预处理：为了模仿人类视觉系统光适应性等光学特征，采用图像匹配、色度空间转换等方法来对输入的图像进行预处理。

视觉的非线性：人眼的感光的自适应性反应了视觉的自适应性，对亮度光强变化的响应是非线性的，可以用上一步空间频率分解的数据来计算局部的对比度，使人眼具有从暗到亮的快速自动调节功能。

CSF 滤波：CSF 称为对比敏感性函数，一般定义为对比度门限的倒数，对比度门限是指人眼所能观察到的最小亮度变化的对比度。因为人眼的自适应能力，所以在人眼可以感知的光强度范围内，对比度门限值几乎为常数。CSF 的计算见式(4-7)：

$$\begin{cases} f = \sqrt{f_x^2 + f_y^2} \\ A(f) = 2.6(0.192 + 0.114f)\exp\left[-(0.114f)^{1.1}\right] \end{cases} \tag{4-7}$$

其中，f 为空间频率，f_x 为水平方向上的空间频率，f_y 为垂直方向上的空间频率。CSF 滤波可以分为三个步骤：首先对图像进行傅里叶变换，求取图像的频谱；其次用 CSF 对图像的频谱做加权处理，将会降低高频分量的幅度；最后对处理后的图像的频谱进行傅里叶逆变换，得到 CSF 滤波后的图像。

视觉多通道：研究表明人眼具有多通道的特性，视觉系统的多条通道之间并不是相互独立的，它们之间相互作用以便产生最佳的视觉效果，而且它们对不同的频率有不同的响应，不具有同向性，在水平方向和水平方向上最敏感，而在对角线方向上最不敏感。正因为人的视觉系统具有多通道性，在进行图像质量评价的时候，都试图模拟人类视觉系统的这种多通道选择的特性，采用某种复杂的方法，比如说小波分解或者是DCT分解将视觉激励分成不同的时空子带。

掩蔽效应：可以反映各种不同激励之间的相互作用关系，即一个本来可见的激励由于另一激励的存在而变得不可见。为了使图像评价结果更有效，在图像质量评价的过程中，常常把原始图像作为背景，而考虑干扰或者噪声被背景掩盖的程度。

误差综合：采用对所有子带Minkowski求和或者是对所有子带误差求平方和的方法来模拟人脑对来自各个视觉通道的神经信号的处理和认知过程，来实现将各空间频带失真综合为单一的指标这一目的，进而得到评价的结果。

人在观察图像的时候常常会把注意力放到自己感兴趣的区域上，图像中感兴趣区域部分图像的质量决定了整幅图像的质量。人们对同一幅图像的理解虽然由于文化背景、情绪或者是其它因素的影响而有所不同，但是对于集中体现了整幅图像所要表达的大部分信息的感兴趣区域来说却具有一定的共同点。文献[50]中提出了一种基于视觉兴趣的图像质量评价方法，通过对图像中不同区域的加权突出人眼对感兴趣区域的兴趣程度，研究发现人眼对感兴趣区域的兴趣程度与其面积成反比。这种方法对图像中只有一个感兴趣区域的情况适用。设图像中感兴趣区域为A_1，其面积为S_1，不感兴趣区域为A_2，其面积为S_2，图像的总面积为S，则$S=S_1+S_2$，测量公式见式(4-8)：

$$\begin{cases} IMSE=\dfrac{1}{S}\left[\lambda_1\sum\limits_{(i,j)\in A_1}\left(f_{ij}f_{ij}^{'}\right)^2+\lambda_2\sum\limits_{(i,j)\in A_2}\left(f_{i,j}f_{ij}^{'}\right)^2\right] \\ \lambda_2=1-\dfrac{2k}{S}\sqrt{S_1\left(S-S_1\right)} \\ \lambda_1=\dfrac{S}{S_1}\left(1-\lambda_2\right)+\lambda_2 \end{cases} \tag{4-8}$$

其中，λ_1、λ_2分别为A_1和A_2的加权值，并满足$\lambda_1S_1+\lambda_2S_2=S$，加权值越大，表示人眼对该区的兴趣程度越大。$k$为调整因子，$k\in[0,1]$，它反映了人眼对不感兴趣区域或者感兴趣区域的重视程度。

4.4.2 基于评价指标的图像质量优化

图像质量的优化是将处理后的结果经过图像质量的评价指标评价，同时给图

像的处理结果一个评分，根据评分和评价后的反馈信息选择最合适的图像处理算法，或者是调整算法的参数，对图像做进一步的处理。基于评价指标的图像质量优化主要是图像经过质量评价指标的评价，会根据评价的结果将处理结果进行排序，系统按结果的排列顺序，选择结果最好的进行下步操作。

图 4-8 为图像质量优化的反馈调节机制示意图，评价所用到的知识库是根据大量的实验所建立的，在进行评价的时候，利用知识库中的信息来对处理结果进行评价。

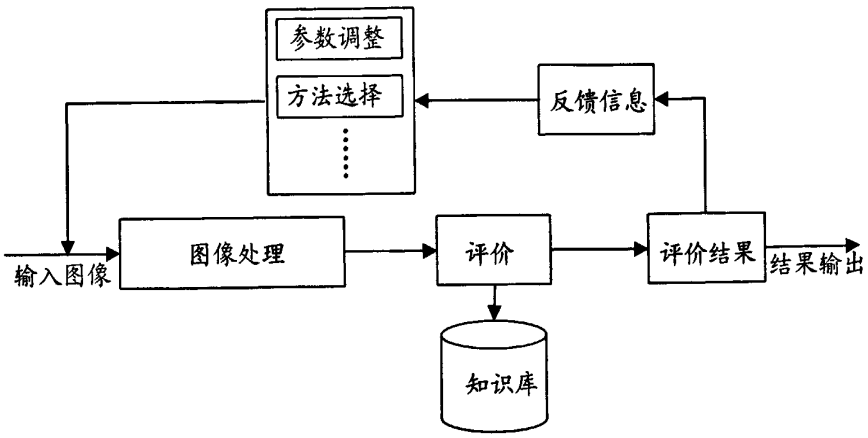


图4-8 图像质量优化过程

用户可以根据评价系统给处理结果评定的等级或者是处理结果的得分情况决定下步的操作。图像质量的优化使用 4.3.3 中介绍的并列复合控件实现，在运行了并列复合控件中所有复选的算法后，根据评价指标的评价结果，自动选择最好的结果继续运行下面的操作，相当于选择一条最优的路径继续运行。下面用带噪声的图像二值化过程来说明图像质量的优化过程。因为不同的滤噪方法适用于含有不同噪声类型的图像，所以对于不同的噪声类型选择的滤噪算法是不一样的，对于有经验的图像处理人员根据经验可能很快能选取合适的滤噪算法，但是对于图像处理初学人员来说，需要在运行不同的滤噪算法对图像进行处理，经过比对之后才能确定所用的算法，这样要么需要反复修改组装图，将不同的滤波方法添加到组装图中去，要么就需要用并列结构的组装图来实现。为了避免繁琐的操作，可以运用并列复合图形控件来对图像进行滤噪操作，然后再对图像进行二值化处理。图 4-9 为带噪声的图像二值化处理的组装图，此组装图的滤波控件并列复合控件。

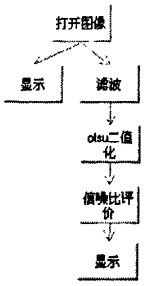


图4-9 图像二值化实验组装图

图 4-10 为两幅具有不同噪声类型的图像在分别经过不同滤波方法后的结果。

(a) 原图 1

(b) 对(a)平均滤波

(c) 对(a)高斯滤波

(d) 对(a)中值滤波

(e) 原图 2

(f) 对(e)平均滤波

(g) 对(e)高斯滤波

(h) 对(e)中值滤波

(i) 标准图

图4-10 带噪声图像滤波结果图

分别计算图 4-10 滤波后的图像的峰值信噪比(PSNR)，相应的值如表 4-5：

表4-5 带噪声图像滤波后峰值信噪比(PSNR)

图像	平均滤波	高斯滤波	中值滤波
原图 1	21.01348	22.36503	19.94809
原图 2	23.38547	23.7170	25.94119

从表 4-5 可以看到，对于原图 1 高斯滤波的 PSNR 最高，平均滤波的 PSNR 次之，中值滤波的 PSNR 最低，说明对于这幅图高斯滤波的效果最好；而对于原图 2，中值滤波的 PSNR 值最高，平均滤波的 PSNR 值次之，高斯滤波的 PSNR 最低，则说明对于这幅图中值滤波的效果最好。所以，对于原图 1 用平均滤波的结果进行二值化，而图像 2 用中值滤波的方法进行二值化，二值化结果如图 4-11 所示。

(a) 高斯滤波结果二值化

(b) 中值滤波结果二值化

图4-11 二值化结果图

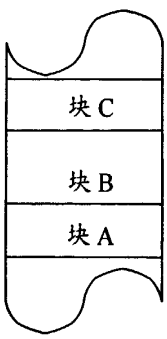
4.5 内存优化

目前，常用的内存分配方法为最先匹配算法[51]和伙伴算法[52]两种。最先匹配算法需要定义两个链表，一个是用来管理内存中空闲块的空闲链表，另一个是用来管理内存中已经分配的内存块的分配链表。当需要分配一块内存的时候，首先搜索空闲链表，直到找到一个满足内存请求的空闲块，然后对两个链表进行更新。相反，当系统释放内存的时候，首先在分配链表中找到将要释放内存块，将其从分配链表中删除，然后更新空闲链表。这种算法的缺点是搜索一个内存块的时间与链表的长度成正比，当一个链表很长时，搜索的时间也会很长。同时，释放一个内存块的时候，要搜索两个链表，所以释放一个内存块的时间比分配一个内存块的时间长。伙伴算法的基本原理是按照 2 的幂次方的大小对内存进行分配的。比如，如果要申请一块大小为 20KB 的内存，按照伙伴算法的基本原理，系统会分配大小为 32KB 的内存空间来满足这个请求，因为内存都是按照 2 的幂次方来分配，32KB 是满足 20KB 要求的最小空间。因为这种算法是按照 2 的幂次方来分配内存的，所以，这种算法的缺点是对内存空间浪费很大，导致后面的内存请求得不到满足，严重的时候会影响到程序的正常运行。假如一共有 1M 的内存空间可以使用，利用伙伴算法对空闲内存进行分配情况如表 4-6，可以看到，内存的利用率只有 50%。

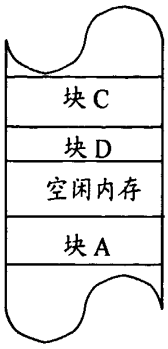
表4-6 用伙伴算法进行内存表

请求	请求空间	分配空间	剩余空间
A	260K	512K	512K
B	260K	512K	0K
C	10K	0K	0K

程序在运行的时候，由于内存管理不当，可能会出现大量的内存碎片，当内存中的内存碎片很多时，程序的运行速度就会减慢，情况严重时程序还会停止运行。所谓内存碎片是指一大片没有使用的内存，即空闲内存，在空闲内存中散布着正在使用的内存。如图 4-12 为出现碎片前和出现内存碎片后的内存的情况。



(a) 出现碎片前的内存



(b) 出现碎片后的内存

图4-12 出现碎片前后的内存

图 4-12(a)中分配了三块连续的内存 A、B、C。在某一时刻，释放内存块 B 后，又有需要申请一块比块 B 小的内存块 D 的请求，(b)为内存块 D 重新分配内存后内存的使用情况，在原来块 B 处空闲的内存空间就越来越小，这样内存块的用处就越来越小了，就会出现很多内存碎片。如果程序分配和释放的内存块的频率很高的话，内存碎片产生的速度也会很快，特别是内存分配和释放时，使用的内存块的大小不同时，这是因为当使用程序的块大小不同时，程序释放的内存块能够被完全应用于新内存请求的机会就越来越小了。所以，如果在运行时动态地为程序分配和释放内存空间，重新释放的内存空间就很难再形成连续的、大块的内存区域。

本系统在绘制组装图时，组装图中图形控件以及图像控件之间连线的生成和保存采用动态分配的方式。动态内存的分配时通过操作系统来完成的，并且用栈结构来保存分配的那部分内存，同时内存的释放也是通过操作系统来完成的。因为频繁地使用操作系统，所以就会使程序的运行速度降低。为了避免在系统的运行过程中形成大量的内存碎片，同时提高系统的运行速度，用一个称之为可利用空间表的表结构来记录空闲内存的路径，当有申请新的内存空间的请求时，就去查找空闲表，将查找到的第一块空闲地址分配给新的请求，同时更新可利用空间

表的信息。图 4-13 为系统的内存分配流程图。

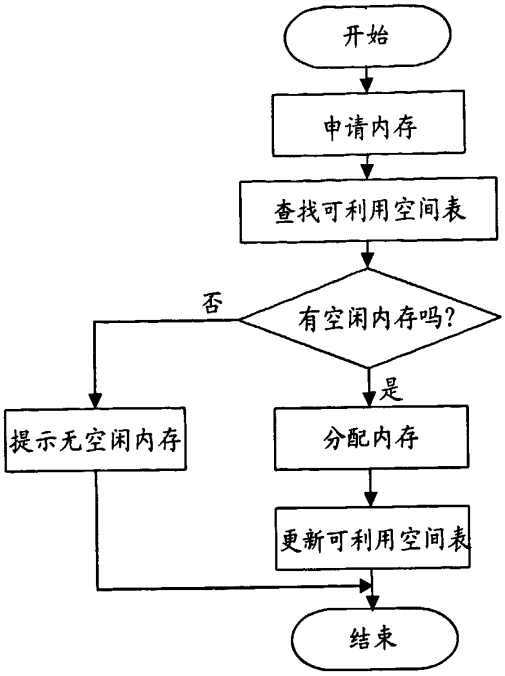


图4-13 系统内存分配流程图

由于图形控件结构和图形间控件连线的结构大小不一样，系统中用到了两个指针用于查询可利用空间表的空闲位置。其中一个用于指向图形控件结构的内存可利用位置，另一个用于指向图形控件间连线结构的内存可利用位置。在进行组装图绘制的时候，当组装图中要添加一个图形控件或者一条连线的时候，就会根据指向图形控件结构的内存可利用位置的指针去查找可利用的空间表，查找到的第一个空闲内存块分配给新添加的图形控件结构，若没有找到空闲内存块则申请一块图形控件结构大小的内存分配给新添加的图形控件结构。同理，用同样的方法给新添加的连线分配一片内存。相反，当要删除图形控件或者图形控件之间连线的时候，将删除的图形控件或者连线的那部分内存空间加到可利用空间表中。

4.6 显示优化

在进行结果显示的时候，系统的显示面板最多能同时显示六幅图像，图像在显示面板上显示的位置是由显示控件在属性面板中设置的显示位置这一属性决定的。如果要显示的图像数目大于六幅，或者两幅或两幅以上的图像被设置在了同一位置则会发生冲突，若发生冲突则按以下原则解除。如果要显示的图像数目大于六幅，那么就用最后一幅将要显示的图像代替第一幅显示的图像，显示在第一幅显示图像的位置；若两幅或者两幅以上的图像被设置在了同一位置，则自动查

找显示面板中空位置，如果找到了空位置，则将后面显示的图像显示在空位置上，如果没有找到空位置则代替第一幅显示的图像显示在第一幅显示图像的位置上。

4.7 小结

本章对组装图的编译过程、组装图结构、图像质量、内存等方面进行了优化。其中，编译过程优化可以减少组装图的运行时间，组装图结构优化不仅可以减少组装图的运行时间也可以减少内存的使用，图像质量优化可以得到较好的处理结果。

第五章 系统实例分析

用户可以根据实验的需求，利用本论文所研究的实验系统，选取合适的图形控件，搭建图形化的组装图，进行实验过程和结果的分析。本章将利用实验系统实现车牌定位这一过程进一步说明系统的运用和所实现的功能。

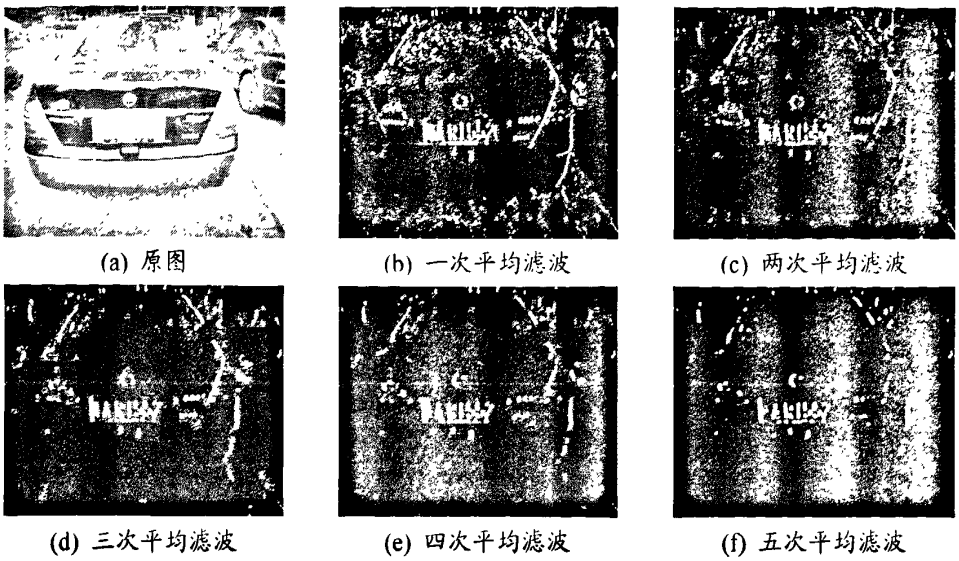
5.1 车牌定位实验组装过程

随着交通业的迅速发展，智能交通系统也在迅速发展，并且已经应用到了交通监控、高速公路管理、小区车辆管理等许多方面。同时，智能交通系统也成为了现在研究的一个热门课题。因为车牌可以对车的身份进行标记，并且它具有唯一性，所以车牌的自动识别是智能交通系统研究中所要研究的一个重要问题。对车牌的识别首先要找到车牌的位置，然后对所找到的区域中的图像进行相应的处理，从而实现车牌的自动识别。下面用实验系统来实现车牌的定位。

(1) 图像预处理与垂直边缘检测

图像的模糊程度会影响图像的处理结果，如果模糊程度太小图像的局部变化会影响实验结果；如果模糊程度太大会将特征点模糊掉，降低实验的准确性。如图 5-1 为图像预处理的组装图，对灰度化后的图像进行平均滤波处理，滤波次数分别取 1、2、3、4、5。因为车牌部分含有大量的垂直边缘，所以对滤波后的图像进行垂直边缘检测。从图 5-2 的处理结果可以看出经过一次滤波和两次滤波后的垂直边缘检测结果含有大量的干扰信息，经过三次、四次、五次滤波后的垂直边缘检测结果比前两幅的干扰点有所减少，但是经过五次滤波后的图像的边缘检测结果车牌部分的边缘信息相对不完整，取四次滤波后的结果进行后面的操作。

图5-1 预处理组装图



(a) 原图 (b) 一次平均滤波 (c) 两次平均滤波

(d) 三次平均滤波 (e) 四次平均滤波 (f) 五次平均滤波

图5-2 滤波后垂直边缘检测

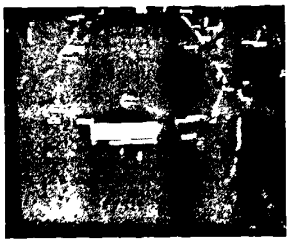
(2) 桥接

从图 5-2 的(e)可以看到，结果中仍然含有一些干扰信息，为了消除干扰信息，可以对垂直边缘检测后的图像设定一定的阈值进行水平桥接和垂直桥接。采用水平桥接和垂直桥接可以生成连通域。同时，在进行完水平桥接后消除水平桥接后水平方向上较短的线，同理，进行完垂直桥接后消除垂直方向上较短的线。图 5-3 为桥接的组装图。

图5-3 桥接组装图

桥接阈值的选取很重要，如果阈值选取太小待检测的车牌区域可能不能连接在一起，如果阈值选取过大则会将干扰信息连接在一起，对检测结果造成影响。图 5-4 为对图 5-2 中(e)进行桥接的结果，其中(a)为取阈值 37 的水平桥接结果，(b)为消除水平桥接的结果中水平长度小于 20 的短线后的结果，(c)为取阈值 20 对(b)

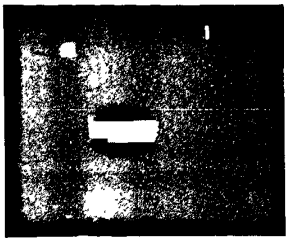
进行垂直桥接，(d)为消除垂直桥接结果中垂直长度小于 18 的直线后的结果。



(a) 水平桥接（阈值=37）

(b) 消除水平短线（阈值=20）

(c) 垂直桥接（阈值=20）



(d) 消除垂直短线（阈值=18）

图5-4 桥接及消除短线

经过桥接等一系列的操作后，图像上形成了几个大小不等的联通区域，接下来可以对联通区域进行提取，分析每个连通域的特征，提取出可能会是车牌区域的连通域。

(3) 联通区域的分析及车牌的粗定位

对图像进行桥接处理以后会得到一些大小不等的相互连通的区域，将这些区域进行标号化，同时记录每个区域的面积、最左端的坐标、最右端的坐标、最上端坐标、最下端坐标以及连通域的标号。然后根据连通区域的几何特征进行筛选，选出复合车牌几何特征的联通区域。车牌区域的特点包括：车牌的长度和宽度的比值大于 3；区域的填充度应该大于一定的阈值，将此阈值设定为 0.7；车牌区域的面积应该大于一定的阈值，这一阈值与摄像头拍摄的位置与车辆之间的距离有关，当它们之间的距离较小时，车牌区域的面积比较大，相反，摄像头的位置与车辆之间的距离较大时，车牌区域的面积比较小，设定车牌区域面积的阈值为 4500。

图 5-6 中(a)为对进行桥接后的图像进行连通域的求取，(b)为连通域的分析，求取满足车牌区域特征的区域，(c)为根据所求取的满足车牌区域特点的连通域的相关信息在原图上对车牌进行定位。经过“求连通域”图形控件后，将经过水平桥接和垂直桥接后的结果图中的连通域求取出来。而在求得的这些连通域中，有的区域是不符合车牌区域所具有的几何特征的，需要经过“求符合条件连通域”图形控件的操作，将不符合条件的连通域排除。最后，根据求得的满足条件的连通域的位置可以在原图中得到含有车牌的子图像。

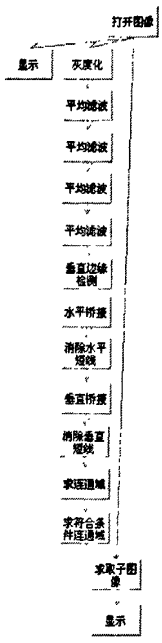
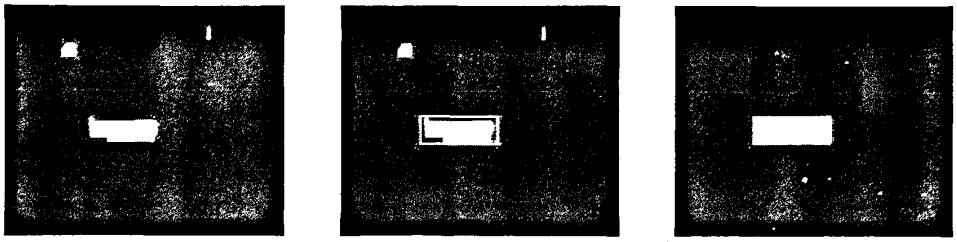


图5-5 车牌定位组装图



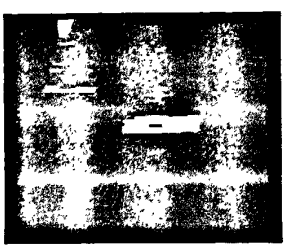
(a) 求连通域 (b) 求符合条件连通域 (c) 车牌子图像

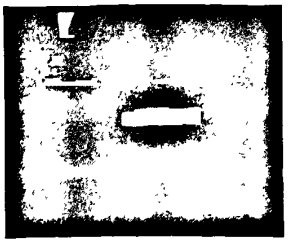
图5-6 连通域求取及车牌初定位

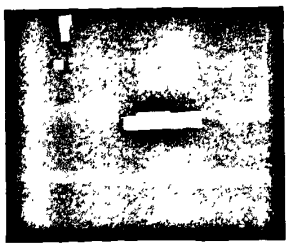
5.2 车牌定位组装图应用

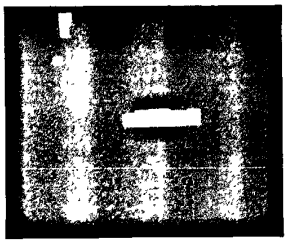
可以用车牌定位实验组装图来分析同类问题。图 5-7 为用图 5-5 的组装图对两幅不同的图像进行处理后的结果。图 5-7 中的(a)是一幅背景比较单一的图，(j)中的背景比较复杂，但是背景的垂直边缘对处理结果的影响不大，两幅图像经过图 5-5 的组装图处理后可以初步确定车牌的区域。

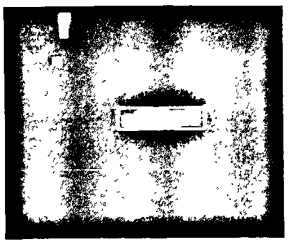
(a) 原图 1 (b) 垂直边缘检测 (c) 水平桥接（阈值=37）

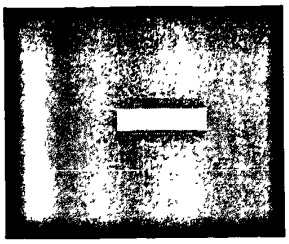

(d) 消除水平短线（阈值=20）


(e) 垂直桥接（阈值=20）


(f) 消除垂直短线（阈值=18）

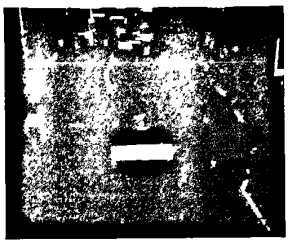

(g) 求连通域

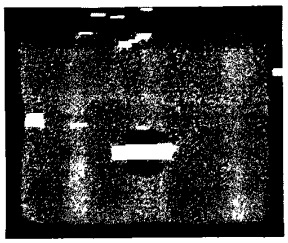

(h) 求符合条件连通域

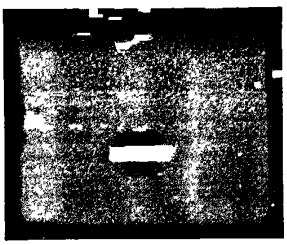

(i) 车牌子图像

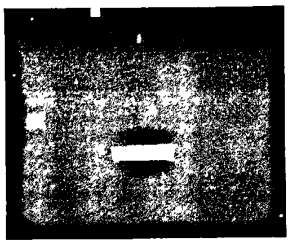

(j) 原图 2

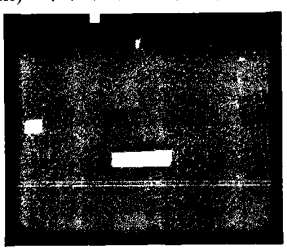

(k) 垂直边缘检测

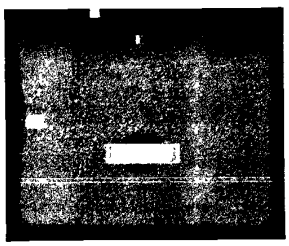

(l) 水平桥接（阈值=37）


(m) 消除水平短线（阈值=20）


(n) 垂直桥接（阈值=20）


(o) 消除垂直短线（阈值=18）


(p) 求连通域


(q) 求符合条件连通域

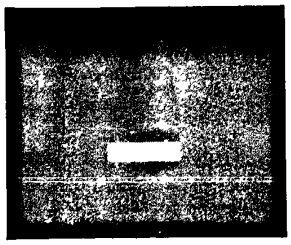
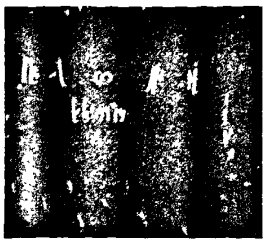
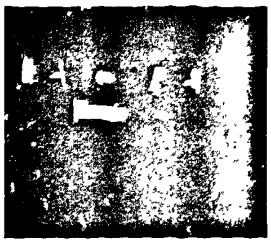

(r) 车牌子图像

图5-7 车牌定位实验过程

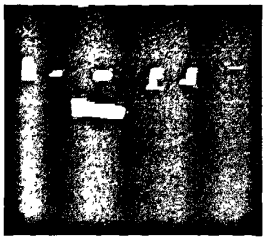
如图 5-8 为用图 5-5 所示的组装图对车牌倾斜情况下的图片进行处理的结果图。从处理结果可以看出，因为原图像的车牌倾斜，造成车牌的长宽比变小，长宽比为 2.9，不满足车牌联通区域连通域的长宽比大于 3 这一条件，所以用图 5-5 的组装图不能检测出车牌。

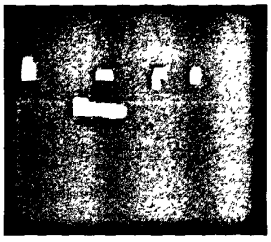

(a) 原图

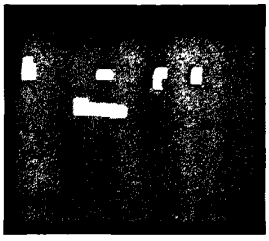

(b) 垂直边缘检测

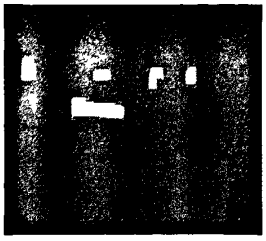

(c) 水平桥接（阈值=37）


(d) 消除水平短线（阈值=20）


(e) 垂直桥接（阈值=20）


(f) 消除垂直短线（阈值=18）


(g) 求连通域


(h) 求符合条件连通域

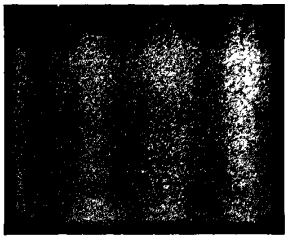
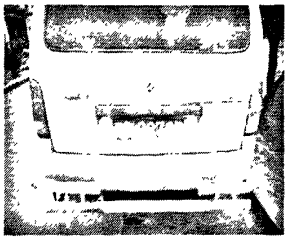
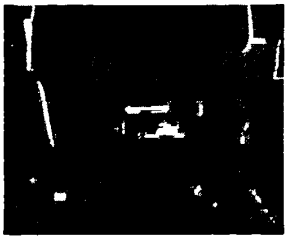

(i) 车牌定位

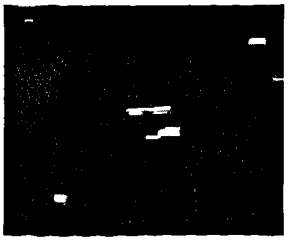
图5-8 倾斜车牌定位实验过程

如图 5-9 为用图 5-5 所示的组装图进行处理的结果，由于在进行图像预处理的时候对图像进行了 4 次平均滤波处理，由于滤波次数太多，造成图像过度模糊，经过垂直边缘检测后车牌部分的一些边缘信息缺失，进而定位不到车牌。

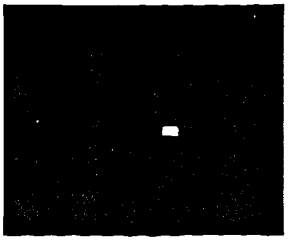

(a) 原图

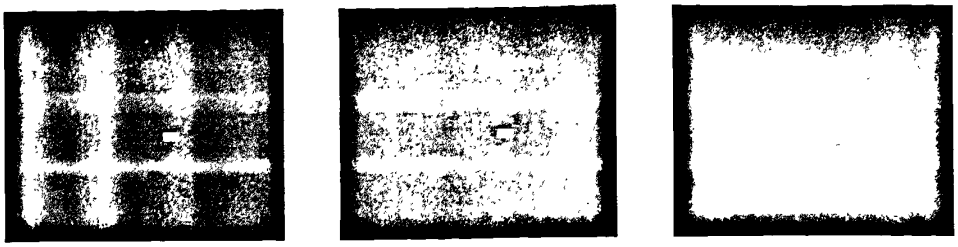

(b) 垂直边缘检测


(c) 水平桥接（阈值=37）


(d) 消除水平短线（阈值=20）


(e) 垂直桥接（阈值=20）

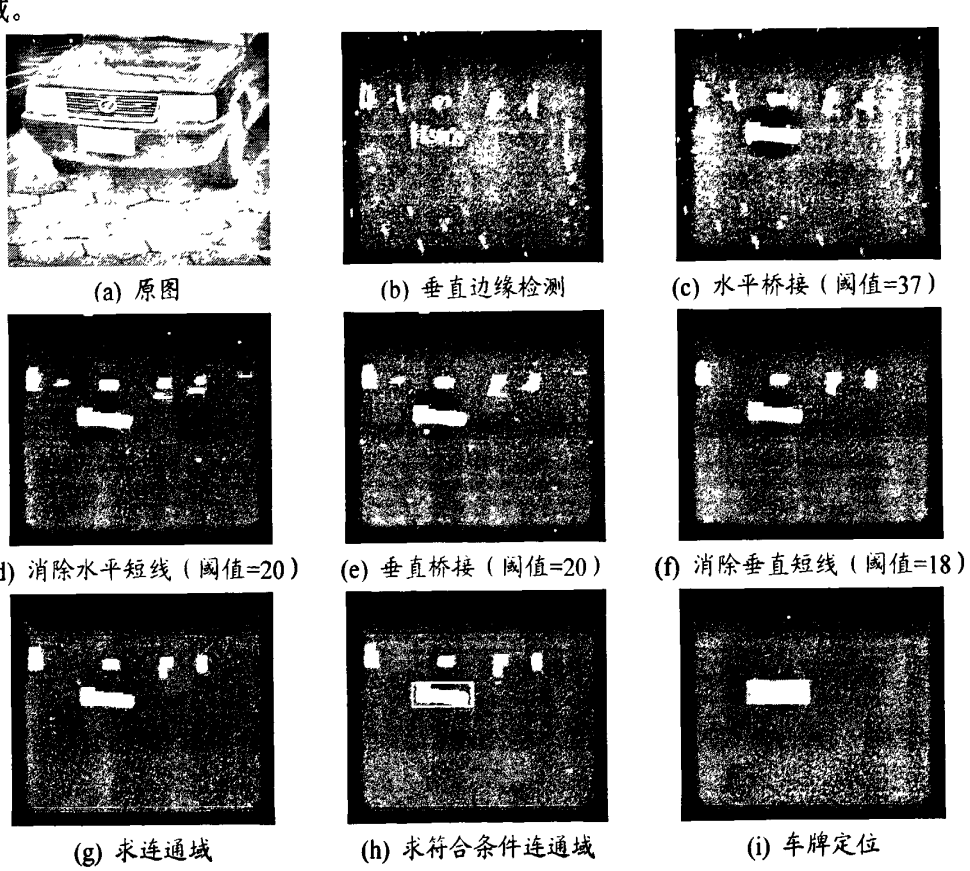

(f) 消除垂直短线（阈值=18）



(g) 求连通域 (h) 求符合条件连通域 (i) 车牌定位

图5-9 滤波次数过多车牌定位实验过程

图 5-10 为将图 5-5 组装图中将“求符合条件连通域”图像控件的连通域的长宽比的输入参数修改为 2.7 后的处理结果，经过修改后可以定位到车牌所在的区域。



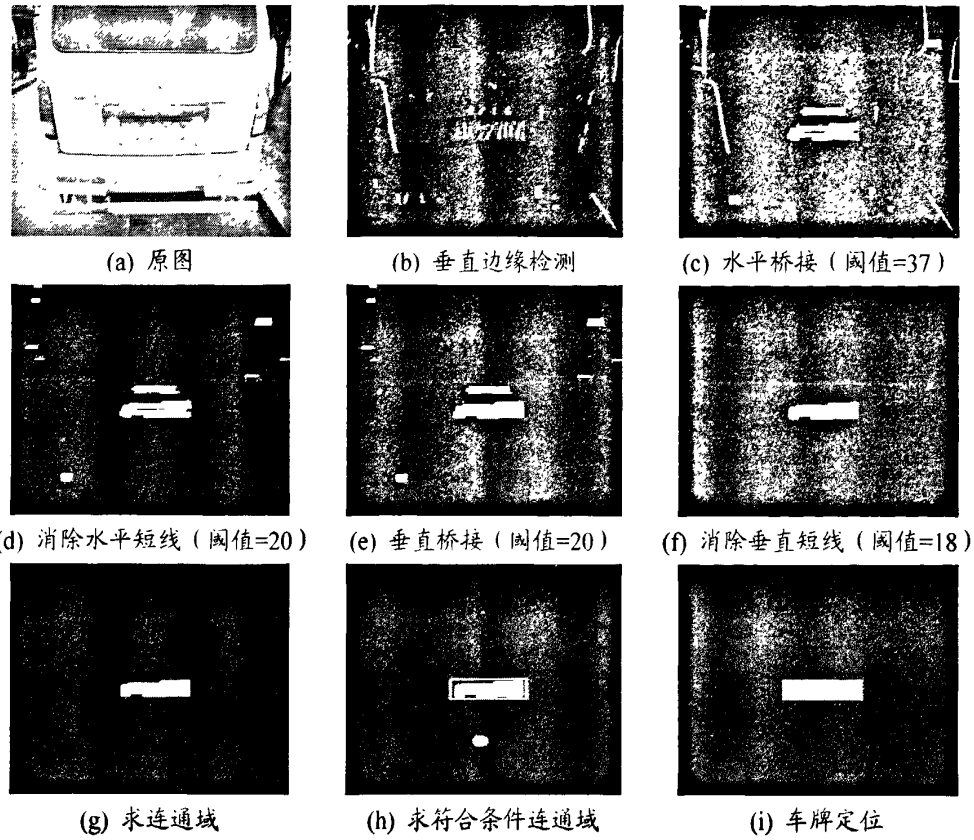
(a) 原图 (b) 垂直边缘检测 (c) 水平桥接（阈值=37）

(d) 消除水平短线（阈值=20） (e) 垂直桥接（阈值=20） (f) 消除垂直短线（阈值=18）

(g) 求连通域 (h) 求符合条件连通域 (i) 车牌定位

图5-10 调整车牌宽高比后车牌定位实验过程

图 5-11 为将图 5-5 中的组装图中对图像进行平均滤波处理的循环结构中的循环次数设置为 2，即对灰度化处理以后的图像进行两次平均滤波后再进行垂直边缘检测。从(b)可以看出，经过垂直边缘检测后车牌部分含有大量的垂直边缘信息，可以在此基础上进行水平桥接和垂直桥接等一些操作，初步确定车牌所在的位置。



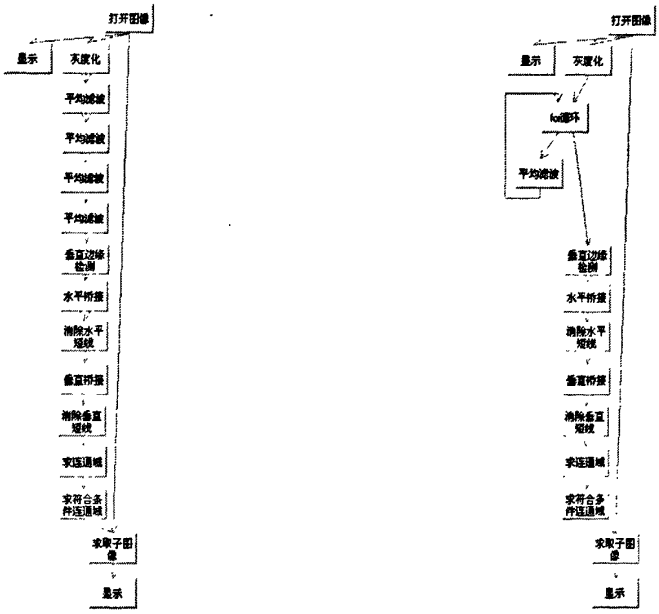
(a) 原图 (b) 垂直边缘检测 (c) 水平桥接（阈值=37）

(d) 消除水平短线（阈值=20） (e) 垂直桥接（阈值=20） (f) 消除垂直短线（阈值=18）

(g) 求连通域 (h) 求符合条件连通域 (i) 车牌定位

图5-11 调整滤波次数后车牌定位实验过程

四次平均滤波用循环结构实现组装图实现，优化前后的组装图如图 5-12。



(a) 优化前组装图 (b) 优化后组装图

图5-12 优化前后车牌定位实验组装图

从上述的实验过程可以看出，对于不同的图像所需的滤波次数是不同的，用循环结构来代替组装图中的滤波过程可以避免因为滤波次数的不同，频繁地改变组装图的结构。表 5-1 为结构优化前后组装图所占的内存大小，从表中的数据可以看出，随着循环次数的增加，优化后组装图所占用的内存数目远远小于优化前组装图所占用的内存数目。

表5-1 优化前后车牌定位实验组装图所占内存大小

	图形控件数目	图形控件所占内存大小（B）	连线数目	连线所占内存大小（B）	共占内存（B）
优化前（循环次数=1）	13	416	13	260	676
优化后（循环次数=1）	14	448	15	300	748
优化前（循环次数=2）	14	448	14	280	728
优化后（循环次数=2）	14	448	15	300	748
优化前（循环次数=3）	15	480	15	300	780
优化后（循环次数=3）	14	448	15	300	748
优化前（循环次数=4）	16	512	16	320	832
优化后（循环次数=4）	14	448	15	300	748
优化前（循环次数=5）	17	544	17	340	884
优化后（循环次数=5）	14	448	15	300	748

由上述的实验分析过程可以看出，在用实验系统进行实验的过程中组装图中为了得到较好的处理结果，图形控件的输入参数可以根据需要进行调整。进行输入参数调整后的组装图再次运行的时候，从给参数进行了调整的图形控件开始运行，而不需要运行整个组装图。表 5-2 为编译优化前后的运行时间表。

表5-2 编译优化前后车牌定位实验组装图运行时间

图像大小（像素）	共运行时间（s）	改变水平桥接参数后运行时间（s）	改变垂直桥接参数后运行时间（s）
720 × 540	16.6	6.3	3.6
768 × 576	19.8	8.9	5.2
800 × 600	22.5	9.7	6.1

5.3 小结

本章主要以车牌的初定位实验在系统中的实现过程为例子，分析了实验在系统上的实现过程，以及在本例的分析中所采用的优化技术，并且对优化前后进行了对比。

第六章 总结与展望

6.1 工作总结

随着计算机技术、信号处理技术以及成像技术的发展，数字图像处理技术运用到了工业、生活、安全、军事等诸多方面，在科技研究中起着越来越重要的作用。本文研究了一种基于图形化组态的数字图像处理系统的设计过程和设计方案，为了进一步提高图像处理的速度，方便用户利用此平台来进行图像处理实验，本文还从组装图的结构、图像的质量等方面研究了系统的优化过程，并且通过实例介绍了系统的使用过程。

论文所做的工作主要有以下几点：

(1) 在分析了基于图形化组态的软件设计思想以及这种设计思想特点的基础上，进行了图像处理实验系统的需求分析，确定了系统的功能需求和其它方面的需求，并且根据系统的需求分析，对系统进行了总体设计。

(2) 对系统进行详细设计。其中包括：分析了组装图的主要组成元素图形控件以及图形控件间连线的结构设计，并且介绍了组装图的绘制方法；分析了系统主要模块的设计方法；分析了顺序结构组装图、分支结构组装图、循环结构组装图以及子程序结构组装图四种组装图结构的特点，并且提出了这四种组装图的设计方法；在分析图形化组态软件常用的运行机制的基础上，提出了本系统对组装图解析机制。

(3) 从组装图编译时间、组装图结构、图像质量、内存管理、显示等方面对系统进行优化，并且通过实例对优化前后进行了对比，证明了经过优化后，可以减少组装图的运行时间和内存的使用。

6.2 工作展望

本文对基于图形化组态的图像处理实验系统的设计以及系统的优化进行了研究，并且用实例证明了系统的实用性。然而还是存在一些问题需要继续深入研究，这主要包括：

(1) 在进行优化设计的时候，通过大量的图像处理实验，形成一个组装图算法选择和算法组合的知识库。用户只需要输入处理结果所要达到的指标，系统可以根据用户的需求，自动选取合适的算法组成合理的组装图进行实验。

(2) 将此系统做成基于浏览器的形式，可以通过网页直接打开，用户不需要

下载此软件安装，可以在网上直接打开使用。

(3) 开放控件中封装函数的源代码，可以在线修改编译。

参考文献

[1] 冈萨雷斯. 数字图像处理(第二版) [M]. 北京: 电子工业出版社, 2002.

[2] Guoxia Yu, Tanya Vladimirova, Martin N Sweeting. Image compression systems on board satellites. Acta Astronautica, 2009, 64: 988~1005.

[3] Samuel W K Chan, K S Leung, W S Felix Wong. An expert system for the detection of cervical cancer cells using knowledge-based image analyzer. Artificial Intelligence in Medicine, 1996, 8(1): 67~90.

[4] 王建新, 张丽媛, 盛羽, 等. 基于组件的计算机组成原理虚拟实验室的设计与实现. 系统仿真学报, 2008, 20(9): 2469~2474.

[5] Barber Jack. Labview an implementation of data flow programming in a graphical language. Advances in Instrumentation, 1989, 44(3): 1259~1266.

[6] Jamal Rahman, Wenzel Lothar. Applicability of the visual programming language Labview to large real-world applications. IEEE, 1995, 9: 99~106.

[7] N Kehtarnavaz, C Gope. Dsp System Design Using Labview and Simulink: A Comparative Evaluation. IEEE , 2006, 6 : 985~988 .

[8] Baroth Ed, Hartsouqh Chris, Holst Amy, etal. Evaluation of Labview 5.0 and HP VEE 5.0 –part 2. Evaluation Engineering, 1999, 38(5): 5~6.

[9] Hewlett-Packard Company. HP VEE Advanced Programming Techniques.

[10]Hanjun Jin, Shidong Mei, Lei Chen. The application of Authorware education. IEEE, 2008, 6(3): 1174~1178.

[11]LaFon Ren. Microsoft Office Visio 2003. Cadalyst, 2004, 21(7) : 36.

[12]D Paulus, T Dickscheid, K D Berg. Design of an Image Analysis System. IEEE, 2005, 9: 135~141.

[13]Chang-le Li, Ji-zhuang Fan, Jie Zhao. Research on the reconfigurable image processing system. IEEE, 2008,8: 284~288.

[14]陈志华, 张洪涛, 陈坤. 基于TIDSP的红外图像采集预处理系统的软硬件实现. 红外, 2006, 27(7) : 16~19.

[15]苏光大. 微机图像处理系统. 北京: 清华大学出版社, 2000.

[16]Zhao Yao-Hong, Xiang Wei, Luo Hai-Bo, etal. Application of SOPC technology in real-time infrared image processing. Infrared and Laser Engineering, 2005, 34(6) : 747~751.

[17]Almudena Lindoso, Luis Entrena, Juan Izquierdo, etal. Coarse-grain dynamically reconfigurable coprocessor for image processing in SOPC. IEEE, 2008: 539~542.

[18]Czarnut P, Ciereszko A, Frzak M. Towards efficient parallet image processing on cluster grids using GIMP. Computational Science, 2004, 2:451~458.

[19]邝国枝, 卞静. 医学图像处理系统的设计与实现. 现代计算机, 2005, 5 : 56~59.

[20]鲁剑峰. 基于 DSP 的实时红外图像处理系统的设计. 红外与激光工程, 2008, 37 : 622~625.

[21]Konstantions Konstantinides, John R Rasure. The Khoros Software Development Environment for Image and Signal Processing. IEEE, 1994, 3(3): 243~252.

[22]金维香. 图形化程序设计 G 语言—Labview 与虚拟仪器. 长沙电力学院学报(自然科学版), 2002, 17(1) : 14~17.

[23]Kodosky J, MacCrisken J, Rymar G. Visual Programming Using Structured Data Flow. IEEE, 1991, 9(3): 34~39.

[24]周泓, 徐小良, 汪乐宇. VPP 虚拟仪器元件库的实现架构. 工程设计学报, 2003, 10(2): 75~79.

[25]Schreier. Softwire Capitalizes on Graphical/Textual language. EE: Evaluation Engineering, 2000, 39(2): 1~2.

[26]Lecklider, T. Get Connected with Softwire. EE: Evaluation Engineering, 2001, 40(4) : 36~45.

[27]Dieter Muller, J M F. Online labs and the Marvel. experience International Journal on Online Engineering, 2005: 1~4.

[28]Kovac, V S K. Virtual Instrumentation and Distributed Measurement Systems. Journal of Electrical Engineering, 2004, 55(1) : 50~56.

[29]Mir, M, M A Al-Saleh. A digital simulator for determining the performance limits of computer relays. IEEE Transactions on Power Delivery, 2002, 17(1): 60~67.

[30]Al-Dhaher, A H G. Applied virtual instrumentation Instrumention. IEEE, 2001,4(1) : 59~59.

[31]耿晨歌. 面向虚拟仪器的可视化编程语言研究: [博士学位论文]. 杭州: 浙江大学, 1999.

[32]徐小良, 刘阳, 周泓, 等. 图形化编程平台的结构设计及实现. 计算机工程与应用, 2001, 4: 4~6.

[33]Goldberg, H. What is Virtual Instrumentation. IEEE, 2000, 3(4): 10~13.

[34]Spanias Andreas, Atti Venkatraman. Interactive online undergraduate laboratories

using J-DSP[J]. IEEE Transaction on Education, 2005, 48(4): 735~749.

[35]Josep Prieto, Joan Arnedo, Jordi Herrera. An Integrated Structure for a Virtual Networking Laboratory. IEEE, 2008, 55(6): 2334~2342.

[36]Enrique Mandado, Jacinto G Dacosta. Virtual Electronics Laboratory: A new tool to improve Industrial Electronics Learning. IEEE, 2004, 6: 644~649.

[37]王建新, 莫秋菊. 基于 Internet 的通信系统虚拟实验室环境设计与实现[J]. 中南大学学报(自然科学版), 2006, 37(2): 330~353.

[38]何伟, 李徽, 张玲. 基于计算机图像处理的电路印刷板缺陷监测. 计算机测量与控制, 2007, 15(10): 1295~1297.

[39]陈若珠, 于小宁, 李战明. 基于 DSP 的啤酒瓶缺陷识别系统的研究. 微计算机信息, 2007, 23(4): 189~191.

[40]Aarabi P, Lam J C L, Keshavarz A. Face detection using information fusion. IEEE, 2007, 11(4) : 1~8.

[41]马国华. 监控组态软件及其应用. 北京: 清华大学出版社, 2001.

[42]杨亚罗, 王润孝, 库祥臣, 等. 组态概念发展的新趋势. 计算机应用研究, 2006, 9: 13~16.

[43]朱洪波. Visual C++ 6.0 完全自学宝典. 北京: 清华大学出版社, 2008.

[44]王建新, 陆炜妮, 王伟平. 基于组建的数字图像处理仿真系统的设计与实现. 系统仿真学报, 2004, 16(6):1213~1216.

[45]徐小良, 周鸿, 刘阳, 等. 图形化编程平台运行算法的研究. 计算机应用研究, 2001, 10: 33~34.

[46]H R Sheikh, A C Bovik. Image Information and Visual Quality. IEEE Transactions on Image Processing, 2006, 36(15) : 430~444.

[47]王鸿南, 钟文, 汪静, 等. 图像清晰度评价方法研究. 中国图像图形学报, 2004, 9(7): 828~831.

[48]Sheikh H R, Bovik A C. Image Information and Visual Quality. IEEE Transactions on Image Processing, 2006, 15: 430~444.

[49]Thomas, C W, Gilmore, G C, etal. Models of contrast sensitivity in human vision. IEEE Transaction on Systems, 1993, 23(3) : 857~864.

[50]汪孔桥, 沈兰荪, 邢昕. 一种基于视觉兴趣性的图像质量评价方法. 中国图像图形学报, 2000, 5(4): 300~303.

[51]倪西钧, 汤可夫, 吴大为. 一个应用于动态内存管理算法中的数据结构[J]. 兰州理工大学学报, 2004, 30(6): 90~91.

[52]董庆丰, 黄迪明. 一种适用于嵌入式系统的动态内存管理技术[J]. 微型机遇

应用, 2004, 8: 53~54.

[53]L Benetazzo, M Bertocco, F Ferraris. A Web-Based Distributed Virtual Educational Laboratory. IEEE, 2000, 49(2): 349~356.

[54]王瑞荣, 汪乐宇. 面向图形化编程的事件触发并发数据流模型. 浙江大学学报, 2002, 36(5): 535~539.

[55]T B Brown, T D Kimura. Completeness of a Visual Computation Model[J]. Software Concepts and Tools, 1994: 34~48.

[56]Whiting P G, Pascoe R S V. A history of data-flow language[J]. IEEE, 1994,16(4): 38~59.

[57]殷飞, 丁维明. 组态软件设计中的模式研究. 计算机测量与控制, 2005, 13(3) : 298~300.

[58]何东健. 数字图像处理. 西安: 西安电子科技大学出版社, 2003.

[59]王建新, 裴惠民, 陈松乔. 基于 Internet 的虚拟实验室平台架构设计. 中南工业大学学报, 2002, 33(5): 530~533.

[60]王忠诚, 程福, 马英庆. 机器人足球决策程序的开放式图形化编程平台.计算机系统应用, 2007, 11:79~82.

[61]Oliver L Wang, J Huang. Developing a Visual Component Library for a Graphical Programming Platform using Object Orientation. IEEE, 2001: 672~678.

[62]Prashant Waknis, Gabor Karsai, Janos Sztipanovits. A graphical programming environment for simulation of control and signal processing systems. IEEE. 1992: 447~450.

[63]蔡悦华, 魏承辉, 罗颂荣. 常用优化设计可视化软件系统开发. 机床与液压, 2005, 2: 160~162.

[64]李玮, 陈炜, 朱博勤. 基于优化理论的图像处理与分析研究. 计算机工程与应用, 2005, 14: 23~27.

[65]Güld Mark O, Thies Christian. A platform for distributed image processing and image retrieval. Proceedings of SPIE, 2003, 5150: 1109~1120.

[66]汤晓安. 一种对虚拟仪器软件 Labview 进行功能扩展的方法. 计算机自动测量与控制, 2000, 8(6): 64~66.

[67]杨亚罗, 王润孝, 库祥臣, 等. 组态概念发展的新趋势. 计算机应用研究, 2006, 9: 13~17.

致谢

转眼间三年的研究生学习生涯就要结束了，在这三年里，老师的恩情、同学之间的友谊、良好的学习环境给我留下了深刻的印象。三年来，也有很多人走进了我的生活，他们关心我、帮助我，在此我要向他们表示最真挚的祝福和深深地谢意。

衷心感谢我的导师罗三定教授，在学习上罗老师给了我许多悉心的指导，在毕业论文的进行阶段给我提出了许多宝贵的指导意见和建议，这些都使我受益匪浅。罗老师渊博的学识、严谨的治学态度、对新技术敏锐的洞察力、兢兢业业的探索钻研精神给我留下了深刻的印象，我也从导师身上学到了许多难以学到的东西。

感谢我的第二位仁师，沙莎教授。她不仅在生活中给了我无微不至的关怀，而且在我撰写论文的过程中给了我很多中肯的建议。

在此，我谨向我的恩师们表示衷心的感谢！谢谢你们对我的指导和帮助！

感谢赵燕、吕敏、朱海军、常卢峰、鲁晋、秦岭、方晓敏等同学在学习和生活中对我的帮助，感谢实验室的师兄师姐、师弟师妹以及各位同门在论文进行前期对我的帮助，很幸运能和他们相识。感谢我的父母和家人，一直关心我、鼓励我，感谢他们为我所做的一切，在我即将毕业之际，我想对他们说一声，你们辛苦了，谢谢你们！

三年的学习生活，我身边需要感谢的人还有很多，再次对帮助过我的人说一声谢谢你们！

陈江婷于长沙

2010年4月

攻读学位期间主要的研究成果

论文发表：

[1] 陈江婷. 图形化组态的数字图像处理实验系统的设计. 科技广场. 已录用.