密级：　　保密期限：

北京郵電大學

硕士研究生学位论文

题目：人脸识别相关技术及其嵌入式应用

学　　号：105386

姓　　名：熊金水

专　　业：通信与信息系统

导　　师：赵衍运

学　　院：信息与通信工程学院

2013 年 01 月 15 日

独创性（或创新性）声明

本人声明所呈交的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，除了文中特别加以标注和致谢中所罗列的内容以外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得北京邮电大学或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

申请学位论文与资料若有不实之处，本人承担一切相关责任。

本人签名：_____ 日期：_____

关于论文使用授权的说明

学位论文作者完全了解北京邮电大学有关保留和使用学位论文的规定，即：研究生在校攻读学位期间论文工作的知识产权单位属北京邮电大学。学校有权保留并向国家有关部门或机构送交论文的复印件和磁盘，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，可以允许采用影印、缩印或其它复制手段保存、汇编学位论文。（保密的学位论文在解密后遵守此规定）

保密论文注释：本学位论文属于保密在__年解密后适用本授权书。非保密论文注释：本学位论文不属于保密范围，适用本授权书。

本人签名：_____ 日期：_____

导师签名：_____ 日期：_____

人脸识别相关技术及其嵌入式应用

摘 要

几十年来，计算机视觉、机器学习领域取得了重大进步，从基于规则的学习方法逐渐转变为基于统计学习的方法。另外，随着计算机硬件技术的不断提升和成本的不断降低，越来越多的场合提出了对人脸识别的实际应用需求。例如，门禁系统、各种移动终端的应用、智能监控等。本文深入研究了人脸识别相关技术及其嵌入式应用，主要包括系统的算法设计、算法优化、针对特定平台优化等方面。

具体工作的描述如下：

1. 本文深入研究了 Viola-Jones 人脸检测的各个关键步骤，包括基于 Haar 特征的弱分类器、基于 AdaBoost 算法构造的强分类器和级联结构，并提出了细致的优化策略。实验结果表明，这些优化策略，不仅能有效地提升算法的执行效率，而且能够小幅度的提高检测率。
2. 本文设计了一种人眼检测框架。该框架把人眼检测分为粗定位和细定位两个步骤。粗定位可以使用 Viola-Jones 人眼检测子。细定位可以采用 Viola-Jones 人眼验证子或 mean shift 方法，两种方法各有优劣，分别适用于不同的场合。
3. 本文在深入研究仿射变换的基础上，提出了一种基于相对位置的人脸归一化算法，该算法充分利用了相邻像素间的相关性，使人脸归一化的执行效率得到了极大的提升。
4. 本文针对已有的 Gabor 方向直方图（GOH）的人脸比对算法进行优化，并创造性的把 FFTW 应用于该算法的实现过程中。实验结果表明，FFTW

的引入，使人脸比对的执行时间减少至原来的 18%。

5. 针对嵌入式平台，本文将人脸识别系统的 C++语言实现修改为 C 语言，同时将浮点运算优化至定点运算。另外，针对 Android 系统，本文使用 Android NDK，完成了基于 Android 平台的人脸识别应用。

关键词：人脸检测 AdaBoost 人眼检测 人脸识别 嵌入式系统

FACE RECOGNITION TECHNOLOGY AND ITS EMBEDDED APPLICATIONS

ABSTRACT

In the past several decades, significant progress has been made in the field of computer vision and machine learning. The learning algorithm has been changed from rule-based to statistics-based. What's more, as the rapid increase in computer hardware performance and the rapid decrease in the computer hardware cost, face recognition are being used in more and more applications. For example, access control systems, the application of a variety of mobile terminals, smart monitoring and so on. Based on the face recognition technology, we did intensive research in the field of algorithm design, algorithm optimization, optimization for specific platforms.

Firstly, the key steps of the Viola-Jones face detector, including the Haar-feature-based weak classifier, the AdaBoost-based strong classifier and the attentional cascade are studied. And we optimized the algorithm in detail. The experimental results show the great improvement in the efficiency, and a slight increasement in the detection rate.

Secondly, a framework for eye detection is presented,which divide eye detection into two steps, including coarse detection and fine detection. The coarse detection can

be implemented by Viola-Jones detector. The fine detection can be implemented by either Viola-Jones-based verification or mean shift method. These two different methods are suitable for different scenarios respectively.

Thirdly, after deep-study of the affine transformation, a novel method for face normalization based on correlation between adjacent pixels is proposed. The experimental results show significant improvement in the efficiency of the normalization.

Fourthly, to speed up the proposed Gabor-Orientation-Histogram-based face verification method, FFTW is used in the implementation. Experimental results show that the FFTW-based method increase the processing speed to 18% of the original time.

Fifthly, the implementation of the face recognition in the C++ programming language and the C programming language is completed. And we optimize the calculus from float-point to fixed-point. In addition, by the use of Android NDK, the face recognition system can be invoked by Android applications.

KEY WORDS: face detection, AdaBoost, eye detection, face recognition, embedded system

目录

第一章　引言

近年来，随着我国经济的不断发展，城乡之间的人口流动性越来越大，由此带来的公共安全问题也越来越突出。为此，能否自动鉴别人的身份，也成为科学研究和工业应用的重要问题。如何鉴别人的身份呢？传统的方法往往是通过身份证、学生证等各类卡片来鉴别身份的，而此类方法可行的依据是建立在如下假设之上的：即每个人与每张卡片是一一对应的关系。也就是说，这种鉴别方法，仅仅是鉴别了卡片，而非真实的人，它并不能保证鉴别的真实可靠性，只要犯罪分子等获取了他人的卡片，便可以为非作歹，导致卡片失去了鉴别身份的作用。另一方面，对于卡片持有人而言，忘记携带，丢失，挂失等等一系列问题，也是这种传统方法的缺点之一。因此，一种能够自动针对个体本身的身份鉴别方法亟待提出。

随着计算机硬件技术的不断发展，生物特征识别技术的成本逐渐降低，促进了生物特征识别技术在各个方面的应用。例如，在上文中讲到的公共安全领域，智能监控，银行，机场，政府等重要场所；还有，在智能人机交互领域，人们希望机器能结合人的面部表情、身体姿势和手势等进行人的行为分析和理解等。随着这些领域的不断发展，生物特征识别技术已经成为计算机视觉研究领域的热点和前沿课题。

在生物特征识别技术中，常用的生物特征有：人脸识别、虹膜识别、指纹识别、声音识别等[1]。虹膜和指纹对每个个体而言具有唯一性，但是采集时不方便。近年来，随着统计学习方法带来的人脸识别和声音识别的飞速发展，使人脸识别和声音识别在生物特征识别技术中占据了重要的位置。

1.1 人脸识别概述

人脸识别技术依次包括：人脸检测、人脸归一化、人脸比对三个部分[1]。如图 1-1 所示。

图 1-1　人脸识别概述

对人脸检测，输入为图像或视频，输出为人脸的位置、大小和姿态等。常用的方法有[2]：知识的方法、不变特征的方法、模板匹配的方法和统计学习的方法。各种方法的详细内容将在下一节介绍。

人脸归一化，主要是为人脸识别提供一个更加精确的、归一化的人脸图像。目前主流采用的方法是定位人脸的某一器官的位置，然后依据器官的位置对人脸检测的图像作几何变换。在我们的方法中，我们使用了双眼的位置作为归一化的依据。依据人脸检测的输出，我们如何快速准确的找到人眼的位置，也是一个值得深入研究的问题。

人脸比对，就是对人脸归一化后的图像做特征比对。人脸比对，又称为“半自动人脸（partially automatic face recognition）”，这个名称来源于 FERET 人脸识别库[3]。如果输入包含人脸的图像和人眼坐标，则称为半自动人脸识别；如果输入仅为包含人脸的图像，则称为全自动人脸识别（fully automatic face recognition）。在实际应用中，一般使用的是全自动人脸识别；而早期由于技术水平的限制，同时也是为了更准确的比较不同方法的性能优劣，研究人员常使用半自动人脸识别。

不管是半自动人脸识别，还是全自动人脸识别，都存在两种应用场景[3]：（1）1 对 1 的应用场景，在该场景中，我们只需要判断给定的两幅图像，是否来自于同一个人，输出为一个或真或假的布尔值；（2）1 对 N 的应用场景，在该场景中，给定的输入为一个包含 N 个不同人的人脸库，需要我们判断，给定的某一幅图片，是否来自于这个人脸库的人物，如果是，是哪一个人物。显然，1 对 N 的应用场景要复杂的多。

在 1 对 N 的应用场景中，使用的半自动人脸识别方法莫过于建立在 1 对 1 的基础之上。当前，主要使用的方法如下[4, 5]

1. 基于几何特征的人脸识别[6]

该方法属于早期的人脸识别方法，算法主要利用了眼睛、鼻子、嘴巴、下巴等脸部特征的形状、大小、结构等几何特征的差异，这些几何特征描述出来可以作为人脸识别的重要特征。该方法稳定性不高，识别效果一般。

2. 基于弹性图匹配的人脸识别[7]

该方法来源于图匹配算法。图中的顶点表示人脸中的某个具体部位，这个部位对于识别不同人脸来说起到关键性作用。图中的边表示人脸各部位之间的拓扑关系。该方法考虑了人脸不同部位之间的空间分布信息，但是忽略了人脸从二维空间投影到三维空间引起的变形，所以对人脸的旋转很敏感，而且在匹配时计算量较大。

3. 基于神经网络的人脸识别[8]

该方法主要是利用神经网络的学习能力和分类能力，将其应用在人脸识别领域。但是，在训练时需要设置大量的参数，实现起来比较困难。

4. 隐马尔可夫模型(Hidden Markov Model，HMM)方法[9]

该方法把人脸图像作为观测状态，真实人脸作为不可客观测状态，相同人使用同一个隐马尔科夫模型表示，不同人使用不同参数的隐马尔可夫模型表示。这个方法起源较早，允许人脸表情有较大的变化，头部转动适应性较强。但是在训练模型时计算量较大。

5. 子空间方法

该方法的基本思路是，寻找一个线性或非线性变换，把像素空间映射到子空间，使不同人脸在子空间中有更强的群聚性。这个方法的核心问题是寻找线性或非线性变换，经典的方法有：PCA 方法[10]，LDA 方法[11]，ICA 方法[12]以及核方法[13]等。其中 ICA 方法不但考虑了样本的一阶二阶统计信息，还考虑了高阶统计信息，PCA 方法仅仅考虑了二阶统计信息，故 PCA 可以看做是 ICA 的特例。核方法将线性不可分的样本映射到高维线性可分空间，然后用线性分类方法在核空间中进行分类。

子空间方法计算复杂度低，识别效果好，已成为人脸识别算法的主流。

1.2 人脸检测概述

对人脸检测，输入为图像或视频，输出为人脸的位置、大小和姿态等。随着人脸检测算法在检测精度和检测速度上的不断改进，同时微处理器的运算能力不断地提高，人脸检测越来越多的走向实际应用，例如，全新人机界面、基于内容的检索、数字视频处理、视频监测等许多领域[14]。另一方面，它又是许多其它人脸分析研究课题的基础。如，人脸对齐（face alignment）、人脸验证（face verification/authentication）、姿态跟踪（head pose tracking）、表情识别（ facial expression recognition）、性别年龄识别（ gender/age recognition）等[2]。

人脸检测的研究工作主要面临以下一些难题，不同的尺度的人脸（scale）；不同朝向的人脸，分为 orientation(in-plane rotation)和 pose (out-of-plane rotation)两种； 不同表情的人脸（facial expression）；处在不同光照下的人脸（ lighting conditions），偏光，强光和弱光等；可能存在部分遮挡（occlusions）[2]。

2002 年，Yang[15]对人脸检测的方法做了如下分类，由于其前瞻性而沿用至今。

1. 基于知识的人脸检测[16]

这种方法的关键，是需要定义一个合理的规则，该规则根据人脸独有的特征，以最大限度的区分非人脸。在复杂背景下，定义这样一个规则往往是困难的。

2. 基于不变特征的人脸检测[17]

这种方法的出发点是：计算机能像人一样，找到区分人在不同环境下的不变特征。因此，这种方法的主要任务在于找出这种不变特征。可惜这种方法最终并没有找到一个合适的不变特征。

3. 基于模板匹配的人脸检测[18]

首先定义标准的模板，然后计算检测区域和模板的相关值（相似值）来进行人脸检测，这种方法发展到后来出现了弹性模板，但效果一般。

4. 基于统计学习的人脸检测[15]

许多基于统计学习的方法，一类称为生成模型，可以理解为一个概率模型，

将从图像中获得的特征向量看作随机变量 x，在训练样本上训练出条件概率密度函数 $P(x|face)$ 和 $P(x|nonface)$，通过 Bayes 分类器或者最大似然分类的方法进行人脸检测。另一类称为判决模型，是在人脸和非人脸之间寻找一个判别边界。当前，统计学习的方法主要包括：神经网络的方法，子空间的方法，Boosting 的方法和支撑向量机的方法（Support Vector Machines,SVM）等。

在计算机的计算能力和存储能力得到快速发展的条件下，基于统计学习的人脸检测正逐渐表现出它的优势，尤其是 Boosting 方法，不论是在检测的精度上还是检测速度上，都有良好的表现[19]。

1.3 人眼检测概述

在本文中，我们只讨论输入为图像的人眼检测，并不涉及视频输入。人脸检测的输出为人眼检测提供一个人脸区域，人眼检测的目的就是在这个区域内定位眼睛的精确位置，分为常光源下的人眼检测和特种光源下的人眼检测。特种光源下的人眼检测需要凭借特殊光源照射人眼，如红外线，因此，对系统的硬件有较高要求。常光源下的人眼检测使用自然光照射下取得的图像，应用广泛，是人眼检测的研究重点。常光源下的人眼检测常用的方法有[20]：

1. 模板匹配的方法[21]

分为几何模板匹配和灰度投影。几何模板匹配，利用人眼的几何信息，构造人眼模板，该方法需要较多的先验知识。灰度投影将人眼图像投影至水平和竖直两个方向，并分别找到一个特定变化的点，最终确定人眼的横纵坐标，该方法容易受到眉毛等因素的干扰。

2. 统计的方法[22]

首先确定一个基于统计学习的分类模型，然后用大量的人眼样本和非人眼样本训练，得到模型参数，最后使用训练好的模型预测测试样本。该方法检测率高，但计算复杂度高。

3. 知识的方法[23]

模板匹配的方法利用人眼的几何信息构造模板，实施人眼检测。知识的方法

直接利用这些几何信息实施人眼检测，算法简单，易实现，但由于效果一般，常作为辅助检测方法。

由于统计的方法在检测精度上远远高于模板匹配和知识的方法，因此已成为人眼检测的研究重点。

1.4 结构和安排

论文总共分为五个部分，具体研究内容及组织安排如下：

第一部分为引言，介绍相关的研究领域，以及不同领域里使用的不同方法。最后给出论文的结构安排。

第二部分为人脸检测，介绍了经典的 Viola-Jones 人脸检测算法，包括特征的快速计算，分类器以及一种级联结构。并给出了自己在实现中使用的一些细致的优化技巧，用于提高算法的性能，最后给出了实验结果。

第三部分为人眼检测与人脸归一化，提出了一种人眼检测框架，并详细介绍了人眼检测的方法，最后给出了人眼检测和人脸归一化的实验结果。

第四部分为人脸识别系统实现及优化，首先介绍整个系统的模块划分，其次介绍了人脸比对的优化，最后介绍了针对嵌入式平台，将 C++版本修改为 C 版本，浮点运算修改为定点运算的方法。

第五部分为总论与展望，总结了硕士期间所作的工作，并对未来可能的研究方向做出了预测。

第二章　人脸检测

人脸检测是人脸识别系统的第一步，它的可靠性直接决定着整个系统的性能。给定一幅静态图像或者一段视频，通过一个理想的人脸检测算法，应该能够成功的判断其中是否存在人脸，以及人脸的位置，而不管其中人脸的尺度、方向、年龄、表情、光照等如何变化。

人脸检测的实现，通常从下面几个角度出发：皮肤的颜色（主要是指彩色图像或彩色视频）[24]、脸部动作（仅适用于视频）、脸部的形状、脸部的表观（appearance），或者同时结合以上几种角度。当前最成功的人脸检测算法，大部分仅仅从表观出发，而未考虑其它角度。

这些成功的人脸检测算法，都遵循这样的思路：对于给定的测试图像（或者视频中的一帧），根据不同的尺度，构成一座图像金字塔；对于金字塔每一层，遍历其中所有可能存在人脸的子窗口，将这些子窗口（或者从这些子窗口提取的特征），作为一个模式输入到一个分类器中，这样，人脸检测问题便成为一个二分类问题。这里常用的分类器，第一种是神经网络[25]，第二种是基于核函数的方法，主要应用是支持向量机（SVM-Support Vector Machine）[26]，第三种方法是 AdaBoost（Adaptive Boosting）[27]。基于 AdaBoost 的方法，不管是在检测的精度还是检测的实时性，不管是在正面人脸图像，还是侧面人脸图像上，都优于其它两种方法。因此，我们的系统选择了 AdaBoost 方法。

本章将着重以 AdaBoost 方法为主，将 AdaBoost 方法在人脸检测领域的应用做详细的介绍。前三节主要介绍 Viola-Jones 人脸监测子的三个关键的贡献，第四节将重点介绍我们在算法实现时做了哪些优化，第五节给出了实验结果。

2.1 Viola-Jones 人脸检测算法概述

我们回顾 21 世纪最重要的人脸检测算法，Paul Viola 和 Michael Jones 于 2001 年提出的算法恐怕[27]。该算法结合了 Haar 特征、AdaBoost 分类器和级联结构，

是目前公认的最快速、最准确的人脸检测算法，其它人脸检测算法大抵是这个方法的扩展。

首先，Viola-Jones 人脸检测子提出了一种新的图像描述方法——积分图，这种表示方法能够加速特征的计算速度。Viola 和 Jones 声称他们的灵感来源于 Papageorgiou 等人[28]，Papageorgiou 等人在原始图像的基础上，使用 Haar 基函数作为滤波器，得到图像的特征。与之相似，Viola 和 Jones 也使用相关的滤波器，但是比 Haar 基函数更复杂，不仅如此，为了在不同尺度空间上，能够尽可能的提高特征提取的速度，他们提出了使用积分图表征图像的方法。若原始图像的像素总数为 N，则可以在 O(N)的时间复杂度内计算出积分图，并同时使用 O(N)的空间存储积分图。而一旦计算好了积分图，任何尺度的 Haar 特征，均可以在常数时间 O(1)内计算得到。

其次，Viola-Jones 人脸检测子使用 AdaBoost(Adaptive Boosting)算法构造分类器。对于给定的图像，我们知道，为了进行人脸检测，我们的第一步工作就是把原始图像扩展到不同尺度的图像金字塔空间中，而对于金字塔空间中的某一个尺度，我们又需要扫描很多的子窗口，同时，在这些子窗口中将有非常密集的 Haar 特征需要提取，这样下来，我们知道总的 Haar 特征的数量将等于不同尺度的数量，乘以不同尺度下子窗口的平均数量（之所以说平均，是因为不同尺度下子窗口的数量并不相同），再乘以一个子窗口中可以计算的 Haar 特征的数量，这个数量将是非常庞大的！为了快速执行分类器算法，我们不可能把所有特征都加入到分类器中，而必须剔除掉其中的绝大部分 Haar 特征，以便将分类器集中在具有较强分类效果的 Haar 特征上。因此，Viola 和 Jones 使用了 AdaBoost 算法的一个简单的变更，让每一个弱分类器的构造都仅仅来源于一个单独的 Haar 特征，这样，AdaBoost 的每一次迭代，本质上是从众多的 Haar 特征中选择了一个具有较强分类效果，而 AdaBoost 算法执行多少次迭代，就意味着多少个 Haar 特征被加入到分类器算法中。这样，AdaBoost 算法不仅达到分类的效果，也间接地完成了特征选择的过程。

最后，Viola-Jones 人脸检测子使用了一种级联结构，用于进一步加速算法的

执行速度。在级联结构中，每一级都是一个完整的 AdaBoost 分类器，而且，AdaBoost 分类器中弱分类器的个数，随着级数的增加而增加。，在级联结构中的任何一级，被分类为非人脸的子窗口，将直接被判定为非人脸，而判定为人脸的子窗口，将需要经过后续更复杂的 AdaBoost 分类器，进一步确认该子窗口是否为人脸。这样的机制之所以有效，是因为人脸检测的任务中，子窗口绝大部分为非人脸，而这些非人脸在级数较少时就可以被淘汰，避免了在后面的更复杂的 AdaBoost 分类器中的计算，这样也就把整个检测子的计算重心放在了人脸的可疑位置。

2.2 Haar 特征及其快速计算

Viola-Jones 人脸检测子提出了四种基本的 Haar 特征[27]，如图 2-1 所示。

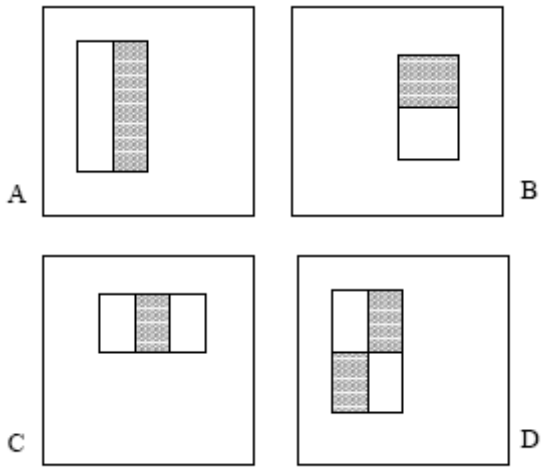


图 2-1　Viola-Jones 最初提出的四种 Haar 特征。计算方法为：黑色矩形中像素值的总和，减去白色矩形中像素值的总和。其中 A、B、C 为两个矩形的 Haar 特征，D 为三个矩形的 Haar 特征。

这些 Haar 特征块，被放置在子窗口(sub window)中的不同位置，而可以使用不同的尺度，最小的尺度可以小到一个 Haar 特征块的局部（如图 2-1 中的黑色部分）只有一个像素，最大的尺度可以达到整个子窗口即为一个 Haar 特征块。当然，并不是所有的尺度都可以大到一个子串口那么大，而与他们的中心所处的

位置有关，最大的时候 Haar 块的边缘应该处于子窗口的边缘。

对于一个 24×24 的子窗口，这些 Haar 特征块的数量能够达到 180 000 多个[27]。我们知道，对于 24×24 的灰度图像而言，总共有 24×24=576 个非线性相关的标量值，而我们知道，每一个 Haar 特征块对应一个标量，故整个 Haar 特征空间中，将有 180 000 多个标量值，是原始非线性相关标量值的 312 倍。因此，整个 Haar 特征空间中，必然存在这大量的线性相关的特征。

为什么选择这些矩形特征块？这些简单的矩形块又如何能够构造弱分类器并通过 AdaBoost 结合起来，构成至今为止最出色的人脸检测子呢？通过下面的图示我们就能很清楚的知道了。

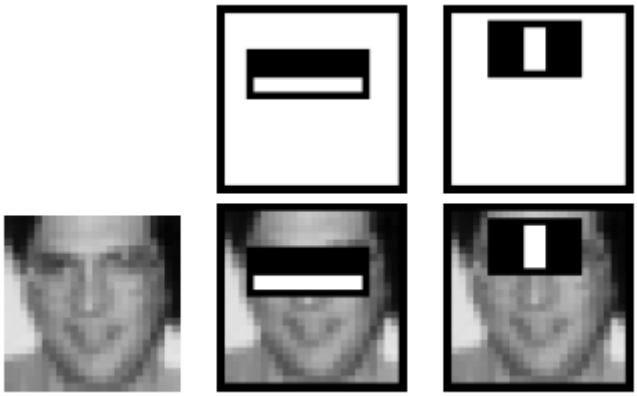


图 2-2　矩形特征块的直观解释

通过图 2-2 的直观解释，我们知道，Haar 特征块看似简单，却能很好的刻画人脸的局部特征，诸如：双眼、双眉、鼻子、嘴巴等。例如，通常双眼所在的位置像素值都较小，而双眼之间的鼻梁，像素的灰度值又较大，经过图 2-1 中 C 类 Haar 特征块的滤波之后，能够得到较大的输出值。而双眼这样的局部特征，在非人脸图像中又很少出现，因此，这样的 Haar 特征块能够非常好的描述人脸。

知道了 Haar 特征块的原理以及直观解释，下面我们重点介绍 Haar 矩形的计算问题。

设原始图像为 I，高度和宽度分别为 H 和 W，处在第 i 行、第 j 列的像素灰度值为 $I(i, j)$。接下来，我们构造一个 $(H+1)\cdot(W+1)$ 的积分图像 II，第一列和第

一行的所有值均置为 0。积分图的第 i 行、第 j 列的元素值 $II(i, j)$，可以通过公式 2-1 计算。

$$\mathrm{II(i, j)} = \sum_{i'<=i, j'<=j} I(i', j') \qquad (2\text{-}1)$$

通过使用下面的公式 2-2 和公式 2-3，我们可以遍历一遍图像的所有像素，就可以得到积分图像 II。

$$S(i, j) = S(i, j-1) + I(i, j) \qquad (2\text{-}2)$$

$$II(i, j) = II(i-1, j) + S(i, j) \qquad (2\text{-}3)$$

其中，$S(i, j)$ 表示第 i 行第 1 列至第 i 行第 j 列 j 个像素值之和。

有了积分图，我们就可以快速的计算出图 2-1 中列举的四个 Haar 特征块的值了。如图 2-3 所示，我们计算下图中矩形 D 内所有像素的灰度值的和为：D = (A + B + C + D) – (A + B) – (A + C) + A = 4 – 3 – 2 + 1。把这个简单的原理应用在我们前面的四个基本的 Haar 特征块时，我们就得到了图 2-4 的结果。

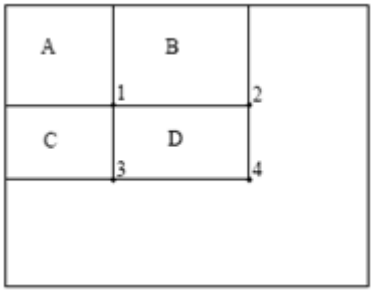


图 2-3　利用积分图计算矩形 D 内所有像素的灰度值之和

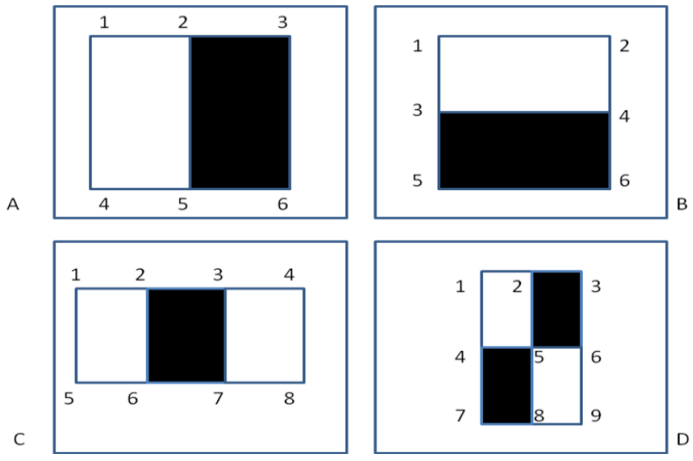


图 2-4　利用积分图计算 Haar 特征

1. Haar 特征块 A，特征值为：(6 – 5 – 3 + 2) – (5 – 4 – 2 + 1) = 6 – 2 * 5 + 4 – 3 – 2 * 2 – 1 ；
2. Haar 特征块 B，特征值为：(6 – 5 – 4 + 3) – (4 – 3 – 2 + 1) = 6 – 5 – 2 * 4 + 2 * 3 + 2 – 1；
3. Haar 特征块 C，特征值为：2 * (7 – 6 – 3 + 2) – (8 – 5 – 4 + 1) = -8 + 2*7 – 2 * 6 + 5 + 4 – 2 * 3 + 2 * 2 – 1；
4. Haar 特征块 D，特征值为：2 * (8 – 7 – 5 + 4) + 2 * (6 – 5 – 3 + 2) – (9 – 7 – 3 + 1) = -9 + 2 * 8 – 7 + 2 * 6 – 4 * 5 + 2 * 4 – 3 + 2 * 2 – 1。

最后，我们简要的介绍扩展的 Haar 特征[2]。这些特征主要应用于非正面人脸检测，包括纸面内的旋转和纸面外的旋转。为了方便快速计算，这些扩展的 Haar 特征块，常采用 45° 的旋转，如图 2-5 所示。

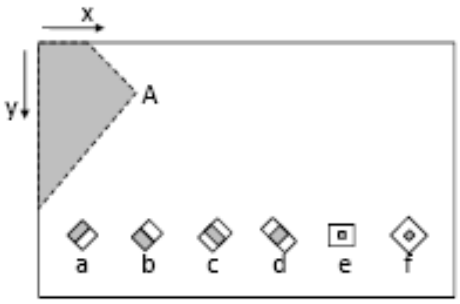


图 2-5　扩展的 Haar 特征块

2.2 Adaboost 算法

对于 AdaBoost 算法，最常用的是离散 AdaBoost(Discrete AdaBoost)算法和连续 AdaBoost(Real AdaBoost)算法[29]。Viola-Jones 人脸检测子采用的是离散 AdaBoost 算法，而在我们的系统中，使用了分类精度更高的连续 AdaBoost 算法。在这一节中，我们将从离散 AdaBoost 算法出发，以连续 AdaBoost 算法为主，详细的介绍 AdaBoost 算法。

2.2.1 离散 AdaBoost 算法

在二分类问题中，假设我们有训练样本 $(x_1, y_1),...(x_N, y_N)$，其中，x_i 表示第 i

个训练样本的特征向量，$y_i \in \{-1,+1\}$ 表示第 i 个训练样本所属的类别标签。在离散 AdaBoost 算法中，定义强分类器为：

$$F(x)=\sum_{t=1}^{T}\alpha_t * f_t(x)-b \qquad (2\text{-}4)$$

其中，$f_t(x)$ 为弱分类器，输出值为-1 或+1，α_t 为标量权值，表示弱分类器 $f_t(x)$ 在整个强分类器 $F(x)$ 中所占的比重，b 是一个可以调整的参数，用于权衡检测率和误检率。最终强分类器的预测结果为 sign(F(x))[29]。

一般化的 AdaBoost 算法假定，我们已经有一种机制 P，能够在带权样本上，训练一个最小化带权错误率的分类器 $f(x)\in\{-1,+1\}$[30]。使用机制 P，AdaBoost 算法在训练 $f_t(x)$ 后，给每个样本一个权值，当前错误分类的样本权值较大，正确分类的样本权值较小，并在此基础上进一步训练下一个弱分类器。最终，强分类器由这些弱分类器加权求和而成。

详细的离散 AdaBoost 算法如下：

算法 2-1：离散 AdaBoost 算法[30]

输入：训练样本集 S={$(x_1,y_1),...(x_N,y_N)$}，弱分类器空间 ~~H~~，其中，x_i 表示第 i 个样本的特征向量，$y_i \in \{-1,+1\}$ 表示类别标签，N 为样本个数。

步骤：

初始化样本的权值$\{w_i^{(1)}\}$，$w_i^{(1)}$ = 1/N，i=1,2,…N

Repeat for t = 1, 2, … T:

1) 使用机制 P，在样本权值$\{w_i^{(t)}\}$的基础上，通过最小化带权错误率：

$$err_t=\sum_{i=1}^{N} w_i^{(t)} I(f_t(x_i)\neq y_i) \qquad (2\text{-}5)$$

训练得到弱分类器 $f_t(x)$。其中，$I(f_t(x_i)\neq y_i)$ 为指示函数，当 f_t 正确预测时，函数值为 0；错误预测时，函数值为 1。

2) 计算弱分类器权值：

$$\alpha_t = \ln(\frac{1-err_t}{err_t}) \tag{2-6}$$

3) 更新样本权值：

$$w_i^{(t+1)} = w_i^{(t)}(\frac{err_t}{1-err_t})^{1-I(f_t(x_i)\neq y_i)} \tag{2-7}$$

并归一化为：

$$w_i^{(t+1)} = \frac{w_i^{(t+1)}}{\sum_{j=1}^{N} w_j^{(t+1)}} \tag{2-8}$$

输出：强分类器：

$$F(x) = sign(\sum_{t=1}^{T} \alpha_t * f_t(x) - b) \tag{2-9}$$

其中，x 为测试样本的特征向量。b 为一个可调参数，用于平衡系统中的检测率和误检率。sign(a)为符号函数，当 a>0 时，结果为 1，当 a<0 时，结果为-1。

观察公式 2-6，我们发现，弱分类器 $f_t(x)$ 越优越，即 err_t 越小，则 α_t 越大。也就是说，性能优越的弱分类器，在强分类器中比重更大。这也是很符合我们直观感受的规律。

从上面的算法 2-1 我们还可以看到，第一个弱分类器训练的时候，所有训练样本的权值是相同的，均为 1/N，这与普通的训练分类器的方法相似。从公式 2-7 和 2-8 我们发现，在后续的迭代过程中，样本权值不断的发生变化。我们知道，对于弱分类器 $f_t(x)$，其带权错误率小于 0.5（否则这将是一个逊于随机猜测的弱分类器），因此公式 2-7 中幂底 $(\frac{err_t}{1-err_t})$ <1。而对于正确分类的样本，公式 2-7 中的幂指 $1-I(f_t(x_i)\neq y_i)$ 为 1，则该正确分类的样本权值将减小；而对于错误分类的样本，幂指为 0，权值暂且不变。但是，我们发现，所有的权值要么减小，要

么不变，那么在最后，这些样本的权值总和将小于 1，也就是说，经过 2-8 的归一化处理，错误分类的样本权值将增加。这样，在下一次迭代时，更多的权值将放在前一次迭代错误分类的样本上。

Christopher M. Bishop 给出了离散 AdaBoost 的流程图[30]，如图 2-6 所示。图中的箭头所指方向，表征了整个算法的运行过程。

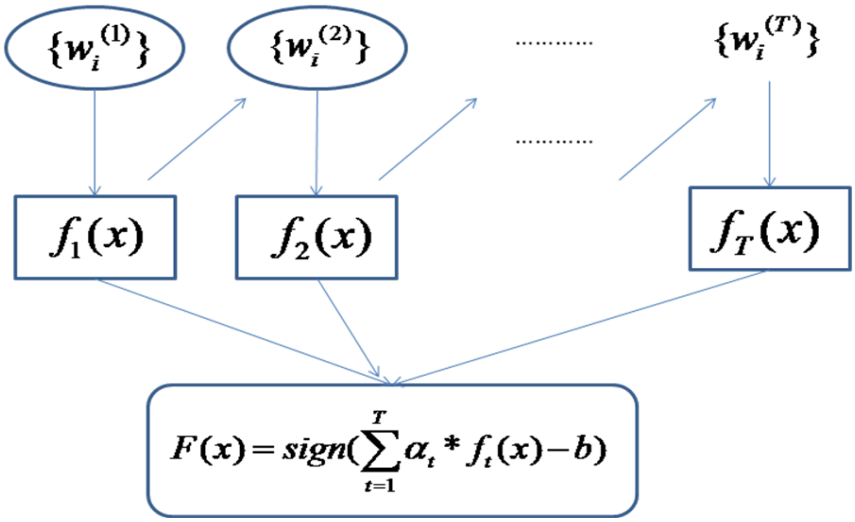


图 2-6　图解离散 AdaBoost 算法[30]

实践证明，AdaBoost 算法能够成功的生成精确的分类器。已经有越来越多的研究人员使用 AdaBoost 算法设计分类器，使用决策树作为 AdaBoost 算法中的弱分类器（事实上，Viola-Jones 人脸检测子中的弱分类器就是只有一个节点的决策树）。通过 AdaBoost 算法构造的强分类器，其性能总是能够显著的优于由单一决策树构造的分类器。Breiman 甚至直言赞扬基于决策树的 AdaBoost 算法是“世界上最好的现成算法（best off-the-shelf classifier in the world）[29]”。更加值得注意的是，在许多实例中，随着弱分类器的不断加入，整个强分类的分类性能持续上升，最后趋于平稳。这说明，存在着某些原因，使 AdaBoost 算法先天具有抵御过拟合（over fitting）的能力[29]。

2.2.2 连续 AdaBoost 算法

上面，我们介绍了离散 AdaBoost 算法的基本原理，在 Viola-Jones 人脸检测子中，使用的学习算法就是离散 AdaBoost。然而，由于连续 AdaBoost 在算法上

更优的性能，我们的系统采用了连续 AdaBoost 算法。当然，知道了离散 AdaBoost 算法，进一步理解连续 AdaBoost 算法也就变得水到渠成了（事实上，所有的 AdaBoost 算法的框架都相同，只是有些细微差别）。

连续 AdaBoost 算法首先由 Freund 和 Schapire 于 1996 年提出[31]，并由 Schapire 和 Singer 于 1998 年完善[32]。连续 AdaBoost 算法中弱分类器的输出不再是类别标签，而是一个实数。即对于特征向量 x，弱分类器构成一个映射 $f_t(x):\chi\mapsto R$，其中 χ 为特征空间。$f_t(x)$ 的正负符号表征了类别，对应于离散 AdaBoost 里的弱分类器；绝对值 $|f_t(x)|$ 表征了分类的置信度，对应于离散 AdaBoost 里的弱分类器权值。

在连续 AdaBoost 算法中，弱分类的构造方法采用类概率密度估计的方法[29]。对于正样本，类概率估计为：

$$p_t(x)=\hat{P}_w(y=1|x)\in[0,1] \qquad (2\text{-}10)$$

弱分类器就是由类概率估计值的对数变换构造获得。

算法 2-1：连续 AdaBoost 算法[29]：

输入：训练样本集 S={$(x_1,y_1),...(x_N,y_N)$}，弱分类器空间 ~~$\mathcal{H}$~~，其中，x_i 表示第 i 个样本的特征向量，$y_i\in\{-1,+1\}$ 表示类别标签，N 为样本个数。

步骤：

初始化样本的权值$\{w_i^{(1)}\}$，$w_i^{(1)}$ = 1/N，i=1,2,…N

Repeat for t = 1, 2, … T

1) 在样本权值$\{w_i^{(t)}\}$的基础上，估计类密度：

$$p_t(x)=\hat{P}_w(y=1|x)\in[0,1] \qquad (2\text{-}10)$$

2) 在类密度的基础上，构造弱分类器：

$$f_t(x)=\frac{1}{2}\log\frac{p_t(x)}{1-p_t(x)} \qquad (2\text{-}11)$$

3) 更新样本权值：

$$w_i^{(t+1)} = w_i^{(t)} \exp[-y_i f_t(x_i)] \qquad (2\text{-}12)$$

并归一化为：

$$w_i^{(t+1)} = \frac{w_i^{(t+1)}}{\sum_{j=1}^{N} w_j^{(t+1)}} \qquad (2\text{-}13)$$

输出：强分类器：

$$F(x) = sign(\sum_{t=1}^{T} f_t(x) - b) \qquad (2\text{-}14)$$

其中，x 为测试样本的特征向量。b 为一个可调参数，用于平衡系统中的检测率和误检率。sign(a)为符号函数，当 a>0 时，结果为 1，当 a<0 时，结果为-1。

连续 AdaBoost 和离散 AdaBoost 最关键的不同之处，在于公式 2-10 和公式 2-11 所反映的弱分类器的构造方法上。在公式 2-10 中，类概率 $p_t(x) = \hat{P}_w(y = 1 | x \in)$ [（注意：这里不是 $\hat{P}_w(x | y = 1)$）所反映的是特征值为 x 的样本，属于正样本的概率。假如正样本中，特征值为 x 的样本有 n_+ 个，负样本中，特征值为 x 的样本有 n_- 个，那么，$p_t(x) = \frac{n_+}{n_+ + n_-}$。如果 $n_+ > n_-$，那么，$p_t(x)$>0.5，这时，弱分类器 $f_t(x)$>0，且 $\frac{n_+}{n_-}$ 越大，$|f_t(x)|$ 也越大，这也就是说，样本中特征值为 x 的样本越多，特征值为 x 的测试样本被分类为正样本的置信度就越高，这是非常符合我们直观感受的。反之，当 $n_+ < n_-$，也有着同样的直观感受。

与离散 AdaBoost 类似，初始的时候，所有样本的权值相同，这意味着，在训练弱分类器 $f_1(x)$ 时，所有样本被视为同等重要。在后续迭代过程中，从公式 2-12 和公式 2-13 我们看到，错误分类的样本权值增加，正确分类的样本权值减小，也就是说，训练其它弱分类器 $f_t(x_i)$ 时，更加重视弱分类器 $f_{t-1}(x)$ 未能正确分类的那些样本。

2.2.3 离散 AdaBoost 算法和连续 AdaBoost 算法之比较

J. Friedman、T. Hastie 和 R. Tibshirani 曾经做过一个有趣的实验[29]，使用 CART（Classification And Regression Tree）决策树算法[33]构造弱分类器，比较了离散 AdaBoost 算法和连续 AdaBoost 算法在二分类问题中的性能差异，结果如图 2-7 所示。值得注意的是，试验中所有的训练样本和测试样本均为人工生成，总共 2000 个，每个样本特征包含 10 维数据。纵坐标表示预测的错误率，横坐标表示 AdaBoost 算法中迭代的次数，0 表示只有一棵 CART 树。

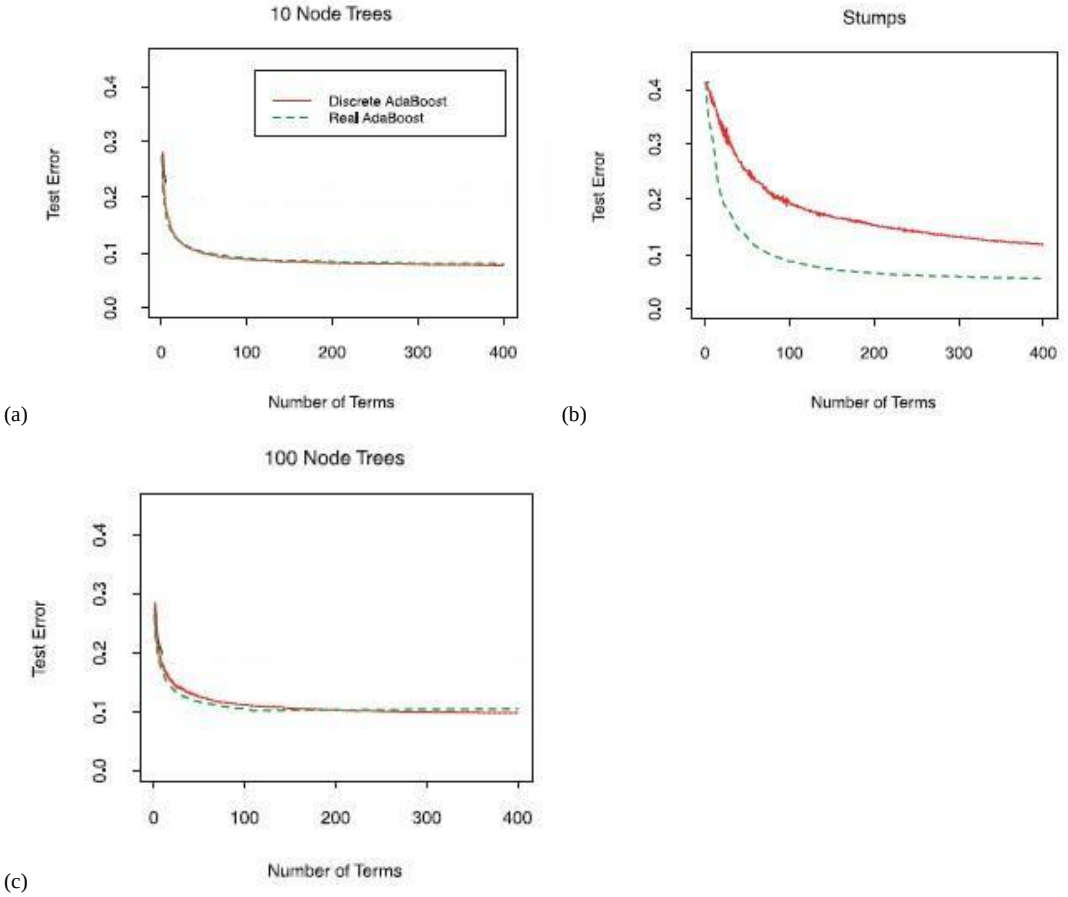


图 2-7　离散 AdaBoost 算法和连续 AdaBoost 算法之比较

（a）弱分类器使用 10 个节点的 CART；（b）弱分类器使用 1 个节点的 CART；（c）弱分类器使用 100 个节点的 CART[29]

从图中我们可以看到，当弱分类器中 CART 树的节点个数为 10 时，离散 AdaBoost 和连续 AdaBoost 有着相似的性能；而当弱分类器中 CART 树中节点只有一个时，离散 AdaBoost 算法明显逊于连续 AdaBoost 算法，而且离散 AdaBoost

算法在迭代了 400 次以后仍然没有收敛的趋势，而连续 AdaBoost 在迭代了 100 次以后已经基本收敛了；当弱分类器中 CART 树的节点个数为 100 时，当迭代次数小于 200 时，连续 AdaBoost 算法要稍微好一些，当迭代次数大于 200 时，离散 AdaBoost 算法又要稍微好一些，但是，两者差别并不大。

因此，综合考虑，使用 CART 作为弱分类器的构造方法时，连续 AdaBoost 算法的性能要优于离散 AdaBoost。而如果使用其它的方法构造弱分类器，孰优孰劣，并没有一个准确的说法。

2.3 级联结构及其优势

本小节介绍一种结构，该结构能够提高算法的性能，而且能够极大的提高算法的执行效率。首先，我们够构造一个非常简单的强分类器 $F(x)=sign(\sum_{t=1}^{T} f_t(x)-b)$，所谓简单，在于该分类器中含有的弱分类器非常少，一般 1 个或 2 个，这样一来，该分类器的计算效率必然非常高。另一方面，我们调整参数 b 使该分类器有非常高的检测率（TPR-True Positive Rate），几乎不会把人脸子窗口分类为非人脸，同时又具有一个较低的误检率（FPR-False Positive Rate），可以把非人脸的子窗口分类为非人脸，这样，进入下一级的将是所有子窗口的一部分。级联结构示意图如图 2-8 所示。

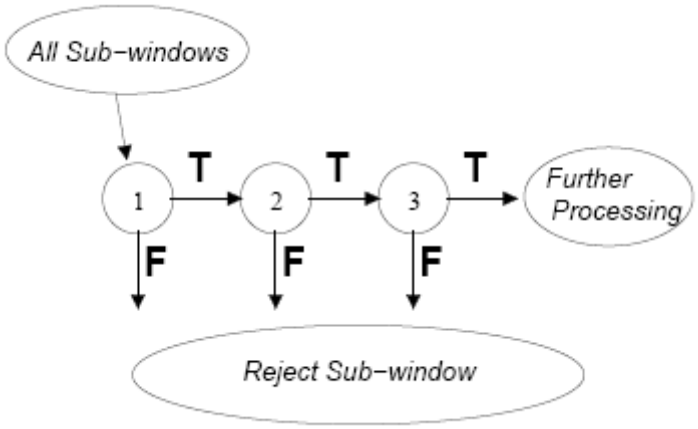


图 2-8　Viola-Jones 人脸检测子级联结构

从图 2-8 中我们可以发现，级联结构本质上是一棵退化的决策树。当某一级

连续 AdaBoost 算法判定子窗口为非人脸时，则该子窗口将被判定为非人脸，而且不会进入下一级检验而直接返回。当某一级连续 AdaBoost 算法判定子窗口为人脸时，并不会立即返回该判定结果，而是将该子窗口送入下一级连续 AdaBoost 算法，作进一步的分类判定。我们知道，经过前一级的处理，进入后一级判定的子窗口的分类判定工作会变得更加困难，需要更多的弱分类器的组合。有了这样的级联结构，绝大部分非人脸将只经过级联结构中的少数几级，而只有人脸或者非常类似人脸的子窗口才需要经过所有级，这样，也就使整个算法的执行效率得到了极大的提升。

该结构之所以有效，是基于这样一个事实，在任意静态图像或视频中，绝大部子窗口为非人脸。因此，级联结构能够使用少量的级数，尽早的将绝大部分非人脸子窗口判断为非人脸并排除。与此同时，少量的人脸子窗口会经过所有的级数，并最终被分类判定为人脸。

级联结构在训练时，需要特别的注意。在前面我们曾经提到，级联结构是一种退化的决策树模型。而决策树模型训练时，非根节点的训练样本来自于通过了所有父节点的样本。同样，在级联结构中，每一级的训练样本也需要经过前面级数并被判定为人脸。这是级联结构在训练中关于样本来源的考虑。具体到整个算法本身，训练需要综合考虑下面的各个参数。

第一，级联结构的级数，即节点数 M，如何确定。

我们知道，每一级本身是一个强分类器，而这一个强分类器需要满足怎样的性能，即检测率（TPR-True Positive Rate）和误检率（FPR-False Positive Rate）需要达到怎样的标准；

第二，每一级强分类器中弱分类器的数量 T 如何调整。

在大部分情况下， 强分类器中的弱分类器数量越多，那么这个强分类器的检测率 TPR 就越高，相应的误检率 FPR 越低。然而，弱分类器数量的增加也会带来计算成本的增加，因此，确定弱分类器的个数对算法整体性能有着举足轻重的作用；每一级的强分类器中的可调参数 b 如何调整。在我们的系统实现中，我

们为每一级设定一个目标值，其中第 i 级的目标检测率为$\mathrm{TPR}_i^{(target)}$，目标误检率为$\mathrm{FPR}_i^{(target)}$，则整个 Viola-Jones 人脸检测子的目标检测率为：

$$TPR^{(target)} = \sum_{i=1}^{M} \mathrm{TPR}_i^{(target)} \tag{2-15}$$

其中,M 表示级联结构中的节点数。整个 Viola-Jones 人脸检测子的目标误检率为：

$$\mathrm{FPR}^{(target)} = \sum_{i=1}^{M} \mathrm{FPR}_i^{(target)} \tag{2-16}$$

由于每一级的目标检测率$\mathrm{TPR}_i^{(target)}$都非常接近于 100%，实际上，其中很多级的目标检测率$\mathrm{TPR}_i^{(target)}$我们直接设定为 100%，因此最终的目标检测率$TPR^{(target)}$非常接近于 100%。而对于整体目标误检率$\mathrm{FPR}^{(target)}$，我们希望它越小越好，而这并不意味着我们要求每一级的目标误检率$\mathrm{FPR}_i^{(target)}$都非常小。实际上，如果我们设定每一级的误检率$\mathrm{FPR}_i^{(target)}$为 50%，如果级联结构一共有 10 级，即 M=10，则最终的目标误检率仅仅只有$\mathrm{FPR}^{(target)} = 0.5^{10} = 0.098\%$。

有了每一级的目标检测率$\mathrm{TPR}_i^{(target)}$和目标误检率$\mathrm{FPR}_i^{(target)}$之后，我们开展对每一级的训练。每次，我们加入一个弱分类器$f_t(x)$，调整参数 b，使强分类器的实际检测率$TPR_i^{(actual)}$大于等于目标检测率$\mathrm{TPR}_i^{(target)}$。而如果此时的实际误检率$\mathrm{FPR}_i^{(actual)}$小于目标误检率$\mathrm{FPR}_i^{(target)}$，则停止对该级的训练；如果实际误检率大于目标误检率，则需要继续添加弱分类器，直到实际检测率$TPR_i^{(actual)}$和实际误检率$\mathrm{FPR}_i^{(actual)}$均满足要求为止。

级联结构的加入，使算法的评估变得稍微有些不同。

对于人脸检测，常用的算法评估方法为 ROC（Receiver Operating Characteristic）曲线[34]。一个 ROC 曲线的横坐标为误检率 FPR（False Positive Rate），取值范围为 0.0 到 1.0，纵坐标为检测率 TPR（True Positive Rate），取值范围为 0.0 到 1.0。为了画出完整的 ROC 曲线，我们需要调整分类器的参数，获得 0.0 到 1.0

之间的误检率 FPR，并获得相应检测率 TPR，这样得到一系列的点，并连接这些点（曲线或折现均可），得到最终的 ROC 曲线。点（0.0,1.0），即左上角的点，构成一个“完美分类器”，越接近该点，说明分类器的性能越好。然而，实际上，并不存在“完美分类器”，因此，在衡量分类器性能时，往往采用 ROC 曲线的右下方区域的面积作为衡量指标，该区域面积越大，认为该分类器越好，否则认为分类器性能较差。

当我们需要画出 Viola-Jones 人脸检测子的 ROC 曲线时，我们首先调整级联结构最后一级的阈值 b。当 b 从 $-\infty$ 到 $+\infty$ 变化时，记录对应的检测率 TPR 和误检率 FPR。当 b 为 $-\infty$ 时，所有进入最后一级的测试子窗口，都将被分类为人脸，因此，该级的检测率 $TPR_M^{(actual)}$ 为 1.0，此时整个检测子的检测率 $TPR^{(actual)}$ 为 $\sum_{i=1}^{M-1}\mathrm{FPR}_i^{(actual)}$；该级的误检率 $\mathrm{FPR}_M^{(actual)}$ 为 1.0，整个检测子的误检率 $\mathrm{FPR}^{(actual)}$ 为 $\sum_{i=1}^{M-1}\mathrm{TPR}_i^{(actual)}$。我们发现，当 b 为 $-\infty$ 时，该级对检测率和误检率而言，并未产生影响，即等效于从整个分类器上移除了最后一级强分类器。当 b 为 $+\infty$ 时，所有进入最后一级的测试子窗口，都将被分类为非人脸，因此，该级的检测率 $TPR_M^{(actual)}$ 为 0.0，此时整个检测子的检测率 $TPR^{(actual)}$ 为 0.0；该级的误检率 $\mathrm{FPR}_M^{(actual)}$ 为 0.0，整个检测子的误检率 $\mathrm{FPR}^{(actual)}$ 为 0.0。这样，通过调整最后一级的阈值 b，我们可以画出误检率从 0.0 到 $\sum_{i=1}^{M-1}\mathrm{FPR}_i^{(actual)}$ 内的一段 ROC 曲线，为了画出检测率大于 $\sum_{i=1}^{M-1}\mathrm{FPR}_i^{(actual)}$ 且小于 $\sum_{i=1}^{M-2}\mathrm{TPR}_i^{(actual)}$ 的 ROC 曲线，我们需要移除最后一级强分类器，并减小倒数第二级的阈值 b。依次类推，直至第一级强分类器，则可以画出整个 ROC 曲线。

2.4 算法实现及优化

在系统实现中，我们需要更进一步考虑更加细致的内容。

2.4.1 光照预处理

第一种，照度梯度修正（illumination gradient correction）。为了消除不均匀的光照的干扰，需要对原始图像进行照度梯度修正。对于输入图像，设其包含 N 个像素，各个像素的灰度值 $I(x_i, y_i), i = 1, 2...N$。对整个图像拟合出一个校正平面，假设该平面表达式为 $z = a_1 x + a_2 y + a_3$，则修正后的像素灰度值为原始灰度值减去校正平面的值。因此，照度梯度修正的关键在于获得校正平面，常用的方法为最小化校正平面和原始灰度图像的均方误差。即：

$$(a_1^*, a_2^*, a_3^*) = \arg\min_{(a_1,a_2,a_3)} \{\sum_{i=1}^{N} [a_1 x_i + a_2 y_i + a_3 - I(x_i, y_i)]^2\} \quad (2\text{-}17)$$

公式 2-16 是典型的凸优化问题，可以使用经典的最小二乘法解出 a_1^*, a_2^*, a_3^*，并求得校正平面。最后，用像素的灰度值和校正平面的差值作为分类器的输入，即：

$$I'(x_i, y_i) = I(x_i, y_i) - (a_1^* x_i + a_2^* y_i + a_3^*) \quad (2\text{-}18)$$

照度梯度修正能够消除图像的一阶变化，但对于图像整体偏暗或偏亮，该方法却无能力，因此，需要引入第二种光照校正办法。

第二种方法，统计光照校正。为了消除人脸图像整体偏暗或偏亮的影响，同时对于像素变化范围进行归一化，我们需要对图像的统计特性进行校正。考虑一幅高 H，宽 W 的图像，像素 (x, y) 的灰度值为 $I(x, y), 0 < x < H + 1, 0 < y < W + 1$，它的基本统计量有平均值和方差分别为：

$$\mu = \frac{1}{WH} \sum_{x=1}^{H} \sum_{y=1}^{W} I(x, y) \quad (2\text{-}19)$$

$$\sigma^2 = \frac{1}{WH} \sum_{x=1}^{H} \sum_{y=1}^{W} [I(x, y) - \mu]^2 = \mu^2 - \frac{1}{WH} \sum_{x=1}^{H} \sum_{y=1}^{W} I(x, y)^2 \quad (2\text{-}20)$$

这样，输入为 $I(x, y)$ 的灰度值，经过校正的灰度值为：

$$I'(x, y) = \frac{I(x, y) - \mu}{\sigma} \quad (2\text{-}21)$$

给定一个待检测的子窗口，为了快速计算平均值和方差，我们需要对原图维护一个基本的积分图，还需要维护一个平方积分图，对照公式 2-1，平方积分图

的计算满足下面的规则：

$$II(i, j) = \sum_{i'<=i, j'<=j} I(i', j')^2 \qquad (2\text{-}22)$$

这样，对于任意的子窗口，就可以在常数时间里计算出平均值和方差了。

2.4.2 对于图像金字塔的优化

在本章开始，我们总结了人脸检测中常用的一种思路。为了方便，重述如下：对于给定的测试图像（或者视频中的一帧），不同尺度的图像，构成一座图像金字塔；对于金字塔每一层，遍历其中所有可能存在人脸的子窗口，将这些子窗口（或者从这些子窗口提取的特征），作为一个模式输入到一个分类器中，这样，人脸检测问题便成为一个二分类问题。如何确定尺度变化，在 Viola-Jones 人脸检测子中，使用了指数变化的方式。在我们的系统中，则使用了线性变化的方式。两种方式的对比从图 2-9 中可以清晰的看出来。

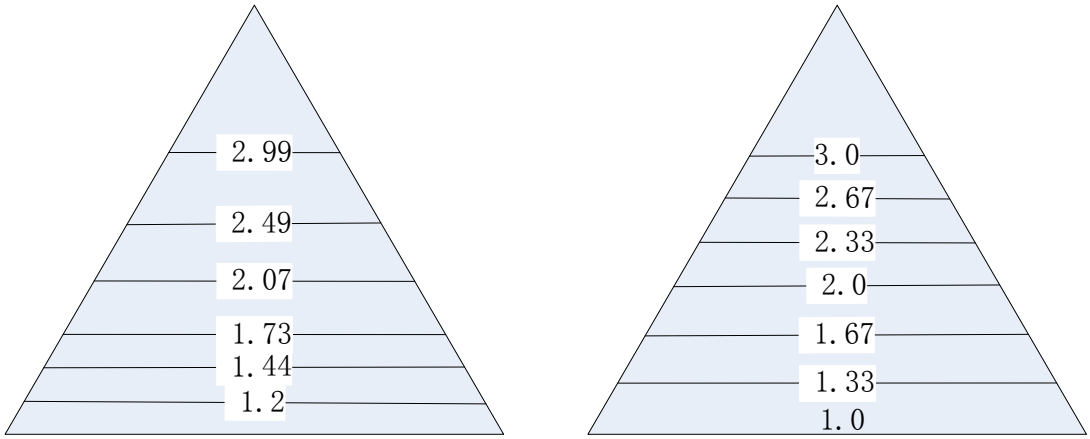


图 2-9　不同变化方式的图像金字塔

（a）指数变化的图像金字塔；（b）线性变化的图像金字塔

从图 2-9 中，我们可以看到，从 1.0 到 3.0 的尺度，不管是指数变化还是线性变化的图像金字塔，都有 7 级（包括 1.0 和 3.0），但是，指数变化的图像金字塔，当尺度越大时，不同尺度之间的间隔就越大，相反，在线性变化的图像金字塔中，不同尺度之间的间隔只与预先设定值有关，而与尺度大小无关。

通过使用不同的尺度变化策略构造图像金字塔，通过在 FERET 人脸库 Fa 测试[3]，检测率从 96%提升至 99%。这或许是因为人脸头像在不同尺度上均匀分布，

而不是在尺度小的时候概率较大。

2.4.3 对于搜索策略的优化

通常，在物体检测（object detection）中，对全图的搜索策略是这样的：从左至右，从上至下，每隔一定的像素间隔搜索一遍，这里像素间隔常常需要试验来获得。

这样的搜索策略把每一个搜索的子窗口的检测过程认为是互相独立的事件。实际上，相邻的子窗口的检测过程的相关性是非常大的。这里，当我们当前检测的子窗口为非人脸，那么我们能否判定下一个子窗口是否存在人脸呢？当然，我们不能。因此，当前子窗口为非人脸，与下一个子窗口是否是人脸，是独立的，我们在搜索子窗口时，需要继续搜索下一个子窗口。然而，如果当前子窗口是人脸呢？

直观上，我们容易想到，如果我们发现一个子窗口是人脸，那么，与之相邻的子窗口是否会是人脸呢？实际上，因为两个子窗口重合区域太大，以至于不重合的区域不可能被人脸检测子识别为人脸，这样，这个子窗口失去了检测的必要。同样，再往右，更远一点的临近子窗口呢，是否存在同样的规律呢。不失一般性，在我们的系统中，做了如下的处理。

使用空间换时间的策略。在每一个尺度下，申请一个二维数组的内存空间，该数组的宽度和高度分别为图像的宽度 W 和高度 H。检测到当前位置为人脸，则认为在该尺度下，周围“该尺度下所有露半个头以内的图像都认为不是人脸”。在别的尺度下，使用同样的策略，即“该尺度下所有露半个头以内的图像都认为不是人脸”，屏蔽掉该 mask。如图 2-10 所示，中间蓝色方形代表人脸位置，蓝色和绿色区域均为屏蔽掉的区域。

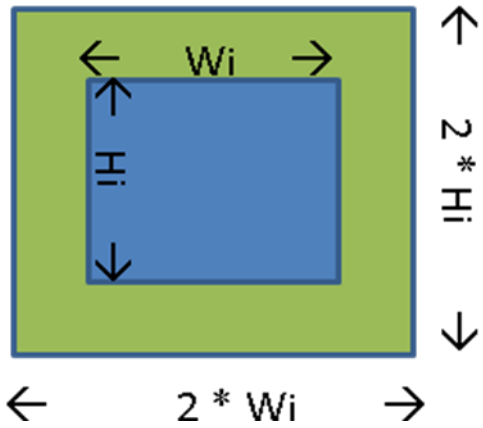


图 2-10　搜索策略的优化

除了上面的方法，我们还有一种进一步提高算法执行效率的方法。我们选择从大尺度开始往小尺度遍历搜索子窗口，这样，某一人脸可以在更大的尺度下被发现，同时更大的区域将被屏蔽，用以提高算法的执行效率。

通过使用这样的策略，显然，我们可以避开对许多相关的子窗口的搜索，能极大的提高算法的执行效率。同样，这样的策略也存在一个小问题，那就是当图像本身的尺寸非常大时，则在图像金字塔中，存在这更多的尺度，更多的层数，这样，上述的二维数组的内存空间将是一个非常大的开销。因此，对于特别大的图像，在输入检测子检测之前，需要做下采样，以免造成检测子过分吞噬系统的内存空间。多大的图像需要进行下采样，这由具体的硬件系统决定，这里不再赘述。

2.5 实验结果

在我们的实验中，我们通过网络抓取、使用标准人脸数据库等途径，获得了 22260 幅正面人脸图像，并将其中的人脸抠出，最后归一化至 24*24 的大小作为训练正样本。将原图中抠去人脸部分，剩余的部分一块一块的取出 24*24 的子窗口，作为训练负样本。最终，我们训练了一个含有 16 级的级联结构，其中每一级含有一个强分类器，前 5 个强分类器包含的弱分类器的个数分别为：5 个，4 个，6 个，8 个，9 个。

在我们的试验中，使用 FERET 人脸库中的 Fa 部分[3]作为测试集，Fa 包含 1196 幅正面人脸图像，如图 2-11 所示。

图 2-11　测试集：FERET 人脸库

使用我们的人脸检测子，实验结果如表 2-1 和图 2-12 所示，试验中用到的硬件平台为 Intel Xeon CPU 2.53GHz，2.00GB 的内存。

表 2-1　人脸检测实验结果

/	检测率 TPR（%）	错检（个）	平均耗时（ms）
优化前	96.49	5	26.32
优化后	99.08	0	18.83

图 2-11　人脸检测结果

我们知道，通过使用搜索策略优化，算法的执行效率得到了很大的提高。但是，由于该方法的引入，也带来了一定的问题，如图 2-12 所示。在这些结果中，虽然人脸也被正确的找出，但是由于给出只是一个非常粗略的框，因此，可能对后期进一步处理带来不便，如人眼检测等。但是，在当前的人脸检测指标中，并

没有什么指标能够用以限制此类问题的出现。因此，或许有必要进一步制定更加精确细致的人脸检测算法的指标。

图 2-12 需要改进的人脸检测结果

2.6 本章小结

在这一章中，我们详细介绍了 Viola-Jones 人脸检测子的原理，包括 Haar 特征、利用积分图快速计算 Haar 特征、使用 AdaBoost 算法从弱分类器中构造出强分类器以及级联结构，另外，我们还详细的给出了我们在系统实现中一些细致的优化策略。最后，给出了实验结果，并进一步提出了当前评价人脸检测算法优劣的指标中存在的问题。

第三章　人眼检测及人脸归一化方法

人脸识别系统通常有两种划分方式[3]。第一种，称为半自动人脸识别（partially automatic face recognition），在半自动人脸识别中，人眼的位置已经人工标注，这样，研究人员只需要根据已经标注好的人眼位置，归一化人脸图像，设计人脸比对算法即可。在本文中，为简单起见，统称半自动人脸识别为人脸比对。实际上，当前大部分的人脸识别算法属于此类。第二种，在实际的系统中，我们不可能先前就知道人脸在什么区域，更不可能知道人眼的位置，因此，这些都需要在算法中给与实现，这样的系统称为全自动人脸识别（fully automatic face recognition）。在本文中，为简单起见，我们简称为人脸识别。我们的人脸识别系统就属于全自动人脸识别。

人脸归一化算法的性能直接影响到人脸识别的准确度，而目前来说，人脸归一化通常是基于双眼的位置进行的，因此，一个精确地人眼检测对整个人脸识别系统将起着举足轻重的作用[35]。

为了佐证上述观点，研究人员做了如下的实验[35]。对于人工标注的人眼位置，引入人工噪声干扰，这些噪声符合某一个半径的圆内的均匀分布。因此，人脸归一化中使用的人眼位置，将是以人工标注人眼位置为圆心的一个均匀分布的圆形。在图 3-1 中，横坐标表示这个圆形的半径相对于左右眼距离之间的比值，纵坐标表示的在 FRGC1.0（Face Recognition Grand Challenge）[36]人脸库中使用 PCA 方法得到的人脸识别率。

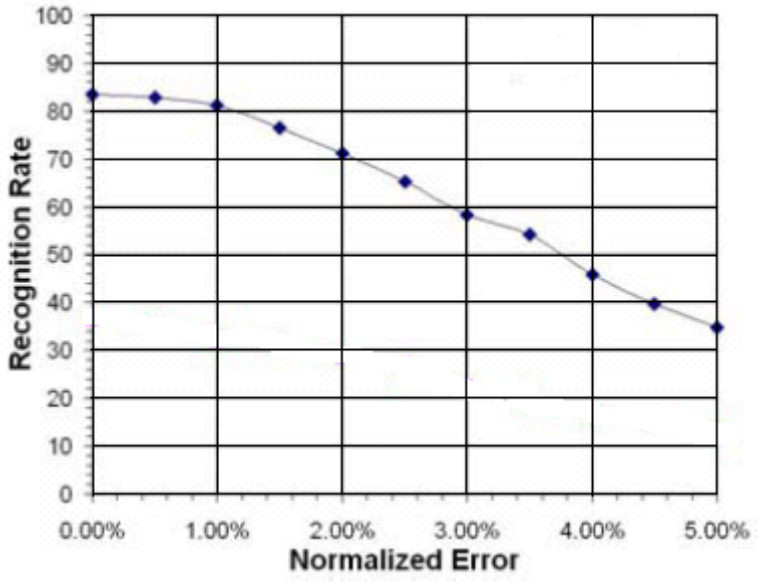


图 3-1　人眼检测精度对人脸识别精度的影响[35]

在我们的系统中，双眼间的距离被归一化至 39 个像素。如果人眼检测有一个像素的偏差，也就是 2.6%的偏差，则意味着如果使用 PCA 作为人脸比对算法，那么识别率将下降约 25%！而如果有两个像素的偏差，即 5.2%偏差，则识别率将下降约 50%！当然，下降比率可能与具体采用的人脸比对算法有关，但这个实验总体上告诉了我们人眼检测精度的重要性。

3.1 人眼粗定位算法

在第二章中，我们使用了经典的 Viola-Jones 人脸检测子用于检测人脸。实际上，这个方法在最初提出时，有一个更加广泛的适用场景——物体检测（object detection），而人脸检测只是 Viola 和 Jones 为了验证算法性能而做的实验[27]。Viola-Jones 人脸检测算法核心是 AdaBoost 算法——一种基于弱分类器构造强分类器的方法。同时，AdaBoost 也是一个很好的特征选择的方法，当成千上万个特征摆在你的眼前，你无法知道哪一个特征具有更好的区分性时，你可以使用 AdaBoost 算法从中筛选出有用的特征。为了提高算法的执行效率，Viola 和 Jones 在 AdaBoost 构造一个强分类器之后，继续对分类器的输出做一次 AdaBoost 分类，这样逐级增加 AdaBoost 分类器，就构成了一个级联结构，级联结构作为一种退化的决策树，简单，物理含义清晰，最重要的是，它能极大的提高算法的执行效

率。最后，对于 AdaBoost 算法，她的弱分类器可以有多种多样的形式，最简单的弱分类器可以由一个特征构成，这就是 Viola-Jones 人脸检测子中使用的基于 Haar 特征的弱分类器，引入积分图之后，Haar 特征在不同位置，不同尺度下的计算都是一个非常高效操作，可以在常数时间迅速完成。复杂的弱分类器，甚至可以是一个 SVM（Supporting Vector Machine）分类器。

在我们的人眼检测中，采用了经典的 Viola-Jones 检测子，为叙述方便，我们称这样的方法为 Viola-Jones 人眼检测子。但是，问题变得稍微有些不同。在人脸检测中，人脸的模式（pattern）非常清晰，且容易被 Haar 特征描述，例如，人脸中的眼睛、眉毛、嘴巴、鼻子等。详见图 2-2。但是，在人眼中，模式变得不那么清晰，如图 3-2 所示。

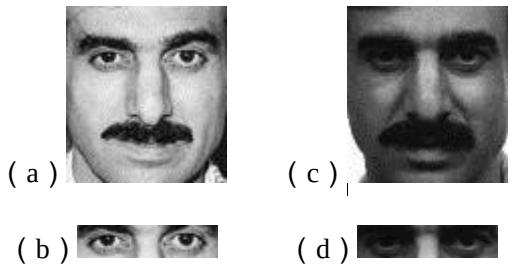


图 3-2　人眼的模式（pattern）

图（a）和图（c）来自同一个人的不同场景，图（b）和图（d）分别是他们人眼模式。从图（b）中，我们会认为双眼中也存在非常适合于用 Haar 特征表达的特征，例如眼球和巩膜之间的黑白交界处，相比于其它地方，鼻梁较亮等等。但是，这些受不同场景影响太大，如图（d），上述所说的特征就不那么明显了。因此，如果简单的套用 Viola-Jones 人脸检测算法，人眼检测算法将很难让人满意。

为此，我们在人眼检测中，将算法拆分成两个步骤，如图 3-3 所示。

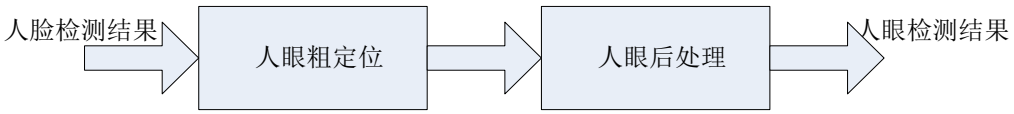


图 3-3　人眼检测流程图

其中，人眼检测的输入为第二章中介绍的人脸检测的输出，具体而言，是一个包

含左上角和右下角的矩形人脸区域。对于输入的人脸区域，首先使用一个人眼粗定位算法，对左眼和右眼分别找到较多的候选点，由于左眼和右眼的对称性，我们不必要对左眼和右眼分别构造粗定位模型，我们只需要构造一个左眼的粗定位模型，使用该模型可以直接粗定位左眼的候选点，而对于右眼，我们只需先将图像关于中心轴作镜像处理，就可以使用该模型粗定位了。然后，使用一个后处理算法，利用较多的候选点，找出精确的人眼位置。

借鉴 Viola-Jones 人脸检测，构造了人眼粗定位算法。具体而言，就是使用 Haar 特征构造弱分类器，并用 AdaBoost 算法将这些弱分类器提升为强分类器，最后用级联结构结合若干强分类器[27]。在这里，由于输入为人脸区域，因此，人眼应该在人脸区域的某一范围内，我们的搜索范围就可以固定在某一个区域内，而不必在整个人脸区域内搜索。如图 3-4 所示。红色矩形表示 Viola-Jones 人脸检测的输出结果，作为人眼检测的输入。左眼周围的紫色矩形表示粗定位左眼时的搜索范围，同理右眼周围的黄色区域为右眼的搜索范围。中间的白色的直线表示人脸区域的中心轴，注意，不管是左眼还是右眼，搜索范围都略微超过了中心轴。

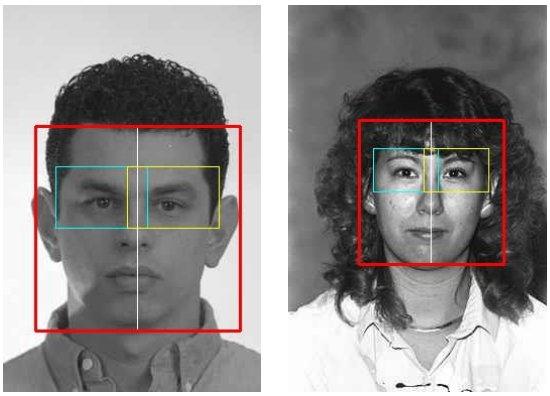


图 3-4　人眼粗定位搜索范围

这样做有以下几个好处，第一，只需要计算搜索范围内的积分图，而不需要计算整个人脸区域的积分图。第二，排除了在搜索区域以外的子窗口的检测，理想情况下，这些子窗口需要经过粗定位检测子中级联结构的若干级，需要一定的时间消耗，划定了搜索区域之后，这些时间消耗都将去除。第三，可以排除许多

远离人眼的人眼候选点，由于 AdaBoost 算法仅对灰度图像处理，因此头发等较阴暗的区域，往往被分类为人眼候选点，假如有了搜索区域的限制，就可以有效的避免这些干扰，提高人眼检测的精度。

但是，由于人脸不同的姿态，也会在很偶然的场合造成该搜索区域并没有有效的包含实际的人眼，如图 3-5 所示。这个问题可以通过调整所搜区域的参数得到解决。如下图所以，最简单的办法可以增加搜索区域的下边界，但是，这又会带来搜索区域的增加使算法的执行效率下降。因此，这是一个需要权衡的问题。实际上，这个问题非常少见，我们使用 FERET 人脸库[3]中的 Fa 部分作为测试集，该测试集共有 1192 幅人脸图片，图 3-5 是其中唯一划分失败的！

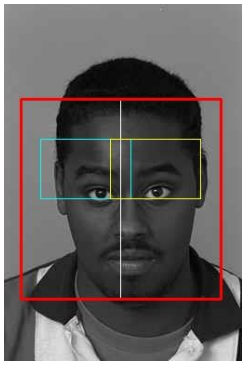


图 3-5　划分失败的人眼粗定位搜索区域

在人眼粗定位算法中，还有一个值得一提的地方，那就是，在训练粗定位算法时，我们目标是尽可能多的找出人眼候选点，而不必要每个候选点都非常精确，因此，我们训练的粗定位算法的要求是：第一，高的检测率 TPR（True Positive Rate），保证把真正的人眼区域分类为人眼候选点；第二，高的误检率 FPR（False Positive Rate），这样会引入一些非人眼候选点，但这些候选点往往位于人眼附近，对后处理算法并无太大的干扰，实际上，只有高的误检率，才能保证达到高的检测率。

图 3-6 给出了人眼粗定位的实验结果。

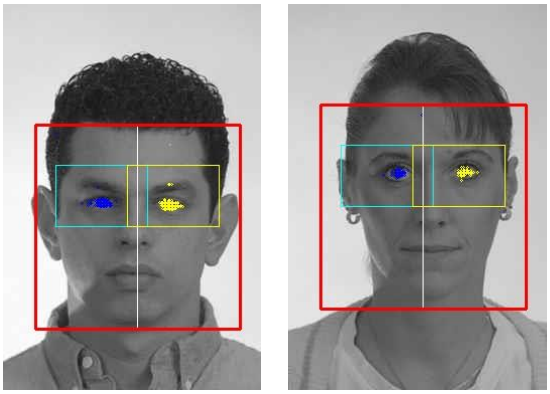


图 3-6　人眼粗定位的结果

3.2 基于 Adaboost 的后处理算法

在我们的系统中，有两种人眼后处理算法。第一种，借助于大量的人眼候选点和双眼的模式，继续使用 Viola-Jones 人脸检测的思想，使用 Haar 特征作为弱分类器，使用 AdaBoost 算法训练强分类器，并最终用级联结构验证左右眼的精确位置，为叙述方便，我们称这样的方法为 Viola-Jones 人眼验证子。第二种，仅仅借助于大量的人眼候选点，用核密度估计的方法估计人眼位置的概率分布，使用 mean shift 方法找到概率最大的位置，作为人眼的精确定位。在这一节里，将着重介绍第一种方法。

图 3-6 展示了人眼粗定位的结果。从图中我们可以看到，不管是左眼，还是右眼，都有大量的候选点。我们用 Viola-Jones 人眼验证子作为验证算法，验证左眼和右眼能够组成一双眼睛的置信度，并最终将置信度最高的三双眼睛的平均值作为最终的精确定位的人眼位置。

利用 AdaBoost 算法做双眼的验证，需要另外重新训练一个模型。这个模型的部分正样本如图 3-7（a）所示。负样本则来自于人脸区域中非双眼模式部分，如何 3-7（b）所示。

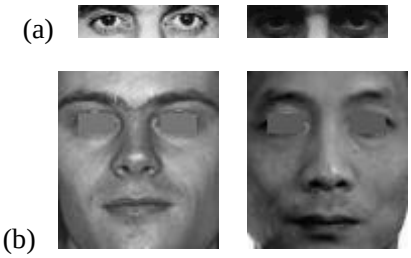


图 3-7　AdaBoost 验证双眼置信度的正样本和负样本

（a）正样本；（b）负样本

使用 Viola-Jones 人眼验证子作为人眼检测后处理的方法，存在以下几个问题。

第一，Viola-Jones 方法的初衷是为了加速算法的执行效率，在人脸检测子窗口中，绝大部分为非人脸子窗口，这些子窗口在级联结构中只需要经过少量的级数，甚至是一级就可以被判定为非人脸，这是 Viola-Jones 人脸检测子运行高效的三大原因之一。然而，对于 Viola-Jones 人眼验证子，每一个待验证的“左眼-右眼”对，由于他们都处在真实眼睛的附近，因此，他们组成的双眼，即使不是精确的双眼，也与真实的双眼模式（见图 3-2）十分相似，这必然导致每一个待验证的“左眼-右眼”对，验证时都需要经过级联结构的大部分级数甚至是所有级数，这必然使算法的执行效率大打折扣。

第二，从图 3-6 中我们可以清楚的看到，不管是左眼，还是右眼，都有大量的候选点，设左眼共有候选点 Nleft 个，右眼有候选点 Nright 个。在我们的试验中，左眼和右眼的候选点有时候能多达 100 个。如果把每一对都输入到 Viola-Jones 人眼验证子中验证，则一共需要验证 Nleft*Nright 次，如果左眼和右眼的候选点平均为 50 个，那么总共需要验证 2500 次！这是非常可怕的。为了降低算法的时间复杂度，我们有必要对这些候选点先做下采样，再输入到 Viola-Jones 人眼验证子中验证。于是，引入了另一个问题，如何下采样，才能保证算法的精度不受影响？这是一个非常难回答的问题。在我们的系统中，下采样方法是：如果候选点个数不大于 10 个，则可以直接输入到 viola-Jones 人眼验证子验证；如果候选点个数大于 10 个，则首先按照候选点的置信度从大到小排序，然后以 Nleft/10

或者 Nright/10 为采样间隔均匀采样，最终获得 10 个候选点。这样，最终需要验证 10*10=100 个“左眼-右眼”对。时间开销依然庞大。

经过 Viola-Jones 人眼检测子和 Viola-Jones 人眼验证子之后，我们就可以得到精确的人眼位置。由于 Viola-Jones 作为验证算法时的一些列问题，这个算法的执行效率并不高，在为 Intel Xeon CPU 2.53GHz，2.00GB 内存的硬件平台上，检测一双眼睛大约需要 11.5ms。

3.3 基于 mean shift 的后处理算法

在上一节中介绍的 Viola-Jones 人眼验证子，在嵌入式系统中无法达到实时，为此，我们有必要引入一种新的后处理方法，避免 Viola-Jones 人眼验证子的诸多弊端。

在物体检测中，一般在目标物体周围，会有好几个子窗口被判定为候选目标，为了最终输出一个准确的物体的位置，需要对多个候选目标做融合处理。检测结果融合有以下三条原则[37]：①局部极值点对应的置信度越高，说明该位置出现目标的可能性越大；②局部极值点附近对应的窗口越多，说明该极值点是目标的可能性越大；③临近的检测结果应该融合在一起，但是，在各个尺度上都临近的检测目标可能对应相邻的两个目标，不应该融合。目前，物体检测中满足着三条融合原则的较好的融合方法是变带宽的均值漂移算法（variable bandwidth mean shift）[37, 38]。为此，我们考虑是否可以把 mean shift 算法引入到我们的人眼后处理算法中来。

3.3.1 mean shift 基本原理

Mean shift 最初在 1975 年由 Fukunaga 等人提出[39]，并且最初提出时是为了解决概率密度梯度函数估计问题。最初，mean shift 指向偏移的均值向量，后来，随着研究的逐步深入，mean shift 被赋予了更多的含义。现在，mean shift 主要指一种通过特定的迭代过程求解最优解的算法，该迭代方法首先指定一个初始值，在初始值的基础上，计算出一个均值偏移，并把初始值和该偏移的和作为下一次迭代的起点，如此往复，直至收敛条件满足为止。

给定 N 个样本点$\{x_i, x_i \in R^n, i=1,2,...N\}$，其中$R^n$表示 n 维向量空间。在点$x_0$处的均值偏移（mean shift）为：

$$m(x_0)=\frac{1}{k}\sum_{x_i\in S}(x_i-x_0) \qquad (3\text{-}1)$$

其中，S 表示x_0的邻域，常见的有圆形邻域、矩形邻域，邻域的大小（即带宽 bandwidth）的选择方法需要事先约定。k 表示落入邻域 S 内的样本点个数。(x_i-x_0)表示第 i 个属于邻域 S 的样本点相对于迭代起点x_0的偏移量。从公式 3-1 我们可以看出，$m(x_0)$实际上是对 S 邻域内所有样本点相对于迭代起点x_0的求和平均，这样就是为什么$m(x_0)$称为均值偏移的原因。

图 3-8 给出了 mean shift 的直观物理含义。图中，圆形框表示领域大小。最终，均值偏移向量将指向右侧样本点集中的方向。

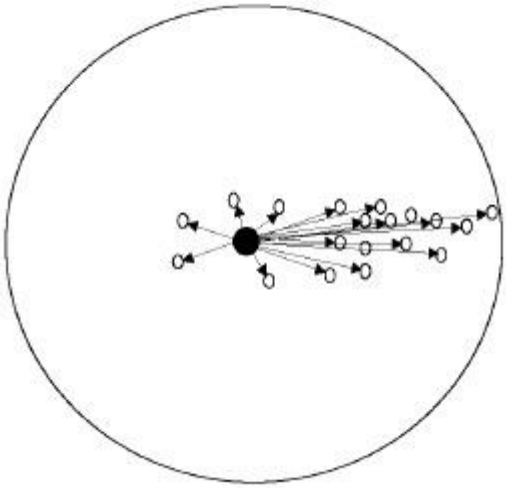


图 3-8　mean shift 的直观物理含义[40]

公式 3-2 实际上给出了最朴素的 mean shift 算法的解析式，实际上，更加具有实际应用价值的 mean shift 方法，是一种基于核函数的扩展形式[41]。由于下一小节将对基于核函数的 mean shift 方法着重展开陈述，因此这里不再赘述。

Mean shift 有许多应用场景。在概率密度估计中，核密度估计作为在当前应用最广泛的估计方法，其极值的求解可以应用 mean shift 快速解决；在跟踪问题中，使用基于核函数的 mean shift 方法已成为跟踪问题的经典解法[42]；同时，mean

shift 也为图像分割提供一种行之有效的解决方法[41]。

3.3.2 mean shift 在后处理中的应用

在上一节中，我们简单介绍了最朴素的 mean shift 算法的原理，以及直观的物理解释，在这一节中，我们将看到 mean shift 在寻找精确人眼位置中的应用。正如我们在本章第三节开始提到的三条原则那样，在人眼检测中：一方面，并不是置信度最高的人眼候选点就一定精确的人眼位置，这主要是因为，Viola-Jones 人眼检测子输出的置信度并不能完全客观的反应真实的置信度，有一定的偏差。另一方面，在某一处候选点非常集中，这似乎告诉我们该处更有可能成为精确的人眼位置。最后，候选点本质山上是从不同的尺度得到的结果，那么，最终人眼的精确位置所在的尺度，它临近的尺度上也一定存在这较多的候选点，加入尺度因子，还有另一个好处，那就是当候选点非常集中，但是在尺度维度上很分散时，说明这些候选点即有可能是头发等干扰因素，尺度维度的加入可以避免此类干扰。

通过上面的分析，我们可以得到一个基于 mean shift 的人眼后处理算法。

设 $\mathbf{y}=[x,y,s]$，其中 x，y 表示候选点的横纵坐标，s 表示候选点所在的尺度。为了使相邻尺度间具有相同的间隔，当尺度变化遵循指数变化时，如图 2-9（a）所示，

$$s=\log(s') \tag{3-2}$$

其中 s’为真实的尺度因子。当尺度变化遵循线性变化时，如图 2-9（b）所示，

$$s=s' \tag{3-3}$$

这样，每一个候选点，都成为一个三维的点。

此外，对于每一个候选点，我们还有一个信息，那就是该候选点的置信度 c。候选点的置信度并不像前面所述的横纵坐标和尺度那样，越密集，说明候选点越有可能处于真实的人眼位置。置信度的性质是，置信度越高，说明候选点越有可能处于真实的人眼位置。因此，置信度不能简单的加到前面说的三维空间中，作为第四维，而应该作为每一个候选点的权值 ω。置信度越高，则候选点在概率密

度估计中所占的比重就越大。

当然，作为权值，必须是一个大于零的数，如果置信度有小于零的情况出现，则我们有必要通过某种映射 t，使 $\omega = t(c)$，把置信度空间映射为权值空间。不同的映射方法有不同的效果，常用的映射方法有[37]：硬裁剪（hard clipping），软裁剪（soft clipping）和 sigmoid 裁剪（sigmoid clipping）。幸运的是，前面提到的 Viola-Jones 人眼检测子的输出置信度均大于零，这一点从连续 AdaBoost 算法（算法 2-1）中可以看出来，因此，在这里，$\omega = c$。我们并不详细展开每种映射的方法，Navneet Dalal 在他的博士论文[37]中对此有详细的评述。

令 $\mathbf{y}_i = [x_i, y_i, s_i], i = 1,2,...N$ 表示 N 个候选点，每个候选点的置信度（权值）为 ω_i，则当迭代中心位于 $\mathbf{y}$ 处时，使用高斯核函数的核密度估计方法，得到候选点的概率密度估计为：

$$\hat{f}(\mathbf{y}) = \frac{1}{N(2\pi)^{3/2}} \sum_{i=1}^{N} \frac{\omega_i \exp(-\frac{D^2[\mathbf{y}, \mathbf{y}_i, \mathbf{H}_i]}{2})}{\sqrt{|\mathbf{H}_i|}} \qquad (3\text{-}4)$$

其中，$D^2[\mathbf{y}, \mathbf{y}_i, \mathbf{H}_i]$ 表示候选点 $\mathbf{y}_i$ 相对于迭代中心 $\mathbf{y}$ 的偏移量，用马氏距离（Mahalanabois Distance）度量，表达式为：

$$D^2[\mathbf{y}, \mathbf{y}_i, \mathbf{H}_i] = (\mathbf{y} - \mathbf{y}_i)^T \mathbf{H}_i^{-1} (\mathbf{y} - \mathbf{y}_i) \qquad (3\text{-}5)$$

$\mathbf{H}_i$ 表示候选点 $\mathbf{y}_i$ 处的协方差矩阵，表达式为：

$$\mathbf{H}_i = \begin{bmatrix} (\exp(s_i)\delta_x)^2 & 0 & 0 \\ 0 & (\exp(s_i)\delta_y)^2 & 0 \\ 0 & 0 & \delta_s^2 \end{bmatrix} \qquad (3\text{-}6)$$

其中，δ_x 表示水平方向搜索子窗口跳变的像素数，δ_y 表示竖直方向搜索子窗口跳变的像素数，δ_s 表示图像金字塔中相邻的尺度之间的比值（在此重申，由于尺度变化常用指数规律，因此这里采用指数规律的形式，如果采用线性规律的形式，则需要作对应的修改）。$\mathbf{H}_i$ 表示候选点 $\mathbf{y}_i$ 处的协方差矩阵，即不确定度。为了减少预设参数，简化模型，假设候选点所在的三维坐标（横坐标，纵坐标和尺度坐

标）之间相互独立，即$\mathbf{H}_i$为对角矩阵。当候选点的尺度坐标为s_i，结合公式 3-2，我们知道，候选点所在的图像金字塔平面中水平方向上相邻搜索子窗口之间的像素间隔为$\exp(s_i)\delta_x$，因此，候选点横坐标方向上的不确定度为$(\exp(s_i)\delta_x)^2$。这也非常符合我们的直观思维，遍历子窗口时，子窗口间的间隔越大，则候选点的不确定就越大。

明白了公式 3-4 中各参数的物理意义后，公式 3-4 的梯度可求得为：

$$\begin{aligned}
&\nabla\hat{f}(\mathbf{y}) \\
&= \frac{1}{N(2\pi)^{3/2}} \sum_{i=1}^{N} \frac{\mathbf{H}_i^{-1}(\mathbf{y}_i - \mathbf{y})\omega_i \exp(-\frac{D^2[\mathbf{y},\mathbf{y}_i,\mathbf{H}_i]}{2})}{\sqrt{|\mathbf{H}_i|}} \\
&= \frac{1}{N(2\pi)^{3/2}} \left[\sum_{i=1}^{N} \frac{\mathbf{H}_i^{-1}\mathbf{y}_i\omega_i \exp(-\frac{D^2[\mathbf{y},\mathbf{y}_i,\mathbf{H}_i]}{2})}{\sqrt{|\mathbf{H}_i|}} \right] \\
&- \frac{1}{N(2\pi)^{3/2}} \left[\sum_{i=1}^{N} \frac{\mathbf{H}_i^{-1}\omega_i \exp(-\frac{D^2[\mathbf{y},\mathbf{y}_i,\mathbf{H}_i]}{2})}{\sqrt{|\mathbf{H}_i|}} \right] \mathbf{y}
\end{aligned} \tag{3-7}$$

我们定义ϖ_i为：

$$\varpi_i(\mathbf{y}) = \frac{|\mathbf{H}_i|^{-1/2}\omega_i \exp(-\frac{D^2[\mathbf{y},\mathbf{y}_i,\mathbf{H}_i]}{2})}{\sum_{i=1}^{N}|\mathbf{H}_i|^{-1/2}\omega_i \exp(-\frac{D^2[\mathbf{y},\mathbf{y}_i,\mathbf{H}_i]}{2})} \tag{3-8}$$

显然，ϖ_i满足：

$$\sum_{i=1}^{N}\varpi_i = 1 \tag{3-9}$$

使用公式 3-6,3-7 和 3-8，我们可以得到$\nabla\hat{f}(\mathbf{y})$与$\hat{f}(\mathbf{y})$的比值为：

$$\frac{\nabla\hat{f}(\mathbf{y})}{\hat{f}(\mathbf{y})} = \sum_{i=1}^{N}\varpi_i\mathbf{H}_i^{-1}\mathbf{y}_i - \left(\sum_{i=1}^{N}\varpi_i\mathbf{H}_i^{-1}\right)\mathbf{y} \tag{3-10}$$

若令：

$$\mathbf{H}_h^{-1}(\mathbf{y}) = \sum_{i=1}^{N} \varpi_i \mathbf{H}_i^{-1} \tag{3-11}$$

则可以得到当迭代中心为$\mathbf{y}$时，可变带宽均值均值偏移为：

$$\mathbf{m}(\mathbf{y}) = \mathbf{H}_h(\mathbf{y}) \frac{\nabla \hat{f}(\mathbf{y})}{\hat{f}(\mathbf{y})} = \mathbf{H}_h(\mathbf{y}) \left(\sum_{i=1}^{N} \varpi_i(\mathbf{y}) \mathbf{H}_i^{-1} \mathbf{y}_i \right) - \mathbf{y} \tag{3-12}$$

在核密度估计达到极大值处时，有$\nabla \hat{f}(\mathbf{y}) = 0$，结合公式 3-12，我们可以知道，$\mathbf{m}(\mathbf{y}) = 0$。因此，通过公式 3-12 我们可以通过迭代的方式：

$$\mathbf{y}_{m+1} = \mathbf{H}_h(\mathbf{y}_m) \left(\sum_{i=1}^{N} \varpi_i(\mathbf{y}_m) \mathbf{H}_i^{-1} \mathbf{y}_i \right) \tag{3-13}$$

设定一个初始迭代中心$\mathbf{y}_0$，一步一步的迭代，直到$\mathbf{y}_{m+1}$与$\mathbf{y}_m$几乎不变，则找到了核密度估计的极大值点，也就找到了 N 个候选点所处的精确的人眼位置。

在使用 mean shift 作为人眼后处理算法时，有以下几个更细致的考虑。

首先，如何选择初始迭代位置。mean shift 作为一种求极值而非最大值的方法，会有陷入局部极值的风险。迭代的终点是全局极值还是局部极值，完全取决于初始迭代位置，因此，初始迭代位置的选择至关重要。一种方法是从 N 个候选点中随机选择一个候选点作为初始迭代位置，为了避免陷入局部极值点，则需要多次随机选择初始迭代位置，由于在精确的人眼位置的周围的候选点一般远多于头发、眉毛等噪声点周围的候选点，因此，应用这种方法，最终从众多的收敛点中选择出现次数最多的那一个，就是最终的精确人眼位置。这个方法的理论依据是概率论中的大数定理，即当随机选择的次数足够多时才有效，而随机选择次数太多，又会造成算法计算复杂度的提升。最终，我们使用了一种简便又实用的方法，由于每一个候选点都对应一个三维坐标$\mathbf{y} = [x, y, s]$和一个置信度ω，通过对所有 N 个候选点作加权求和，得到初始迭代位置：

$$\mathbf{y}_0 = \sum_{i=1}^{N} \omega_i \mathbf{y}_i \tag{3-14}$$

实验证明，只要人眼位置周围的候选点个数多于噪声点的个数，该方法最终总能

收敛到正确的人眼位置。

3.4 两种后处理方法的比较

在算法的时间复杂度上，mean shift 要明显优于 Viola-Jones 人眼验证子。在本章第二节中，我们曾经分析过为什么 Viola-Jones 人眼验证子是一个非常耗时的方法，而对于 mean shift 方法，采用加权求和的方法设置迭代初始值，通常只需要 2~3 次的迭代，算法就可以收敛。在同样的测试集下，基于 Viola-Jones 人眼验证子的算法耗时 11.3ms，而基于 mean shift 的人眼后处理算法只需要 1.88ms。

在算法的检测率上，二者并没有太大的差异。使用同样的人脸检测算法，即第二章所述的 Viola-Jones 人脸检测子，作为人眼检测的输入，剔除人脸检测失败的测试图片，在同样的标准下（具体标准见本章第六节），二者都能达到 93% 的检测率。

在归一化效果上，基于 Viola-Jones 人眼验证子的后处理算法要略优于基于 mean shift 的人延后处理算法。所谓归一化效果，是指采用同样的人脸比对算法（在我们的系统中，采用的算法详见第四章第二节所述），使用不同的人眼后处理方法，得到的人脸比对的正确率。由于基于 Viola-Jones 人眼验证子考虑了双眼之间的模式，如鼻梁等，而基于 mean shift 的方法只考虑了单眼的模式，也就导致了最终精确定位的双眼位置略逊于 Viola-Jones 的方法。而之所以二者的检测率不相上下，主要是由于目前还没有一个合适的标准，既能用来衡量人眼检测的精度，又能兼顾到人脸归一化的效果。

3.5 人脸归一化算法

人脸归一化算法，主要是为了矫正不同姿态的人脸图像，并使进入人脸比对系统的输入图像尺寸一致。人脸归一化的原理如图 3-9 所示。

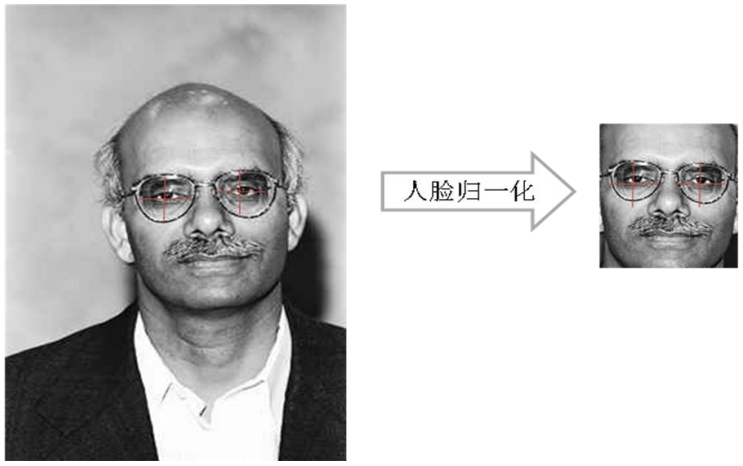


图 3-9　人脸归一化原理

设归一化前，左眼位置为(lx_0, ly_0)，右眼位置为(rx_0, ry_0)，归一化后，左眼位置为(lx_0', ly_0')，右眼位置为(rx_0', ry_0')。为了适应人脸比对的输入规则，通常需要对归一化后的图像做一定的限制，满足$rx_0' - lx_0' = c$，其中 c 是与归一化图像宽度相关的常数；$ly_0' = ry_0'$，即图像中的人脸均为竖直方向。

人脸归一化的目的，就是为了确定一个仿射变换 T，使得：$T((lx_0', ly_0')) = (lx_0, ly_0)$，并且$T((rx_0', ry_0')) = (rx_0, ry_0)$。这样，在仿射变换 T 的作用下，归一化图像中的某点(x', y')映射为原图中的(x, y)，注意，这里的 x 和 y 不一定是整数，即 x 和 y 是超像素。因此，要求点(x', y')处的灰度值$I(x', y')$，需要对(x, y)所处的周围四点做双线性内插。原理如图 3-10 所示。

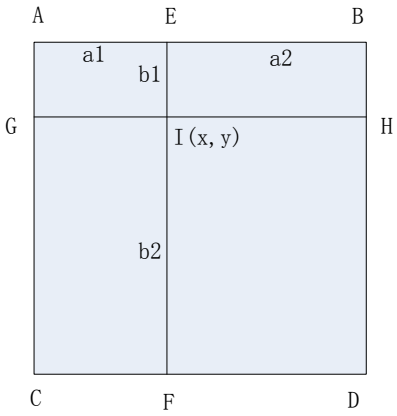


图 3-10　双线性内插原理

像素 A、B、C、D 为紧紧环绕在超像素 (x,y) 周围的四个像素点。显然：

$$\begin{aligned} a_1 &= x - \lfloor x \rfloor \\ a_2 &= 1 + \lfloor x \rfloor - x \\ b_1 &= y - \lfloor y \rfloor \\ b_2 &= 1 + \lfloor y \rfloor - y \end{aligned} \tag{3-15}$$

首先，通过像素点 A 和 B 线性内插，得到 E 点处的灰度值为：

$$I_E = (1-a_1)I_A + a_1 I_B \tag{3-16}$$

同理得到 F 点处的灰度值为：

$$I_F = (1-a_1)I_C + a_1 I_D \tag{3-17}$$

最后，通过 E 点和 F 点，内插得到超像素 (x,y) 的灰度值为：

$$I(x,y) = (1-b_1)I_E + b_1 I_F \tag{3-18}$$

综合公式 3-16，3-17 和 3-18 得：

$$I(x,y) = (1-a_1)(1-b_1)I_A + a_1(1-b_1)I_B + (1-a_1)b_1 I_C + a_1 b_1 I_D \tag{3-19}$$

这样，就得到了双线性内插的结果。实际上，先内查出 G 点、H 点的灰度值，再内插超像素 (x,y) 的灰度值，结果与公式 3-19 结果相同。

这样，在人脸归一化算法中，剩下的问题就是如何求解仿射变化 T 了。下面，分别介绍两种方法，分别称为基于绝对位置的方法和基于相对位置的方法。

3.5.1 基于绝对位置的人脸归一化方法

对于图像的仿射变换，有三种变换方式：旋转变换、尺度变换和平移变换。我们可以用数学的语言描述为：

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{b} \tag{3-20}$$

其中，$\mathbf{x}$ 为原位置坐标，$\mathbf{y}$ 为仿射变换后的坐标，$\mathbf{A}$ 代表旋转变换和尺度变换，$\mathbf{b}$ 代表平移变换。如果我们对坐标点加入一维，该维的值为 1，那么上面的描述方式可以变形为：

$$\begin{bmatrix}\mathbf{y}\\1\end{bmatrix}=\begin{bmatrix}\mathbf{A} & \mathbf{b}\\0 & 1\end{bmatrix}\begin{bmatrix}\mathbf{x}\\1\end{bmatrix} \tag{3-21}$$

这样，一个二维平面的仿射变换，就可以用一个 3 维方阵 $\mathbf{T}$ 表示。

那么，如果一个点连续做多次仿射变换，结果又是怎样的呢？设有一个点 $\mathbf{x}_0$，经过仿射变换 $\mathbf{T}_1$ 得到点 $\mathbf{x}_1$，若对 $\mathbf{x}_1$ 再做一次仿射变换 $\mathbf{T}_2$，得到的点为 $\mathbf{x}_2$。容易证明：

$$\begin{bmatrix}\mathbf{x}_2\\1\end{bmatrix}=\mathbf{T}_2\bullet\mathbf{T}_1\bullet\begin{bmatrix}\mathbf{x}_0\\1\end{bmatrix}=\begin{bmatrix}\mathbf{A}_2 & \mathbf{b}_2\\0 & 1\end{bmatrix}\begin{bmatrix}\mathbf{A}_1 & \mathbf{b}_1\\0 & 1\end{bmatrix}\begin{bmatrix}\mathbf{x}_0\\1\end{bmatrix} \tag{3-22}$$

这样，任意多次的仿射变换，我们都可以用仿射变换矩阵连乘的形式求得变换后的坐标值。

简单介绍了最简单的二维平面的仿射变换后，我们回到初始的问题中来。在人脸归一化问题中，我们需要把归一化人脸图像中的某点 (x',y')，经过一定的仿射变换 $\mathbf{T}$，映射为原图中的某个超像素 (x,y)。这里的限制条件为，需要把归一化后左眼和右眼分别映射到原图中的左眼和右眼，即：$\mathbf{T}((lx_0',ly_0'))=(lx_0,ly_0)$，并且 $\mathbf{T}((rx_0',ry_0'))=(rx_0,ry_0)$。为了满足上述两个限制条件，有多种变换方式。其中一种变换方式为，首先，平移图像，使左眼和右眼的中点变换到坐标原点，该变换称为 $t\mathbf{T}_1$，

$$t\mathbf{T}_1=\begin{bmatrix}1 & 0 & -\dfrac{lx_0'+rx_0'}{2}\\0 & 1 & -\dfrac{ly_0'+ry_0'}{2}\\0 & 0 & 1\end{bmatrix} \tag{3-23}$$

再经过一次尺度变换，使左眼和右眼之间的距离等于原图像中左眼和右眼的距离。该尺度变换矩阵为 $s\mathbf{T}$，

$$s\mathbf{T}=\begin{bmatrix}l/l' & 0 & 0\\0 & l/l' & 0\\0 & 0 & l/l'\end{bmatrix} \tag{3-24}$$

其中，l和l'分别为原图和归一化后图像中的左眼和右眼的距离。

然后，经过一次旋转变换，使左眼和右眼所在的直线与原图像中左眼和右眼平行，该旋转变换矩阵为$r\mathbf{T}$，

$$r\mathbf{T}=\begin{bmatrix}\cos\theta & -\sin\theta & 0\\ \sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix} \tag{3-25}$$

其中，θ为上述两条直线的夹角。

最后，经过一次平移变换，使左眼和右眼的中点变换到原图像中的左右眼中心，该变换称为$t\mathbf{T}_2$，

$$t\mathbf{T}_2=\begin{bmatrix}1 & 0 & \dfrac{lx_0+rx_0}{2}\\ 0 & 1 & \dfrac{ly_0+ry_0}{2}\\ 0 & 0 & 1\end{bmatrix} \tag{3-26}$$

综上，经过四次仿射变换，最终的仿射变换$\mathbf{T}$为，

$$\mathbf{T}=t\mathbf{T}_2\bullet r\mathbf{T}\bullet s\mathbf{T}\bullet t\mathbf{T}_1 \tag{3-27}$$

这样，通过仿射变换$\mathbf{T}$，就可以找到归一化后的图像在原图中的超像素的位置。虽然可以一次求得仿射变换矩阵$\mathbf{T}$，归一化图像中的每个点都可以通过这个仿射矩阵$\mathbf{T}$求到原图中的超像素位置，但是，每一次计算，都要用到公式 3-24 中的三角函数。在浮点运算中，这并不会有问题，但是对于定点运算而言，这里的三角函数运算就会带来精度上的灾难，使人眼检测定点运算的性能远逊于浮点运算。

3.5.2 基于相对位置的人脸归一化方法

根据流程，已知原图像的左上角和右下角的点位置，以及归一化后目标图像上的左上角和右下角的点位置。人脸归一化的目标就是希望获得目标图像中每个像素坐标在原图像中所对应的超像素位置，从而内插出相应位置的像素值。除了上一节提到的方法，该仿射变换可以通过如下两步获得：

第一步，对目标向量先做旋转和伸缩，使其与原向量平行相等，如图 3-11 所示。

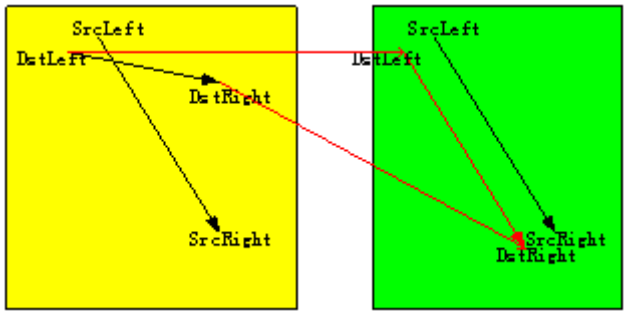


图 3-11　旋转和伸缩变换

第二步，对第一步得到的向量进行平移使其与原向量重合，如图 3-12 所示。

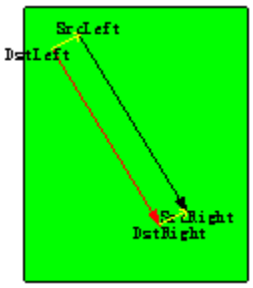


图 3-12　平移变换

记原图像的左上角和右下角的点位置 $u = DstRight.x - DstLeft.x$ 和 $v = DstRight.y - DstLeft.y$，以及归一化后目标图像上的左上角和右下角的点位置 $x = SrcRight.x - SrcLeft.x$ 和 $y = SrcRight.y - SrcLeft.y$，设旋转伸缩矩阵为：

$$RotateMatrix = \begin{bmatrix} m\cos\alpha & -m\sin\alpha \\ m\sin\alpha & m\cos\alpha \end{bmatrix} \tag{3-28}$$

通过解方程：

$$\begin{bmatrix} y \\ x \end{bmatrix} = \begin{bmatrix} m\cos\alpha & -m\sin\alpha \\ m\sin\alpha & m\cos\alpha \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \tag{3-29}$$

解得旋转伸缩矩阵为：

$$RotateMatrix = \frac{1}{u^2 + v^2} \begin{bmatrix} xu + yv & -yu + xv \\ yu - xv & xu + yv \end{bmatrix} \tag{3-30}$$

上面求到了第一步中的旋转伸缩矩阵，此时，将两图像中左上角点平移重合，

所以得平移向量为：

$$TransVector = \begin{bmatrix} SrcLeft.x \\ SrcLeft.y \end{bmatrix} \tag{3-31}$$

这样，归一化图像中任一点坐标 $(Dst.x, Dst.y)$ 所对应的原图像中点坐标 $(Src.x, Src.y)$ 的关系为：

$$\begin{bmatrix} Src.x \\ Src.y \end{bmatrix} = RatateMatrix \begin{bmatrix} Dst.x - DstLeft.x \\ Dst.y - DstLeft.y \end{bmatrix} + TransVector \tag{3-32}$$

至此，我们也只是用到了绝对坐标，即每一个归一化图像中的任一点坐标都需要独立计算一次，并没有利用到相邻像素间的相关性。

对于归一化图像中，水平方向相邻的两个像素点 $(Dst.x, Dst.y)$ 和 $(Dst.x+1, Dst.y)$，其对应原图像点的坐标分别为 $(Src.x, Src.y)$ 和 $(Src.x', Src.y')$，根据公式有 3-31，容易得到：

$$\begin{bmatrix} Src.x' \\ Src.y' \end{bmatrix} - \begin{bmatrix} Src.x \\ Src.y \end{bmatrix} = RatateMatrix \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} RatateMatrix[0][0] \\ RatateMatrix[0][1] \end{bmatrix} \tag{3-33}$$

可见，每一行的像素变换可以通过前一个像素点坐标平移一个常值向量获得。

对于归一化图像中，竖直方向相邻的两个像素点 $(Dst.x, Dst.y)$ 和 $(Dst.x, Dst.y+1)$，其对应原图像点的坐标分别为 $(Src.x, Src.y)$ 和 $(Src.x', Src.y')$，根据公式有 3-31，容易得到：

$$\begin{bmatrix} Src.x' \\ Src.y' \end{bmatrix} - \begin{bmatrix} Src.x \\ Src.y \end{bmatrix} = RatateMatrix \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} RatateMatrix[1][0] \\ RatateMatrix[1][1] \end{bmatrix} \tag{3-34}$$

可见，每一列的像素变换可以通过上一个像素点坐标平移一个常值向量获得。

进一步观察公式 3-29，3-30 和 3-31，我们发现，由于输入的坐标都是整型数据，所以唯一出现浮点型数据仅在求旋转伸缩矩阵时做了除法（除以 u^2+v^2），又由于归一化后图像的尺寸是定值，因此，u^2+v^2 也是个定值，这样，定点运算中所有的误差来源仅仅是在这一次除法。实践证明，使用定点运算的基于相对位置的人脸归一化方法，在计算结果上，与浮点运算的基于绝对位置的人脸归一化

方法完全相同。

3.6 实验结果

在人眼检测领域，如何评价是否找到了正确的眼睛位置？目前，国内外的学者普遍采用下面的规则作为统一的评价标准。

定义归一化误差为：

$$\text{err}=\frac{\max(d_{left\_detect},d_{right\_detect})}{d_{standard}} \tag{3-35}$$

其中，d_{left_detect} 表示人眼检测子得到的左眼位置与人工标注的左眼位置之间的距离，d_{right_detect} 表示人眼检测子得到的右眼位置与人工标注的右眼位置之间的距离，$d_{standard}$ 表示人工标注的左眼和人工标注的右眼之间的距离。当归一化误差满足 $err<0.25$ 时，认为人眼检测输出了正确的人眼位置[43] [44]。如果测试库有 N 幅人脸图像，我们得到人眼检测子的检测率为[44]：

$$rate=100\%\times\sum_{i=1,err_i<0.25}^{N}1/N \tag{3-36}$$

这个评价标准使用了归一化距离，这样就去除了人脸图像尺寸不一致的影响。

为了评价我们的算法性能，在实验中我们与主流的人眼检测子进行了比较。BioID[45]测试库共有 1521 幅包含人脸图片，图像尺寸为 320*240。测试结果如表 3-1 所示。其中 SVM+边缘特征的方法由 Turkan M 等人[46]提出，MIC + Mean shift 的方法由 Valenti R 等人[47]提出，AdaBoost + AdaBoost 和 AdaBoost + mean shift 为本文使用的方法。由于不同的参数选择，可能导致不同的检测率和时间，表中列出的是两者的折中选择。

表 3-1　BioID 测试库上人眼检测性能比较

算法	检测率（%）	时间（ms）	CPU 主频（GHz）
SVM + 边缘特征	99.46	80	-
MIC + mean shift	97.45	0.4	2.4
AdaBoost + AdaBoost	98.90	11.32	2.5
AdaBoost + mean shift	98.31	1.8	2.5

从表中我们可以看出，我们的算法，不管是检测率，还是时间复杂度，都处在另外两种主流方法的中间。

另外，AdaBoost + mean shift 方法和 AdaBoost + AdaBoost 方法相比，检测率只有细致的下降，而时间下降了 6 倍以上。因此，AdaBoost + mean shift 方法在嵌入式系统中有广泛的应用前景。

为了更好的对算法进行调优，在我们的系统中，使用了一套新的评价标准，该标准中使用的归一化误差与公式 3-34 相同，这样，仍然可以以去除人脸图像尺度不同的影响。但是，只有当$err < 0.125$时，我们才认为人眼检测是正确的。

使用新的评价标准主要基于下面几个方面的考虑。首先，通过表 4-1 我们看到，人眼检测子的检测率已经达到了 97.96%，已经非常接近 1，这样的话，很多改进对算法性能的提升，并不能直接反应在检测率指标上。其次，在图 3-1 中，我们已经看到人眼检测的精度对后续人脸精度的影响巨大。在我们实验过程中，我们认为使用$err < 0.25$作为正确检测人眼的标准，对人脸比对来说，是一个非常粗糙的精度。因此，我们选择了更加精细的$err < 0.125$。

表 3-2 列出了对 AdaBoost + mean shift 进行算法调优过程中几个具有代表性的实验结果。该表中的检测率均使用了新的评价标准。

其中，第一列表示图像金字塔的变化方式，有指数（exp）和线性（linear）两种，数字表示尺度变化系数。从表 3-2 我们可以看出，指数变化的方式比线性变化的方式能得到较高的检测率。第二列表示人眼检测的搜索范围，第三列表示

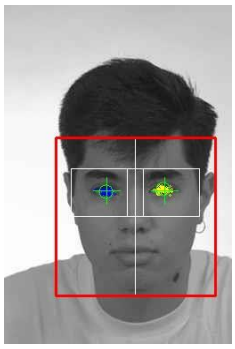
表 3-2　AdaBoost + mean shift 算法调优

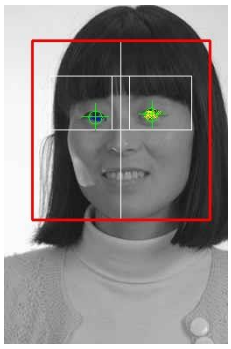
scale	Search area	search stribe	$\delta_x \backslash \delta_y \backslash \delta_s$	Rate & time	Data set
exp 0.8	(0,0)- (1,1)	xstep = 1 ystep = 1	2\1 \log1.25	95.86% 11.03ms	FERET- Fa
exp 0.8	(0.1,0.2)- (0.55, 0.5)	xstep = 2 ystep = 2	2\1 \log1.25	93.42% 1.88ms	FERET- Fa
linear 0.5	(0.1,0.2)- (0.55,0.5)	xstep = 2 ystep = 2	2\1 \0.5	91.90% 1.41ms	FERET- Fa
exp 0.8	(0.1,0.2)- (0.55,0.5)	xstep = 2 ystep = 2	2\1 \log1.25	88.36% 1.84ms	BioID

水平方向（xstep）和竖直方向（ystep）搜索时跳变的距离（search stribe），从第二行和第三行的结果我们可以看出，限制搜索范围，增大 search stribe 后，检测率仅仅下降了 2%，而速度增加了 5.8 倍！为嵌入式系统的实时应用提供了算法保证。第四列表示公式 3-6 中对应的参数。

图 3-13 给出了 FERET-Fa 测试库和 BioID 测试库上的部分人眼检测结果。

(a)





(b)

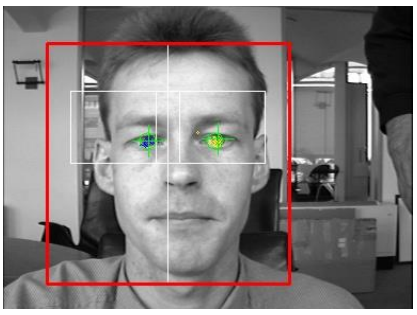


图 3-13　部分人眼检测结果

（a）FERET-Fa 测试库；（b）BioID 测试库

3.7 本章小结

本章详细介绍了人眼检测的方法和基于人眼位置的人脸归一化方法。人眼检测中，使用了两种不同的方法，两种方法均采用 Viola-Jones 人眼检测子作为人眼粗定位的方法，第一种方法采用 Viola-Jones 人眼验证子作为后处理的方法，第二种方法采用 mean shift 作为后处理的方法。人脸归一化中，分别介绍了基于绝对位置和相对位置的人脸归一化方法，并比较算法的优劣。最后，给出了实验结果，体现了我们的算法在实时系统中的优势。

第四章 人脸识别系统实现及优化

人脸识别有两种应用场景：1 对 1 和 1 对 N。所谓 1 对 1，指判断两幅包含人脸的图片，是否来自同一个人。所谓 1 对 N，指给定一幅包含人脸的图像，是否来自于人脸库中的某一个人，如果是，是哪一个人。对于 1 对 1 的情况，人脸识别系统需要两次人脸检测、两次人眼检测、两次人脸归一化和两次人脸比对，两次人脸比对包括两次人脸特征提取和一次相似度的计算。对于 1 对 N，人脸识别系统需要一次人脸检测、一次人眼检测、一次人脸归一化和 N 次人脸比对，N 次人脸比对包括一次人脸特征提取和 N 次相似度的计算。其中，人脸检测、人眼检测和人脸特征提取是整个系统中最耗时的部分。在前面两章中，主要介绍了人脸检测和人眼检测的实现和优化问题，在这一章中，将简述人脸比对的优化，尤其是人脸特征提取的优化。

另外，我们的系统已广泛应用于嵌入式系统中，如 DPS、ARM 等。为此，本章还将阐述详细阐述一个复杂的算法如何移植到嵌入式系统中。

4.1 模块划分

在前面两章中，我们介绍了人脸检测和人眼检测的算法原理。在这一章的开始，我们将首先按照功能划分系统模块。虽然，对于 1 对 1 和 1 对 N 两种应用场合，算法的核心思想大致相同，但他们的系统模块却不尽相同，分别见图 4-1 和图 4-2 所示。

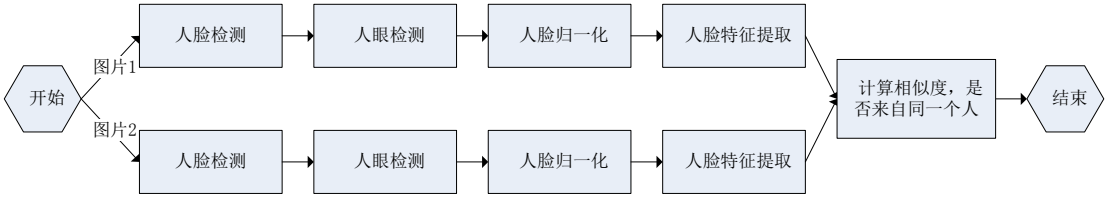


图 4-1　1 对 1 人脸识别

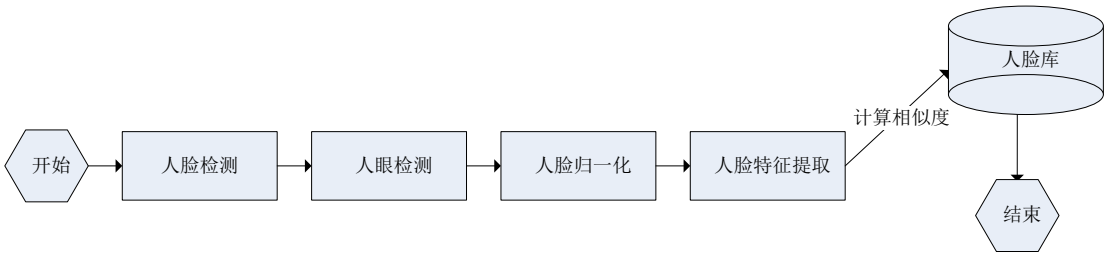


图 4-2　1 对 N 人脸识别

在图 4-1 和图 4-2 两个系统中，人脸检测、人眼检测和人脸归一化算法均采用前面两章中提到的方法，而人脸特征提取的方法和相似度计算的方法，将在本章第二节中作简单介绍。另外，从图 4-1 中，我们发现，设计良好的系统，应该可以使图片 1 和图片 2 的处理过程并行起来，以充分利用计算机的计算能力。

4.2 人脸比对优化

人脸比对，通常包含两个部分：特征提取和相似度计算。特征提取的目的，是找到一个人脸描述子，这个描述子具有这样的性质，首先，能最大限度的缩小同一个人在不同场景中的图像，例如不同光照，不同表情，不同姿态等，其次，能够最大限度的区分来自不同人的图像。特征提取往往得到一个描述人脸图像的特征向量，如何定义一个计算方法，使同一个人的人脸图像相似度尽量高，而不同人的人脸图像相似度尽量低，就是相似度的计算。

早期，往往使用几何特征来描述人脸，该方法首先定位脸部特征，如眼睛、眉毛、鼻子、下巴等，然后利用这些面部特征之间的相互关系来生成人脸描述子，如直接计算这些脸部特征之间的距离作为描述子。但是，使用几何特征来描述人脸，首先需要精确的定位脸部特征，这本身就是一项非常具有挑战性的任务，其次，几何特征对人脸表情、姿态等十分敏感，并不能很好的描述人脸。

当前，主流的人脸识别算法普遍采用统计特征来描述人脸，如弹性图匹配的方法[48]，使用马尔科夫模型的方法[49]和使用形状、纹理的方法[50]。最近十年来，研究发现，Gabor 滤波器对光照、表情、姿态变化等具有较强的鲁棒性[51]，因此，基于 Gabor 滤波器的方法得到越来越多的关注。在我们的系统中，正是使用了一

种基于 Gabor 方向直方图的人脸描述子，称为 GOH(Gabor Orientation Histogram)[52]。

4.2.1 人脸比对算法简介

GOH 人脸描述子的生成主要分成两步。第一步，使用一组 Gabor 滤波器与原始图像作卷积，得到 Gabor 幅度图像 GMI(Gabor Magnitude Image)。第二步，根据 Gabor 幅度图像，得到 Gabor 方向直方图 GOH。下面对这两部分别做阐述。

Gabor 滤波器组的表达式如公式 4-1 所示。

$$G(\mathbf{x}) = \frac{\|\mathbf{k}\|^2}{\sigma^2}\exp(-\frac{\|\mathbf{k}\|^2\|\mathbf{x}\|^2}{2\sigma^2})\left[\exp(i\mathbf{k}\bullet\mathbf{x}) - \exp(-\frac{\sigma^2}{2})\right] \qquad (4\text{-}1)$$

其中，$\|\bullet\|$ 表示 2-范数，$\mathbf{k}\bullet\mathbf{x}$ 表示两个维度相同的向量的内积。向量 $\mathbf{k} = [k_x, k_y]^T = [k_v\cos\varphi_u, k_v\sin\varphi_u]^T$，其中，$k_v = 2^{-(v+2)/2}\pi, v = 0,1,...V-1$ 表示 Gabor 滤波器的尺度因子，V 表示滤波器组中不同尺度的个数。角度 $\varphi_u = \pi u / U, u = 0, 1, U.$ 表示 Gabor 滤波器的方向性因子，U 表示滤波器组中不同方向的个数。σ 用于确定高斯包络振动频率。

Gabor 滤波器组的直观意义非常明确，如下图所示。

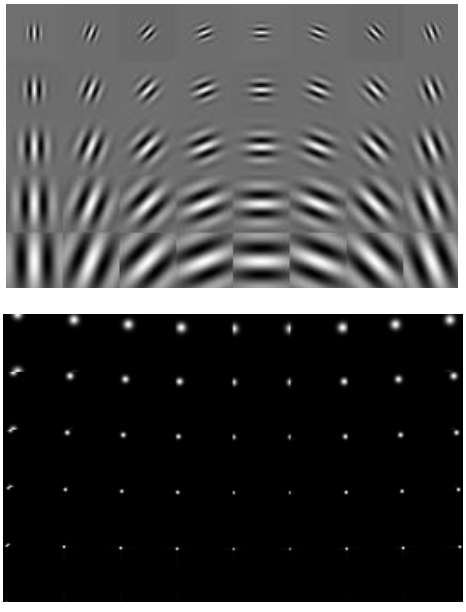


图 4-3　Gabor 滤波器组的直观意义（5 个尺度、8 个方向）

（a）时域特性；（b）频域特性

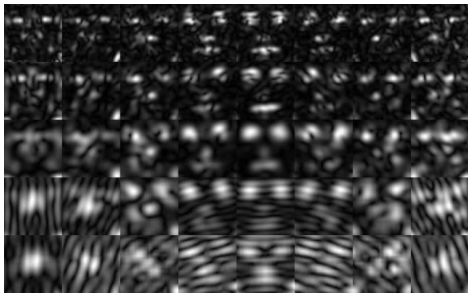
图像中某一个像素 $I(x,y)$ 经过 Gabor 滤波器得到的响应可以表示为：

$$O_{u,v}(x,y)=I(x,y)*G_{u,v}(x,y)=\rho_{u,v}(x,y)e^{j\theta_{u,v}(x,y)} \qquad (4\text{-}2)$$

其中，*为卷积运算符，$\rho_{u,v}(x,y)$ 表示幅度响应，$\theta_{u,v}(x,y)$ 表示相位响应。一幅人脸图像，经过 Gabor 滤波器组得到的响应如图 4-4 所示。

(a)

(b)



(c)

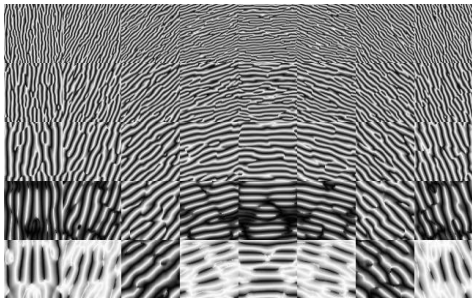


图 4-4　Gabor 滤波器组的响应

（a）归一化的人脸图像；（b）幅度响应；（c）相位响应

从图中可以看出，Gabor 滤波器的幅度响应比相位响应能更好的表达人脸，在幅度响应中加入相位响应，仅仅可以细微的提高识别的精度，却使特征维度增加了一倍，计算复杂度相应的大幅度提升。因此，GOH 中仅仅利用了幅度响应。

在我们的系统中，Gabor 滤波器组共有 5 个不同的尺度，平面内的 360° 被均匀量化成 8 个不同的方向，因此，对于一幅高 H，宽 W 的图像，幅度响应的维数为 $5\times8\times W\times H=40WH$。为了减少向量的维度，我们把图像均匀的划分成 B 个块，对于每一个块 b，$b\in\{1,2,...B\}$，我们得到他在尺度 v 和方向 u 上的幅度响

应为：

$$S_{u,v}(b)=\sum_{(x,y)\in b}\rho_{u,v}(x,y) \qquad (4\text{-}3)$$

其中，$(x,y)\in b$ 表示像素 (x,y) 在块 b 内部。这样，幅度响应的维数下降至 $40B$。

另外，我们知道，我们使用的 8 个不同的方向，覆盖了平面内所有的角度，即构成了一个划分，因此，我们可以认为，在某一个尺度 v 下，所有方向 $0,1,...U$ 上的幅度响应构成一个概率分布，某一个方向 u 的概率值正比于它的幅度响应。因此，我们对方向因子做 1-范数归一化，可以得到对应的概率分布为：

$$H_v(b)=norm1([S_{0,v}(b),S_{1,v}(b),...S_{U-1,v}(b),]) \qquad (4\text{-}4)$$

其中，$norm1(\bullet)$ 表示 1-范数归一化。接下来，再把分块 b 内不同尺度 v 都连接起来，构成分块 b 的特征向量为：

$$H(b)=[H_0(b),H_1(b),...H_{V-1}(b)] \qquad (4\text{-}5)$$

最后，把所有 B 个分块连成一个完整的 GOH 描述子为：

$$HS=[H(1),H(2),...,H(B)] \qquad (4\text{-}6)$$

至此，我们得到了 GOH 描述子所描述的特征向量 HS，整个过程如图 4-5 所示。

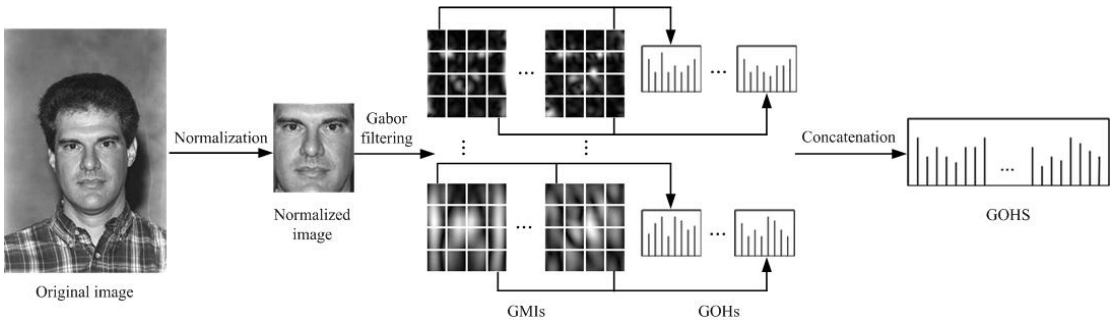


图 4-5　GOH 人脸描述子的提取过程[52]

有了特征向量 HS，就可以使用多种多样的形式对特征向量计算相似度了。在我们的系统中，为了增加算法鲁棒性，我们采用了 FLDA（Fisher Linear Discriminant Analysis）[53]对 GOH 描述子进行降维，并对降维后的特征向量取余弦距离（即两个特征向量的夹角的余弦值）作为两个特征向量相似性的度量。最后，我们需要设定一个阈值（该阈值也可以通过训练得到），大于该阈值，认为两幅

人脸图像来自同一个人，小于该阈值则认为不是同一个人。

在我们的早期的系统实现中，提取一幅 88*80 的归一化人脸图像的 GOH 特征向量，需要耗费 280ms（硬件平台为 Intel Xeon CPU 2.53GHz，2.00GB 的内存），而相似度的计算所需要的时间微乎其微。通过分析发现，在提取 GOH 特征向量时，主要时间耗费在 Gabor 滤波器组的卷积计算上。由于归一化人脸图片的尺寸（我们的系统中为 88*80pixel）较大，因此，把时域的卷积运算转换成频域的乘积运算能得到较快的结果。我们一共有 40 个 Gabor 滤波器，因此，我们提取一幅人脸图片的特征，共需 1 次 FFT 变换和 40 次 IFFT 变换。为了对系统实现进行优化，有必要对 FFT 算法的实现作进一步的调查和研究。

4.2.2 FFTW 开源库简介

FFTW（Fastest Fourier Transform in the West）[54]一个颇为古怪的名字，是由 MIT 的 Matteo Frigo 博士和 Steven G. Johnson 博士开发的一个完全免费的软件包。FFTW 最初的 release 版本于 1997 年发布，最新的 release 版本 3.2.2 于 2009 年 7 月发布。它是一个 C 语言开发的库，支持任意大小的、任意维数的数据的离散傅里叶变换(DFT)，并且还支持离散余弦变换(DCT)、离散正弦变换(DST)和离散哈特莱变换(DHT)。

根据两位博士的测试，FFTW 在计算速度上远远优于目前其它计算 DFT 的免费的库，甚至可以和收费的 DFT 库相媲美。但是，和收费的 DFT 库相比，FFTW 能够轻松的在不同的平台上移植。我们熟悉的 MATLAB，就是调用了 FFTW 来实现 DFT/IDFT 变换的。

另外，值得一提的是，FFTW 还获得了 1999 年的 J. H. Wilkinson Prize for Numerical Software 奖[55]。J. H. Wilkinson Prize for Numerical Software 每四年颁发一次，用于奖励那些“最好的解决了高质量的数值计算问题的软件设计”。

FFTW 有以下优越的特性[56]：

1. 高速——远优于目前其它的免费 DFT 库。
2. 支持任意维度的变换。

3. 支持任意大小的变换——FFTW 对 N = 2^a * 3^b * 5^c * 7^d * 11^e * 13^f 的变换处理的最好，其中 e + f = 0 或 1，其它指数可以为任意值。
4. 支持快速的输入为实数的 DFT 变换。
5. 支持多线程。
6. 支持并行处理。
7. 可移植性——任意包含 C 编译器的平台都可以使用 FFTW。

FFTW 是当今世界公认的最快的 FFT 算法，已经受到越来越多的科学研究和工程计算工作者的青睐，并为量子物理、光谱分析、音视频流信号处理、石油勘探、地震预报、天气预报、概率论、编码理论、医学断层诊断等领域提供切实可行的大规模 FFT 计算。

4.2.3 FFTW 的在人脸比对中的应用

为了将 FFTW（Fastest Fourier Transform in the West）[54]引入到我们的算法实现中，需要注意以下两个方面。

首先，我们需要对归一化人脸图像作一次 FFT 变换，得到它的频域表达。显然，人脸图像的输入都是实数，对于输入均为实数的二维数组，FFT 的输出具有共轭对称的性质，在 FFTW 中，作者利用该性质来节省内存的使用。而我们在使用的时候，需要把这种内存压缩的表示形式还原为普通的表示形式。

对于一维的情况，假设输入序列为 x(n),n=0~N-1，对应的 DFT 为 X[k],k=0~N-1。我们知道，X(n)有共轭对称性，即：

$$X[k] \leftrightarrow X[N-k]$$

当输入序列均为实数时，FFTW 的存储策略如图 4-6 和图 4-7 所示。

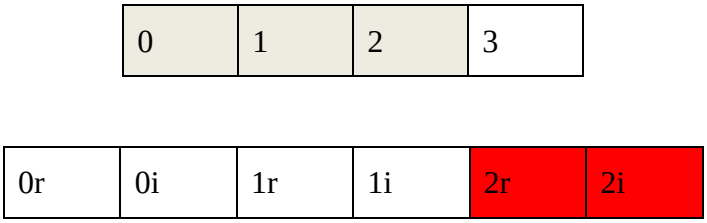


图 4-6　输入序列为偶数个实数时，FFTW 输出的存储策略

（a）时域表示；（b）频域表示

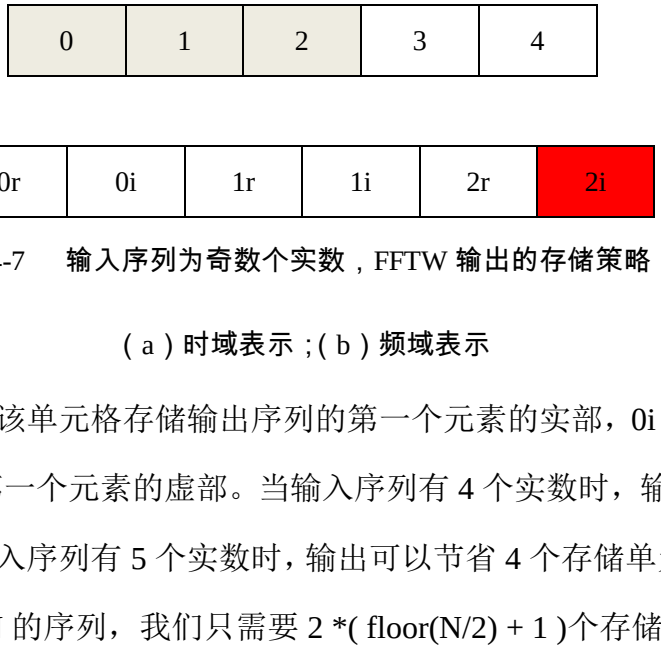


图 4-7　输入序列为奇数个实数，FFTW 输出的存储策略

（a）时域表示；（b）频域表示

图中，0r 表示该单元格存储输出序列的第一个元素的实部，0i 表示该单元格存储输出序列的第一个元素的虚部。当输入序列有 4 个实数时，输出可以节省 2 个存储单元，当输入序列有 5 个实数时，输出可以节省 4 个存储单元。更一般的，对于输入长度为 N 的序列，我们只需要 2 *(floor(N/2) + 1)个存储单元。相比于输入长度，N 为奇数时，输出长度增加一个单元；N 为偶数时，输出长度增加两个存储单元。

其次，GOH 特征向量的生成过程中，使用 40 个 Gabor 滤波器对原图像作卷积，相当于使用滤波器的频域模板和归一化人脸图像的频域表达相乘后，作 40 次 IFFT 变换，才能得到归一化人脸图像的幅度响应。而我们知道，这 40 次 IFFT 变换，他们的尺度相同。FFTW 针对这种情况，使用 wisdom 策略[56]，做了进一步的优化。

最终，通过 FFTW 的引入，提取一幅归一化人脸图像的特征向量的时间，由原来 280ms 降至 50ms，效果十分显著！

4.3 针对嵌入式平台的优化

至此，我们已经把人脸识别系统的每一个部分的原理、实现以及如何优化做了详细介绍。目前，这个系统能够非常实时的运行在 PC 平台上。然而，随着移动互联网的迅猛发展，人们对移动终端上的人脸识别需求也越来越强烈。因此，我们把人脸识别系统移植到嵌入式平台上。

嵌入式平台有别于 PC 平台，它的便携性和机动性，给用户带来了前所未有的体验，使嵌入式的人脸识别有着非常强的现实意义和广泛的潜在应用价值。但

是，嵌入式平台由于其固有的特性，使我们的算法移植至嵌入式平台时，需要作具有针对性的优化。

4.3.1 嵌入式平台的特点

嵌入式平台有其自身的，不同于 PC 平台的特点，这些特点能够用来指导我们针对嵌入式平台的优化。

首先，嵌入式平台的操作系统多种多样。常见的嵌入式操作系统有嵌入式 Linux，Windows CE，Windows XP Embedded，Windows Vista Embedded，VxWorks，uCOSII 等。不同的操作系统，有不同的底层接口可供调用，因此，不同操作系统间的底层接口是无法移植的。然而，不同操作系统之间，并非没有共性。目前，至少据我们所知的，所有的已知操作系统，对标准开发语言的支持是一样的，虽然不同的操作系统，使用不同的开发工具，但同样的标准代码表达同样的含义。

其次，几乎所有的嵌入式应用，都要求应用的实时性。例如，我们的系统，目前正运行在一个门禁系统中，可以想象，如果一个用户站在摄像头前面，无法实时的得到正确的识别结果，用户是无法容忍的。因此，一个可用于嵌入式平台的人脸识别系统，必须具备实时的运行效率。

最后，绝大部分的嵌入式平台，其硬件水平远低于 PC 平台。嵌入式平台往往具有较少的 RAM 可用，而且微处理器的主频往往低于 PC 平台。因此，在 PC 平台上能够实时、流畅的运行的应用，往往不能保证在嵌入式平台仍然可以实时流畅的运行。

为此，除了在算法层面作细微的调整外，我们对算法的实现还作了如下的处理。

首先，我们把所有的 C++语言实现修改为 C 语言。虽然，Bruce Eckel 曾在《Thinking in C++》中指出，好的 C++代码的执行效率应该只比对应的 C 代码慢 5%以内，但是，我们并不能保证我们的代码是否符合 Bruce Eckel 所说的“好的 C++代码”，另外，更重要的一点是，很多针对嵌入式平台的编译器，往往能够对 C 代码做更好的优化，这是 C++代码所不具备的。

其次，我们把所有实现中的浮点运算，优化成定点运算。这个优化的目的是显而易见的，浮点运算所需要的时钟周期，比定点运算所需要的时钟周期要大不少。由于嵌入式平台硬件水平的限制，这个优化所带来的收益往往是非常可观的。

4.3.2 C++版本到 C 版本的优化

C++语言与 C 语言最大的区别，在于 C++使用类，使用面向对象的思想，而 C 使用面向过程的思想。为了把 C++版本修改为 C 版本，需要把 C++的类退化成 C 的结构体。有两种方法，第一，阅读并理解所有 C++代码，使用 C 语言重写，该方法工作量大。第二，不需要理解 C++代码，只需要按照类的特性，逐一把类中成员退化为结构体成员即可。该方法只需要逐个修改 C++类，几乎不会出错，该方法修改迅速，但程序的可读性降低。我们采用了第二种方法，具体的修改方法如下。

第一，我们考虑类的成员函数和成员变量。注意，我们这里提到的成员函数仅仅指最普通的成员函数，不包含构造函数、复制构造函数、析构函数等，这些函数将在下文有更详细的介绍。

对于类的成员变量，可以直接转为 C 中结构体的数据成员。这样做仅仅是去了该成员变量在 C++类中的访问权限，而不会影响到代码的执行。

对于类的成员函数，则需要转化为 C 中结构体的函数指针。如果在 C++中的成员函数中出现了 virtual、inline 等修饰符，在 C 结构体中也要去掉。另外，我们知道，类的成员函数的形参中有一个隐含的 this 指针，该指针指向该类的对象，为此，我们在 C 结构体中，函数指针需要增加一个形参，用于显示的指明该指针。如类类型 B 的成员函数为 void fun(int a)，则在 C 结构体中的函数指针应该声明为 void (*fun)(struct B *ths, int a)。

对于类的静态成员，包括静态成员函数和静态成员变量，由于该类的所有对象公用一个成员，因此，需要把他们定义为全局函数或全局变量。

第二，考虑类的构造函数。C++类在实例化的过程中，会隐式的调用类的构造函数。因此，一方面，我们需要在 C 结构体中定义一个函数指针 void (*A)(struct

B *ths, …)，该函数指针指向一个函数，该函数的功能相当于 C++类的构造函数。另一方面，我们在 C 代码中定义了一个结构体之后，需要显示的调用函数指针 A，用于模拟 C++中构造函数的隐式调用。最后，需要注意的是，在 C++中内存管理的函数，如 new，delete 等，均需替换成 C 语言中的内存管理函数，如 malloc 和 free 等。

第三，考虑类的析构函数。类的析构函数与类的构造函数类似，在类的对象被销毁的时候，被隐式的执行。因此，一方面，我们需要在 C 结构体中定义一个函数指针 void (*U)(struct B *ths, …)，该函数指针指向一个函数，该函数的功能相当于析构函数。另一方面，在 C 代码中，当结构体被销毁时，或结构体的生命周期结束时，我们需要显式的调用函数指针 U。

第四，考虑类的复制构造函数。类的复制构造函数一般出现在下面三种情况中：作为参数传给函数、实例化类时作为参数、作为函数的返回值。对于前两种情况，我们可以把这些函数参数修改为指针。对于最后一种情况，也可以将返回值包装成指针的形式，作为函数的形参传递。这样一来，复制构造函数在 C 结构体中就可以不必考虑了。

第五，考虑类的重载。我们知道，重载分为函数重载和操作符重载。对于函数重载，在 C 结构体中，只能给重载的函数取不同的名字。对于操作符重载，这里仅仅以‘=’号的重载为例。值得注意的是，这是一个非常容易出错的地方，在我们实现过程中，就因为疏忽了‘=’号重载，导致系统的调试多费了一周左右的时间。

在 C++语言中，表达式 C=D，其中 C 和 D 为类对象，是调用了对象 C 的‘=’重载操作符，这个重载函数中的实际操作，并不仅仅限于类成员的赋值，完全可能还有别的内容。在 C 语言中，C=D，其中 C 和 D 为结构体对象，仅仅是把 C 的所有成员赋值为 D 对应的成员，这里的成员不仅包括成员变量，还包括成员函数指针。因此，C++语言的‘=’所对应的实际操作，要远远丰富于 C 语言中的‘=’，极端的时候，这两个操作甚至可以没有交集。为此，我们需要把 C++语言中的‘=’操作，替换成 C 语言中的一个函数指针 void operator_Equal(struct

B *c, struct B *d)，才能保证 C++程序与 C 程序的等价性。

最后，考虑类的继承。由于多继承一般不用，或者很少使用，因此，这里仅仅介绍单继承。首先把基类改成 C 程序，然后把 C 实现的基类拷贝到子类的实现中。如果子类中有对基类成员函数的覆盖，则要把函数指针赋值为子类的函数。

4.3.3 浮点运算到定点运算的优化

在许多的嵌入式平台中，由于硬件水品的限制，浮点运算往往效率低下， 甚至在有些 DSP 上，浮点运算无法进行。因此，我们需要把程序中的浮点运算优化成定点运算。

一个整型数的最大表示范围取决于字长，常见的有 16 位和 32 位。在我们的系统中，为了方便，所有的转化都使用了 32 位字长。那么，如何使用 32 位字长表示浮点数呢？关键就在于，通过设定小数点在 32 位字节中不同的位置，就是数的定标，常用 Q 表示。不同的 Q 所表示的数，不仅取值范围不同，而且精度也不相同。Q 越大，数值范围小，但是精度越高；相反，Q 越小，数值范围越大，但精度越低。

浮点运算到定点运算优化过程中，一个主要问题就是定标 Q，即小数点在 32 位数的哪一位。定标 Q 确定后，浮点数 x 和定点数 x_q 之间的转换关系为：

$$x_q = (\text{int})(x \cdot 2^Q) \tag{4-7}$$

$$x = (float)(x_q \cdot 2^{-Q}) \tag{4-8}$$

那么，给定一个浮点型变量 x，如何确定变量的 Q 值呢？确定 Q 值，本质就是确定 Q 的取值范围，知道了 Q 的取值范围，我们就可以在保证变量不溢出的情况下，尽量提高变量的精度。另外，由于定点数在表示正数和负数时具有的对称性，因此，对于同时可能取正值和负值的的浮点数，确定的取值范围，往往是要浮点变量的绝对值的取值范围，或者说最大值。

假设，浮点型变量 x 的绝对值的最大值为 M，取一个整数 n，使得 n 满足：

$$2^{n-1} < M < 2^n \tag{4-9}$$

显然，一旦确定了 M，那么 n 的取值就是存在且唯一的。对于 32 位字长的定点

数 x_q，我们令：

$$M < 2^n = 2^{32-Q} \tag{4-10}$$

这样确定的 Q 值，就能保证定点数能够表示的数的取值范围 2^{32-Q} 大于浮点数的绝对值的最大值。故：

$$Q = 32 - n \tag{4-11}$$

如何确定浮点数 x 的绝对值的最大值呢？常见的有两种方法。

理论分析法。这种方法适用于有明确含义的物理量，这些物理量往往具有明确的数学表达式，如数字信号处理中的汉明窗，某一汉明窗的表达式为 $y(n)=0.54-0.46\cos\left(\frac{2\pi n}{N-1}\right), 0 \le n \le N-1$，则我们可以轻易的获得 $y(n)$ 的绝对值的取值范围，$|y(n)| \le 1.0$。而对于物理含义不明确的变量，或者没有明确的数学表达式的变量，则需要用到第二种方法，即统计分析法。

所谓统计分析法，就是用足够多的样本作为输入，并使这些样本尽可能覆盖各种可能情况，统计在这些输入的情况下，变量的绝对值的最大值。使用这个方法，一定要使用尽可能多的样本，而且这些样本要尽可能覆盖所有可能的情况。当然，在实际工作中，覆盖所有可能的情况是不可能的，因此，定标 Q 时，往往留出一定的空余，牺牲一些精度，以保证实际运行时定点数 x_q 不会溢出。

确定了定点数的标值 Q，在优化程序的时候，就需要对这些浮点数参与的运算做优化，常见的有加法，减法，乘法，除法，指数，对数等。这里，我们仅仅介绍最常用的加法和乘法，减法与加法原理类似，除法与乘法原理类似。

把浮点数加法优化为定点数加法时，最重要的一点是保证两个操作数在运算前定标值相同，若不相同，则需要将定标值小的数扩大，调整为与另一个数相同的定标值。此外，还需注意加法的结果是否会溢出，如果溢出，需要保存溢出的结果。

假设有是三个浮点数，他们的定标值分别为 Q_x, Q_y, Q_z，满足 $z = x + y$。则对应的定点运算满足：

$$z_q \cdot 2^{-Q_z} = x_q \cdot 2^{-Q_x} + y_q \cdot 2^{-Q_y} \quad (4\text{-}12)$$

不失一般性，我们不妨假设$Q_x > Q_y$，得到定点数z_q为：

$$z_q = [x_q + y_q \cdot 2^{Q_x - Q_y}] \cdot 2^{Q_z - Q_x} \quad (4\text{-}13)$$

这里的三个定标值Q_x, Q_y, Q_z均可以用上面提到的方法提前确定，因此，z_q可以由上式求得。

把浮点乘法优化为定点乘法时，如果两个乘数都大于 1，那么结果的绝对值将是一个很大的值，导致结果的定标值 Q 不得不取的很小，牺牲了很多精度。如果精度确实不够，可以考虑使用两位 32 位字长的整型变量表示一个浮点数。幸运的是， 我们的系统中并没有出现这样的情况。

假设有三个浮点数x, y, z，满足$z = x \cdot y$。他们对应的定标值分别为Q_x, Q_y, Q_z。则对应的定点运算满足：

$$z_q \cdot 2^{-Q_z} = x_q \cdot 2^{-Q_x} \cdot y_q \cdot 2^{-Q_y} \quad (4\text{-}14)$$

得到乘积z的定点表示z_q为：

$$z_q = x_q \cdot y_q \cdot 2^{Q_z - Q_x - Q_y} \quad (4\text{-}15)$$

这里的三个定标值Q_x, Q_y, Q_z，同样应用上面提到的方法可以唯一确定，因此，z_q可以用公式 4-15 唯一确定。

在实际应用中，往往采用下面的策略，对一个浮点程序做定点优化。首先，使用统计分析方法，得到所有浮点数的取值范围，并依此得到他们的定标值 Q，并将这些 Q 值定义为全局可见的宏（或者全局变量，但是这样浪费内存空间），然后，对这些浮点数参与的运算，运用上面提到的原则，逐步转化为定点运算。为了高效的定位优化过程中可能出现的溢出或者精度问题，我们可以对所有的浮点数逐个优化，每次只优化一个浮点数，完成后，重新在测试集上测试一遍，以保证该浮点数的定点化没有溢出或精度问题。

4.4 基于 Android 系统的应用

我们的系统，除了运行在一个 PC 端、一个嵌入式门禁系统中，还运行在一个基于 Android 操作系统的移动终端上。我们知道，Android 操作系统的 SDK 是基于 Java 实现，这意味着，基于 Android SDK 进行开发的第三方应用，都必须使用 Java 语言。而我们的人脸识别系统，使用的全部是 C/C++语言，那么，我们是否需要把所有代码都用 Java 重写一遍吗？

首先，我们不能这么做。由于 Java 代码的低效率，这样做将十分严重的算法的执行效率。我们在本文的大量篇幅中，提到如何优化算法，包括算法层面、C++语言修改为 C 语言、浮点运算优化为定点运算，一旦我们使用 Java 重新实现，所有这些努力将白费。

其实，为了能在 Android 应用程序中调用 C/C++程序，我们可以使用 Google 公司在 2009 年 6 月发布的 NDK（Native Development Kit）。NDK 是一个工具集，这个工具集允许开发者在 Android 应用里使用 C/C++等低级语言，作为应用的一部分。虽然，使用 NDK 可以在应用中使用已有的代码，然而，并不是所有的应用都需要 NDK，NDK 对大多数应用来说是无用的[57]。

把人脸识别的 C/C++程序应用到 Android 应用程序中， 需要经过一层一层的包装，最后才能被应用程序的 Java 代码所调用。包装过程如图 4-8 所示。

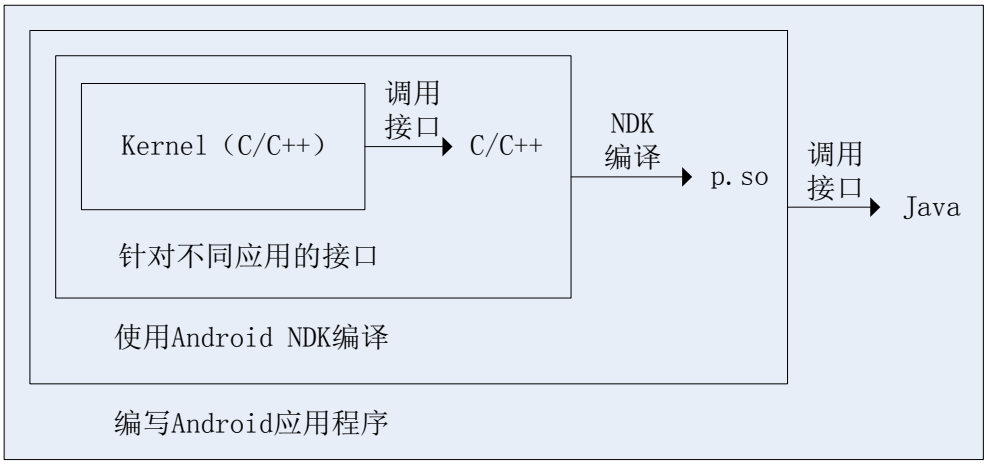


图 4-8　基于 Android 系统的应用

从图 4-8 我们可以看出，我们算法的核心部分使用 C/C++实现，针对不同的项目，不用的应用，我们调用核心算法的接口，得到 C/C++接口。然后，我们使

用 Android NDK 编译上面的 C/C++程序，可以得到一个 p.so 文件，so 文件是 Linux 下的动态链接库文件，类似于 Windows 操作系统下的 dll 文件。有了 so 文件，开发者就可以使用 Java 语言，直接调用我们提供的接口。

使用 Android NDK，一方面，可以利用 C/C++语言高效、灵活的内存管理的特性，编写耗时耗资源的核心算法，另一方面，可以继续使用已有的 C/C++语言核心算法库，而不必重新用低效的 Java 语言重写。

4.5 本章小结

在本章中，我们首先按照功能，对人脸识别系统做了模块划分。紧接着，对人脸比对模块，进行了简要的原理介绍，并详细介绍了 FFTW 开源库在该算法中的应用，效果十分显著。然后，我们针对嵌入式平台的特点，详细的介绍了两种优化方法，一种是在不需要读懂源代码的情况下，把 C++程序修改为 C 程序，一种是把浮点运算优化成定点运算。最后，针对 Android 系统，我们使用 Android NDK，使 Java 程序员可以在 Android 应用程序中调用我们提供的人脸识别系统的接口。

第五章 总结与展望

5.1 总结

随着计算机视觉技术的不断发展，以及计算机硬件技术的突飞猛进，越来越多的场合提出了对人脸识别的应用需求，例如，门禁系统，各种移动终端的应用，智能监控等。各种各样的应用场景，对人脸识别的技术提出了不同的要求。本文以人脸识别技术为基础，从算法设计、算法优化和针对特定平台优化三个角度出发，做了深入细致的研究。综合起来，本文的工作内容有以下几项。

1. 本文深入研究了 Viola-Jones 人脸检测子的各个关键步骤，包括基于 Haar 特征的弱分类器、基于 AdaBoost 算法构造的强分类器以及级联结构，并对算法提出了细致的优化策略，包括图像金字塔线性尺度变化策略，子窗口的屏蔽搜索策略。实验结果表明，我们的优化策略，不仅能有效的提升了算法的执行效率，而且能够小幅度的提高检测率。
2. 借鉴人脸检测和行人检测，本文提出了一种人眼检测的框架。该框架把人眼检测分为粗定位和细定位两个步骤。粗定位采用的方法均类似于 Viola-Jones 人脸检测子。细定位中，一种方法是采用 Viola-Jones 人眼验证子，该方法利用了更多的信息，速度慢，但有利于后面的人脸比对，一种方法是采用 mean shift 寻找核密度估计的极值点，该方法速度快，适用于实时性要求高的嵌入式领域。
3. 本文深入研究了二维平面的仿射变换，提出了一种基于相对位置的人脸归一化算法，该算法充分利用了相邻像素间的相关性，使人脸归一化的执行效率得到了极大的提升。该算法的提出，立即解决了定点运算的人脸归一化精度不高的问题。
4. 本文深入研究了基于 Gabor 方向直方图（GOH）的人脸比对算法，并在熟悉了 FFTW 库（被公认为目前最快的 FFT 算法）的基础上，创造性的

把它应用于人脸比对算法的实现。实验结果表明，FFTW 的引入，使人脸比对的执行时间减少至原来的 18%。

5. 本文针对人脸识别系统在嵌入式领域的应用，将 C++语言实现的人脸识别系统修改为 C 语言的实现，同时将浮点运算优化至定点运算。另外，针对 Android 系统，本文使用 Android NDK 方法，使 Java 程序员可以在 Android 应用程序中调用我们提供的人脸识别系统的接口。

5.2 展望

人脸识别系统是一个有着广泛应用前景的研究课题。本文作者在近两年的时间里，阅读了大量相关文献，对已有的人脸检测和人脸比对算法进行优化，提出了两种人眼检测的方法，同时改进了人脸归一化的方法。虽然系统性能有了长足的进步，但仍然存在着一些值得继续研究的问题。

1. 优化后的人脸检测算法，虽然有些人脸也被正确的找出，但是由于给出只是一个非常粗略的框，因此，可能对后期进一步处理带来不便，如人眼检测等。但是，在当前的人脸检测指标中，并没有什么指标能够用以限制此类问题的出现。因此，或许有必要进一步制定一个更加精确细致的人脸检测算法的指标。
2. 在已有的人眼检测中，使用的特征为 Haar 特征，该特征能很好的用于人脸检测，但并不是最好的人眼的特征，人眼最明显的特征在于眼睑和眼球的弧线，如何表达这些特征，或许是进一步提高人眼检测精度的好方法。
3. 在人脸识别系统中，人脸检测为人眼检测提供一个人脸框，最终目的是定位人眼。那么，我们是否能够找到一种方法，这种方法结合人脸检测和人眼检测，能够直接寻找人眼的位置。如果能够找到这样的方法，人脸识别系统的执行效率有望大幅度提升。
4. 在 C++程序修改成 C 程序的过程中，几乎所有的工作都是根据语法特性而开展。那么，我们是否能够编写一个程序，该程序能够自动的将 C++

程序修改成 C 程序，或者至少是部分 C++特性修改为 C 特性，这样，C++程序修改成 C 程序的工作量将几乎减少至 0.

参考文献

[1] S. Z. Li and A. K. Jain, *Handbook of face recognition*: Springer, 2011.

[2] C. Zhang and Z. Zhang, "A survey of recent advances in face detection," *Microsoft Research, June*, 2010.

[3] P. J. Phillips, H. Moon, S. A. Rizvi, and P. J. Rauss, "The FERET evaluation methodology for face-recognition algorithms," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, pp. 1090-1104, 2000.

[4] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld, "Face recognition: A literature survey," *Acm Computing Surveys (CSUR)*, vol. 35, pp. 399-458, 2003.

[5] 聂祥飞, "人脸检测和识别中若干问题研究," 北京: 北京邮电大学, 2007.

[6] T. A. H. Olstad B, "Encoding of a Priori information in active conntour models," *IEEE Transactions on pattern analysis and machine intelligence*, pp. 863-872, 1996.

[7] V. J. C. Lades M, Buhmann J, et al, "Distortion invariant object recognition in the dynamic link architecture," *IEEE Trans. on Computer*, vol. 42(3), pp. 300-311, 1993.

[8] K. S. Y. Lin S H, Lin L J, "Face Recognition with radial basis function (RBF) neural networks," *IEEE Trans. on Neural Networks*, vol. 8(1), pp. 114-132, 1997.

[9] A. T. Othman H, "A separable low complexity 2D HMM with application to face recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 25(10), pp. 1229-1238, 2003.

[10] P. A. P. Turk M A, "Eigenfaces for recognition," *Journal of Cognitive Neuroscience*, vol. 3(1), pp. 71-86, 1991.

[11] P. K. N. Lu J, Venetsanopoulos A N, "Face recognition using LDA-based algorithms," *IEEE Trans. on Neural Networks*, vol. 14(1), pp. 195-200, 2003.

[12] M. J. R. Bartlett M S, Sejnowski T J, "Face recognition by independent component analysis," *IEEE Trans. on Neural Networks*, vol. 13(6), pp. 1450-1464, 2002.

[13] A. F. Baudat G, "Generalized discriminant analysis using a kernel approach," *Neural Computation*, vol. 12(10), pp. 2385-2404, 2000.

[14] 李月敏, 陈杰, 高文, and 尹宝才, "快速人脸检测技术综述," in *全国第 16 届计算机科学与技术应用 (CACIS) 学术会议论文集*, 2004.

[15] M. H. Yang, D. J. Kriegman, and N. Ahuja, "Detecting faces in images: A survey," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, pp. 34-58, 2002.

[16] T. S. H. G. Yang, "Human face detection in complex background," *Pattern Recognition*, vol. 27(1), pp. 53-63, 1994.

[17] J. S. a. I. Pitas, "Segmentation and Tracking of Faces in Color Images," *Automatic Face and Gesture Recognition*, pp. 236-241, 1996.

[18] M. N. T. Sakai, S. Fujibayashi, "Line Extraction and Pattern Detection in a Photograph," *Pattern Recognition,* vol. 1, pp. 233-248, 1969.

[19] 李春明, "视频图像中的运动人体检测和人脸识别 [D]," 西安电子科技大学, 2005.

[20] Q. J. Dan Witzner Hansen, "In the Eye of the Beholder: A Survey of Models for Eyes and Gaze," *IEEE Trans. on Pattern Analysis and Machine Intelligence,* vol. 32(3), 2010.

[21] P. I. Sobottka K, "A novel method for automatic face segmentation, facial feature extraction and tracking," *Signal Processing and Image Communication,* vol. 12(3), pp. 263-281, 1998.

[22] Y. J.-y. WANG Qiong, "Eye detection in facial images with unconstrained background," *Pattern Reognition Research,* pp. 55-62, 2006.

[23] R. K. KUMAR T, RAMAKRISHNAN A G, "Eye detection using color cues and projection functions," *IEEE International Conference on Image Processing,* pp. 22-25, 2002.

[24] R. L. Hsu, M. Abdel-Mottaleb, and A. K. Jain, "Face detection in color images," *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* vol. 24, pp. 696-706, 2002.

[25] H. A. Rowley, S. Baluja, and T. Kanade, "Neural network-based face detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* vol. 20, pp. 23-38, 1998.

[26] E. Osuna, R. Freund, and F. Girosit, "Training support vector machines: an application to face detection," in *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, 1997, pp. 130-136.

[27] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, 2001, pp. I-511-I-518 vol. 1.

[28] C. P. Papageorgiou, M. Oren, and T. Poggio, "A general framework for object detection," in *Computer Vision, 1998. Sixth International Conference on*, 1998, pp. 555-562.

[29] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: a statistical view of boosting (With discussion and a rejoinder by the authors)," *The annals of statistics,* vol. 28, pp. 337-407, 2000.

[30] C. M. Bishop, *Pattern recognition and machine learning* vol. 4: springer New York, 2006.

[31] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE-*, 1996, pp. 148-156.

[32] R. E. Schapire and Y. Singer, "Improved boosting algorithms using confidence-rated predictions," *Machine learning,* vol. 37, pp. 297-336, 1999.

[33] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and regression trees*: Chapman & Hall/CRC, 1984.

[34] T. Fawcett, "An introduction to ROC analysis," *Pattern recognition letters*, vol. 27, pp. 861-874, 2006.

[35] P. Wang, M. B. Green, Q. Ji, and J. Wayman, "Automatic eye detection and its validation," in *Computer Vision and Pattern Recognition-Workshops, 2005. CVPR Workshops. IEEE Computer Society Conference on*, 2005, pp. 164-164.

[36] Face Recognition Grand Challenge (FRGC): http://www.nist.gov/itl/iad/ig/frgc.cfm.

[37] N. Dalal, "Finding people in images and videos," Institut National Polytechnique de Grenoble-INPG, 2006.

[38] D. Comaniciu, "An algorithm for data-driven bandwidth selection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 25, pp. 281-288, 2003.

[39] K. Fukunaga and L. Hostetler, "The estimation of the gradient of a density function, with applications in pattern recognition," *Information Theory, IEEE Transactions on*, vol. 21, pp. 32-40, 1975.

[40] 胡波, "基于 meanshift 和 Kalman 滤波的视频目标跟踪技术," Jan 10th, 2010.

[41] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, pp. 603-619, 2002.

[42] P. Meer, "Kernel-based object tracking," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 25, 2003.

[43] O. Jesorsky, K. Kirchberg, and R. Frischholz, "Robust face detection using the hausdorff distance," in *Audio-and video-based biometric person authentication*, 2001, pp. 90-95.

[44] Z. H. Zhou and X. Geng, "Projection functions for eye detection," *Pattern recognition*, vol. 37, pp. 1049-1056, 2004.

[45] The BioID Face Database. https://www.bioid.com/support/downloads/software/bioid-face-database.html.

[46] M. Türkan, M. Pardas, and A. Cetin, "Human eye localization using edge projection," *Comp. Vis. Theory and App*, 2007.

[47] R. Valenti and T. Gevers, "Accurate eye center location and tracking using isophote curvature," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, 2008, pp. 1-8.

[48] L. Wiskott, J. M. Fellous, N. Kuiger, and C. von der Malsburg, "Face recognition by elastic bunch graph matching," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 19, pp. 775-779, 1997.

[49] F. S. Samaria, "Face recognition using hidden Markov models," University of Cambridge, 1994.

[50] C. Liu and H. Wechsler, "Face recognition using shape and texture," in *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, 1999.

[51] L. Shen and L. Bai, "A review on Gabor wavelets for face recognition," *Pattern Analysis & Applications*, vol. 9, pp. 273-292, 2006.

[52] J. Yi, "A Novel Feature Descriptor for Face Representation and Recognition," *International Conference on Biotechnology*, 2012.

[53] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, "Eigenfaces vs. fisherfaces: Recognition using class specific linear projection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 19, pp. 711-720, 1997.

[54] M. Frigo and S. G. Johnson, "The design and implementation of FFTW3," *Proceedings of the IEEE*, vol. 93, pp. 216-231, 2005.

[55] J. H. Wilkinson Prize for Numerical Software. http://en.wikipedia.org/wiki/J._H._Wilkinson_Prize_for_Numerical_Software.

[56] FFTW Home Page. http://www.fftw.org/.

[57] Android NDK. http://developer.android.com/tools/sdk/ndk/index.html.

致　谢

回首即将结束的硕士生涯，许多老师和同学都对我给予了极大的帮助。至此论文完稿之际，谨以有限的篇幅对他们表示感谢。

两年前，我有幸保送至多媒体通信与模式识别实验室攻读硕士研究生，开始了我两年的硕士学习生涯。在攻读硕士学位期间，从课程学习、论文选题、研究工作的开展到学位论文的写作等每一个环节无不凝聚了苏菲教授的心血。同时，在生活方面，苏老师也给予我非常大的关心。苏教授开阔的科学视野、严谨的治学态度以及平易近人的待人之道，都对我产生了深远的影响。在此，我谨向苏菲教授致以最衷心的谢意。

感谢赵衍运副教授和庄伯金副教授，赵老师和庄老师在课程学习、科研生活以及论文审稿阶段都给予了非常大的帮助。

感谢易军师兄，易军师兄细致、缜密的思维习惯，用最朴素的现象解释数学公式的能力深深的影响了我，在我科研生涯的自始至终，易师兄都给予了至关重要的指导，在此表示最衷心的谢意。

感谢刘丽师姐，刘丽师姐在我科研生涯初期给予了最直接的帮助，让我快速熟悉了我硕士期间的科研课题，少走了许多弯路。

感谢信通院 2010 级 5 班所有同学，尤其是多媒体通信与模式识别实验室 2010 级全体同学，和他们在一起的两年时光将是我人生一段难忘的经历。

感谢陈晓龙同学、白志渊同学，和他们在一起的时光，让我找到了另一片天地，让我得以在紧张的科研生活中释放自己。

感谢李晓晴同学，在我最艰难的时候，李晓晴同学总能给我最强有力的支持；在我最低落的时候，李晓晴同学总是用发生在身边的故事鼓励我；在我自满的时候，李晓晴同学又能提醒我静下心来。

最后，感谢我的父母亲，感谢我的姐姐、哥哥，没有他们的汗水与付出，甚至是牺牲，我不会有机会走出大山，不会有机会经历如此精彩的人生。

攻读学位期间发表或已录用的学术论文

[1] 熊金水，苏菲，张建. 一种结合 AdaBoost 与 mean shift 的人眼检测方法[OL]. [2013-01-14]. 中国科技论文在线.