单位代码 10006

学 号 13061129

分类号 TP391

密 级

****

毕业设计(论文)

针对微博平台的文本细粒度

情感分析研究与实现

|  |  |
| --- | --- |
| 院（系）名称 | 计算机学院 |
| 专业名称 | 计算机科学与技术 |
| 学生姓名 | 张海龙 |
| 指导教师 | 牛建伟 |

2017年6月

针对微博平台的文本细粒度情感分析研究与实现 张海龙 北京航空航天大学

北京航空航天大学

**本科生毕业设计（论文）任务书**

Ⅰ、毕业设计（论文）题目：

针对微博平台的文本细粒度情感分析研究与实现。

Ⅱ、毕业设计（论文）使用的原始资料（数据）及设计技术要求：

数据分两部分，第一部分是维基中文百科的数据，第二部分是从微博上爬虫下来等等数据。

Ⅲ、毕业设计（论文）工作内容：

1、熟悉和了解基本的基于内容的图像搜索技术。学习了解SIFT算法。

2、编写基于SIFT算法的图像处理程序。

3、手动采集一些常见的商标，提取商标的特征点信息，生成包含特征点信息、商标相关信息的数据文件。

4、编写Android平台的APP，在APP上实现基于图像识别技术的商标信息检索系统。

Ⅳ、主要参考资料：

[1] 李向阳,庄越挺,潘云鹤.基于内容的图像检索技术及系统[J].计算机研究与发展,2001,38(3)

[2] DG Lowe. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision, 2004, 60(2):91-110

[3] 祝晓斌,刘亚奇,蔡强,曹健. 基于内容的图像检索技术研究[J].计算机仿真, 2015, 32(5):1-4

计算机 学院（系） 计算机科学与技术 专业类 130615 班

学生 张海龙

毕业设计（论文）时间： 2016 年 10 月 20 日至 2017 年 6 月 1 日

答辩时间： 2016 年 6 月 6 日

成 绩：

指导教师： 牛建伟

兼职教师或答疑教师（并指出所负责部分）：

系（教研室） 主任（签字）：

注：任务书应该附在已完成的毕业设计（论文）的首页。

**本人声明**

我声明，本论文及其研究工作是由本人在导师指导下独立完成的，在完成论文时所利用的一切资料均已在参考文献中列出。

作者：张海龙

签字：

时间：2017年 6 月

针对微博平台的文本细粒度情感分析研究与实现

学 生：张海龙

指导老师：牛建伟

摘要

俗话说“一图胜千言”，形容的就是图像包含众多的信息，远非简单几句话就能描述清楚的，传统的图像搜索方式是基于文字进行搜索，而这种搜索方式在搜索图像时由于输入的搜索文字很难全面、准确的描述出整个图片的信息而显得不是很适用，所以基于内容的图像搜索（content based image retrieval, CBIR）技术在上个世纪90年代便随着实际的需求应运而生了。但这项技术发展至今仍然不是很成熟，虽然产生了一些实际可以使用的CBIR系统，但是其准确率一般较低，很难用于精确查找。

传统的CBIR系统一般用图像的颜色直方图进行图像匹配，而经过学习相关知识后，使用SIFT算法作为系统的关键技术编写一个小型的CBIR系统便作为了这次毕业设计。SIFT算法是一个优秀的图像匹配算法，生成的图像匹配点具有尺度不变性，十分适合做图像匹配。使用SIFT算法处理图像，得到的SIFT特征点用于图像的之间的匹配，判断不同的两幅是否含有相同的图标。

最终的成果是制作一个基于Android平台的APP实现一个商标信息检索系统，系统以SIFT算法为核心编写图像处理程序，建立存储商标图像特征点的数据文件用于匹配，从摄像头获取图像进行识别，判断是否存在商标图像，识别到商标后可得到该商标相关信息包括商标视频、商标二维码。

关键词：基于内容的图像搜索，图像识别，SIFT算法

Trademark retrieval based on image recognition technology

Author: Deng Pan

Tutor: Niu Jian-wei

**Abstract**

As the saying goes, "one picture is worth thousand words", the image contains massive information is very difficult to use simple words to accurately describe, traditional way of image search is based on the text search, and this kind of search is not very applicable, due to the input of the search words are difficult to comprehensively and accurately describe the whole picture’s information, so content-based image search (content based image retrieval, CBIR) technology firstly proposed in the 1990s has made great development. But the technology is still not very mature, although there are a number of practical CBIR system can be used, but its accuracy is generally low, it is difficult to accurately find a matched image by a given image.

Traditional CBIR systems generally use image color histogram for image matching, and after I learned the relevant knowledge of CBIR, I decided to use the SIFT algorithm as the key technology of the system to write a small CBIR system. The feature points generated by using SIFT algorithm have scale, rotation, illumination invariance, so SIFT algorithm is very suitable for image matching. Using the SIFT algorithm to deal with the image, the SIFT feature points are used for the matching between the image to judge whether the different two pieces contain similar information.

The final result is making an android APP to achieve a trademark information retrieval system, system uses sift algorithm as the core to build the image processing procedures, establish a database storage trademark image feature points for matching, from the camera image acquisition to identify, determine the existence of trademark image to identify the mark and can get the trademark related information.

**Key words**: Content Based Image Retrieval，Image Identification，SIFT

目 录

[1 绪论 3](#_Toc452824242)

[1.1 研究背景及意义 3](#_Toc452824243)

[1.2 国内外研究现状 4](#_Toc452824244)

[1.2.1 基于内容的图像搜索技术 4](#_Toc452824245)

[1.2.2 SIFT算法 7](#_Toc452824246)

[1.3 本文研究内容 8](#_Toc452824247)

[1.4 论文组织 10](#_Toc452824248)

[2 相关理论基础及技术 11](#_Toc452824249)

[2.1 SIFT算法综述 11](#_Toc452824250)

[2.1.1 使用高斯卷积核构建尺度空间 11](#_Toc452824251)

[2.1.2 关键点定位 14](#_Toc452824252)

[2.1.3 去除不好的关键点 14](#_Toc452824253)

[2.1.4 为特征点赋一个方向参数 15](#_Toc452824254)

[2.1.5 特征点描述子的生成 16](#_Toc452824255)

[2.2 CBIR技术综述 17](#_Toc452824256)

[2.3 本章小结 20](#_Toc452824257)

[3 基于图像识别的商标检索系统设计 21](#_Toc452824258)

[3.1 系统功能需求 21](#_Toc452824259)

[3.2 系统设计原则 22](#_Toc452824260)

[3.3 系统详细设计 22](#_Toc452824261)

[3.3.1 读取图像处理模块设计 22](#_Toc452824262)

[3.3.2 信息读取模块设计 25](#_Toc452824263)

[3.3.3 商标模板图像信息存储文件设计 25](#_Toc452824264)

[3.4 本章小结 26](#_Toc452824265)

[4 基于图像识别的商标检索系统实现 27](#_Toc452824266)

[4.1 系统开发环境 27](#_Toc452824267)

[4.1.1 开发环境和开发语言 27](#_Toc452824268)

[4.1.2 技术基础 27](#_Toc452824269)

[4.2 系统实现概述 28](#_Toc452824270)

[4.2.1 图像信息获取实现 30](#_Toc452824271)

[4.2.2 图像信息处理的实现 30](#_Toc452824272)

[4.2.3 图像存储及搜索实现 37](#_Toc452824273)

[4.3 本章小结 39](#_Toc452824274)

[5 系统测试 40](#_Toc452824275)

[5.1 测试环境 40](#_Toc452824276)

[5.2 系统测试 40](#_Toc452824277)

[6 总结与展望 43](#_Toc452824278)

[6.1 工作总结 43](#_Toc452824279)

[6.2 工作展望 43](#_Toc452824280)

[致谢 45](#_Toc452824281)

[参考文献 46](#_Toc452824282)

图目录

[图 1.1 CBIR系统的结构 4](#_Toc452875157)

[图 1.2 百度识图 6](#_Toc452875158)

[图 1.3 bing图像匹配界面 6](#_Toc452875159)

[图 1.4 SIFT匹配示例 8](#_Toc452875160)

[图 2.1 高斯模糊示例 12](#_Toc452875161)

[图 2.2 图像金字塔 13](#_Toc452875162)

[图 2.3 高斯差分金字塔 13](#_Toc452875163)

[图 2.4 采样点比较 14](#_Toc452875164)

[图 2.5 计算特征点方向参数 16](#_Toc452875165)

[图 2.6 生成128维的特征描述子 16](#_Toc452875166)

[图 2.7 颜色直方图示例 18](#_Toc452875167)

[图 2.8 图像纹理特征 19](#_Toc452875168)

[图 3.1 系统运作流程 21](#_Toc452875169)

[图 4.1 Android APP工程目录 28](#_Toc452875170)

[图 4.2 Scanner类的UML图 29](#_Toc452875171)

[图 4.3 ProcessImg类的UML图 29](#_Toc452875172)

[图 4.4 SIFT算法匹配流程 34](#_Toc452875173)

[图 4.5 商标匹配流程示意图 37](#_Toc452875174)

[图 5.1 系统进入首界面 40](#_Toc452875175)

[图 5.2 图像获取界面 41](#_Toc452875176)

[图 5.3 信息显示界面 41](#_Toc452875177)

[图 5.4 视频浏览 42](#_Toc452875178)

表目录

[表格 3.1 三个native函数功能介绍 23](#_Toc452824237)

[表格 3.2 SiftFeature数据结构定义 24](#_Toc452824238)

[表格 3.3 商标信息介绍 25](#_Toc452824239)

[表格 3.4 商标模板图像信息存储文件存储元素介绍 26](#_Toc452824240)

[表格 4.1 APP中包含的Activity功能描述 29](#_Toc452824241)

# 绪论

## 研究背景及意义

微博，是一种以博客形式存在的广播媒体。微博不同于传统博客，因为他的实际文件，诸如图片、视频等信息，和文字内容通常都比较少。微博允许用户发表140字的短文本、最多9张图像、或视频链接的小内容元素，这也是微博能够流行的主要原因。根据第三十九次中国互联网络发展状况的统计报告[]显示，直至2016年12月，中国网民数量突破7亿人，其中95%的用户为手机用户，移动互联网的发展，同样带动了微博用户数量的增长，人们可以随时随地的表达自己的状态及心情。同样，不同年龄段的用户人数也向两极分流，年轻人使用越来越早，越来越多的中老年人使用微博工具。微博集中了人们日常生活中是诸多功能——手机短信、社交网络、以及博客发表等。这也吸引了大量研究人员对于微博的研究，同时分析、监测在微博中所涵盖的情感信息，也有助于开发者和决策者了解群众对事件的关注度以及情感的变化，从而掌握事件的发展进展。所以出现了针对微博平台的文本信息研究，和微博情感的分析，这也是自然语言处理研究的一部分。

一些知名人士或者政府可以通过微博来了解他们的关注者，以及他们在微博上对观众的影响力，这一点非常重要。例如在每年的两会工作报告中，政府会做出一些计划和决策，这时候微博上会实时更新一些言简意赅的、让大众可以深入了解的文件解读，做出各种图表、数据来支持他们的观点，群众也可以通过下面的评论来及时反映自己的观点、看法、情感。而对于这些评论的情感分析可以作为确定群众情绪的指标，为影响力提供新的见解，更好地了解用户。此外，这可以用于评估产品营销的效果，并评估客户对名人，政客，公司等的回应。

但是因为微博的一些独有的特性，使得研究微博情感具有一定的挑战性。

首先，微博的文字内容属于短文本，字数会限制在140个字符以内，对比于传统的博客，微博的文本长度非常短，大众不需要对于文章有什么思考，可以很便捷的会表达出自己的观点、想法、经历的事件等等想要给其他人看的内容，写一条微博并发送出去。对于英文的微博来说，尽管文字数量都会限制在140个字符以内，但是英文中，每个单词之间都需要空格符号以示区分，所表达的信息量也就是中文微博的70%左右。

其次，微博具有实时性。微博可以通过许多种不同的渠道来访问，可以使用电脑，利用浏览器进行网页访问，利用PC客户端，也可以通过移动设备上的应用程序进行访问。这也极大的增强了用户发表微博的便捷性，用户可以随时随地的发布一条微博，比如在发生一些重大突发事件后不就，用户就可以在微博上发布相关信息和自己的评论，再者，由于直播发展迅速，除却视频直播，经常也会有文字直播。因此，微博的内容更新有很强的实时性。

再者，微博中发表的内容不仅只含有文字，还有许多符号。微博上不需要长篇大论，不需要很正式的语言，格式，因此符号可以表示出多重意思，例如表示提醒某人阅读的“@”符号，或者表示话题、标签的“#”符号，表示对转发其他微博的“//”符号，以及许许多多网络上用于表达心情的表情符号等等。表达方式的不同使得微博用户可以用不同的方式表达出自己的观点和感情。

## 国内外研究现状

下面，本文将分别从情感信息抽取的相关研究工作和微博情感分类的相关研究工作两方面，介绍目前的国内外研究现状。

### 情感信息抽取研究情况

情感信息抽取，作为情感分析的一个重要子任务，旨在抽取情感文本中有价值的情感信息，是一种关于细粒度文本的情感分析，根据预先设定的倾向性单元定义，从文本中抽取所表达的单元要素。该任务在近年来受到了广大学者的关注，相继出现了大量的抽取方法。例如基于情感词典的判别方法。词典的建立可以极大的支持极性分析任务，利用词典能够直接判断相对应的词语极性，同时也能够通过语义相似度、建立同义词典等方式对于未登录词进行极性判断。文献[]提出基于HowNet词汇语义相似度和语义相关场的情感词极性计算方法，其中基于语义的相似度方法有比较理想的效果，词频加权后的准确判别率可以达到80%以上。文献[]基于PMI\_IR和HowNet词汇提出了Ontology模型，将极性词图像化为类似地球仪，越倾向于两极效果越明显，计算词汇相似度与其在Ontology模型中的映射来确定情感词极性。这种方法的难点在于，中文里有许多一词多义的现象，在不同的语言环境中表达的意思会有所不同。此外，微博上有许许多多的网络新词层出不穷，现在也缺少与之对应的网络用语情感词库。

而随着基于监督学习的情感信息抽取方法研究的开展，监督学习方法表现出了其良好的任务独立性和抽取性能。因此，基于监督学习的情感信息抽取方法是情感信息抽取方法研究的一种趋势。而基于监督学习的情感信息抽取方法需要依靠大量的人工标注语料作为基础，因此，语料库建设问题也成为情感信息抽取任务中的一个重要研究方面。随着网络在中国的普及，越来越多的中文文本的出现，解决面向中文文本的情感信息抽取任务的需求日益增多。目前，面向中文文本的情感信息抽取任务的研究较少，相应的语料资源较匮乏。因此，建设一个规模较大的、标注类型丰富的中文情感信息抽取语料库是目前中文情感信息抽取任务的一个重点。此外，已有的中英文语料库所标注的信息都忽视了一些表达特点，而这些情况本身也具有一定的研究价值。如以下例句：

例１　我很喜欢，很好看。

例２　这个电影，哎，都看的睡着了。

例３　前台服务员态度不是很好。

三个例句都表达了评论作者的情感倾向，但是较以往处理的情感信息抽取问题有其特殊性。如例句１中，作者所表达的情感倾向是正面的，有着明显的情感词“喜欢”，“好看”，但评价对象在句中未有出现；例句２中的评价对象为“这个电影”，虽然句子表示了这个电影不好看的含义，但并没有情感词出现；例句３中由于否定词“不”的出现，整句话的情感倾向相对句中情感表述“好”所表达的情感倾向发生了翻转。本文分别称这三种现象为评价对象的省略现象、隐含情感及情感极性转移现象。这三种现象同情感信息抽取任务密切相关，对情感信息抽取任务的完成造成了一定的影响。例如，现有的评价对象抽取方法部分依赖于评价对象与情感词的关系来帮助抽取评价对象，若句中无情感词，一些之前表现良好的特征便不适用了。情感极性转移现象会使整句的情感倾向相对于情感词的情感倾向发生反转等。而其中评价对象的省略现象也是中文文本的一个非常特殊的表达现象。因此，在建设新的中文情感信息抽取语料库时，考虑以上的现象是有必要的。本文将面向中文文本标注一个用于情感信息抽取任务的大规模语料。该语料的特色在于，除了包括了一些传统的情感信息，例如，评价对象、评价词等，还包括了评价对象的省略现象、隐含情感及情感极性转移现象的标注。

### 文本情感分类研究情况

文献[]指出只有涵盖主观观点的文本才会蕴含情感，所以需要对语料（即原素材）进行主客观的文本分类处理。

现阶段中文微博的情感分类方法大致被分为两种：一类是依据语义词典的情感计算法，另一类是以及以机器学习为基础的情感分类法。目前的研究状况如下：

1.依据语义词典对文本进行的情感计算

在对语义词典情感的规则性进行研究时，其间的情感判断具有一定的规律性。文献[]就是通过定义态度词典，权重词典，否定词典，程度词典和感叹词典从而对每一条微博的情感指数进行计算。

这种方法产生的问题主要有两个：

（1）即使注意到了感叹词对评价词强度的增加，否定词及多种否定给情感表达带来的影响，在一句表达句中同时辩证地含有正与反两面的叙述，或是在一句话中某个情绪词多次出现的时候，只是通过计算权重的方法来测量会出现很大的误差。

（2）测试时所使用的词语通常是具有明显情感特点的情感词，使得构建词典时产生许许多多的歧义，同时也会导致对那些可以表达情感的词语的忽略。

2.以机器学习为基础的情感分类法

这一方法主要是通过运用支持向量机，朴素贝叶斯，最大熵等机器学习的模型，对训练集的特点进行研究，创建模型，并运用于对测试集合的分类型判断。

文献[]就是使用了三种机器学习算法，三种特征选取算法和三种特征项权重计算方法，证实研究了微博的情感分类。通过研究，结合考虑三方面因素，应用支持向量机（SVM）和信息增益（IG），以及TF-IDF（Term Frequency-Inverse Document Frequency)作为特征项权重，结合这三者对微博的情感分类的结果是最近人意的。

## 本文研究内容

课题的目的是根据某一部分情感词在不同领域下或不同场景下表达的情感极性或情感强度不同，来建立针对不同领域的细粒度的情感词典，然后将其作为文本分析的特征来提取出来，根据人们的言论评价来分析其中所包含的情感。

为能够使本课题可以进行顺利，首先需要从互联网中获取大量关于人们的言论信息。在获取大规模数据之后，本课题将提出一种情感词典扩充算法，对获得的文本数据进行训练和分析，抽取不同领域文本中的所有可能的情感词汇。然后根据算法得出候选词汇在不同领域里的极性和强度，来扩充现有的基础情感词典。最后提取语句中的特征词，并将特征词转换为向量模式，进行神经网络模型的训练，得出其文本的情感倾向，分析得到人们对该领域的情感态度。此外，为了使用的便捷，同时会对模型进行界面化调用，方便对语句进行操作处理。

尽管目前已经有很多情感词典扩充的方法，但最终情感分析效果的好坏与数据所属领域、情感词等特征密切相关。本课题将特别针对新浪微博数据集的各种特点提出相适应的情感词典扩充算法。

此外，数据可视化已成为数据分析科学中的一种重要手段。当文本数据的情感被标注出来之后，将其以直观的方式展现出来方便人们分析社会舆情，成为辅助人们分析数据的重要手段。本课题将采用多种可交互的可视化展现方式，展示采集到的数据，同时将情感趋势、舆情态势等信息呈现在人们面前。

## 论文组织

第一章 绪论

介绍本文的研究背景，之后，论文从情感信息的抽取和文本情感的分类两个方面，全面介绍了目前国内外有关情感分析的研究现状。最后，本章提出了论文研究的具体内容和目标。

第二章 相关理论基础及技术

介绍本次毕设涉及到的主要的理论及技术，详细介绍了本文提出的情感词典扩充算法和情感分类算法的理论基础。主要介绍了分词算法、点互信息（PMI）、LSTM（Long Short Time Memory）模型等理论基础。

第三章 基于领域情感词典扩充

提出一种情感词典扩充算法，对获得的文本数据进行训练和分析，抽取不同领域文本中的所有可能的情感词汇。然后根据算法得出候选词汇在不同领域里的极性和强度，来扩充现有的基础情感词典。

第四章 文本情感分类

详细介绍最后如何提取语句中的特征词，并将特征词转换为向量模式，用其情感词性强度来对特征进行处理，最后进行神经网络模型的训练，得出其文本的情感倾向，分析得到人们对该领域的情感态度。

第五章 系统试验

利用图形界面输入文字，输出其领域性关系、情感词词向量、文字情感值。同时可以进行批处理文件。

第六章 总结与展望

对全文进行总结，分析了本文提算法的优势与不足，以及算法对于情感分析的意义，并说明了未来可以继续深入开展的相关工作。

# 相关理论基础及技术

本章主要介绍研究所需要使用到的相关技术和理论基础，分词算法、点互信息（PMI）、长短期记忆（Long Short Time Memory）模型等。

## 分词算法

对于中文而言，词是承载语义的最小单元，由词构成语句，又由语句构成篇章。但是，中文文本是由连续的字序列构成，词与词之间是没有天然的分隔符。在自然语言处理领域，国外已经做出了很多卓有成效的研究，但是那些研究大多基于英文（存在天然的分隔符），也就是说是以正确切分出单词为前提的。于是，NLP对于中文而言要想取得较好的科研成果，就需要准确识别词与词之间的边界，也就是分词。

现阶段中文分词的算法多种多样，并且已经较为成熟。常用的分词算法主要分为两大类，即基于词典的分词算法和统计分词算法。下面将对这两种分词算法进行简要的介绍。

|  |  |  |  |
| --- | --- | --- | --- |
| 分词方法 | 基于词典的分词 | 基于理解的分词 | 基于统计的分词 |
| 歧义识别 | 差 | 强 | 强 |
| 新词识别 | 差 | 强 | 强 |
| 需要词典 | 需要 | 不需要 | 不需要 |
| 需要语料库 | 否 | 否 | 是 |
| 需要规则库 | 否 | 是 | 否 |
| 算法复杂性 | 容易 | 很难 | 一般 |
| 技术成熟度 | 成熟 | 不成熟 | 成熟 |
| 实施难度 | 容易 | 很难 | 一般 |
| 分词准确性 | 一般 | 准确 | 较准 |
| 分词速度 | 快 | 慢 | 一般 |

表格 2.1 分词方法的优劣对比

### 基于词典和规则的分词算法

这种方法按照一定策略将待分析的汉字串与一个“充分大的”机器词典中的词条进行匹配，若在词典中找到某个字符串，则匹配成功。识别出一个词，根据扫描方向的不同分为正向匹配和逆向匹配。根据不同长度优先匹配的情况，分为最大（最长）匹配和最小（最短）匹配。根据与词性标注过程是否相结合，又可以分为单纯分词方法和分词与标注相结合的一体化方法。常用的方法如下：

(1)最大正向匹配法(MM, Maximum Matching Method)

通常简称为MM法。其基本思想为：假定分词词典中的最长词有i个汉字字符，则用被处理文档的当前字串中的前i个字作为匹配字段，查找字典。若字典中存在这样的一个i字词，则匹配成功，匹配字段被作为一个词切分出来。如果词典中找不到这样的一个i字词，则匹配失败，将匹配字段中的最后一个字去掉，对剩下的字串重新进行匹配处理…… 如此进行下去，直到匹配成功，即切分出一个词或剩余字串的长度为零为止。这样就完成了一轮匹配，然后取下一个i字字串进行匹配处理，直到文档被扫描完为止。

其算法描述如下：

step1: 从左向右取待切分汉语句的m个字符作为匹配字段，m为大机器词典中最长词条个数。

step2: 查找大机器词典并进行匹配。若匹配成功，则将这个匹配字段作为一个词切分出来。若匹配不成功，则将这个匹配字段的最后一个字去掉，剩下的字符串作为新的匹配字段，进行再次匹配，重复以上过程，直到切分出所有词为止。

(2)逆向最大匹配法(Reverse Maximum Matching Method)

通常简称为RMM法。RMM法的基本原理与MM法相同 ,不同的是分词切分的方向与MM法相反，而且使用的分词辞典也不同。逆向最大匹配法从被处理文档的末端开始匹配扫描，每次取最末端的2i个字符（i字字串）作为匹配字段，若匹配失败，则去掉匹配字段最前面的一个字，继续匹配。相应地，它使用的分词词典是逆序词典，其中的每个词条都将按逆序方式存放。在实际处理时，先将文档进行倒排处理，生成逆序文档。然后，根据逆序词典，对逆序文档用正向最大匹配法处理即可。

由于汉语中偏正结构较多，若从后向前匹配，可以适当提高精确度。所以，逆向最大匹配法比正向最大匹配法的误差要小。统计结果表明 ,单纯使用正向最大匹配的错误率为 1/169，单纯使用逆向最大匹配的错误率为1/245。例如切分字段“硕士研究生产”，正向最大匹配法的结果会是“硕士研究生/产”，而逆向最大匹配法利用逆向扫描，可得到正确的分词结果“硕士 / 研究 / 生产”。

当然，最大匹配算法是一种基于分词词典的机械分词法，不能根据文档上下文的语义特征来切分词语，对词典的依赖性较大，所以在实际使用时，难免会造成一些分词错误，为了提高系统分词的准确度，可以采用正向最大匹配法和逆向最大匹配法相结合的分词方案。

(3)双向匹配法

将正向最大匹配法与逆向最大匹配法组合。先根据标点对文档进行粗切分，把文档分解成若干个句子，然后再对这些句子用正向最大匹配法和逆向最大匹配法进行扫描切分。如果两种分词方法得到的匹配结果相同，则认为分词正确，否则，按最小集处理。

这种基于词典和规则的分词算法仅仅是跟一个电子词典进行比较，故不能进行歧义识别；无法正确识别未登录词，因为这种算法仅仅与词典中存在的词语进行比较；基本思路就是与电子词典进行比较，故电子词典是必须的。并且词典越大，分词的正确率越高，因为词典越大，未登录词越少，从而可以大大减少未登录词识别的错误；仅仅进行字符串的比较操作，故算法简单；是最早出现也是最成熟的算法；是一种"浅理解"的分词方法，不涉及真正的含义理解，故可能会出现错误，难以达到100%的准确性；算法简单，操作容易，故分词速度快，所以这种算法常常作为另外两种算法的预处理，进行字符串的粗分。

### 基于理解的分词算法

这种分词方法是通过让计算机模拟人对句子的理解，达到识别词的效果。其基本思想就是在分词的同时进行句法、语义分析，利用句法信息和语义信息来处理歧义现象。它通常包括三个部分：分词子系统、句法语义子系统、总控部分。在总控部分的协调下，分词子系统可以获得有关词、句子等的句法和语义信息来对分词歧义进行判断，即它模拟了人对句子的理解过程。这种分词方法需要使用大量的语言知识和信息。由于汉语语言知识的笼统、复杂性，难以将各种语言信息组织成机器可直接读取的形式，因此目前基于理解的分词系统还处在试验阶段。

理解字符串的含义，从而有很强的新词识别能力；规则是计算机进行理解的基础，故准确、完备的规则库是这种分词算法的前提；需要充分处理各种规则，故算法非常复杂；事实上到目前为止，还没有成熟的这类算法；不过从理论上说，基于理解的分词算法有最高的分词准确性，理论上有100%的准确性。

### 基于统计的分词算法

基于统计的分词模型如下所示：

令C=C1C2...Cm ，C 是待切分的汉字串。W=W1W2...Wn ，W 是切分的结果。

设P(W|C)是汉字串C切分为W的某种估计概率。

Wa，Wb，⋯．Wk是C的所有可能的切分方案。那么，基于统计的切分模型就是这样的一种分词模型，它能够找到目的词串W ，使得W 满足：

即估计概率为最大之词串。我们称函数P(W|C)为评价函数。一般的基于统计的分词模型的评价函数，都是根据贝叶斯公式．同时结合系统本身的资源限制，经过一定的简化，近似得来的。

## 本章小结

## 本章小结

## 本章小结

# 基于图像识别的商标检索系统设计

本章在前两章的基础上，分析基于图像识别的商标检索系统的功能需求，从功能需求入手，得到系统的设计原则；在功能需求和设计原则的指导下提出对APP详细确的设计。本章着重分析系统的设计和运作流程，具体的实现方法在下一章中具体介绍。

## 系统功能需求

本系统是一个商标检索系统，其功能是通过手机获得图像在手机中进行图像检测，如果检测到商标便输出商标的相关信息，所以系统的工作流程可以分为三个步骤，即获取图像、图像检测、获得商标信息。为实现这一工作流程需要系统能够实现以下的三个功能。

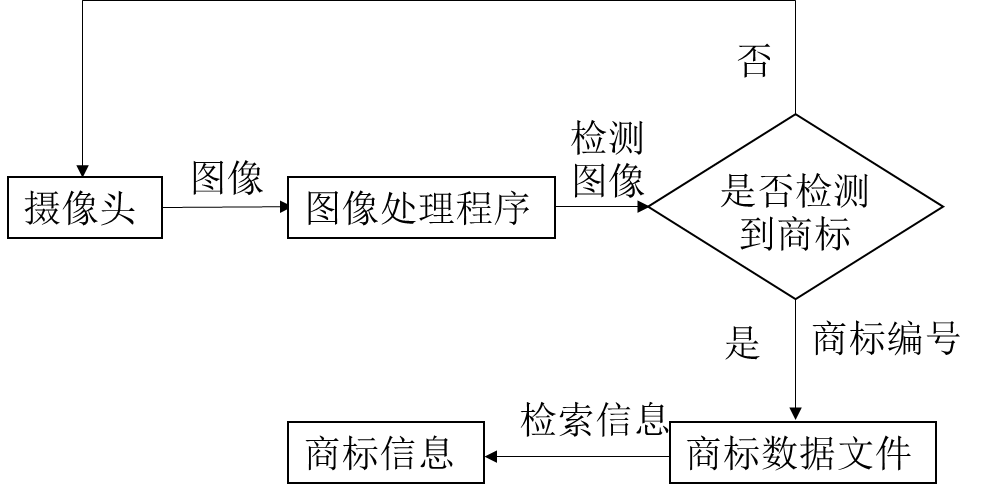


图 3.1 系统运作流程

* + - 1. 移动端获取图像

虽然在电脑上也可以设计商标检测系统，且速度会更快，但是考虑到实际的应用需求中，如果开发在电脑中的系统，则获取图像只能靠电脑摄像头或手动传入图像，使用电脑摄像头会受限于电脑的不便捷、手动传入会受限于时间上的无法立即处理。所以基于实际使用场景需要在手机上开发APP，在移动端实现该系统，使用手机摄像头获取图像。

* + - 1. 实现商标检测

得到图像后要对图像进行分析处理，判断是否含有商标图像，所以说商标检测是该系统的关键，既要保证识别的准确度较高，又要保证识别的速度较快，受限于手机的硬件能力较弱，必须对传统电脑使用的图像识别算法进行优化，而且由于图像识别算法普遍采用速度较快的C++语言编写，在使用基于Java语言的Android APP时还要考虑两种语言的粘合使用。

1. 得到商标信息

图像检测之后能够识别出图像中的商标，这时要获取商标的信息，必须有一个存放商标信息的数据库或者数据文件，其中存储着商标的相关信息，能够快速查询获得。

## 系统设计原则

明确了系统的功能需求之后，我们可以得到系统的三点要求：1）本系统要再移动端运行；2）系统要有较高的检测速度和识别精度；3）系统要能够迅速的获得商标信息。所以为了达到这些要求，系统的设计原则如下：

* + - 1. 模块化设计

这里的模块化设计主要指APP内模块化设计，将系统按照需求所述分为不同的功能模块，每一块功能之间实现高内聚、低耦合，对每个进行独立设计、实现，还要对SIFT算法使用到的算法模块进行封装，使其可以在Java语言环境下使用。

* + - 1. 体验优良原则

本系统是一个检索系统，设计到检索必然是用户输入通过检索得到输出，为了保证用户的体验优良，必须提升算法的速度，使用户在一个可以忍受的时间范围内等待；还要提升识别算法的精确度，使用户能够得到一个尽可能正确的识别结果

## 系统详细设计

本节通过系统需要实现的各个功能和APP的运行流程，给出APP内各模块的详细设计说明。

### 读取图像处理模块设计

读取图像处理模块需要实现的功能是从摄像头获取图像，使用SIFT算法寻找图像内是否存在待检测的商标。

这里关键的算法是SIFT算法，对于SIFT算法采用的OpenCV库中直接封装好的算法，OpenCV是一个很不错的图像处理库，其中的图像处理算法都是是经过优化的算法，运行速度较快，识别率较高，所以图像处理相关的代码使用OpenCV库中集成的代码实现。OpenCV早期普遍使用的是其C++版本，但是从2.0版本之后，OpenCV开始拥有Java版本，但是Java版的OpenCV库并不如C++版本完整，只有部分较简单的算法，没有一些复杂算法的代码，而SIFT算法作为一个经典的算法也没有包含在其中，一方面是因为Java在运行效率上不如C++高效，另一方面是因为SIFT算法本身在C++版本的库中所在的文件夹便是nonfree文件夹，证明这个算法不是免费的，为了防止该算法被免费使用，自然也就没有被包含在Java库中。

但是获得图像之类的简单图像处理功能还是可以用OpenCV中已经封装好的各种简便函数。在这里获取图像使用的是CameraBridgeViewBase.CvCameraViewListener2接口，该接口自动封装了Android的相机功能，使用起来十分方便。在APP实现该接口，调用图像获取函数从摄像头获得Mat类的图像，将图像传入图像处理活动（Scanner Activity）进行检测与处理。

图像处理使用经典的SIFT算法进行处理，但是由于SIFT算法不包含在Java版的OpenCV库中，所以必须要使用C++版的代码。由于现在需要跨代码使用，所以便需要用到Java语言的JNI （Java Native Interface）功能，JNI是JAVA标准平台中的一个重要功能，它弥补了JAVA的与平台无关这一重大优点的不足，在JAVA实现跨平台的同时，也能与其它语言（如C、C++）的动态库进行交互，给其它语言发挥优势的机会。

**表格 3.1 三个native函数功能介绍**

|  |  |
| --- | --- |
| 函数名称 | 功能介绍 |
| createSiftFeature | 构造SIFT特征点 |
| freeSiftFeature | 释放SIFT特征点 |
| findIcon | 在图像中寻找商标 |

使用JNI功能调用C++代码的步骤如下：使用Java定义相关的SIFT函数，分别是createSiftFeature、freeSiftFeature、findIcon这三个native函数，函数介绍见表格3.1，这三个函数分别实现构造SIFT特征点、清除SIFT特征点、和寻找商标的功能。再根据JNI的使用规则构造相应的.h文件，在.h文件对应的.cpp文件中用C++实现这个三个函数，这样使得Java可以在使用这三个方法时其内部调用的其实是这三个函数对应的C++代码，这也就实现了通过Java调用C++代码的功能，使得Android平台可以使用C++版本的SIFT算法。

为了方便后续的图像匹配和框出商标操作，在createSiftFeature函数中定义一个名为SiftFeature的数据结构，该数据结构包括三个数据元素：图像的指针，图像的关键点（KeyPoint）、图像的特征点，SiftFeature的数据介绍见表格3.2。createSiftFeature函数中使用OpenCV库中的方法实现图像检测，存储在SiftFeature数据结构中。

**表格 3.2 SiftFeature数据结构定义**

|  |  |
| --- | --- |
| 数据元素 | 描述 |
| Mat \* img | Mat指针，指向被处理的图像 |
| vector<KeyPoint> keypoints | vector容器储存的是KeyPoint类型的数据，该数据代表被处理图像的关键点 |
| Mat feature | 存储的是图像的特征点，特征点在这里是用Mat类型存储的 |

在findIcon中探测商标的原理是将照相机传入的图像与系统数据库中存储的图像信息进行匹配，匹配成功则说明探查成功。所以说需要在系统内部建立一个存储图像信息的数据库。但是如果数据库中存储的是图像，则每次匹配都要重新处理图像，对图像提取特征点，计算其关键点，这样会加重运算负担，也会带来没有必要的运算量，造成程序运行过慢。所以为了提升运算速度，好的做法应该是提前将待匹配的商标模板图像进行预处理，提取出图像匹配时需要使用到的数据，存储在数据文件中，以后匹配可以从中直接读取进行使用，为系统运行节约时间。所以在这里我将待匹配的模板商标图像进行了特征预提取，将其存储在XML文件中，数据文件的具体介绍见3.3.3节。

在图像与模板商标图像进行匹配的过程中，findIcon函数中使用createSiftFeature函数创建照相机传入图像的特征点，从XML数据文件中逐一提取各商标的图像信息，使用库中的SIFT算法进行匹配操作。但是由于OpenCV中的匹配代码中并没有对是否匹配到进行判断，且由于SIFT算法匹配采用的是特征点匹配，两张完全不同的图像也有可能有较多的特征能够相匹配，这就造成了SIFT算法在使用过程中极易发生误匹配，所以在实际使用中需要在其中修改匹配代码，对特征点进行择优筛选，对匹配是否正确加入判断标准。

对特征点进行择优筛选采用的条件是（distance<3\*minDist），即关键点间的距离小于3倍最小关键点间距离，在匹配完成后在各个匹配点中寻找最短的匹配点距离，以三倍的最短距离为标准对匹配点对进行筛选，小于三倍最小距离的视作好的匹配点对，其余的匹配点对全部移除，这样做便能很大程度的保证留下的匹配点对是正确匹配得到的匹配点对。匹配正确后，将两幅图像的信息进行计算，生成仿射矩阵，通过模板图像的四个角点，将探测出的商标在原图中框出，并将探测得到的商标的内部编号传入信息读取模块。而这里为了进一步提升匹配的准确率，需要对生成的框进行计算，判断四边形的方框其四个角的角度，规定角度在60至120度至今才是正确的匹配，这两步检测加入后基本可以保证匹配的正确性。

### 信息读取模块设计

信息读取模块的工作是使用读取图像处理模块得到的图像编号进行商标信息的检索和处理。

信息读取模块接收由读取图像处理模块传入的商标编号，根据商标编号查询对应商标的信息。由于本身模板商标的数量并不是很多，所以在这里为了提升速度也为了压缩程序的大小便不再使用数据库，而是直接将相关信息写在代码段里，直接用case语句判断读取相关信息。通过case判断，找到对应的商标信息，能够读取到的信息包括商标的名称（icon\_name）、商标介绍视频的网址信息（icon\_website）、商标二维码名称（icon\_QRcode），详细介绍见表格 3.2。每次传入商标编号后，系统根据编号查到对应的商标信息，然后跳转到信息显示页面，显示匹配商标名称、商标网址链接、商标对应二维码，其中二维码图像是存储在APP中，得到商标二维码名称后，通过名称从/res/layout文件夹中读取并进行显示。

**表格 3.3 商标信息介绍**

|  |  |
| --- | --- |
| 商标信息元素 | 元素介绍 |
| icon\_id | 存储的是商标的编号，用于提取信息时的匹配作用 |
| icon\_name | 存储的是商标的名称 |
| icon\_website | 存储的是商标的网址信息 |
| icon\_QRcode | 存储的是商标的二维码图像名称 |

### 商标模板图像信息存储文件设计

商标模板图像信息存储文件需要包含商标图像的全部信息，用作搜索和查询使用，所以其中应该包含的条目有：商标的行像素（icon\_rows）、商标的列像素（icon\_cols）、图像的关键点（icon\_keypoint）、图像的描述子（icon\_descriptor）、商标的编号（icon\_id），商标模板图像信息存储文件的具体介绍如表格3.2所示。OpenCV中的数据推荐存储在XML文件中，所以这里的数据信息也都存储在了XML文件中，不同的商标信息以商标编号进行区分，每个商标编号下存储着这些子信息，得到商标编号后便可以读取商标的全部信息。

**表格 3.4 商标模板图像信息存储文件存储元素介绍**

|  |  |
| --- | --- |
| 存储元素名称 | 元素介绍 |
| icon\_rows | 存储商标图像的行像素大小 |
| icon\_cols | 存储商标图像的列像素大小 |
| icon\_keypoint | 存储商标图像的关键点 |
| icon\_descriptor | 存储商标图像的特征点 |
| icon\_id | 存储商标图像的编号 |

## 本章小结

本章重点介绍了系统的需求和需要实现的功能，根据这些需求对系统提出设计原则，在原则的指导下对系统各个模块进行详细设计，阐述了其内部运行机制和数据传输流，介绍了一些数据结构及其存储的信息。

# 基于图像识别的商标检索系统实现

本章着重讲述系统的具体实现，描述各个模块之间的接口、内部函数的功能及其接口，包括关键的数据结构及算法，以及模块读取文件的具体实现。

## 系统开发环境

### 开发环境和开发语言

1．运行环境

本系统为Android的APP应用，运行在Android平台的手机上

* 操作系统：Android 4.2.2

2．开发环境

系统开发环境是Windows个人计算机。在Android APP开发中使用Java语言进行编写，在编写图像处理相关算法时使用C++语言进行编写，图像处理使用OpenCV库中函数进行操作，模板商标图像相关信息存储在XML文件中。

* 操作系统：Windows 10
* APP开发IDE：Android Studio，版本2.1.1
* APP开发语言：Java，JDK版本1.8.0\_73
* 图像处理开发语言：C++
* Android SDK版本：level：23，revision：1
* Android NDK版本：android-ndk-r10c
* OpenCV库版本：2.4.10

### 技术基础

为了更好的完成系统的设计和实现，必须了解其所用的相关技术方法。

1.OpenCV技术：OpenCV的全称是：Open Source Computer Vision Library，也就是开源计算机视觉库，OpenCV是一个跨平台的计算机视觉库，可在在Linux、Windows和Mac OS上运行，它由一系列C/C++函数以及C++类构成，提供C++、Java、Python等多种语言的接口，库中包括图像处理领域很多常用而经典的算方法实现。是计算机视觉领域十分重要而经典的开源库。

2.Android APP开发技术：Android操作系统作为全世界占有率最高的手机操作系统，应用于其上的APP已达上百万个，所以Android APP开发越来越热门，甚至已经成为了计算机专业人才必须掌握的技能。Android APP开发使用Java语言，开发用的IDE主要有eclipse和Google开发的Android Studio，目前在界面开发上已实现可视化开发，还可以通过AVD（Android Virtual Device）在电脑上调试，只需掌握Java语言和基本的Android知识便可进行APP开发。

3.Android NDK技术及JNI技术：由于Android是使用Java语言开发的，但是现有很多算法及库类都是使用C或C++语言写就的，为了使用现有的算法，必须要使用JNI技术来调用C/C++语言的实现的算法，但是安卓早期没有对JNI的任何帮助文档，即使APP开发者使用JNI完成了自己的C动态链接库（即so文件）开发，仍然存在不知如何打包文件这一技术难题。由于没有官方支持，程序员单方面的使用造成了程序的不稳定和不安全，为了改变这一困境，Google发布了NDK，使得对于C/C++的使用终于受到了正式的支持。使用Android提供的NDK能够将so文件和java文件自动打包生产apk，极大的方便程序员进行编程，也提高了使用原有库类的效率和便捷程度。

## 系统实现概述

本系统实现了商标信息检索功能，包括从手机摄像头获取图像、处理图像、探测是否存在商标、获得商标的相关信息的一系列的运行流程。



图 4.1 Android APP工程目录

图4.1展现了APP的工程文件结构，可以看出此APP中用到的activity有3个：MainActivity、ScannerActivity和ShowActivity。MainActivity是APP的主活动在这里启动APP，显示开始界面，开始扫描图像后跳转到ScannerActivity；ScannerActivity用来显示图像获取时的摄像头画面，并在此活动中探测商标，找到后跳转到ShowActivity；ShowActivity用来显示获得商标信息的界面，运行搜索图像信息的活动。三个活动的具体说明如下表格4.1：

**表格 4.1 APP中包含的Activity功能描述**

|  |  |
| --- | --- |
| 活动名称 | 功能描述 |
| MainActivity | 主活动。在这里启动APP，显示开始界面，在开始界面按下按钮跳转到ScannerActivity |
| ScannerActivity | 负责扫描的活动。在这里获取摄像头传来的图像，并探测其中是否有商标存在，如果找到则跳转到ShowActivity |
| ShowActivity | 负责显示商标信息。显示找到后的商标的相关信息，如二维码、网址等 |

APP中还有两个程序运行中使用到的类， Scanner类中定义了三个Java native函数，即图像处理所使用的、通过JNI调用C++代码的函数，其静态结构如图4.2所示；ProcessImage类是一个线程类，被ScannerActivity调用处理图像，ProcessImage调用Sacnner类的实例使用图像处理函数来处理图像，进行商标探测，其静态结构如图4.3。

|  |
| --- |
| Scanner |
| -context: Context  -am: AssetManager |
| + Scanner(context: Context)  #finalize()  - createSiftFeature(imageAddr:long): long  - freeSiftFeature(featureAddr:long): long  -findIcon(frameMatAddr: long, featureAddr: long, am: AssetManager): int  - scannerIcon(frameMatAddr:long): int |

图 4.2 Scanner类的UML图

|  |
| --- |
| ProcessImg |
| -frame: Mat  -mScanner: Scanner  -context: Context |
| +ProcessImg(context: Contex)  +setImg(f: Mat)  +run() |

图 4.3 ProcessImg类的UML图

接下来将按照系统运行流程详细描述全部的系统实现。

### 图像信息获取实现

APP运行的第一步是获取图像信息，这里采用的是使用手机后置摄像头获取图像信息，关键代码如下：

|  |
| --- |
| public class ScannerActivity extends AppCompatActivity implements CameraBridgeViewBase.CvCameraViewListener2 {  public Mat onCameraFrame(CameraBridgeViewBase.CvCameraViewFrame inputFrame){  Mat frame = inputFrame.rgba();//frame是从摄像头获取的图像  Mat mt=new Mat();  frame.copyTo(mt);  pimg.setImg(mt);  }  } |

这里获取图像用的是OpenCV库自带的函数，ScannerActivity类定义onCameraFrame方法，该方法用来获取摄像头的图像。方法的参数中传入一个OpenCV库中定义的CameraBridgeViewBase.CvCameraViewFrame接口的实例inputFrame，调用inputFrame.rgba()函数用来获得摄像头得到的RGBA图像，定义一个用来储存图像的Mat类变量frame储存RGBA图像，将图像拷贝给一个新的Mat对象mt，这样做是因为将图像传给线程类进行处理后，摄像头还会不断获取新的图像，而每获得一个图像，就新建Mat对象开辟新的内存空间存储摄像头图像，这样就不会让后来的图像覆盖掉原有图像，保证处理的正确。创建一个ProcessImg线程类的实例pimg，调用其setImg方法传入mt图像开始进行图像处理工作。

### 图像信息处理的实现

将图像传入ProcessImg类的实例进行图像处理，该类是一个线程类，设置为线程主要是为了提升处理速度，获取图像和处理图像能够同时进行，这也是为了在手机这样一个运算资源有限的地方尽可能的提高运算效率，该类的关键代码如下：

|  |
| --- |
| public void setImg(Mat f) {frame=f;}  public void run() {  while (true) {  if (frame != null)  //mScanner是scanner类的实例  Int img\_no = mScanner.scannerIcon(frame.getNativeObjAddr());  }  } |

setImg函数用来接收传入的图像并赋给frame变量，线程中一直运行在等待图像的传入，当图像传入、frame非空后，调用scanner类的实例mScanner。frame是一个Mat类的实例，调用其getNativeObjAddr()方法，获得图像所在地址，也就是图像的指针，传给mScanner的scannerIcon()函数去处理图像。

接下来便是进入scanner类继续处理图像。scanner类定义了如下三个native函数及scannerIcon函数，三个native函数代表这些函数是JNI函数，在这里进行声明实际调用的C++的代码，而scannerIcon函数其本质调用的也是native函数findIcon，所以说scannerIcon这个图像扫描函数只是在Java中的一个类似接口的函数，真正实现图像处理的还是即将调用的C++代码。

|  |
| --- |
| private native long createSiftFeature(long imageAddr);  private native void freeSiftFeature(long featureAddr);  private native int findIcon(long frameMatAddr , AssetManager am);  public int scannerIcon(long frameMatAddr){  return findIcon(frameMatAddr, iconSiftFeatureAddress, am);  } |

createSiftFeature函数是用来为传入的图像创建SiftFeature的函数；freeSiftFeature函数是用来为传入的SiftFeature清空该数据结构的函数；findIcon是用来寻找图标的函数；scannerIcon是一个供Java内部使用的函数，但是其本质仍是JNI函数findIcon。这四个函数的各个参数含义如下定义：

* imageAddr：代表传入的是long类型图像的地址，最终传入C++代码中，在C++中作为图像指针使用。
* featureAddr：代表传入的是SiftFeature结构的地址，最终传入C++代码中，在C++中作为SiftFeature数据结构指针使用。
* frameMatAddr：代表传入的是long型的摄像头获取的图像的地址，最终传入C++代码中，在C++中作为SiftFeature数据结构指针使用。
* am：代表传入的是一个AssetManager类型的对象，该对象是用来管理APP目录下Asset文件夹的一个类，通过该对象在C++代码中可以访问Asset文件夹中的XML数据文件。

这里使用AssetManager类型对象的原因是由于Asset目录中的文件和Res中的文件类似，打包生成apk时不会被转化为二进制文件而是继续以原文件形式存在，图像处理程序需要在C++代码中访问APP内部的数据文件，但C++代码无法访问Res文件中的内容，所以需要使用AssetManager传给C++代码，使其可以访问Asset中的数据文件。

三个native函数实现的关键代码如下，由于其实现方法类似，所以这里以其中的createSiftFeature函数为例进行说明：

|  |
| --- |
| // com\_zzu\_me\_jniUtils\_Scanner.h，.h文件中声明该函数  JNIEXPORT jlong JNICALL Java\_com\_zzu\_me\_jniUtils\_Scanner\_createSiftFeature (JNIEnv \*, jobject, jlong);  // com\_zzu\_me\_jniUtils\_Scanner.cpp，.cpp文件中定义该函数，这里其定义中引用了另  //一个函数所以其真正运行的代码在其引用的IconScanner.h中  JNIEXPORT jlong JNICALL Java\_com\_zzu\_me\_jniUtils\_Scanner\_createSiftFeature (JNIEnv \* env, jobject obj, jlong ImgMatAddr){  return (jlong)createSiftFeature((Mat \*) ImgMatAddr );}  //IconScanner.h  SiftFeature \* createSiftFeature(Mat \* image); |

首先根据JNI的使用规则，需要对写好的Scanner.class文件即包含native函数的文件在cmd中使用“javah com\_zzu\_me\_jniUtils\_Scanner”命令进行编译，自动生成com\_zzu\_me\_jniUtils\_Scanner.h头文件，上述代码所示的便是生成的.h文件中createSiftFeature函数的声明，接下来写com\_zzu\_me\_jniUtils\_Scanner.cpp文件，补全函数的实现代码，这里函数的实现返回的是createSiftFeature函数，这个函数在IconScanner.h中声明，在IconScanner.cpp中实现，所以Scanner类中这3个native函数真正调用的是IconScanner.cpp中的3个函数。

在IconScanner.h中还定义了一个数据结构SiftFeature，SiftFeature中包含三个内容，已在上一章进行过介绍，其中的\* img是一个指向图片的指针；vector<KeyPoint> keypoints是KeyPoint类型的一个容器，keypoint类型是在OpenCV中定义的关键点数据结构，包含两个变量即点的X、Y坐标值，但keypoints并不是真正用来匹配的特征点，而是不包含方向参数的匹配点；而feature中储存的才是真正用来匹配的特征点，是keypoints经过进一步计算得来的，计算得到的特征点储存在Mat数据结构中。所以说SiftFeature数据结构中存储的信息是接下来在IconScanner.cpp中实现数据处理中会使用到的重要信息，SiftFeature是重要的数据结构。

接下来传入的图像进入scanner类的实例，此实例调用了scannerIcon方法用来处理图像，其中的是摄像头获取图像的地址。最终调用的函数是IconScanner.cpp中的findIcon函数，将图像地址传入，作为图像指针使用，开始进行图像处理和匹配。使用SIFT算法进行图像处理的关键部分代码如下：

|  |
| --- |
| //createSiftFeature  SiftFeatureDetector detector;  detector.detect(\*(p->img), p->keypoints);  SiftDescriptorExtractor extractor;  extractor.compute(\*(p->img), p->keypoints, p->feature);  //findIcon  SiftFeature \* sceneFeature = createSiftFeature(scene);  vector< DMatch > allMatches, goodMatches;  FlannBasedMatcher matcher;  matcher.match(featureHandle->feature, sceneFeature->feature, allMatches);  for (int i = 0; i < featureHandle->feature.rows; i++){  if (allMatches[i].distance < 3 \* minDist)  goodMatches.push\_back(allMatches[i]);  } |

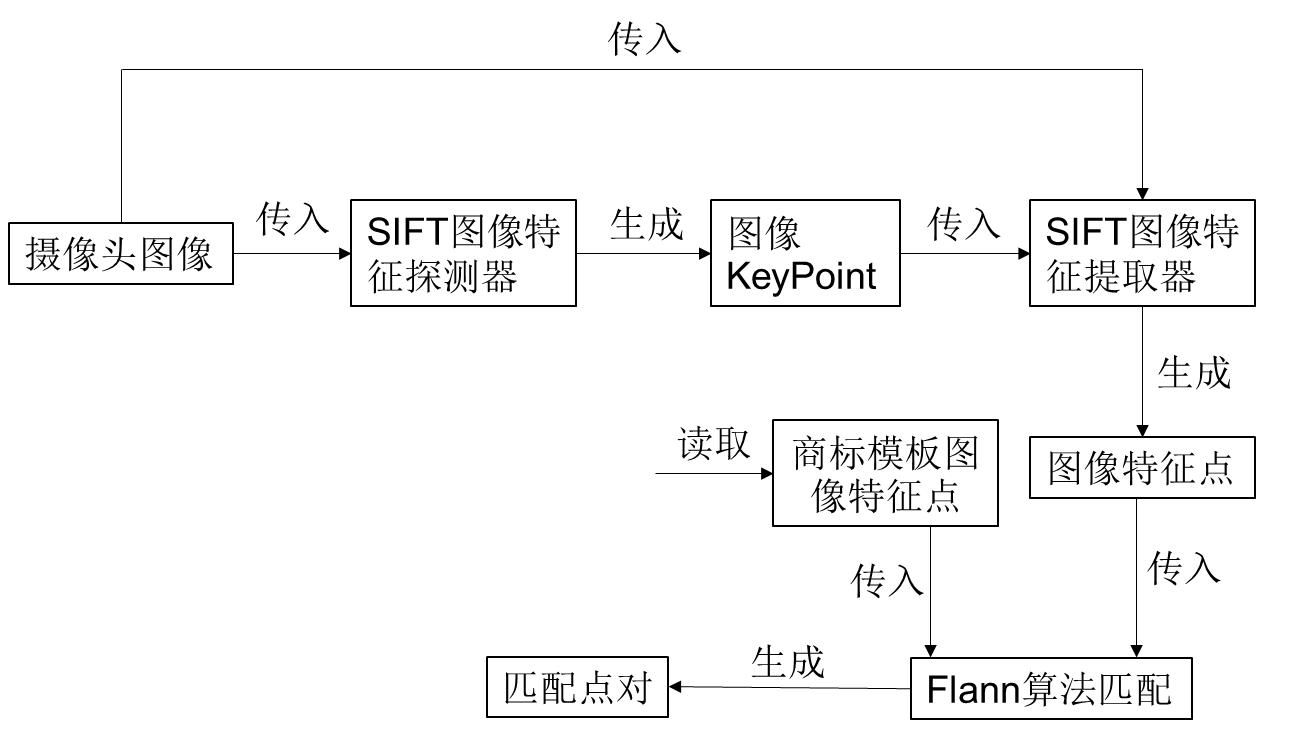


图 4.4 SIFT算法匹配流程

SIFT算法的处理流程如图4.4所示。具体的步骤如下可以归结为以下几步：

1. 摄像头获取的图像传入SIFT图像特征探测器，生成图像的关键点（KeyPoint）；
2. 图像的关键点及图像指针传入SIFT图像特征提取器，提取生成SIFT图像特征点，该特征点将用于图像匹配；
3. 逐一读取商标模板图像的特征点，与摄像头获取的图像特征点进行匹配，使用的是Flann匹配算法，即近似模糊匹配，这种匹配比常用的另一种BF（BruteForce）算法精确度稍差，但是效率更高，更加适合手机使用；
4. 匹配成功生成匹配点对，通过两个判断条件判断是否匹配正确，如果匹配正确开始进行后续的工作。

以上步骤在代码中体现如下：

1. 实例化SiftFeatureDetector，探测图像的关键点。

在createSiftFeature函数中定义一个新的SiftFeature数据结构p用来储存图像的特征信息，实例化一个OpenCV中实现的SiftFeatureDetector对象，也就是SIFT图像特征探测器，调用其detect方法，将图像指针\*(p->img)传入，探测得到图像的关键店keypoints储存到p->keypoints中；

1. 实例化SiftDescriptorExtractor，提取SIFT特征。

实例化一个OpenCV自带的SiftDescriptorExtractor对象，即SIFT特征点提取器，调用其compute方法，将图像指针\*(p->img)和图像关键点p->keypoints传入，将计算得到的图像特征点储存在p->feature中，得到的便是可用于匹配的图像的匹配点。

1. 创建FlannBasedMatcher类型的匹配器匹配图像与模板图像的特征点

接下来创建两个DMatch类型的容器，DMatch类型是OpenCV中定义的中类型，类型中包含两个元素，是两个相匹配的特征点。创建一个FlannBasedMatcher类型的匹配器，将两个图像的特征点传入进行匹配，匹配得到的特征点对存储在DMatch类型的allMatches容器中。这里并没有使用常用的BFMatcher（Brute Force Matcher，暴力匹配器），因为暴力匹配总是尝试所有的可能性来获得一个最好的匹配结果，但是这就造成了BFMatcher在使用中十分的费时，并不符合设计原则中的体验优良原则，所以这里使用更适合一些的FlannBasedMatcher（Fast Library for Approximate Nearest Neighbors，最近邻近似匹配），从字面上就可以看出这是一个近似匹配算法，算法速度更快但是找到的是最近邻近似匹配，精度相比于BFMatcher有所下降，所以当我们需要找到一个相对好的匹配但是不需要最佳匹配的时候往往使用FlannBasedMatcher，而考虑到手机性能的受限FlannBasedMatcher是一个更好的选择。

1. 对匹配得到的匹配点对进行处理得到好的匹配点对。

allMatch中的匹配点对不一定都是正确的匹配，所有匹配点对接下来还要进行一个判断，通过扫描遍历所有allMatches点对的距离大小得到其中的最小距离，以3\*minDist为标准，点对间的距离小于3\*minDist的匹配点对留下，大于的舍弃，认为留下的都是好的匹配点，留下的点对储存在goodMatches中。如果goodMatches中留存的点对大于等于四个，则这里暂时认为图像与这个商标模板想匹配，即图像中识别出了该商标，进入接下来的判断。如果不满足条件则说明图像与该商标模板不匹配，继续匹配下一个商标模板，直到匹配成功或者匹配到数据文件中最后一个模板图像的特征点为止，如果都不匹配则读取下一张摄像头图片继续进行匹配过程，直到最终匹配成功为止。

通过上面步骤得到goodMatches，两张图像被认为匹配，接下来要进行框出商标的操作。

|  |
| --- |
| vector<Point2f> objectPoint,scenePoint;  for (size\_t i = 0; i < goodMatches.size(); i++){  objectPoint.push\_back(featureHandle->keypoints[goodMatches[i].queryIdx].pt);  scenePoint.push\_back(sceneFeature->keypoints[goodMatches[i].trainIdx].pt); }  Mat H = findHomography(objectPoint, scenePoint, CV\_RANSAC);  vector<Point2f> objCorners(4), sceneCorners(4);  objCorners[0] = cvPoint(0, 0);  objCorners[1] = cvPoint(cols, 0);  objCorners[2] = cvPoint(cols, rows);  objCorners[3] = cvPoint(0, rows);  perspectiveTransform(objCorners, sceneCorners, H); |

定义两个Point2f类型的容器，Point2f是OpenCV中定义的一个数据类型，存储的是两个double类型的数字，用来表示图像中点的x、y坐标。从goodMatches中获取匹配点对中两点的索引，通过索引得到各自的keypoint，再从keypoint中获取点的x、y坐标存储在定义的Point2f类型的容器内，这时也就得到了匹配点对中各点的坐标值，分别存储于objectPoint、scenePoint容器中。使用findHomography函数，传入两个点坐标容器和一个计算参数得到两张图像对应的仿射矩阵H，仿射矩阵的作用是可以将一张图像通过放射矩阵变换到另一个图像上，使用perspectiveTransform()函数传入三个参数，分别是模板商标图像的四个角点构成的容器，另一个空的容量为4的容器，以及仿射矩阵H，使用这个函数可以将模板图像四个角点转化到摄像头图像中去，再讲摄像头中的四个点顺时针连线，构成一个四边形区域，区域框出的部分就被认为是商标在摄像头图像中的位置。

而认为框出部分就是商标图像所在是因为，这里巧妙的使用了SIFT算法的图像匹配首先找到两张图像中的匹配点，得到匹配点对中点的坐标信息，通过这些计算仿射矩阵，仿射矩阵可以理解成一张图片向另一张图片变换的工具，而商标模板图像只含有商标这一个内容，将其向摄像头图像变换得到的角点变换点的连线所框出的区域就一定是商标的区域。但是这里也有一个问题，SIFT匹配点在匹配时不可能是完全正确的，完全不同的两张图像也有可能产生想匹配的匹配点对，进而生成变换矩阵，在上一步的goodMatches判断中已经剔除一部分不正确的匹配，在这里为了保证识别的正确率需要继续添加判断条件如下：由于模板图像都是矩形图像，四个角都为直角，经过仿射变换的角点围成区域的四个角应该也是近似直角的，所以对连线形成的区域判断四个角大小，允许角度在(9030)°之间认为是正确匹配，如果角度过大或过小，则都认为匹配不正确，需要重回上一步继续匹配。所以商标匹配完整流程示意图如图4.5所示。

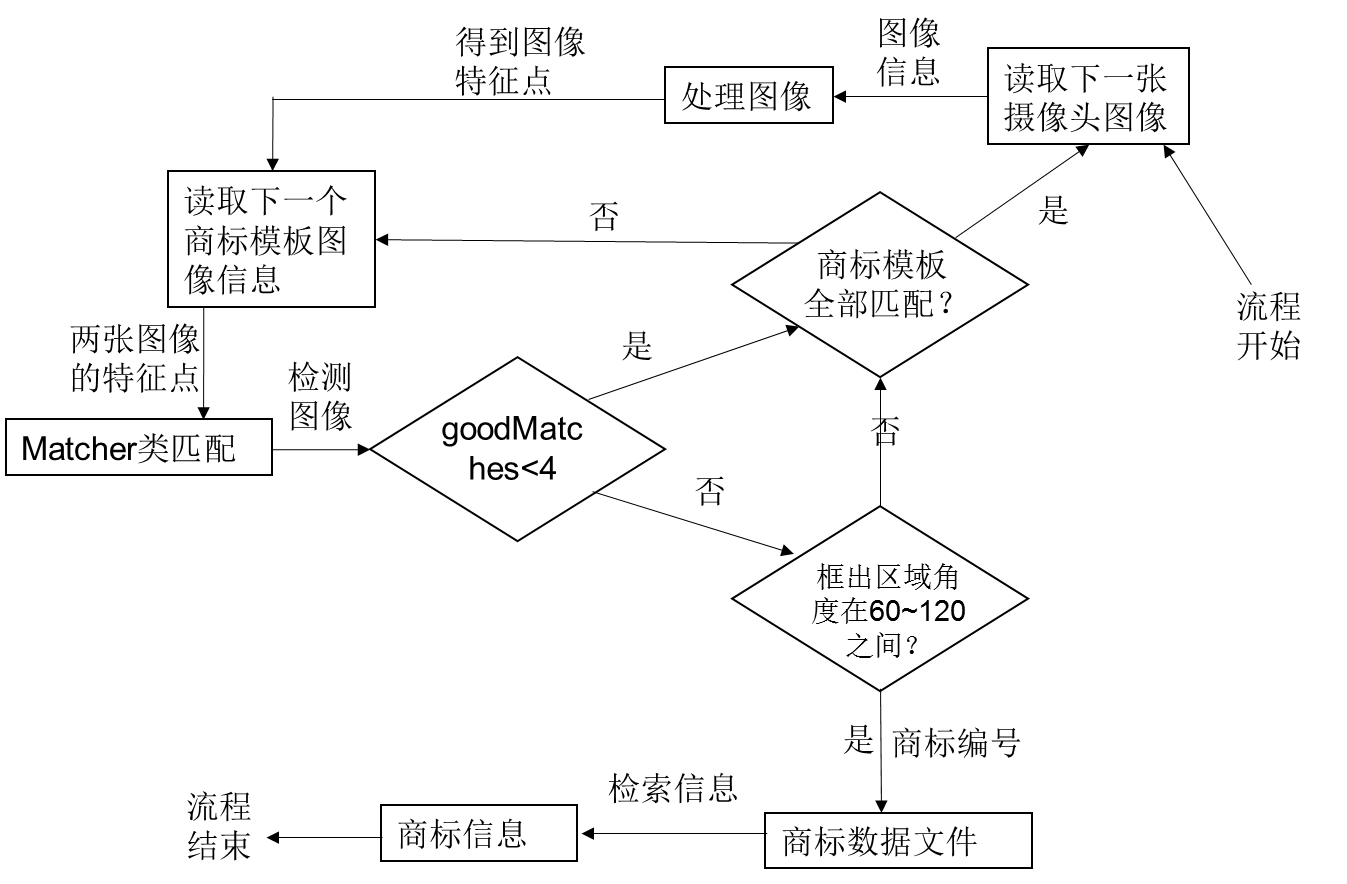


图 4.5 商标匹配流程示意图

### 图像存储及获取实现

在功能实现中为了使匹配的实际尽可能的短，需要缩减不必要的计算，所以图像数据文件中存储的并不是完整的商标模板图像，等到使用时再去计算其匹配点，而是存储的图像相关信息。同时商标的信息也储存在这里，等到匹配到正确商标文件时去进行查询。

这里的数据文件采用的是XML文件，XML文件在传输信息方面十分的便捷和好用，而且OpenCV有直接支持XML文件读取、写入的函数，所以这里就要使用XML文件存储相关信息。XML文件中存储的信息如下：

* icon\_rows：商标的行像素。int类型
* icon\_cols：商标的列像素。int类型
* icon\_keypoint：图像的关键点。vector<keypoint>类型
* icon\_descriptor：图像的描述子。Mat类型
* icon\_id：商标的编号。int类型
* icon\_name：商标的名称。string类型
* icon\_website：商标介绍视频的网址信息。string类型

但是在APP中C++代码无法直接使用FileStorage类读取数据，因为无法得到数据文件放置的地址信息，XML数据文件不能直接放在代码段的文件中，因为这些文件在打包成apk时都会被转换成二进制文件只能放在Res或Asset文件夹中，但是访问Res文件需要Java定义的R类型，C++无法使用，所以想要访问数据文件就只能将数据文件放在Asset文件中，使用AssetManager类传递地址，具体代码如下。

|  |
| --- |
| //scanner.java  private AssetManager am;  private native int findIcon(long frameMatAddr,long featureAddr, AssetManager am);  // Java\_com\_zzu\_me\_jniUtils\_Scanner.cpp  JNIEXPORT jint JNICALL Java\_com\_zzu\_me\_jniUtils\_Scanner\_findIcon (···, jobject assetManager){  AAssetManager\* mgr = NULL;  mgr = AAssetManager\_fromJava(env, assetManager );  AAsset\* asset = AAssetManager\_open(mgr, "matData.XML", AASSET\_MODE\_UNKNOWN);  char\* buffer = (char\*) malloc (sizeof(char)\*size);  AAsset\_read (asset,buffer,size);  jint rs = (jint)findIcon((Mat\*)frameAddr,(SiftFeature\*)featureAddr, buffer);  free(buffer);  AAsset\_close(asset); |

在scanner.java文件中定义AssetManager类型的am变量，am可以理解成一个存储文件储存位置的对象，在findIcom函数中传入这一对象，在JNI函数的定义文件Java\_com\_zzu\_me\_jniUtils\_Scanner.cpp中使用这一参数。使用AssetManager\_fromJava函数，传入JNIEnv\*类型的参数ent，这个参数是系统自动生成的参数；传入assetManager，这是一个jobject类型的参数，代表这是一个java语言中的对象，函数得到一个AAssetManager\*类型的变量mgr，即可用在C++语言中使用的AssetManager类型，也就通过这个数据将Asset的地址传给了C++语言，使用AAssetManager\_open传入mgr变量、文件名称和一个参数便可获得一个AAsset类型的指针，相当于获得了这个文件的操作指针类似于FILE指针，使用AAsset\_read函数，传入文件指针，读取文件的buffer和读指针指向文件的字符数，将文件中的全部内容读到buffer中；在使用buffer中的内容时只需要使用OpenCV自带的FileStorage类来读取文件内容即可。创建FileStorage对象，使用重载的“[]”符号输入想要访问的字段的名称便可成功获得相应的数据。而得到匹配商标编号后，在buffer中使用编号及字段名便可找到每个商标对应的各种信息。这样便可成功从C++代码中读到APP数据文件中存储到的相关信息了。

## 本章小结

本章按照系统实际使用的流程介绍了系统的各个环节的具体实现，讲解了关键代码与技术，分析了实现的考虑和理由，对各个函数也都进行了完整的介绍。

# 系统测试

之前介绍了系统各个模块的设计和具体实现。本章将对系统进行具体的测试，测试系统功能是否正常，测试完整的系统流程，以及测试完成商标识别的时间。

## 测试环境

由于编写的APP是Android平台的APP，所以这里作为测试平台的是华为的安卓智能机荣耀畅玩X3，与测试相关的环境参数如下：

* 出厂日期：2013年7月
* CPU：联发科 MT6592 1.7GHz；
* 后置摄像头像素：1300万像素；
* 内存：2G；
* 系统：Android 4.2.2

## 系统测试

本节进行系统测试，测试系统的完整流程是否正确。系统运行流程如下：



图 5.1 系统进入首界面

1. 首先是点击APP图标，进行此APP，显示系统界面如图5.1所示。

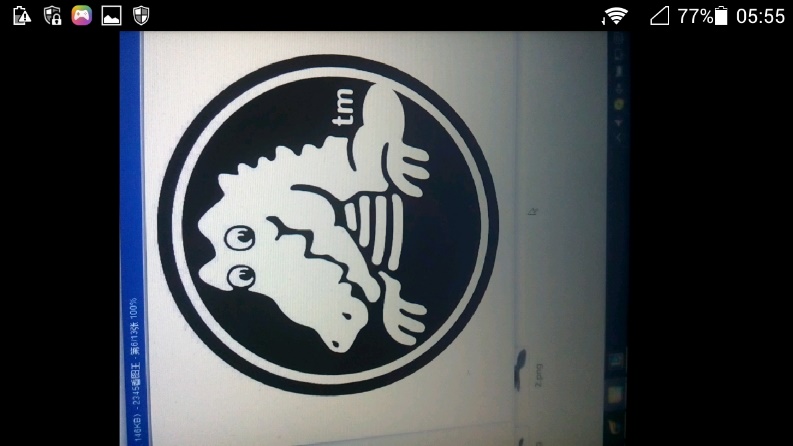


图 5.2 图像获取界面

1. 点击其中的“扫一扫”按钮，进图像获取界面，图像获取界面如图5.2所示。这里以crocs图标为例，检索crocs图标。



图 5.3 信息显示界面

1. 扫描图像后经过几秒钟的运算处理，会跳到crocs图标信息显示界面，如图5.3所示。



图 5.4 视频浏览

（4）点击其中的观看相关视频，会打开系统浏览器进入视频网址播放视频，而对于不太适合播放视频的商标，则打开其相关主页，如图5.4所示。

以上便是系统运行的基本流程。经测试，系统可以正常的运行，能够正确识别出图像中包含的商标，并显示商标的相关信息。

对于系统关键步骤——检测商标的用时一般是在5秒左右，检测用时受手机性能、光照、图像清晰度、图像复杂程度以及商标信息在数据文件中的位置等等多重因素影响，无法得到一个准确的结果。根据多次测试在本安卓手机测试的匹配用时基本不高于5s，匹配时间适宜，不会有匹配过慢的感觉。系统测试基本成功。

# 总结与展望

## 工作总结

本文以研究并设计一个基于SIFT算法的商标识别系统为目标，首先研究CBIR系统的概念及内容，学习系统的基本工作流程，了解之前存在的CBIR系统的运行原理，实际使用一些识图系统进行感受。

确定以SIFT算法作为系统的关键算法，学习掌握SIFT算法的原理，学习一些SIFT的实例，首先在C++上写一些小的程序尝试使用SIFT算法进行图像的匹配工作。确定使用OpenCV库编写SIFT程序，写一个使用OpenCV的SIFT算法进行关键的图像特征提取及匹配工作，并将其封装为一个C++的方法。

为了最终的演示效果，决定做一个Android的APP进行演示，学习Android的编程知识，了解其基本使用方法，开始编写程序界面和其中的逻辑代码，对于关键的图像识别代码，出于效率的考虑将使用C++代码，所以学习JNI的使用，在Android APP中调用封装的C++方法做图像特征提取与匹配。

寻找待匹配的模板商标，思考其存储方式，为了运行的效率首先对其进行特征提取，数据文件中存储特征而不是商标模板图像，但是数据特征不适宜存在数据库中，通过搜索学习SIFT特征的存储和读写方式，将数据最终存储在XML文件中。

完成APP后，进行系统调试，解决运行中出现的bug，优化代码，提升运行速度，最后进行实际测试，检测系统功能及运行速度。最终实现了一个基于图像识别技术的商标检测系统，以SIFT算法为核心，进行商标图像的特征提取和检测，可以识别出数据文件中存储的约十个商标图像，识别之后能够得到商标的相关信息。

## 工作展望

由于本系统中很多技术都是第一次接触，所以实现的基于图像识别技术的商标检索系统虽然有了一定的可用性，但是很多方面还需要进一步的研究和改进。

* + - 1. 系统数据文件无法装载过多的图像信息，否则会造成匹配时间过长的现象，需要研究新的效果更好的特征匹配方法。
      2. 系统从摄像头获取文件的大小很大程度决定了将来匹配所用的时间，所以需要对所获取图像进行进一步的处理再提取特征信息。
      3. 无论数据文件中存储多少模板，都是有限的，今后可以增加网络获取模板图像的功能，遇到无法识别的图像，尝试从网上获取更多的商标模板进行匹配。
      4. 手机在运行图像处理程序时由于自身性能的限制速度不可能很快，为了提升速度，可以考虑搭建一个服务器处理摄像头获取的图像，将处理信息传回手机，这样可以提升匹配速度。

# 致谢

能够顺利完成这次毕设首先我要感谢的就是牛建伟老师，牛老师对待毕设十分严格，对我高标准严要求，没有一丝马虎，经常挤出自己宝贵的时间为我们开会督促我们的毕设进程，毕设遇到困难只要找老师讨论老师也都是来者不拒，不管多忙都会及时做出回复。

其次要感谢李青锋学长，学长待人亲切，平易近人。对待毕设一丝不苟，时刻督促我们抓紧进度，毕设遇到任何难题他都会帮我们想办法、出主意，有任何不会的可用随时向他咨询，做的迷茫的时候他也会给我点拨方向，重新找到思路。在代码调试时遇到困难向他请教也是有求必应，放下自己的工作帮我解决问题，经常和我一讨论就是五、六个小时。

还要感谢我的舍友，在程序调试出现问题的时候是他们陪我一起不厌其烦的一遍一遍调试，APP的调试需要一遍遍重装，是他们陪我一起做、一起等，没有他们的帮助我是不可能顺利完成毕设的。

还要感谢我的父母对我一如既往的支持，感谢计算机学院的各位老师教会我这么多的专业知识，感谢计算机学院、感谢北航给我四年充实的大学生活，谢谢！

# 参考文献

[1] Y Rui, TS Huang, S Mehrotra.Content-based image retrieval with relevance feedback in MARS[C]. International Conference on Image Processing, 1997, 2:815

[2] 李向阳,庄越挺,潘云鹤.基于内容的图像检索技术及系统[J].计算机研究与发展，2001，38(3)

[3]K.Hirata, T.Kato. Query by Visual Example - Content based Image Retrieval [C]. International Conference on Advances in Database Technology - EDBT, 1992, 580(12):56-71

[4] Arnold W.M. Smeulders, Marcel Worring, Simone Santini, Amarnath Gupta, and Ramesh Jain. Content-Based Image Retrievalat the End of the Early Years[C].IEEE Transactions on Pattern Analysis and Machine Intelligence, VOL. 22, NO. 12, DECEMBER 2000

[5] Ying Liu, Dengsheng Zhang, Guojun Lu, Wei-Ying Ma, survey of content-based image retrieval with high-level semantics [J].Pattern Recongection, 2007, 40(1):262-282

[6] S Singh，T Sontakke. An Effective Mechanism to Neutralize the Semantic Gap in Content Based Image Retrieval (CBIR) [J]. International Arab Journal of Information Technology, 2014, 11(2):124-133

[7] 祝晓斌,刘亚奇,蔡强,曹健. 基于内容的图像检索技术研究[J].计算机仿真, 2015, 32(5):1-4

[8] MS Zarchi,A Monadjemi,K Jamshidi. A semantic model for general purpose content-based image retrieval systems [J]. Computers & Electrical Engineering, 2014, 40(7):2062-2071

[9] J Ashley, M Flickner, J Hafner,D Lee, W Niblack. The query by image content (QBIC) system [J]. Acm Sigmod Record, 1995, 24(2):475

[10] Y Li，P Zhang. Content-Based Image Retrieval Using Hybrid Micro-Structure Descriptor[C] Intelligence Science and Big Data Engineering. Image and Video Data Engineering. Volume 9242 of the series Lecture Notes in Computer Science pp 411-419, Springer International Publishing, 2015.

[11] A Bala，T Kaur. Local texton XOR patterns: A new feature descriptor for content-based image retrieval [J]. Engineering Science & Technology an International Journal, 2015, 19(1):101-112

[12] 杨光皓，霍智勇，梁雨，郭云仙. 基于内容的图像检索(CBIR)中图像颜色特征提取方法的研究和改进[J]. 电脑知识与技术, 2014(11):2642-2646

[13] N Shrivastava，V Tyagi. Content based image retrieval based on relative locations of multiple regions of interest using selective regions matching. Information Sciences, 2014, 259(3):212-224

[14] DG Lowe. Distinctive image fearures from scale-invariant keypoints. 《International Journal of Computer Vision》, 2004, 60(2):91-110

[15] 冯嘉. SIFT算法的研究和改进[D]. 硕士学位论文. 吉林：吉林大学，2010

[16] 黄超，齐英剑. SIFT算法研究和应用[J]. 中国传媒大学学报:自然科学版, 2012, 19(1):68-72

[17] H Bay，T Tuytelaars，LV Gool. SURF: Speeded Up Robust Features[J]. Computer Vision & Image Understanding, 2006, 110(3):404-417

[18] K Yan，R. Sukthankar. PCA-SIFT: a more distinctive representation for local image descriptors[C]. CVPR'04 Proceedings of the 2004 IEEE computer society conference on Computer vision and pattern recognition, Pages 506-513

[19] J Luo，O Gwun. A Comparison of SIFT, PCA-SIFT and SURF[J]. Journal of Business Education, 2009, 3(4):143-152

[20] 邹承明，徐泽前，薛栋. 一种基于分块匹配的SIFT算法[J]. 计算机科学, 2015, 42(4):311-315

[21] G Bradski. The Opencv Library [J]. Doctor Dobbs Journal, 2000, 25(11):384-386

[22] 杨秋英.二维条码—QR Code研究及应用[J]. 华北工学院, 2003