华中科技大学文华学院

毕业设计[论文]

题目：指纹识别技术

学 生 姓 名：_____ 学号：_____

学 部 （系）：_____

专 业 年 级：_____

指 导 教 师：_____ 职称或学位：_____

2011 年 5 月 15 日

目 录

指纹识别技术

摘　要

随着现代社会文明的进步，计算机和网络的迅速发展，人们对身份认证的准确性、安全性与实用性提出了更高的要求。基于生物特征识别的智能身份认证技术也逐渐受到广泛的关注。在众多的生物识别技术中，指纹识别技术是发展最早、应用最广泛的一种。指纹识别技术充分利用了指纹的普遍性、唯一性和永久性的生物特征，已逐步取代了传统的基于标志和数字的识别方式，目前在网络、银行、金融、医疗和安检等行业均得到了广泛应用。本文对指纹识别系统的原理和基本过程进行了分析研究，重点研究指纹识别的具体实现方法及其中的算法。

整个软件的编译在 Matla 环境中进行，在对识别的研究过程中，分为五个部分，分别为图像的选取，图像的观察，图像的存储，图像的识别和图像的清除

在指纹图像预处理部分，论文对预处理的各个步骤包括规格化、图像分割、中值滤波、二值化、细化等以及各个步骤的方法进行了深入的分析和研究，选择了一种图像预处理方案。

在指纹特征提取部分，采用基于 Matlab 实现的指纹细节特征提取方法，并给出了去伪算法。指纹特征提取是从细化后的指纹图中得到细节特征点（即端点和分叉点），此特征点含有大量的伪特征，既耗时又影响匹配精度。采用了边缘去伪和距离去伪，使得特征点去伪前后减小了近 1/3，然后提取可靠特征点信息，以便实现指纹匹配。

在指纹匹配部分，本文采用 Gabor 滤波器的指纹匹配算法，并进行研究。

关键词：指纹识别；特征提取； Gabor 滤波器

Fingerprint identification

Abstract

With the progress of modern civilization, the rapid development of computers and networks, the accuracy of people's identity, safety and practicality of a higher requirement. Biometrics-based authentication technology is also smart gradually been widespread concern. Among the many biometric technologies, fingerprint recognition technology is the development of the earliest and most widely used one. Fingerprint recognition technology takes full advantage of the universality of the fingerprint, unique and permanent biological characteristics, has gradually replaced the traditional signs and figures based on means of identification, at present in the network, banking, finance, medical and security industries have been a wide range of applications. In this paper, the principle of fingerprint identification system and the basic process was analyzed, focusing on fingerprint identification method and its concrete realization of the algorithm.

Compile the entire software environment in matla conducted research in the identification process, is divided into five parts, namely the selection of images, image observation, image storage, image recognition and image removal

Part of the fingerprint feature extraction, fingerprint-based Matlab implementation details of feature extraction method, and gives the false and algorithms. Fingerprint feature extraction is refined to get after the fingerprint minutiae (ie, endpoint and bifurcation points), the feature point contains a lot of pseudo-features, time-consuming and affect the matching accuracy. Used and the distance to the edge of the false and counterfeit, makes the false feature points before and after the reduction of nearly 1 / 3, and then extract a reliable feature point information, in order to achieve the fingerprint match.

Part of the fingerprint matching, we use Gabor filters fingerprint matching algorithm, and conduct research.

Keywords: fingerprint identification ； feature extraction ； Gabor filter

第 1 章 绪论

1.1 指纹识别技术的背景

指纹识别是依据人的手指尖表面的脊、谷线分布状态来识别和验证人身份的方法。据考古学家证实，人类对于指纹的应用，可以追溯到公元前 7000 年到 6000 年的古叙利亚和中国。在那个时代的一些粘土陶器等工艺品及日用品上，都留有陶艺匠人的指纹。在中国历史上，也经常在契约等文件上用红印记录下相关人士的拇指印，作为对内容的约定。

尽管指纹在我国应用较早，但由于长期缺乏专门的系统研究，未能上升到一门专门的学科。在 1892 年英国科学家 Galton 出版了《指纹学》和《指纹分析法》，从此奠定了指纹科学研究的基础。在 19 世纪中叶，“没有任何两个手指指纹的纹线形态一致，指纹纹线的形态终生不变”的结论，使得指纹在犯罪鉴别中得以正式应用。而在 1896 年在阿根廷首次也根据这一结论应用了指纹鉴定犯罪技术。1899 年，英国学者亨利将指纹的特征及识别原理加以分析归纳，科学地提出了人的指纹各不相同，并提出了基于指纹特征进行识别的原理和方法，以后衍生出的各种识别方法都是基于该理论。到本世纪初，指纹学的研究基本成熟，这时的指纹鉴别，主要由指纹专家根据指纹知识凭经验判断。世界上许多国家都建立了指纹库，仅美国联邦调查局的指纹库中就存有二亿多张指纹卡。然而传统的指纹识别方法面临着从指纹库中人工查找、对比指纹卡速度慢、效率低、对人员要求高等问题。从二十世纪六十年代开始，随着计算机技术的发展，传统的指纹识别技术发生了重大变化。人们开始利用计算机来建立指纹识别系统。

1.2 指纹识别研究的目的和意义

指纹识别与其他识别方法最大的区别就在于——传统的验证身份信息的方法，借助于一些外物，或者一些特殊的语言信息等，例如：信物、暗语等等，却始终没有办法能在人本身寻找到确认的方法。只要“物”或“信息”的有效性得到确认，那么这个人就可以进行权利范围内的所有操作。但是这种方法没有考虑到“物”的可伪造性(也就是不唯一性)和丢失的可能，也没有考虑到“信息”的被盗或被破解，缺点显而易见。

而生物识别技术却具有不可复制的唯一性，首先他的采集方便，不需要特殊的制造，每个人身上都会有，并且可以随时采样，其次这些信息无法复制也无法轻易的改变，具有很高的唯一性，最后这些信息在很长时间内都不会发生改变，非常有助于长期使用。

也正是因为这些优越性，人们还逐步发展了语音识别、面部识别、虹膜识别、签名识别、指纹识别等多种生物识别技术。而指纹的独特性、持久性和防卫性能都比较高，是众多生物特征中综合性能最高的，成为了生物识别技术的首选。同时，指纹识别也是目前应用的最为广泛的生物识别技术。指纹识别就是用每个人独特的指纹特征构成口令，提高系统的安全性。随着科技的进步，个人电脑和光学扫描仪两项技术不断革新，指纹识别技术不断发展，为指纹识别技术的应用提供了更广阔的空间。现在利用指纹来

完成身份验证和识别任务的系统已经大规模使用。比如：政府机要部门、国家重点实验室、军事要地、高级住宅等重要入口的身份验证，关键设备的启动控制，银行金库、金融系统等，公司、学校等单位的考勤，甚至部分家庭已经使用指纹锁。这种安全、方便、高效的身份鉴别方法会带给人们意想不到的便利：畅想一下，不需要再一记忆繁琐的密码、回家不需要带钥匙等等，那将是一种怎样的情景呢？指纹识别技术的广泛应用必将开创个人身份鉴别的新时代。

1.3 指纹识别在国内外的研究现状及分析

1.3.1 国外研究现状

人类将指纹作为身份识别的依据和验证身份的方法已有悠久的历史。早在公元前7000－6000 年以前，在叙利亚和中国，指纹作为身份鉴别的工具已经开始应用。中国古代就出现了在文契上的“按指为书”—画押。在欧洲，1788 年 Mayer 首次著文指出指纹的两个重要特性：唯一性和稳定性；现代的指纹匹配技术主要是 16 世纪后期产生。1872 年 Francis Galton 提出了分叉点和端点开发人个指纹识别模式，基于这两种特征的指纹识别模式至今都在使用。并且这两种细节特征可以为每一枚指纹构建唯一的信息。Henry Faulds 在 1880 年，第一次科学的提出了指纹的两个重要特征：一是任何两个不同手指的指纹脊线的式样（ridge pattern）不同，二是指纹脊线的式样在人的一生中不会改变。这一发现奠定了现代指纹识别技术的理论基础，也使得指纹识别在罪犯鉴定中得到应用。Francis Galton 对指纹进行深入研究，并于 1888 年引入了特征点的分类技术。1899 年，Edward Henry 学习了 Galton 的指纹科学，建立了著名的“Henry System”用于指纹分类。使用精准的指纹索引给专家指纹识别带来极大的便利。早在 20 世纪初期，司法部门已经正式采用指纹作为有效的身份标记，一些指纹识别机构建立了世界范围的罪犯指纹档案库。

1923 年 Purkinje 首次对指纹进行了分类；19 世纪晚期，F.Glton 开展了关于指纹的广泛研究，并引入了指纹中的细节特征作为指纹识别的依据。这些研究成果和方法为现代指纹识别技术奠定了基础，至今，一些方法仍被广泛使用。1960 年，美国联邦调查局、英国内务部（Home offiee in the UK）和法国巴黎警察局联合开始投巨资研发指纹识别系统，并于 1975 年成功推出了第一个商业化系统，随后，各国研究机构和许多大公司开始指纹识别技术的研究和产品开发工作。国际上著名的指纹识别系统有：美国联邦调查局的 AFIS 系统，日本 NEC 公司的指纹鉴定系统，北美英弗公司的指纹鉴别系统等。目前，随着数字化、信息化社会对自动身份鉴别技术的要求的不断提高和 AFIS 在司法领域取得了巨大成功，随着计算机硬件性能的飞速提高和价格的不断降低，随着普通大众对指纹识别了解的深入和接受，指纹识别技术已经突破了司法、侦探领域进入民用领域，并取得了快速的发展。传统指纹识别算法（主要用于司法、刑侦领域）主要考虑降低拒识率，一般需要人工协助处理，而且存在误识率高、计算速度慢、资源消耗大等问题，并不适合于民用领域。同时，民用市场对指纹识别算法在自动化程度、拒识

率和误识率、响应时间、资源消耗等方面也提出了更高的要求。JAIN 等人于 1998 年提出将指纹与人脸识别的结果融合；于 2000 年提出确定每个用户的特定参数的方法将指纹、脸像和手形的识别结果融合，并在 2001 年对多生物特征识别作了概述。2006 年初，澳大利亚成功发行世界上第一本生物识别护照。2007 年 11 月，美国国土安全部宣布所有入境美国的非美国公民都要接受数字拍照及双手十指指纹扫描。指纹识别即将迎来迅速普及的发展时期。2009 年，美国成功对指纹识别系统进行了更新的研究。

1.3.2 国内研究现状

我国利用指纹识别身份的历史最早可以追溯到秦朝，1903 年,中国青岛市警察局首次应用汉堡式指纹法。此后我国相继开展了指纹的应用及研究,还曾建立过“指纹学会”。刘紫宛编写的《中华指纹法》一书是我国最早的指纹专著。全国解放后,我国对指纹研究一直比较重视。1955 年编制了《中华人民共和国十指纹分析法》。这可以说是我国指纹的科学时期。

在国内，清华大学在 80 年代开始指纹识别的研究。中科院自动化所模式识别国家重点实验室自 90 年代以来，一直致力于“基于生物特征的身份鉴别”的研究，在指纹、虹膜、脸相识别等方面取得了很多的研究成果。北京大学视觉与听觉信息处理国家重点实验室先后承担了国家“七五”和“八五”，科技攻关项目，对指纹识别进行了长期的基础性研究，提出了一整套独创的理论和高效实用的算法。另外，自九十年代初以来，我国的北大方正集团、长春鸿达集团、西安青松集团等机构分别以所在地高校为技术依托，陆续开展了这方面的研究工作。总的来说，国内开展了很多研究，而且取得了很多成果。2002 年，清华大学实现了在海量数据库上的人脸和指纹综合识别系统，在识别的过程采用的融合策略是先用人脸特征进行比对得到前 n 个候选，然后在这个范围内用指纹特征再进行比对。迄今为止，还没有综合生物特征的识别系统的产品问世，综合身份识别系统的研究有待于进一步发展。2009 年中北大学信息与通信工程学院提出了一种基于傅立叶变换的指纹图像增强技术，大大提高了图像的清晰度。为后来的指纹识别技术作出了较大贡献。

1.3.3 研究现状分析

现在国内外指纹识别大都采用基于细节特征点的指纹识别技术，即采用基于图像处理的指纹识别算法，其中比较有代表性的有两种。一种是基于方向滤波增强，并在指纹细化图上提取特征点的算法，另一种是直接从指纹灰度图上提取特征点的算法。指纹识别作为一种热门的生物识别技术受到越来越多人的关注，国内外许多机构和学者都采用了很多不同的算法对指纹图像进行预处理和匹配。但有些算法会由于指纹图像的噪音、皮肤弹性引起的非线性形变等多方面因素，导致在识别过程中出现误差，影响识别率等。

1.4 本文研究的主要内容及工作安排

1.4.1 研究的主要内容

通过阅读大量的文献资料，本文深入研究了如何进行指纹识别技术所包含的主要方面：

指纹图像预处理：全文研究的重点是指纹图像预处理算法。预处理的目的是改善输入指纹图像的质量，以提高特征提取的准确性。本文采用灰度分割法对质问图像进行分割。利用中值滤波进行去噪。通过自适应二值化的方法处理指纹图像，最后再对图像进行细化以及去除毛刺，断裂等干扰。

指纹图像特征提取：对指纹图像的特征点进行提取。由于经过预处理后的细化图像上存在大量的伪特征点，所以提取大量的伪特征点，这些伪特征点的存在，不但使匹配的速度大大降低，还使指纹识别性能急剧下降，造成识别系统的误拒率和误识率的上升，因此在进行指纹匹配之前，尽可能将伪特征点去除，针对提取出指纹细节特征点含有大量的伪特征这一问题，提出了一种边缘信息判别法，有效地去除了边界伪特征点，再根据脊线结构特性去除其毛刺和短脊等伪特征点，显的减少了伪特征点。

指纹匹配：对指纹图像的匹配算法进行研究。特征匹配是识别系统的关键环节，匹配算法的好坏直接影响识别的性能、速度和效率。为了克服指纹图像非线性形变的影响，采用基于结构特征的点匹配算法，对校准后的点集进行匹配，匹配的特征点个数在两个点集中所占比例大约百分之六十五的范围内就可判为匹配成功。

并且重点在matlab环境下编译出可以进行对图像信息的提取进行指纹识别的算法，对其进行研究。

1.4.2 工作安排

本论文共分三章，每章的主要安排如下：

第一章为绪论部分，对指纹识别技术及系统的研究目的、意义及国内外发展动态进行了概述，比对其研究现状进行分析。

第二章为指纹识别技术的组成及原理分析。简单介绍了指纹识别的工作流程，以及指纹识别的基本原理，包括指纹结构特征、分类方式等。

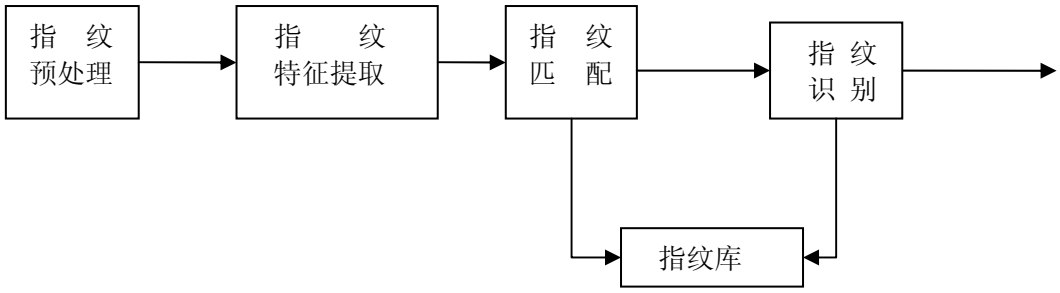
第三章为毕业设计研究的基于 gabor 滤波器的指纹识别软件的研究，操作过程，实验结果及代码程序。

第 2 章 指纹识别的组成及原理分析

2.1 指纹识别系统的组成

指纹识别技术是指使用取像设备读取指纹图像，通过识别软件提取出指纹图像中的特征数据，然后根据匹配算法得到的结果鉴别指纹所有人身份的生物特征识别技术。

指纹识别系统主要涉及三大步骤:指纹图像预处理、特征提取、特征匹配三个部分，其中预处理部分又可分为归一化、图像滤波增强、二值化和细化等几个步骤。系统流程框图如图 1－1 所示。下面对这三个个部分做一下简单的介绍。



指纹识别系统流程图

2.2 指纹识别的基本原理

2.2.1 指纹的基本特征

指纹其实是比较复杂的。与人工处理不同，许多生物识别技术公司并不直接存储指纹的图象。多年来在各个公司及其研究机构产生了许多数字化的算法（美国有关法律认为，指纹图象属于个人隐私，因此不能直接存储指纹图象）。但指纹识别算法最终都归结为在指纹图象上找到并比对指纹的特征。

指纹识别系统中，通常采用全局和局部两种层次的结构特征。两枚指纹可能具有相同的全局特征，但局部特征却不可能完全相同。

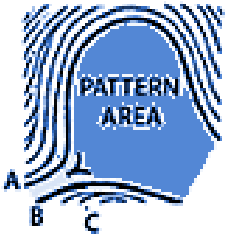
2.2.2 全局特征

全局特征是指那些用人眼直接就可以观察到的特征，包括：基本纹路图案环型（loop）,弓型（arch）,螺旋型（whorl）如图所示。其他的指纹图案都基于这三种基本图案。仅仅依靠图案类型来分辨指纹是远远不够的，这只是一个粗略的分类，但通过分类使得在大数据库中搜寻指纹更为方便。

环型　　弓型　　螺旋型

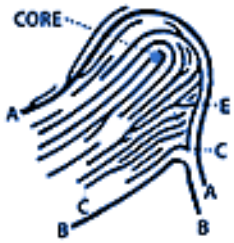
环型、弓型、螺旋型指纹图像

模式区是指指纹上包括了总体特征的区域，即从模式区就能够分辨出指纹是属于那一种类型的。有的指纹识别算法只使用模式区的数据。Secure Touch 的指纹识别算法使用了所取得的完整指纹而不仅仅是模式区进行分析和识别，如图所示。



模式区

核心点位于指纹纹路的渐进中心，它在读取指纹和比对指纹时作为参考点。许多算法是基于核心点的，既只能处理和识别具有核心点的指纹。核心点对于 Secure Touch 的指纹识别算法很重要，但没有核心点的指纹它仍然能够处理，如图所示。



核心点

三角点位于从核心点开始的第一个分叉点或者断点、或者两条纹路会聚处、孤立点、折转处，或者指向这些奇异点。三角点提供了指纹纹路的计数跟踪的开始之处，如图所示。

三角点

指模式区内指纹纹路的数量。在计算指纹的纹数时，一般先在连接核心点和三角点，这条连线与指纹纹路相交的数量即可认为是指纹的纹数,如图所示。

纹数

2.2.3 局部特征

局部特征是指指纹上的节点的特征，这些具有某种特征的节点称为特征点。两枚指纹经常会具有相同的总体特征，但它们的局部特征--特征点，却不可能完全相同。

1、指纹的特征点

指纹纹路并不是连续的、平滑笔直的，而是经常出现中断、分叉或打折。这些断点、分叉点和转折点就称为“特征点”。就是这些特征点提供了指纹唯一性的确认信息。指纹上的节点有四种不同特性：

2、特征点的分类

有以下几种类型，最典型的是终结点和分叉点。

终结点（Ending）：一条纹路在此终结，如图所示。

终结点

分叉点（Bifurcation）：一条纹路在此分开成两条或更多的纹路，如图所示。

分叉点

分歧点（Ridge Divergence）：两条平行的纹路在此分开，如图所示。

分歧点

孤立点（Dot or Island）：一条特别短的纹路，以至于成为一点，如图所示。

孤立点

环点（Enclosure）：一条纹路分开成为两条之后，立即有合并成为一条，这样形成的一个小环称为环点，如图所示。

环点

短纹（Short Ridge）：一端较短但不至于成为一点的纹路，如图所示。

短纹

方向（Orientation）：节点可以朝着一定的方向。

曲率（Curvature）：描述纹路方向改变的速度。

位置（Position）：节点的位置通过(x,y)坐标来描述，可以是绝对的，也可以是相对于三角点或特征点的。

2.3 指纹识别的一般算法

2.3.1 指纹图象预处理

在指纹识别过程中，输入的指纹图像由于各种原因的影响，是一幅含噪声较多的灰度图像，预处理的目的就是去除图像中的噪声，使图像画面清晰，边缘明显，把它变成一幅清晰的点线图，以便于提取正确的指纹特征。指纹图像预处理环节在整个指纹识别系统中具有重要的地位和作用，它的好坏直接影响着指纹识别的效果。预处理一般分为四步进行：图像分割、图像滤波、二值化和细化。

首先，对图像进行分割。由于有的原始图像跟其背景区域相混合，在背景和指纹图像之间存在一道白色区域，所以需要对原始指纹图像进行背景分离，，消除最外面的边框。我们可以根据灰度的大小对图像进行初步处理，得到初步处理然后对指纹图像进行归一化及分割处理，消除剩下的背景区域。

其次，指纹预处理过程中最重要的一步就是对指纹图像进行滤波去噪，它是指纹图

像预处理需要解决的核心问题。图像滤波的目的是在增强脊线谷线结构对比度的同时抑制噪声，连接断裂的脊线和分离粘连的脊线，按特定的需要突出一幅图像中的某些信息，同时削弱或去除某些不需要的信息。

再次，图像经滤波处理后，其中的纹线(脊)部分得到了增强，不过脊的强度并不完全相同，表现为灰度值的差异。二值化的目的就是使脊的灰度值趋向一致，使整幅图像简化为二元信息在指纹识别中，一方面对图像信息进行了压缩，保留了纹线的主要信息，节约了存储空间，另一方面还可以去除大量的粘连，为指纹特征的提取和匹配作准备。

最后，指纹图像二值化后，纹线仍具有一定的宽度，而指纹识别只对纹线的走向感兴趣，不关心它的粗细。细化的目的是为了删除指纹纹线的边缘像素，使之只有一个像素宽度，减少冗余的信息，突出指纹纹线的主要特征，从而便于后面的特征提取。细化时应保证纹线的连接性，方向性和特征点不变，还应保持纹线的中心基本不变。

图像的预处理大致可以划分为以下几步：分割、平滑滤波、二值化和细化。主要流程如下图所示：

预处理主要流程

①对指纹图像进行分割

由于获得的指纹图像跟其背景区域相混合，所以需要对原始指纹图像进行背景分离。

对指纹图像进行分割处理，消除剩下的背景区域。

a．先对初步处理后的指纹图像进行归一化处理，在此利用公式如下：

$$G(i,j)=\begin{cases} M_0+\sqrt{\dfrac{V_0(I(i,j)-M_i)^2}{V_i}} & \text{若I}(i,j)\rangle M; \\ M_0-\sqrt{\dfrac{V_0(I(i,j)-M_i)^2}{V_i}} & \text{若}I(i,j)\langle M; \end{cases}$$

如果 $M_0-\sqrt{\dfrac{V_0(I(i,j)-M_i)^2}{V_i}}\langle M_i$，则把灰度值 $M_0-\sqrt{\dfrac{V_0(I(i,j)-M_i)^2}{V_i}}$ 归一化为 255 背景处理，其中 M_0 和 V_0 为期望的均值和方差，根据实际情况而定，M_i 和 V_i 为指纹图像的均值和方差。

b．对指纹图像进行分块，将其分为 8×8 的小块，如果是背景区域，其灰度的方差较小，而前景区的指纹图像的方差较大，所以对每个小块求其方差，再设定一个阈值，小于阈值的方块区域设置为背景区域，将其灰度值设定为 255，而大于阈值的区域的灰度值保持不变，从而可以将指纹图像从背景区域很好的分离。

②对指纹图像进行二值化处理

由于分割后的图像质量仍然不是很好，所以需要对其进行滤波、消除毛刺、空洞处理和二值化处理，以使指纹图像清晰，消除不必要的噪声，以利于进一步的辨识。

指纹图像二值化作为指纹预处理过程的一部分，是进行指纹图像细化处理的基础。目前指纹细化方法都是基于二值指纹图像进行的。对指纹图像二值化的好处在于使得图像的几何性质只 0 和 1 的位置有关，不再涉及像素的灰度值，使处理变得简单，这给存储和处理带来了很大的方便，同时也提高了系统的经济实用。一个好的算法可以得到一个高质量的二值图像。反之，如果该阶段引入噪声，就会直接降低图像质量，影响识别精度。对指纹图像进行二值化，其基本要求就是二值化后的图像能真实地再现原指纹。具体要求为：

1．脊线中不出现空白；

2．二值化后的脊线基本保持原来指纹的特征；

3．指纹的纹线不应有太多的间断和相连；

4．指纹纹线间的间距应大致相同。

指纹图像首先要进行中值滤波处理，去除噪声。然后进行二值化过程，变成二值图像。由于原始指纹图像不同区域深浅不一，如对整幅图像用同一阈值进行二值分割，会造成大量有用信息的丢失。这里我们使用自适应阈值二值化的思想，对每块指纹图像，选取的阈值应尽量使该块图像内大于该阈值的像素点数等于小于该阈值的像素点数。

一般灰度图像二值化的变换函数 $f(x)$ 用下列公式表示，见式：

$$f(x)=\begin{cases}1, x \geq T\\ 0, x < T\end{cases}$$

公式中 T 为指定的阈值，x 为灰度值。

自适应阈值算法首先是利用固定阈值算法的思想，然后根据图像中每一部分的明暗度来调整阈值。本文首先把图像分为若干个 $w\times w$ 的方块，每一块根据自己的阈值进行二值化。这种算法充分利用了指纹图中脊线与谷线宽度大致相同的特点，即二值化后黑白像素的个数也应大致相同，首先利用固定阈值算法的特点对指纹图像中的每块确定一个大致的阈值，然后再利用自适应的思想对阈值进行准确的调整，即阈值的取值合适时图像是最光滑的，既没有“黑洞”阈值过大，也没有“白点”阈值过小，所以 0—1 之间的转换次数最少。下面为块区域阈值的选取算法：

1．将指纹图像划分为不重叠的大小为 $w\times w$ 的块，求取该区域内所有像素的灰度平均值。在综合考虑算法速度和处理效果两方面的条件下，本文分块尺寸为 8×8；T 为块的灰度平均值．见下式：

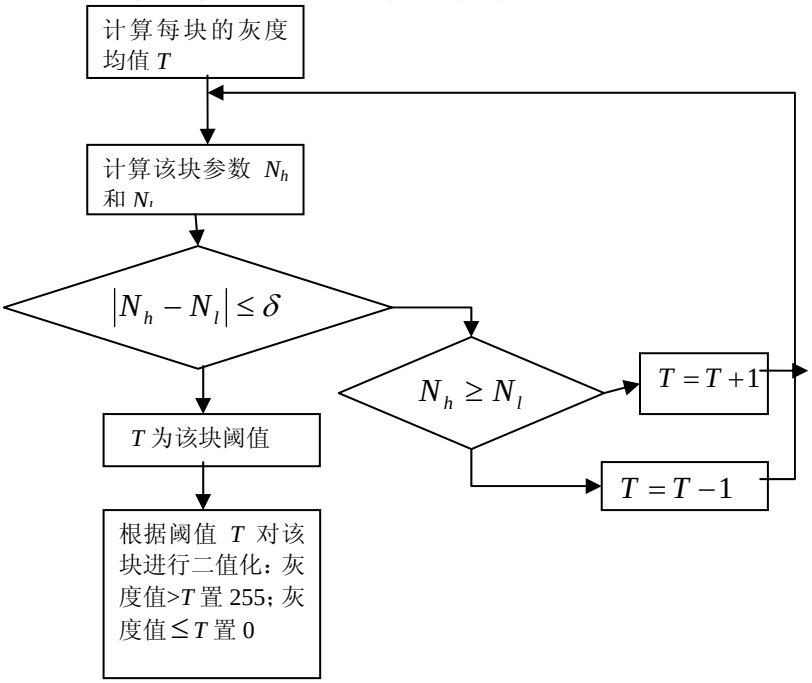
$$T=\frac{1}{w\times w}\sum_{u=i-\frac{w}{2}}^{i+\frac{w}{2}}\sum_{v=j-\frac{w}{2}}^{i=\frac{w}{2}}G(i,j)$$

2．计算区域内的N_h和N_l的值，N_h=灰度值大于等于T的像素点的个数。N_l=灰度值小于T的像素点的个数；

3．如果$|N_h - N_l| \le a(a=w\times w\times 10\%)$，则$T$为阈值；

4．若$N_h > N_l$，则$T=T+1$，否则$T=T-1$，返回第二步。

自适应阈值二值化的流程图如图所示：



自适应阈值二值化流程图

图中T为该块指纹图像的平均灰度值N_h、N_l分别为第(k,l)块指纹图像中灰度值大于等于T和小于T的像素点数，$\delta = w\times w\times 10\%$，$w$是分块尺寸（像素）。

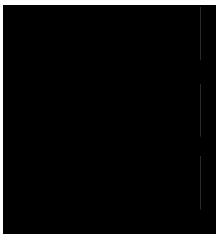
③对指纹图像进行细化处理

指纹图像处理中物体的形状信息是十分重要的，为了提取指纹图像特定区域的特征，对指纹图像通常需要采用细化算法处理，得到与原来指纹图像形状近似的由简单的弧或曲线组成的图形，这些细线处于物体的中轴附近，这就是所谓的指纹图像的细化。

细化方法不同,细化结果就有差异。在指纹识别中要求在不改变原来指纹图像的拓朴连通性的同时,细化的结果应为严格的八邻域图像骨架;纹线中除去特征点以外,每个像素均只与相邻两个像素为八邻域,抹去任意一像素都将破坏纹线的连接性。概括起来说就是纹线细化处理要满足收敛性、连接性、拓朴性、保持性、细化性、中轴性、快速性的要求。目前为止,关于细化方法的研究工作已有很多成果,所采用的方法从使用的观点来看,比较多的是采用模板匹配的方法(如迭代法、OPTA 单连通法等)。这种方法是根据某个像素的局部邻域(如 3×3,5×5 等)的图像特征对其进行处

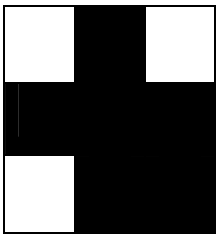
理,此外也有采用边缘搜索编码、外轮廓计算以及神经网络等细化方法。从处理的过程来看,主要可以分为串行和并行两类,前者对图像中当前像素处理依据其邻域内像素的即时化结果,且不同的细化阶段采用不同的处理方法;后者对当前的像素处理该像素及其邻域内各像素的前一轮迭代处理的结果,自始至终采用相同的细化准则。

对于任意形状的区域，细化实质上是腐蚀操作的变体，细化过程中要根据每个像素点的八个相邻点的情况来判断该点是否可以剔除或保留。

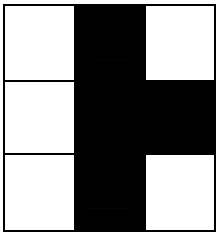


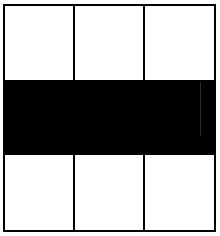
（ 1 ）

（2）

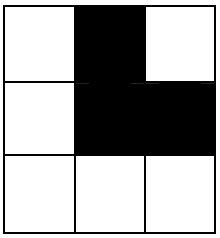


（3）

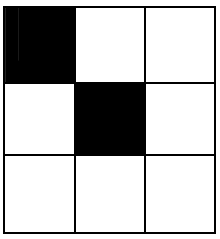




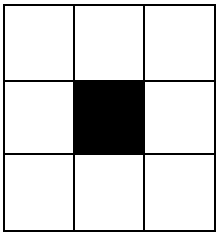
（ 4 ）



（ 5 ）



（6）



（7）

根据某点的八个相邻点的情况来判断该点是否能删除

图中给出了当前需要处理的像素点在不同的八邻域条件下的情况，可以看出：(1)不能删，因为它是个内部点，我们要求的是骨架，如果连内部点也删了，骨架也会被掏空的；(2)不能删，和(1)是同样的道理；(3)可以删，这样的点不是骨架；(4)不能删，因为删掉后，原来相连的部分断开了；(5)可以删，这样的点不是骨架；(6)不能删，因为它是直线的端点，如果这样的点删了，那么最后整个直线也被删了，剩不下什么；(7)不能删，因为孤立点的骨架就是它自身。总结上图，有如下的判据：

(1)内部点不能删除；

(2)孤立点不能删除；

(3)直线端点不能删除；

(4)如果 P 是边界点，去掉 P 后，如果连通分量不增加，则 P 可以删除。

我们可以根据上述的判据，事先做出一张表，从 0 到 255 共有 256 个元素，每个元素要么是 0，要么是 1。我们根据某点的八个相邻点的情况查表，若表中的元素是 1，则表示该点可删，否则保留。查表的方法是，设白点为 1，黑点为 0；左上方点对应一个 8 位数的第一位（最低位），正上方点对应第二位，右上方点对应的第三位，左邻点对应第四位，右邻点对应第五位，左下方点对应第六位，正下方点对应第七位，右下方点对应的第八位，按这样组成的 8 位数去查表即可。考虑当前像素点的各种八邻域的情况，我们可以得到一个细化操作查找表，该表在下面的细化算法中详细介绍。

为了避免分裂指纹图像，细化的过程分为两个步骤，第一步是正常的腐蚀操作，但是它是有条件的，也就是说那些被标记的可除去的像素点并不立即消去；在第二步中，只将那些消除后并不破坏连通性的点消除，否则的话保留这些边界点。以上的步骤是在一个 3×3 邻域内运算，可以通过查表实现细化的操作。算法的实现步骤如下：

（1）定义一个 3×3 模板和一个查找表，模板和查找表分别如表和图所示：

细化模板

1	2	4
128	256	8
64	32	16

erase table[256]={0，0，1，1，0，0，1，1，1，1，0，1，1，1，0，1，1，1，0，0，1，1，1，1，0，0，0，0，0，0，0，1，0，0，1，1，0，0，1，1，1，1，0，1，1，1，0，1，1，1，0，0，1，1，1，1，0，0，0，0，0，0，0，1，1，1，0，0，1，1，0，0，0，0，0，0，0，0，0，0，0，0，0，0，0，0，0，0，0，0，0，0，0，0，0，0，1，1，0，0，1，1，0，0，1，1，0，1，1，1，0，1，0，0，0，0，0，0，0，0，0，0，0，0，0，0，0，0，0，0，1，1，0，0，1，1，1，1，0，1，1，1，0，1，1，1，0，0，1，1，1，1，0，0，0，0，0，0，0，1，0，0，1，1，0，0，1，1，1，1，0，1，1，1，0，1，1，1，0，0，1，1，1，1，0，0，0，0，0，0，0，1，1，0，0，1，1，0，0，0，0，0，0，0，0，0，0，1，1，0，0，1，1，1，1，0，0，0，0，0，0，0，0，1，1，0，0，1，1，0，0，1，1，0，1，1，1，0，0，1，1，0，0，1，1，1，0，1，1，0，0，1，0，0，0}

细化查找表

（2）对二值图像从上到下、从左到右进行扫描；该过程结束后再对图像进行从左到右，从上到下的扫描；如果图像中当前像素点的灰度值为“0”，且其左右（第一次扫描过程考虑左右像素点）或上下（第二次扫描过程考虑上下两个像素点）两个像素点中有任意一个为“255”则转至步骤（3），否则回转到步骤（2）；

（3）该像素点为中心的 3×3 区域内的各个像素值和定义的模板中的权值进行卷积求和，得到查找索引值 k；

（4）根据这个索引值 k 得到表里相应的数据，如果为“1”，那么该像素点的灰度值设为“255”，如果为“0”，则该像素点的灰度值为“0”。

（5）图像从头至尾扫描二遍后，如果该次扫描修改了图像中的点，则跳转至步骤二，开始新的一轮扫描。否则图像细化结束。

2.3.2 特征提取

细节特征提取的方法分为两种：一种是从灰度图像中提取特征，另一种是从细化二值图像中提取特征。直接从灰度图像中提取特征的算法一般是对灰度指纹纹线进行跟踪，根据跟踪结果寻找特征的位置和判断特征的类型。这种方法省去了复杂的指纹图像预处理过程，但是特征提取的算法却十分复杂,而且由于噪声等因素影响，特征信息（位置、方向等）也不够准确。目前大多数系统采用第二种方法，从细化二值图像中提取特征，该方法比较简单，在得到可靠的细化二值图像后，只需要一个 3×3 的模板就可以将端点和分叉点提取出来。

特征点提取的好坏将直接影响匹配的结果。现实中，指纹输入时，由于汗渍、干燥、按压力度不同等影响,得到的指纹图像大都含有断纹、褶皱、模糊、灰度不均匀等质量问题，虽然经过预处理，图像质量会有所改观，但预处理算法对各个指纹的适应性和有效性也会不同，并且会引入新的噪声，因此得到的细化二值图像往往含有大量的伪特征点。伪特征点不仅会影响匹配的速度，严重的会影响整个识别的正确率。所以提取特征点后要进行去伪处理，尽可能滤除伪特征点、保留真特征点。实践中发现，伪特征点的数量一般占总特征数量的一半以上，所以去伪是必不可少的过程。去伪过程可以在两个阶段进行：一是在特征提取之前对细化二值图像进行平滑、去除毛刺、连接断纹等操作，然后提取特征作为真特征；另一种是在特征提取之后，根据特征之间的相互关系,尽可能准确的识别伪特征点并滤除它们。前者直接对图像进行修补，操作比较复杂，容易引入新的伪特征；后者对特征提取后的数据进行判断，识别比较麻烦，但是速度较快本文采用第二种方法，即从已提取的特征点中滤除伪特征，保留真特征。

本文的特征提取算法是在细化的图像基础上采用是模板匹配法。模板匹配法有运算量小、速度快的优点。

主要提取指纹的细节特征即端点和分叉点。端点和分叉点是建立在对 8 邻点的统计分析基础之上的，则在八邻域的所有状态中，满足端点特征条件的有 8 种，满足分叉点特征条件的有 9 种。

对于细化图像而言，像素点的灰度值只有两种情况（即 0 或 1）“0”为背景点灰度，“1”为纹线点灰度。对于细化图像上的任意点 P，其交叉数定义见式 4—1，P 点的八邻域黑点数定义见式：

$$cn=\sum_{i=1}^{8}\left|p_{i+1}-p_i\right|\left(p_9=p_1\right),\ p_i\text{表示该像素点的灰度值}$$

$$sn=\sum_{i=1}^{8}p_i$$

具体算法如下：

(1)从端点出发，端点的八邻域只有一个黑点,该点就是脊线跟踪的下一点；

(2)对脊线中间连续点，因为八邻域只有两个黑点,除去上一个被跟踪的点，下的一点即为下一个待跟踪点；

(3)设集合$\Omega=\{xi,yi,zi,gi\}$，记录下端点或分叉点的横坐标xi，纵坐标yi，及特征点的类型zi,gi是特征点的角度跟踪结束条件。若被跟踪点的八邻域黑点数等于 1 且交叉数等于 2 时如下式 4—3，则认为是端点；若被跟踪点的八邻域黑点数等于 3 且交叉数等于 6 时如下式 4—4，则认为是分叉点；端点的角度取从端点为起点的端线的角度，分叉点的角度取相对最小分支的角度。端线及分支线的角度求法为：即从一个特征的位置出发坐标为(xi,yi)搜索到步长为 7 是最后一点坐标为(x,y)。见式：

$$\begin{cases}cn=\sum_{i=1}^{8}|p_{i+1}-p_i|=2,p_9=p_1\\ sn=\sum_{i=1}^{8}p_i=1\end{cases}$$

$$\begin{cases}cn=\sum_{i=1}^{8}|p_{i+1}-p_i|=6,p_9=p_1\\ sn=\sum_{i=1}^{8}p_i=3\end{cases}$$

$$gi=\arctan(y-yi)/(x-xi)$$

求出特征点后，再根据平均纹线距离等信息对所得特征点进行有效性检验，去除伪特征点，保留真特征点。然后以特征点的坐标(x,y)，及特征点的方向 d，结合其邻域情况(邻域内的特征点数、相对位置、脊线上特征点所处位置的纹曲率、特征点邻域内的脊线纹密度等等)，可以构成该指纹细节特征点的特征向量。将所有的特征向量进行筛选后留下 50 到 80 个特征向量，构成指纹特征模板。

2.3.3 特征匹配

人们对指纹匹配做了很多研究，提出了许多匹配算法，主要可分为两类：一类是基于图形的匹配方式，包括点模式匹配和基于图论的方法；另一类是采用人工神经网络的方法。图形匹配是针对纹线几何形状及其特征点拓扑结构的匹配方式，它的原理是基于相似变换的方法把两个特征点集中的相对应点匹配起来，这些相似变换可以是平移变换、旋转变换、伸缩变换等线性变换，可以在一定程度内允许少量伪特征点的存在、真正特征点的丢失以及轻微的特征点定位偏差，且对图像的平移和旋转也不敏感。但这种方法有两点不足：一是匹配速度比较慢；二是对指纹图像的质量要求比较高，低质量的图像匹配效果不佳，下面对这些算法进行一些简要的介绍。

Ranade 和 R. Seinfeld 提出了点模式匹配的松弛算法，其思想是寻找一对匹配点，

使得反映匹配程度的相似变换最大，则将该点对作为基准点对，然后根据相似变换的计算结果调整待识别图像的位置，统计最终的匹配点对数，给出匹配结果。

Stookinan 等提出的基于 Hough 变换的方法，把点模式匹配转化成对转换参数的 Hough 空间中的峰值检测。这种方法的缺点在于当特征点数目较少（少于 30 个）时，很难在 Hough 空间里积累起足够大的证据来保证一个可靠的匹配，另外，该方法有计算量较大的缺点。

Sparrows 与 A.K.Hrechak 等都提出了基于结构信息的特征匹配方法，而 D.K.ISenor 与 S. G.zaky 使用图来表示指纹特征，并用图匹配的方法来匹配指纹图像。这类方法利用了指纹图像的拓扑结构，允许一般的图像平移旋转、特征点丢失以及伪特征点的存在，但是这类方法的准确性在很大程度上依赖于所提取的指纹特征信息及分类信息的准确性。

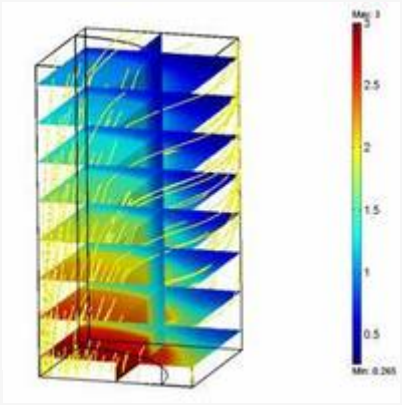
采用人工神经网络的指纹匹配方法也有很多。Vinod 将非对称神经网络应用于指纹匹配中，提出了一种基于非对称神经网络的点模式匹配算法，而田捷等人将遗传算法应用于指纹匹配中，提出了基于遗传算法的指纹图匹配方法，利用指纹图像的结构信息进行初匹配，缩小搜索空间，然后采用遗传算法和补偿算法匹配指纹图像，有较强的抗噪声与非线性形变的能力。但由于神经网络固有的反复处理特性，速度难以得到提高，计算量偏大，因此不适合用于对实时性要求较高的在线指纹识别系统。

第三章 基于滤波器的指纹识别软件

3.1 开发工具简介

MATLAB 是由美国 mathworks 公司发布的主要面对科学计算、可视化以及交互式程序设计的高科技计算环境。它将数值分析、矩阵计算、科学数据可视化以及非线性动态系统的建模和仿真等诸多强大功能集成在一个易于使用的视窗环境中，为科学研究、工程设计以及必须进行有效数值计算的众多科学领域提供了一种全面的解决方案，并在很大程度上摆脱了传统非交互式程序设计语言（如 C、Fortran）的编辑模式，代表了当今国际科学计算软件的先进水平。

MATLAB和Mathematica、Maple并称为三大数学软件。它在数学类科技应用软件中在数值计算方面首屈一指。MATLAB可以进行矩阵运算、绘制函数和数据、实现算法、创建用户界面、连接其他编程语言的程序等，主要应用于工程计算、控制设计、信号处理与通讯、图像处理、信号检测、金融建模设计与分析等领域。



图形处理功能

MATLAB 自产生之日起就具有方便的数据可视化功能，以将向量和矩阵用图形表现出来，并且可以对图形进行标注和打印。高层次的作图包括二维和三维的可视化、图象处理、动画和表达式作图。可用于科学计算和工程绘图。新版本的 MATLAB 对整个图形处理功能作了很大的改进和完善，使它不仅在一般数据可视化软件都具有的功能（例如二维曲线和三维曲面的绘制和处理等）方面更加完善，而且对于一些其他软件所没有的功能（例如图形的光照处理、色度处理以及四维数据的表现等），MATLAB 同样表现了出色的处理能力。同时对一些特殊的可视化要求，例如图形对话等，MATLAB 也有相应的功能

函数，保证了用户不同层次的要求。另外新版本的 MATLAB 还着重在图形用户界面（GUI）的制作上作了很大的改善，对这方面有特殊要求的用户也可以得到满足。

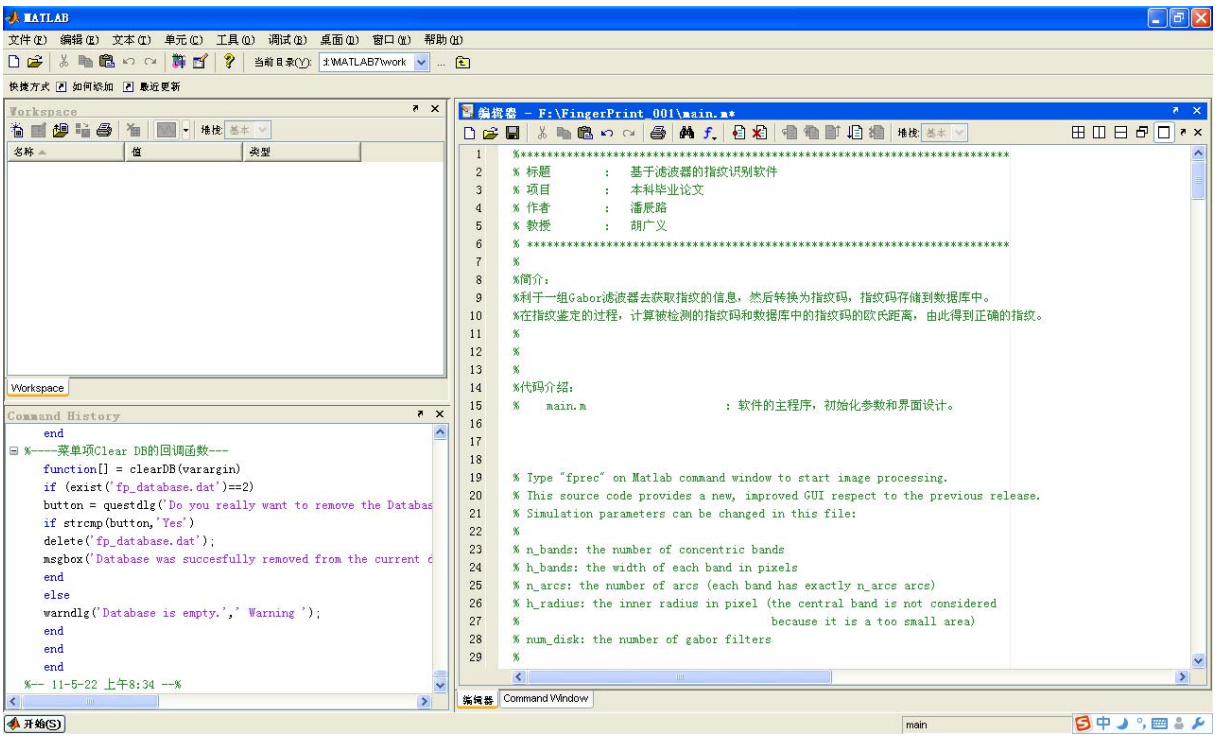
3.2gabor 滤波器

Gabor 变换属于加窗傅立叶变换，Gabor 函数可以在频域不同尺度、不同方向上提取相关的特征。另外 Gabor 函数与人眼的生物作用相仿，所以经常用作纹理识别上，并取得了较好的效果。二维 Gabor 函数可以表示为：

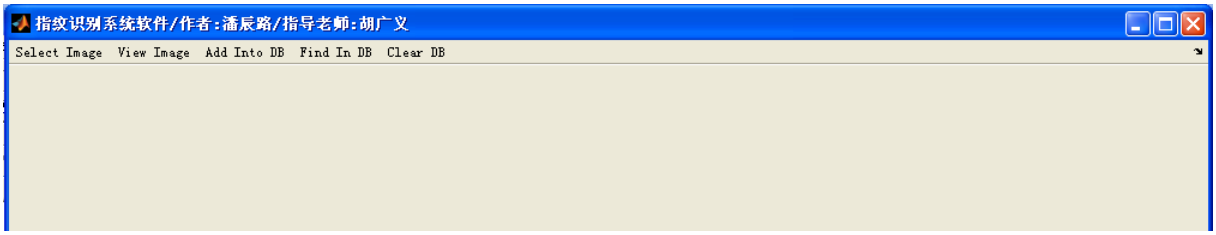
其中：

v 的取值决定了 Gabor 滤波的波长，u 的取值表示 Gabor 核函数的方向，K 表示总的方向数。参数决定了高斯窗口的大小，这里取。程序中取 4 个频率（v=0, 1, …, 3），8 个方向（即 K=8，u＝0, 1, … ,7），共 32 个 Gabor 核函数。

3.3 实验过程及结果

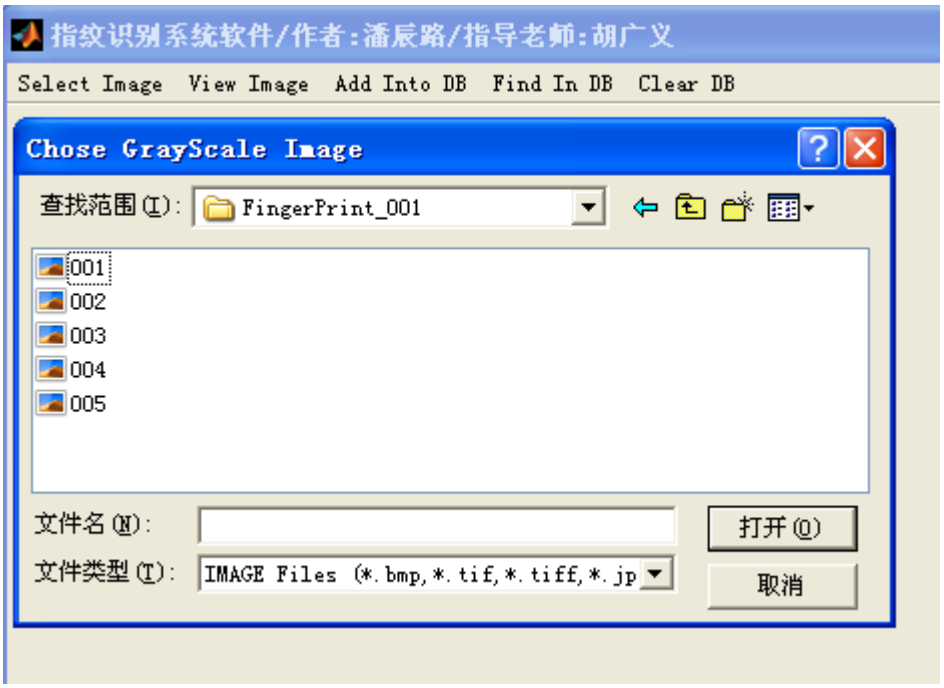


首先第一步 打开 matlab 加入 mian 文件

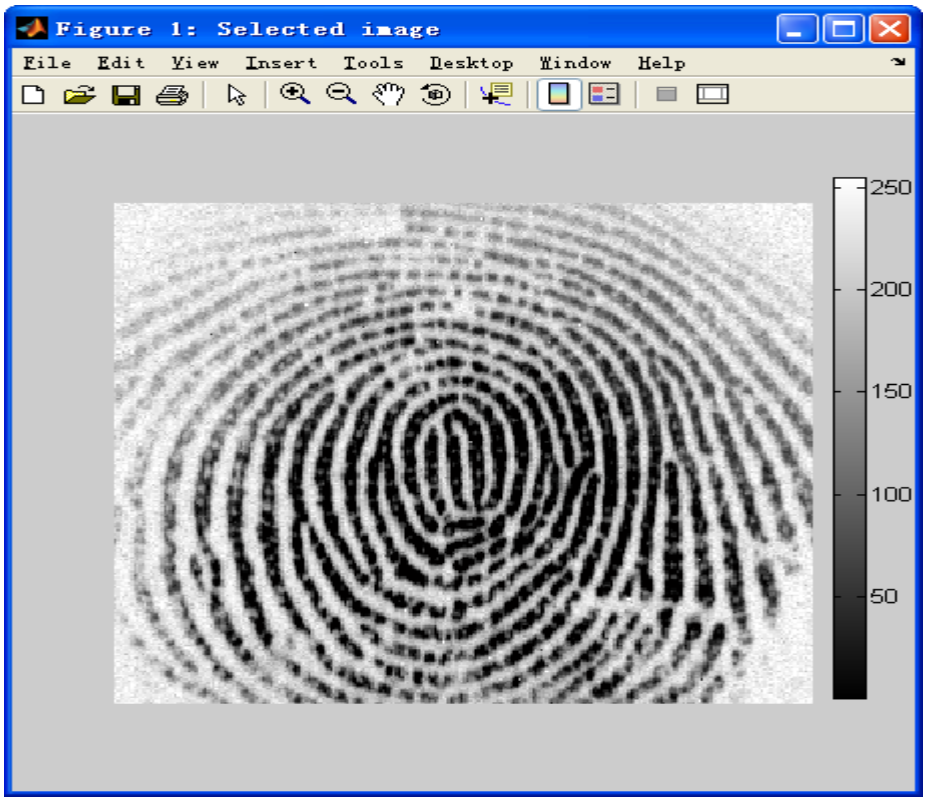


第二步 软件的界面主要包括五个操作部分：

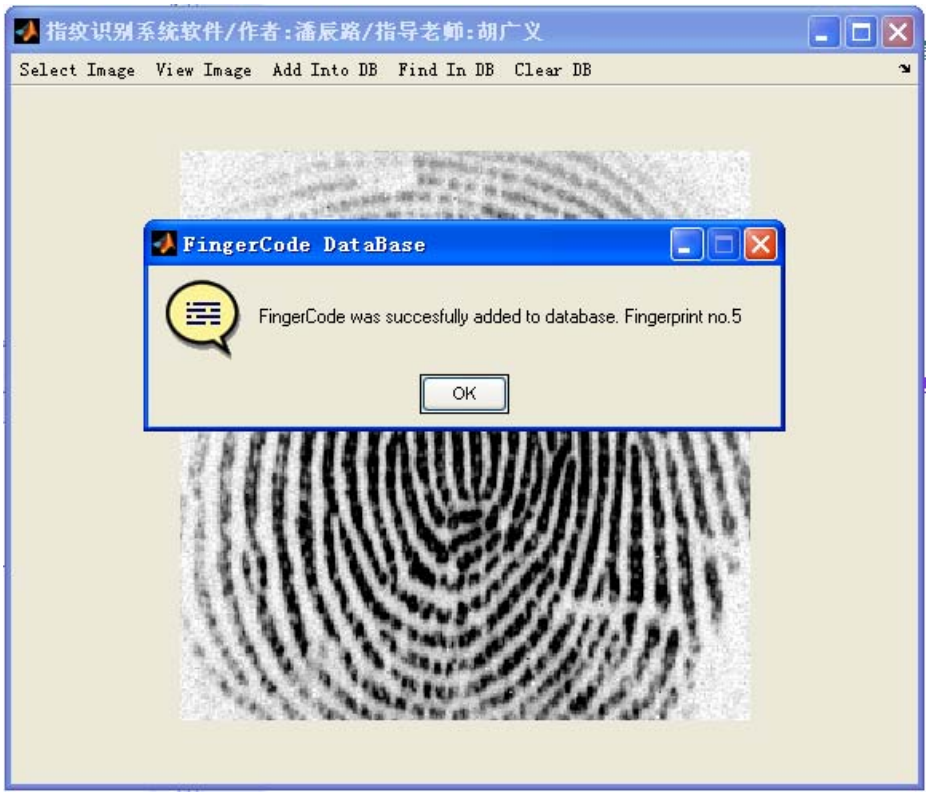
选择图像、观察图像、加入至 DB、在 DB 中寻找、清空 DB



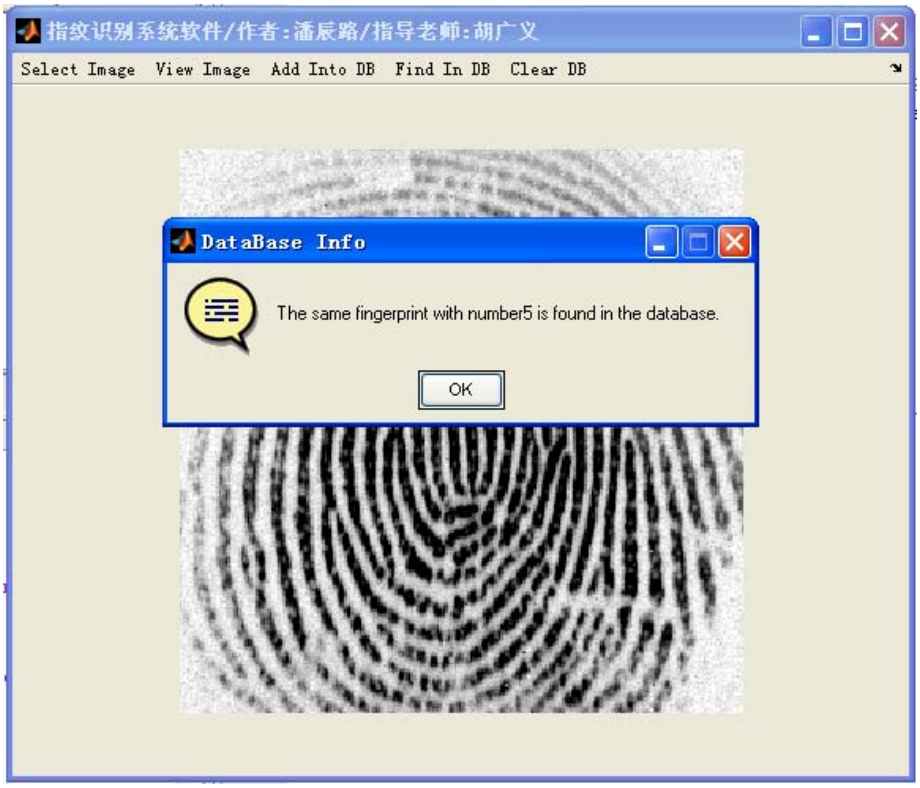
第三步：点击 select image 选择图片，例如选择 001 图片



第四步 可以选择 view image 来进一步观察图像 可放大 缩小 调整灰度等



第五步 在观察完图像后可以选择加入至存储空间 DB 中 此时会出现对话框显示：图片已成功加入至 DB



第六步同样选择 001 图片点击 find in DB 后会出现对话框成功在 DB 中找到

第七步 选择另外一张编号为 005 的照片 但不加入至 DB

第八步 选择 find in DB 就会出现对话框 表示未找到相同的数据

3.4 软件程序代码部分

```
%*************************************************************************
****
% 标题          :   基于滤波器的指纹识别软件
% 项目          :   本科毕业论文
% 作者          :   潘辰路
% 教授          :   胡广义
%
*************************************************************************
%
%简介：
%利于一组 Gabor 滤波器去获取指纹的信息，然后转换为指纹码，指纹码存储到数据库中。
%在指纹鉴定的过程，计算被检测的指纹码和数据库中的指纹码的欧氏距离，由此得到正确的指纹。
%
```

```
%
%
%代码介绍：
%    main.m                  : 软件的主程序，初始化参数和界面设计。

% Type "fprec" on Matlab command window to start image processing.
% This source code provides a new, improved GUI respect to the previous release.
% Simulation parameters can be changed in this file:
%
% n_bands: the number of concentric bands
% h_bands: the width of each band in pixels
% n_arcs: the number of arcs (each band has exactly n_arcs arcs)
% h_radius: the inner radius in pixel (the central band is not considered
%                                      because it is a too small area)
% num_disk: the number of gabor filters
%
% n_sectors and h_lato rapresents respectively the total number of sectors
% (the length of the feature vector associated to each filter of filter-bank:
% there are num_disk gabor filters) and the height of the cropped image in pixels.
% N_secors and h_lato should not be changed.
%
%
%
% Input fingerprint should be 256 x 256 image 8-bit grayscale @ 500 dpi.
% If these conditions are not verified some parameters in m-functions
% should be changed in a proper way (such as, for example, Gabor filter
% parameters in gabor2d_sub function). See the cited references for more
% details.
%
%   M-files included:
%
%      -main.m:              this file. It initializes the entire image processing. The simulation
```

```
%                           parameters can be changed in this main file.
%     -centralizing.m:      a function which accept an input image and
determines the coordinates
%                                 of the core point. The core point is
determinated by complex filtering.
%                           The region of interest is determinated fixing
a minimum threshold value
%                           for the variance. Input image is divided into
non-overlapping blocks and
%                           only blocks with a variance smaller than this
threshold value are considered
%                           background. The logical matrix (associated to
the region of interest) is first
%                           closed (Matlab function imclose), then eroded
(Matlab function imerode) with two
%                               given structuring elements. The image is
"mirrored" before convolution with complex
%                           filter, then it is re-cropped to its original
sizes.
%    -mirror.m:             a function which is used to "mirror" input image
in  order to avoid undesired
%                                   boundary effects (function used by
centralizing.m).
%    -recrop.m:             a function used to resize the mirrored filtered
image (function used by centralizing.m)
%    -conv2fft.m:           this function performs 2D FFT-based convolution.
Type "help conv2fft" on Matlab command
%                           window for more details.
%    -whichsector.m:        a function used to determine (for each pixel of
the cropped image) the corresponding
%                           sectors of the concentric bands (function used
by sector_norm.m).
%    -sector_norm.m:        a function used to normalize input image and to
calculate the features vector
%     -cropping.m:             this function is used to cropp the input
```

```
fingerprint image after the core point is
    %                           determinated.
    %    -gabor2d_sub.m:        a function used to calculate the coefficients
of the gabor 2D filters.
    %     -vedicentro.m:           this simple routines uses the M-function
centralizing.m and it is used to display
    %                           the core point.
    %
    %
    function main(varargin)
    %---清除缓存，命令窗口，对话框---
    clear;
    clc;
    close all;

    global    n_bands h_bands n_arcs h_radius h_lato n_sectors matrice num_disk
fileName pathName;       %定义全局变量

    %----初始化全局变量-------
    n_bands                 = 4;
    h_bands                 = 20;
    n_arcs                  = 16;
    h_radius                = 12;
    h_lato=h_radius+(n_bands*h_bands*2)+16;
    if mod(h_lato,2)==0
        h_lato=h_lato-1;
    end
    n_sectors = n_bands*n_arcs;
    matrice=zeros(h_lato);
    for ii=1:(h_lato*h_lato)
        matrice(ii)=whichsector(ii);
    end
    num_disk                = 8;
    fileName                = 0;
    pathName                = 0;
```

```
    bgColor=get(0,'DefaultUIControlBackgroundColor');        %获取默认的背景颜色

    %--- 计算屏幕的中心位置，设置用户界面的大小，是界面处于屏幕中心------
    screenUnits=get(0,'Units');
    set(0,'Units','pixels');
    screenSize=get(0,'ScreenSize');                                    % 获取屏幕大小
    set(0,'Units',screenUnits);
    figWidth=1000;
           %设置界面的宽
    figHeight=700;                                                     %设置界面的长
    figPos=[(screenSize(3)-figWidth)/2 (screenSize(4)-figHeight)/2  ...
      figWidth                    figHeight];                 %设置界面的位置


    %---创建界面窗口----
    h.Fig=figure(...
      'Color'                    ,bgColor                              ,...
% 设置主窗口背景颜色
      'IntegerHandle'     ,'off'                    ,...
      'DoubleBuffer'      ,'on'                     ,...
      'MenuBar'           ,'none'                   ,...
      'HandleVisibility'  ,'on'                     ,...
      'Name'                ,'指纹识别系统软件/作者:潘辰路/指导老师:胡广义'  ,...      % 主窗口的标题
      'Tag'               ,'Software Analyzer'  ,...
      'NumberTitle'       ,'off'                    ,...
      'Units'             ,'pixels'                 ,...
      'Position'          ,figPos                   ,...
      'UserData'          ,[]                       ,...
      'Colormap'          ,[]                       ,...
      'Pointer'           ,'arrow'                  ,...
```

```
    'Visible'                    ,'off'                                    ...
% 隐藏主窗口
    );


    h.fmSelectImage = uimenu(h.Fig,'label','Select Image');                %
创建菜单项， 命名菜单项为 Select Image
    set(h.fmSelectImage(1),'callback',@selectImage);
                    % 设置菜单项的回调函数

    h.fmViewImage = uimenu(h.Fig,'label','View Image');                    %
创建菜单项， 命名菜单项为 View Image
    set(h.fmViewImage(1),'callback',@viewImage);
% 设置菜单项的回调函数

    h.fmAddIntoDB = uimenu(h.Fig,'label','Add Into DB');                   %
创建菜单项， 命名菜单项为 Add Into DB
    set(h.fmAddIntoDB(1),'callback',@addIntoDB);
% 设置菜单项的回调函数

    h.fmFindInDB = uimenu(h.Fig,'label','Find In DB');                     %
创建菜单项， 命名菜单项为 Find In DB
    set(h.fmFindInDB(1),'callback',@findInDB);
% 设置菜单项的回调函数

    h.fmClearDB = uimenu(h.Fig,'label','Clear DB');                        %
创建菜单项， 命名菜单项为 Clear DB
    set(h.fmClearDB(1),'callback',@clearDB);
% 设置菜单项的回调函数

    set(h.Fig,'Visible','on');


        %----菜单项 Select Image 的回调函数---
        function[] = selectImage(varargin)
```

```
[fileName,pathName]=uigetfile({'*.bmp;*.tif;*.tiff;*.jpg;*.jpeg;*.gif','IMA
GE Files (*.bmp,*.tif,*.tiff,*.jpg,*.jpeg,*.gif)'},'Chose GrayScale Image');
% 获取指纹图片的文件名和路径
        if (fileName~=0)
                % 如果文件名不为空
            [img,map]=imread(strcat(pathName,fileName));
                % 读取指纹图像文件
            if(~isgray(img))
                                        % 如果指纹图像不是黑白图片
                warndlg('Please select a grayscale image.',' Warning ');
             % 显示警告对话框
                fileName = 0;
                                        % 设置文件名为空
            else
               subplot(1,1,1,'replace');

               imshow(img,map);
                            % 显示读取的指纹图像
            end
        else
           warndlg('Please select a fingerprint image.',' Warning ');
        end
    end

      %----菜单项 View Image 的回调函数---
    function[] = viewImage(varargin)
        if (fileName~=0)
                                        % 如果文件名不为空
            [img,map]=imread(strcat(pathName,fileName));
                    % 读取指纹图像文件
            figure('Name','Selected image');
                            % 创建一个 MATLAB 提供的窗口

            imshow(img);
```

```
                                                    % 显示指纹图像文件
            else
                warndlg('Please select a fingerprint image.',' Warning ');
            end
        end

        %----菜单项 Add Into DB 的回调函数---
        function[] = addIntoDB(varargin)
            if (fileName~=0)
                                                        % 如果文件名不为空
                [img,map]=imread(strcat(pathName,fileName));
                                % 读取指纹图像文件
                if(~isgray(img))
                    warndlg('Please select a grayscale image.',' Warning ');
                else
                    fingerprint=double(img);



[BinarizedPrint,XofCenter,YofCenter]=centralizing(fingerprint,0);

[CroppedPrint]=cropping(XofCenter,YofCenter,fingerprint);
                    [NormalizedPrint,vector]=sector_norm(CroppedPrint,0);

                    for(angle=0:1:num_disk-1)
                        gabor=gabor2d_sub(angle,num_disk);

ComponentPrint=conv2fft(NormalizedPrint,gabor,'same');
                        [disk,vector]=sector_norm(ComponentPrint,1);
                        finger_code1{angle+1}=vector(1:n_sectors);
                    end

                    img=imrotate(img,180/(num_disk*2));
                    fingerprint=double(img);
```

```
[BinarizedPrint,XofCenter,YofCenter]=centralizing(fingerprint,0);

[CroppedPrint]=cropping(XofCenter,YofCenter,fingerprint);
                    [NormalizedPrint,vector]=sector_norm(CroppedPrint,0);

                    for (angle=0:1:num_disk-1)
                        gabor=gabor2d_sub(angle,num_disk);

ComponentPrint=conv2fft(NormalizedPrint,gabor,'same');
                        [disk,vector]=sector_norm(ComponentPrint,1);
                        finger_code2{angle+1}=vector(1:n_sectors);
                    end
                    % FingerCode added to database
                    if (exist('fp_database.dat')==2)
                        s =load('fp_database.dat','-mat');
                        fp_number = s.fp_number;
                        data = s.data;
                        fp_number=fp_number+1;
                        data{fp_number,1}=finger_code1;
                        data{fp_number,2}=finger_code2;

save('fp_database.dat','data','fp_number','-append');
                    else
                        fp_number=1;
                        data{fp_number,1}=finger_code1;
                        data{fp_number,2}=finger_code2;
                        save('fp_database.dat','data','fp_number');

                    end

                    message=strcat('FingerCode was succesfully added to
database. Fingerprint no. ',num2str(fp_number));
                    msgbox(message,'FingerCode DataBase','help');
                end
```

```
        else
            warndlg('Please select a fingerprint image.',' Warning ');
        end
    end

      %----菜单项 Find In DB 的回调函数---
    function[] = findInDB(varargin)

        if (fileName~=0)
            [img,map]=imread(strcat(pathName,fileName));


            fingerprint = double(img);


[BinarizedPrint,XofCenter,YofCenter]=centralizing(fingerprint,0);
            [CroppedPrint]=cropping(XofCenter,YofCenter,fingerprint);
            [NormalizedPrint,vector]=sector_norm(CroppedPrint,0);

            % memoria per feature vector d'ingresso
            vettore_in=zeros(num_disk*n_sectors,1);
            for (angle=0:1:num_disk-1)
                gabor=gabor2d_sub(angle,num_disk);
                ComponentPrint=conv2fft(NormalizedPrint,gabor,'same');
                [disk,vector]=sector_norm(ComponentPrint,1);
                finger_code{angle+1}=vector(1:n_sectors);

vettore_in(angle*n_sectors+1:(angle+1)*n_sectors)=finger_code{angle+1};
            end

            % FingerCode of input fingerprint has just been calculated.
            % Checking with DataBase

            if (exist('fp_database.dat')==2)
                s =load('fp_database.dat','-mat');
```

```
%---- alloco memoria ----------------------------------
%...
fp_number = s.fp_number;
data = s.data;
vettore_a=zeros(num_disk*n_sectors,1);
vettore_b=zeros(num_disk*n_sectors,1);
best_matching=zeros(fp_number,1);
valori_rotazione=zeros(n_arcs,1);
% start checking ----------------------------------------
for scanning=1:fp_number
    fcode1=data{scanning,1};
    fcode2=data{scanning,2};
    for rotazione=0:(n_arcs-1)
        p1=fcode1;
        p2=fcode2;
        % ruoto i valori dentro disco
        for conta_disco=1:num_disk
            disco1=p1{conta_disco};
            disco2=p2{conta_disco};
            for old_pos=1:n_arcs
                new_pos=mod(old_pos+rotazione,n_arcs);
                if new_pos==0
                    new_pos=n_arcs;
                end
                for conta_bande=0:1:(n_bands-1)

disco1r(new_pos+conta_bande*n_arcs)=disco1(old_pos+conta_bande*n_arcs);

disco2r(new_pos+conta_bande*n_arcs)=disco2(old_pos+conta_bande*n_arcs);
                end
            end
            p1{conta_disco}=disco1r;
            p2{conta_disco}=disco2r;
        end
        % ruoto i dischi circolarmente
```

```
                            for old_disk=1:num_disk
                                new_disk=mod(old_disk+rotazione,num_disk);
                                if new_disk==0
                                    new_disk=num_disk;
                                end
                                pos=old_disk-1;

vettore_a(pos*n_sectors+1:(pos+1)*n_sectors)=p1{new_disk};

vettore_b(pos*n_sectors+1:(pos+1)*n_sectors)=p2{new_disk};
                            end
                            d1=norm(vettore_a-vettore_in);
                            d2=norm(vettore_b-vettore_in);
                            if d1<d2
                                val_minimo=d1;
                            else
                                val_minimo=d2;
                            end
                            valori_rotazione(rotazione+1)=val_minimo;
                        end
                        [minimo,posizione_minimo]=min(valori_rotazione);
                        best_matching(scanning)=minimo;
                    end
                    [distanza_minima,posizione_minimo]=min(best_matching);
                    beep;
                    message = strcat('');
                    if(distanza_minima == 0)
                        message = strcat('The same fingerprint with number
',num2str(posizione_minimo), ' is found in the database.');
                        %message=strcat('The nearest fingerprint present in
DataBase which matchs input fingerprint is  :',num2str(posizione_minimo),...
                        %           '    with    a    distance    of    :
',num2str(distanza_minima));
                    else
                         message = strcat('Can not find the same fingerprint
```

```
in the database.');
                    end
                    msgbox(message,'DataBase Info','help');
                else
                    message='DataBase is empty. No check is possible.';
                    msgbox(message,'FingerCode DataBase Error','warn');
                end
            else
                warndlg('Please select a fingerprint image.',' Warning ');
            end
        end

          %----菜单项 Clear DB 的回调函数---
        function[] = clearDB(varargin)

            if (exist('fp_database.dat')==2)
                button = questdlg('Do you really want to remove the Database?');
                if strcmp(button,'Yes')
                    delete('fp_database.dat');
                    msgbox('Database was succesfully removed from the current
directory.','Database removed','help');
                end
            else
                warndlg('Database is empty.',' Warning ');
            end
        end

    end
```

结束语

本设计是基于 matlab 应用程序设计的指纹识别技术的研究。通过在图像识别技术和 matlab 环境下的研究还有各种算法和滤波器的研究以及通过查询资料，询问老师以及和同学的交流完成了此次在软件的设计。

在本次的开发过程中，我意识到最重要的环节在于对图像识别的算法面是至为关键的，虽然算法有很多，但针对指纹图像部分我最终还是选择了 gabor 滤波器，这也是整个计算的核心部分。

同时，在这次的软件设计过程中我也看到了自己在新知识的学习，新工具的应用方面的不足。开始时，我对 matlab 的了解是停留在书面上的了解，还不够深入透彻。本次设计的过程也就成为我对 matlab 的深入透彻学习的过程。正因为这样，所以在设计过程中往往会被一些问题卡住。在原地徘徊了一段时间后，我通过查找参考资料，和同学交流，利用互联网向有经验的编程人员请教等各种方式试着克服在设计过程中遇到的困难。随着开发设计的深入，我也逐渐掌握了一部分设计技巧，了解了一些关于设计方面的规律，一些困难也就迎刃而解了。

总之，本设计对我来说不仅仅是一次毕业设计，这也是我一次有意义的学习、提高的过程。通过这次系统开发，我不但初步掌握了 matlab 开发工具，也让我看到了自己知识结构、知识储备、学习和应用能力上的不足。我想，在今后的学习和工作中，我将针对这些发现的问题，不断努力和提高自身各方面的素质，为社会做出自己应有的贡献。

参考文献

[1] 杜彦蕊，李丹．指纹识别技术探究．中国教育技术装备， 2010，3：61

[2] 杨文杉．指纹识别技术的现状与发展．软件导刊，2008，7（10）：187～188

[3] 汤敏.通用指纹图像处理分析平台的研发与应用.计算机工程师与设计，2010，31（4）：791～794

[4] 魏发建，游敏娟，王保帅等．浅谈指纹识别的基本原理．中国科技信息，2009，10

[5] 王莹,苏成利．指纹图像增强算法．研究科学技术与工程，2010，10（1）：94～98

[6] RAVI．J，K．B．RAJA，VENUGOPAL．K．R．Fingerprint Recognition Using Minutia Score Matching，2009，1（2）：35～42

[7] Yi Wang，Jiankun Hu．A Fingerprint Orientation Model Based on 2D Fourier Expansion (FOMFE) and Its Application to Singular-Point Detection and Fingerprint Indexing．IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE，2007，29（4）：573～585

[8] 太艳荣.基于Matlab实现的指纹图像预处理.西南民族大学学报(自然科学版)，2008，34（4）：836～838

[9] 李惠芳．基于方向图的指纹图像自适应二值化算法研究．北京师范大学学报（自然科学版），2009，45（3）：250～253

[10] 陈倩．基于细节特征的指纹识别算法研究．武汉理工大学硕士论文，2008

[11] 高婧婧．指纹识别预处理算法研究．电子科技大学硕士论文， 2007

[12] 郭晶莹，吴晴，商庆瑞．基于Matlab实现的指纹图像细节特征提取．计算机仿真，2007，24（1）：182～185

[13] 孙玉明,王紫婷.基于Matlab的指纹识别系统的研究与实现.电脑知识与技术,2009，34（5）：9803～9804

[14] Eun-Kyung Yun，Sung-Bae Cho．Adaptive fingerprint image enhancement with fingerprint image quality analysis．Image and Vision Computing，2006：101～110

[15] 田纪亚．基于Matlab在指纹识别系统中的应用研究．吉林大学硕士论文，2008

致　谢