分类号 密级

UDC注1



**硕士专业学位论文**

**基于Android平台的服饰图像搜索**

**系统的设计与实现**

**李刚**

**指导教师姓名 赵春霞 教 授**

**学 位 类 别 工程硕士**

**专 业 名 称 计算机应用**

**研 究 方 向**

**论文提交时间**

注1：注明《国际十进分类法UDC》的类号。

**声 明**

本学位论文是我在导师的指导下取得的研究成果，尽我所知，在本学位论文中，除了加以标注和致谢的部分外，不包含其他人已经发表或公布过的研究成果，也不包含我为获得任何教育机构的学位或学历而使用过的材料。与我一同工作的同事对本学位论文做出的贡献均已在论文中作了明确的说明。

研究生签名： 年 月 日

**学位论文使用授权声明**

南京理工大学有权保存本学位论文的电子和纸质文档，可以借阅或上网公布本学位论文的部分或全部内容，可以向有关部门或机构送交并授权其保存、借阅或上网公布本学位论文的部分或全部内容。对于保密论文，按保密的有关规定和程序处理。

研究生签名： 年 月 日

# 摘 要

近年来，随着智能手机的迅速普及和电子商务的蓬勃发展，移动电子商务也越来越吸引人的目光。随着人们生活节奏的不断加快，网上购物以其实惠便利的优势获得越来越多人的青睐。智能手机一个重要功能是拍照。xxxxxxx

本文研究了基于内容的图像检索相关技术，结合服饰图像搜索的特殊要求。最终选择了量化过的HSV颜色分布直方图作为颜色特征，Uniform LBP特征作为纹理特征，SIFT特征作为局部特征，描述图像。并利用GrabCut分割算法对图像进行分割减少背景的干扰。SIFT特征借助视觉词袋技术进行降维后用于匹配。图像特征采用欧式距离进行相似度度量。使用TOP K算法对距离进行排序，得到相似度最高的图像。

本文介绍了基于Android的服饰图像搜索系统的设计和现实过程。系统分为客户端、服务器端、图像集和特征池三大部分。客户端基于Android实现，包括待搜索图像输入模块、搜索结果展示模块、其它辅助功能模块。服务器端基于Python Django实现，包括颜色特征提取、纹理特征提取、SIFT特征提取、视觉词、排序等模块。图像集和特征池部分主要包括网络爬虫模块。

**关键词**：Android，基于内容的图像检索，LBP，SIFT，特征提取

# Abstract

- 4 -

In this dissertation, a research has been made on the surface hydrophilic modification of porous polyvinylidene fluoride(PVDF) membrane based on supramolecular chemistry process and its application in the treatment of low concentration oily wastewater.

**[Document Title]**

First, the mechanism of surface hydrophilic modification of PVDF membrane based on supramolecular

**Key word:**Android, CBIR, LBP, SIFT, Feature extraction

**目录**

[摘 要 - 3 -](#_Toc438219375)

[Abstract - 4 -](#_Toc438219376)

[1 绪论 7](#_Toc438219377)

[1.1课题的研究背景和意义 7](#_Toc438219378)

[1.2 国内外研究现状 8](#_Toc438219379)

[1.3 论文的主要工作 9](#_Toc438219380)

[1.4 论文的组织结构 9](#_Toc438219381)

[2 基于内容的图像检索技术 10](#_Toc438219382)

[2.1 概述 10](#_Toc438219383)

[2.2 颜色特征 11](#_Toc438219385)

[2.3 纹理特征 13](#_Toc438219386)

[2.4 SIFT特征 15](#_Toc438219387)

[2.5 视觉词袋模型 18](#_Toc438219388)

[2.6 图像分割技术 18](#_Toc438219389)

[3 Android平台相关技术 20](#_Toc438219390)

[3.1 Android平台的发展 20](#_Toc438219391)

[3.2 Android平台的组成 21](#_Toc438219392)

[3.2.1 Linux内核 21](#_Toc438219393)

[3.2.2 系统运行库 21](#_Toc438219394)

[3.2.3应用程序框架 22](#_Toc438219395)

[3.2.4应用程序 23](#_Toc438219396)

[3.3 Android开发技术 23](#_Toc438219397)

[3.3.1 Activity 23](#_Toc438219398)

[3.3.2 Service 25](#_Toc438219399)

[3.3.3 Broadcast 26](#_Toc438219400)

[3.3.4 ContentProvider 28](#_Toc438219401)

[4 基于Android平台的服饰图像搜索系统实现 28](#_Toc438219402)

[4.1系统架构 28](#_Toc438219403)

[4.2 Android客户端实现 30](#_Toc438219404)

[4.2.1 逻辑架构 32](#_Toc438219405)

[4.2.2 图像输入模块 33](#_Toc438219406)

[4.2.3 搜索结果展示模块 36](#_Toc438219407)

[4.2.4 其它辅助模块 41](#_Toc438219408)

[4.3 图像集及特征池的建立 47](#_Toc438219409)

[4.3.1 图像集的建立 47](#_Toc438219410)

[4.3.2 特征池的建立 48](#_Toc438219411)

[4.4 服务器端实现 48](#_Toc438219412)

[4.4.1 Python简介 48](#_Toc438219413)

[4.4.2 Django简介 50](#_Toc438219414)

[4.4.3 GrabCut模块 52](#_Toc438219415)

[4.4.4 颜色特征提取模块 53](#_Toc438219416)

[4.4.5 纹理特征提取模块 54](#_Toc438219417)

[4.4.6 SIFT特征提取模块 55](#_Toc438219418)

[4.4.7 视觉词模块 56](#_Toc438219419)

[4.4.8 排序模块 57](#_Toc438219420)

[致 谢 58](#_Toc438219421)

[参考文献 58](#_Toc438219422)

# 1 绪论

传统的服饰搜索通常采用关键词搜索技术对服饰信息进行整合，忽略了电子商务信息中重要的图片信息。另一方面，用户并不总能对商品的关键词进行准确描述，而随着智能手机的普及，随时随地用手机拍摄的服饰照片则对实物具有更直观、准确的描述。本文基于以上问题，提出移动设备上基于图片搜索的服饰检索系统，构建“以图搜图”的应用平台。本文采用图像识别技术，基于移动互联网实现服饰检索功能，通过提取图像特征，返回最相似的服饰列表，同时给用户提供即拍即搜的服饰搜索体验。目前市场上仅有很少的APP实现此功能，因此具有广阔的应用前景和推广价值。

## 课题的研究背景和意义

随着互联网的进一步普及和人们生活节奏的日益加快，网上购物受到了越来越多消费者的青睐。

2007年11月5日Google公司正式发布基于 Linux内核的开源手机操作系统 Android。自发布以来，该系统以其开放、自由的特性赢得了各大厂商、开发人员、以及用户的青睐。经过八年的发展，Android已经从最初的智能手机领域进入教育、医疗、军事、汽车、家居等重要行业。

随着智能手机前所未有的发展和迅速的普及以及时下流行的网络购物应用的日渐增多，移动电子商务逐渐成为一个新兴的领域。网上购物已经成为一种时尚与流行，网上购物者的数量在不断的增长，购物网站相继涌现、发展和壮大,比如淘宝网、天猫商城、京东、苏宁易购等等。和实体店购物不同的是，网络购物者都是利用计算机和网络来浏览所需要的商品。由于缺乏现实购物过程中的视觉和触觉体验，网络购物者很难挑选到合适的商品。因此，一个电子商务平台的商品搜索系统就显得尤为重要。

多数购物网站设有自己的信息检索系统，像淘宝、拍拍网等大型专业购物网站依赖于自己开发的商品搜索引擎。一般的购物网站都是嵌入了百度、搜狗等搜索功能来进行搜索。由于技术的原因，大部分购物网站对于商品图片信息的检索都是基于文本标注的形式来实现的。

图像信息在数据库中的检索方式有两种，即全文索引和关键词索引。其基本步骤就是在网页信息自动采集和标引作为搜索引擎的重要组成的基础上，建立全文索引和关键词索引。全文索引就是用图像所在网页的全部文字信息作为图像的注释，网页上任何文字信息都认为与图像相关，显然这种图像标注查全率高但是查准率却是很低；基于关键词的索引使用若干关键词来表示图像信息，这些关键词一般都是从卖家对商品的图像的描述中获得，相比之下查准率高，但是查全率低。

电子商务购物网站和一般网站相比具有其独特性，即为了直观呈现商品的信息，多使用图片作为信息载体。商品本身具有自己的特性，比如对时间的敏感性和对色彩的依赖性。这些因素导致图像数量的飞速增长和图像信息数量的爆炸。

采用上述两种检索方式存在着以下问题：

1. 商品销售方对商品信息分类方式的多样性。销售人员是网上信息的发布者，他们不可能按照有关分类法来进行信息的分类，而是根据自己的理解对图片文件加注标签，具有很强的主观性。因而必然会增加标引词的多样性，增加了检索的范围。
2. 消费者对商品信息标签理解的歧义性。即由于个人认识的差异，不同的人对待相同的标签可能就会有不同的理解。
3. 视觉依赖性强的商品类型划分不同。即对于这种视觉依赖性强的商品，不同的角度划分就会导致所属的类型不同。
4. 商品信息标注的不完全性。即商品的文本标注不能全部反映图像的内容。

由于上述各种主观因素的存在导致基于文本的信息检索不能够很好的满足消费者所需商品的检索匹配。用户通过文本搜索检索不到自己心仪的商品，这在一定程度上将会降低购物者的消费需求。如果有一种检索方式可以避免图像文字标注的主观性和差异性所带来的匹配不精确问题，而直接客观地从商品的图像来检索匹配，不仅可以提高检索的查准率和查全率，还可以提高搜索的时间效率。于是基于内容的图像检索在购物网站中的应用成为一种迫切的需要与必然趋势。

## 1.2 国内外研究现状

基于内容的图像检索逐渐成为图像理解和计算机视觉领域的热门研究课题，国内外的研究结构已经投入了大量人力物力开展了对该课题的广泛研究，并且研制了一些商业系统和实验系统。比较知名的基于内容的图像检索系统有IBM公司的QBIC系统、由哥伦比亚大学研究开发的VisualSEEK和WebSEEK系统、由美国Virage公司开发的Virage系统、由美国MIT媒体实验室开发的Photobook系统等。

同时近年来，计算机视觉技术和电子商务得到了迅猛的发展。基于图像外部特征描述的商品检索方式已经不能很好的满足用户的需求，因此，国内外的研究机构开始将基于内容的图像检索技术引入到了商品的检索和浏览中，其中比较有代表性的图像搜索引擎如QBIC、谷歌图像搜索以及移动端图像检索系统Google Goggles。

（1）QBIC

QBIC系统是由IBM Almaden研究中心开发的第一个商品化的基于内容的图像检索系统，它的系统框架、结构和技术对后来的图像检索系统有着深远的影响。QBIC系统支持基于例子图像、手绘略图、选择的颜色、纹理等查询，不仅支持图像检索，还支持视频、文本和语音等多种形式的信息检索。QBIC系统是少数几个考虑高维特征索引的系统。QBIC系统使用的颜色特征是颜色直方图。纹理特征采用粗糙度、对比度和方向性描述。形状特征包括面积、离心率、主轴方向和不变炬。颜色、纹理和形状均采用加权的欧氏距离比较。

（2）谷歌图像搜索系统

2011年6月份谷歌发布了以图搜图的图片搜索服务，谷歌以其强大的搜索功能而著称，以图搜图的搜索技术也相对比较成熟。谷歌图像搜索融合了图像分析、模式识别、人机交互等多种技术，提供关键字、视觉特征相结合的检索模式，提高了查询的准确性，提升了用户体验。其操作流程比较简单，用户通过选择查询图像的相关特征如颜色、纹理等进行检索。Google图像检索具有理想的检索效率，谷歌图像检索系统结合了图像特征和关键字对查询图像进行分析进而匹配出最相似的图片集合，提高了图像检索的准确率，优化搜索结果，改善了用户体验。

（3）Google Goggles

Google Goggles是谷歌推出的一款移动端的图像搜索工具，利用手机摄像头拍摄当地地标建筑、书籍封面、艺术作品、酒类标签以及产品商标等物体的照片后，软件就将会自动在Google上搜索相关信息，并予以识别显示。不过Google Goggles对查询图像的类别有一些限制，不能检索家具、服装之类的东西。

## 1.3 论文的主要工作

本论文围绕着基于内容的图像检索系统，进行了如下工作：

阅读大量资料，研究基于内容的图像检索系统的特征提取方法及特征匹配方法。最终选择了HSV颜色直方图、Uniform LBP、SIFT作为本系统的特征提取方式，以计算各图像之间欧式距离作为特征匹配方法。

研究Android平台的组成与特性，以及应用层开发技术。

搭建图像检索系统，实现拍照、结果展示功能。系统拍照后检索出相似的图片结果，并在用户界面展示给用户。~~最后，测试图像检索系统，得出并分析实验数据。~~

## 1.4 论文的组织结构

本文共划分为四个章节，每章的主要内容如下所示。

第一章，绪论。主要介绍了系统上基于内容的图像检索研究的背景与意义，说明了当前国内外对基于内容图像检索的研究实践现状。给出了本论文的主要工作以及论文组织结构。

第二章，基于内容的图像检索技术。本章主要介绍了基于内容的图像检索原理，着重解析了HSV颜色直方图、Uniform LBP、SIFT这三种特征提取方式，并介绍了本文项目中使用到的GrabCut算法和视觉词袋模型。

第三章，Android平台相关技术。本章首先介绍了Android的发展历程和发展现状。然后~~说明了~~Android平台的组成。最后~~解析了~~Android应用开发的关键技术。

第四章，系统实现。详细介绍了Android客户端、服饰图像集及特征池、服务器端的实现过程。认真阐述了工程工作中遇到的问题及解决办法。

第五章，实验结果与讨论。对实现的图像检索系统进行测试，分析检索效果。

第六章，研究总结及展望。在这一章将总结前面的研究及实践成果，讨论并展望下一步研究方向。

# 2 基于内容的图像检索技术

为了搭建上的图像检索系统，本论文首先研究了基于内容的图像检索原理。基于内容的图像检索包含图像特征抽取和特征匹配两大关键技术。在本章节中，会对基于内容的图像检索原理进行概述。在小节中介绍了本论文选用的特征描述符的原理，详细解析了的颜色、纹理信息提取及的执行与量化。在小节中会对本论文使用的三种特征匹配方式：谷本系数、余弦相似度和曼哈顿距离进行介绍。

## 概述

随着互联网设备的发展以及图像采集设备的普及，使得快速、大量的采集、传输图片信息成为可能。由于一张图片往往难以用几个关键词准确形容，原来简单的基于文本的图像检索已经不能满足用户的需求。为了解决基于文本的图像检索中的诸多不足，人们想到了利用图像的内容进行检索。基于内容的图像检索简称应运而生。基于内容的图像检索是指由计算机自动提取图像的某些特征信息，并采用某种相似度的方法对这些特征进行匹配，并最终返回用户所需结果的技术。不同于基于文本的图像检索，基于内容的图像检索要直接从图片中提取特征值，而不需要标注关键字。这之后，基于内容的图像检索将所有图片特征值集合起来形成一个图片特征库。对于待检索图像，提取相同的特征值与图片特征库中的特征值逐一进行相似度测量，最后按照相似度的大小返回给用户，以方便用户的使用、查询。

相比于基于文本的图像检索，基于内容的图像检索具有以下三个特点。第一是，直接从图像提取特征值，突破了传统图像检索中基于文本关键词检索的局限。第二是，其本质是一种近似匹配技术，不同于常规的精确匹配方法，通过度量相似度，最后排列检索结果进行输出。第三是，图片特征提取通过计算机自动完成，减少了人工加注标示的工作量，并且消除了人主观性的干扰。基于内容的图像检索系统的工作流程示意图如图中所示。包含两个阶段：一是特征值预先提取阶段，二是图像搜索阶段。在特征值预先提取阶段，抽取图像数据库中的每一张图片的特征，存放在特征数据库中。在图像搜索阶段，利用同种特征提取手段提取待搜索图像的特征值，之后和特征数据库中的特征值逐一进行匹配，计算特征相似度，以获得最佳检索结果。

基于内容的图像检索选用的特征必须能够区别目标并且足够形容目标。基本上，实现一个成功的检索系统的关键就是选择一组特征，这组特征描述能力要强而且要独特以便利于区分判别。关于系统的类型，它们可以分为基于色彩特征的，基于纹理特征的以及基于形状特征的。仅用一种特征种类很难以达到理想的效果。很多图像检索技术都釆用提取多种图像特征放到一个直方图中的方法来进行图像特征抽取。

~~本文选用的特征提取方式为，这是一种将颜色信息和纹理信息结合于一个直方图的描述符。的优点是限制了描述符的大小，将每张图提取出的特征限制在字节。保证其仅需较少的时间就可以完成特征提取工作。~~

~~关于的具体内容将在中加以详细叙述。~~



## 颜色特征

在基于内容的图像检索中，图像的内容由图像的特征表示。图像的特征主要分为低层视觉特征和高层语义特征两大类。目前，受到计算机视觉、心理学、生物学等学科发展水平的制约，基于高层语义特征的图像检索技术还很不成熟。因此，目前基于内容的图像检索技术的研究热点是基于低层视觉特征的图像检索技术。

颜色特征具备旋转不变性和尺度不变性，而且相比其它特征容易获取，是描述一幅图像最简单有效的特征。因此颜色特征在基于低层视觉特征的图像检索中应用广泛。

RGB颜色空间是通过红绿蓝三原色来描述颜色的颜色空间，是最基本、最常用的颜色空间。数字图像一般均采用RGB颜色空间来表示。然而，RGB颜色空间的分量与亮度密切相关，即只要亮度改变，3个分量都会随之相应地改变。所以，RGB颜色空间适合于显示系统，却并不适合于图像处理。

HSV颜色空间是通过色调（Ｈ）、饱和度（Ｓ）、量度（Ｖ）３个分量来描述颜色的颜色空间，是一种面向视觉感知的颜色空间。其中，色调是指图像的主色，与混合光谱中主要光波长有关，如黄、绿、青代表不同的色调；饱和度是指色彩的深浅程度，与一定色调的纯度有关；亮度是指人眼感受到的光的明暗程度。HSV颜色模型与人眼的视觉特征比较接近，所以HSV颜色空间在图像颜色特征提取与分析中应用广泛。

从RGB颜色空间到HSV颜色空间的转换是一个计算简单的非线性变换。

RGB颜色空间到HSV颜色空间的转换公式如下所示：





其中，，，， 。

为了计算的便捷，需要对颜色进行量化，即将颜色空间映射到一个给定的子集中，使其总体误差最小。

1. 根据人的视觉分辨能力，把色调 H 空间分成 8份，饱和度S空间分成3 份，亮度V空间分成3份。







2) 根据色彩的不同范围和主观颜色感知进行量化

3) 构造一维特征矢量。按照以上的量化级，把各颜色分量合成为一维特征矢量：



其中，和分别是S、V的量化级数，，。

公式x实际上为：



这样，H,S,V三个分量在一维矢量上分布开来。W的取值范围是。计算 W获得72 bin一维直方图。

|  |  |
| --- | --- |
| 原始图像 | 颜色直方图 |
| Z:\170_0.jpg | \\Mac\Home\Documents\cf_170.png |

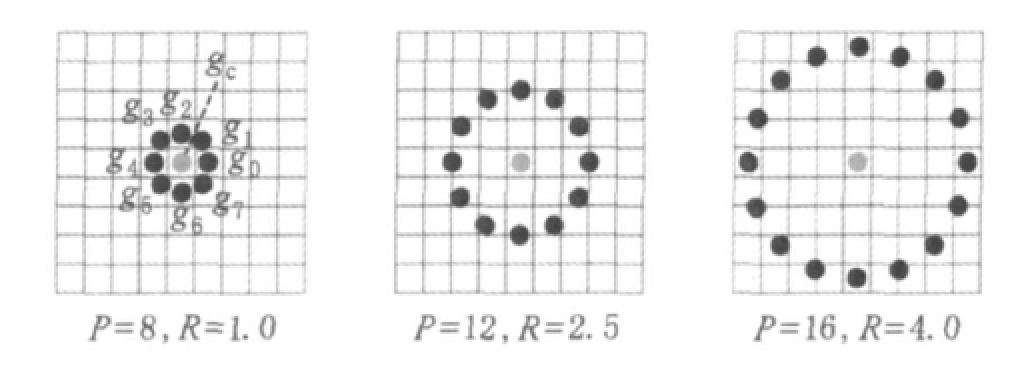
## 2.3 纹理特征

Castlema等人认为[2]：纹理是一种反映图像中一块区域的像素灰度级的空间分布属性，这种空间结构的固有属性可以通过邻域像素间的相关性刻画。常用的纹理分析方法有四种：统计分析方法、结构分析方法、模型分析方法和频谱分析方法。由于纹理特征提取计算成本过高，影响了图像纹理处理的实际应用，一些研究人员在图像纹理处理的计算简化上做了许多工作。Ojala 等人[15]提出了具有开创性的纹理特征提取方法，其计算简单可行，已经在许多研究领域内取得了实效。

局部二元模式（local binary pattern，简称LBP）是一种局部纹理描述算子。由于具有计算简单、对光照变化不敏感等优势，在纹理分类、人脸识别、医学图像处理等领域应用广泛。

基本LBP算子计算过程为：将中心像素点3\*3邻域内的8个像素点的灰度值分别与中心像素点的灰度值比较大小，根据大小关系对8个相邻像素点进行二值化。即如果相邻像素点大于中心像素点的灰度值，则将其置为0，否则置为1。然后按顺时针方向对这些值进行加权求和，得到该邻域的LBP特征值。

基本LBP算子具有无法提取大尺度纹理特征的局限性。为了提高LBP算子的有效性和完整性，Ojala等人对基本LBP算子进行了改进，LBP算子的计算不在局限于3\*3窗口内的相邻的8个像素点，而是给定一个采样半径R和采样点数P，在中心像素点为圆心、R为半径的圆周上等间隔的采样P个点。常用的P、R取值组合有P=8，R=1.0、P=12，R=2.5、P=16，R=4.0，如图x所示。

现在介绍P=8，R=1.0时改进LBP算子的计算过程，其它取值组合依次类推。设中心像素点的灰度值为，相邻的8个像素点的灰度值依次为，那么该中心像素点的LBP算子可表示如下：





LBP8,1.0有种不同的取值，如果我们采用LBP8,1.0特征直方图来表示一副图像的话，需要统计种类别，这个向量将是bin的，不便于计算。Ojala等人经过对大量纹理图像进行研究之后发现，如果将LBP算子二进制值首尾相连组成环，绝大多数的环至多存在2次0和1之间的跳变。以LBP8,1.0为例，00000000、01110000、01010000分别包含0、2、4次0和1之间的跳变。基于这样的统计规律，Ojala等人提出了均匀模式的LBP算子，即将至多存在2次0和1之间的跳变的LBP算子定义为均匀模式，分别归类。将存在2次以上0和1之间的跳变的LBP算子定义为混合模式，归为一类。压缩之后，LBP算子的种类可由原来的减少为。均匀模式占所有模式中的绝大多数，这样利用均匀模式和混合模式求统计直方图时，可以在不损失太多信息的情况下，而大大减少统计的类别。

本文纹理特征的提取使用均匀模式的LBP8,1.0，将原来的28种LBP特征压缩为8\*(8-1)+3=59种LBP特征，方便计算图像的LBP直方图。

|  |  |
| --- | --- |
| 原始图像 | 均匀模式LBP特征直方图 |
| Z:\139.jpg | \\Mac\Home\Documents\nlbp_139.png |

## 2.4 SIFT特征

图像的高斯尺度空间

尺度空间理论的出现是为了模拟图像数据的多尺度特征。Koenderink与Lindeberg证明了高斯卷积核是唯一的线性尺度核。一副图像的尺度空间被定义为：



其中为尺度可变高斯函数：



式中：为尺度坐标；为尺度因子，决定图像的平滑程度，大尺度对应图像概貌特征（低分辨率），小尺度对应图像细节特征（高分辨率）。

SIFT特征

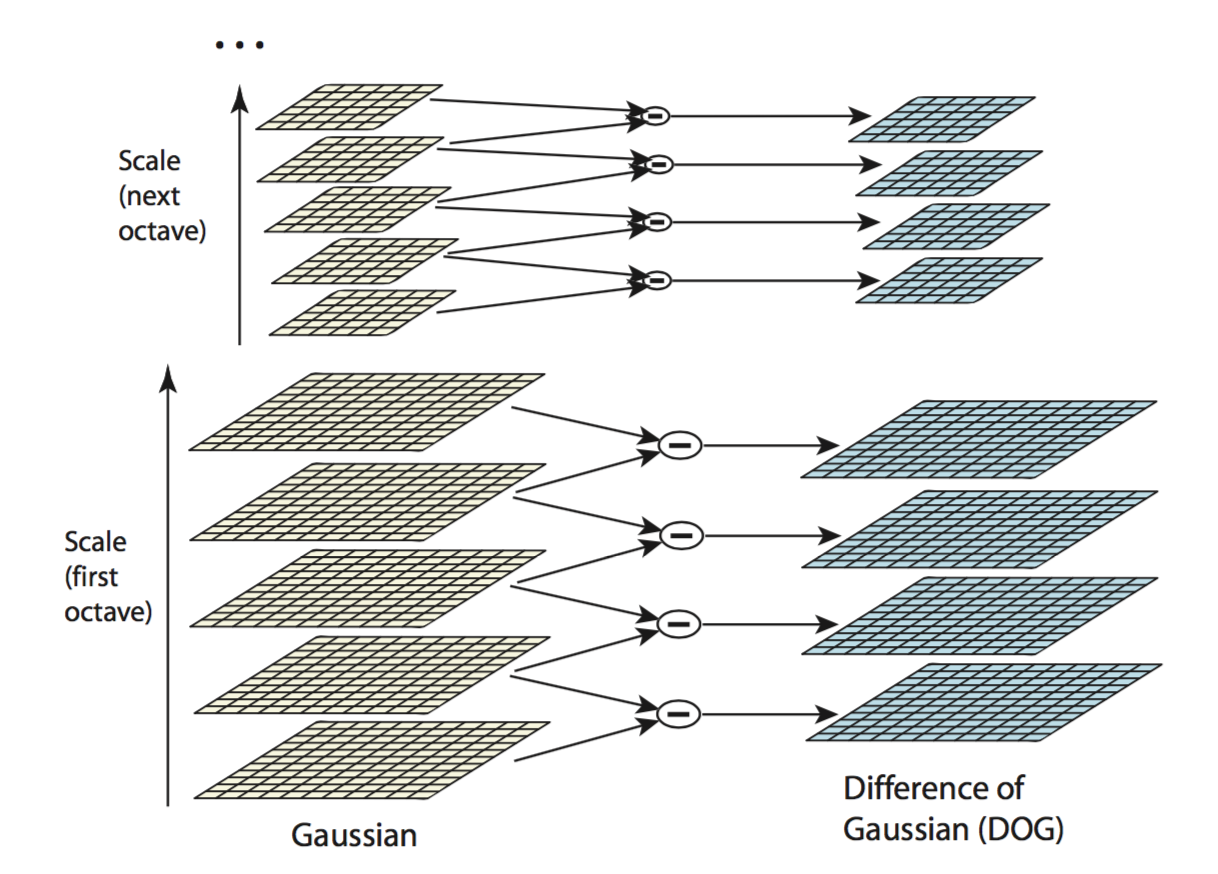
SIFT(Scale-Invariant Feature Tansform)描述子由Lowe于1999年提出，2004年总结完善。SIFT对旋转、尺度缩放、亮度变化具有不变性，对视角变化、仿射变换、噪声的容忍度较高。应用范围包含物体辨识、机器人地图感知与导航、影像缝合、3D模型建立、手势辨识、影像追踪和动作比对。SIFT特征提取主要分为4个步骤：

构建高斯差分尺度空间（DOG scale-space）

高斯差分尺度空间利用不同尺度高斯差分核与图像卷积生成。是两个相邻尺度图像的差。



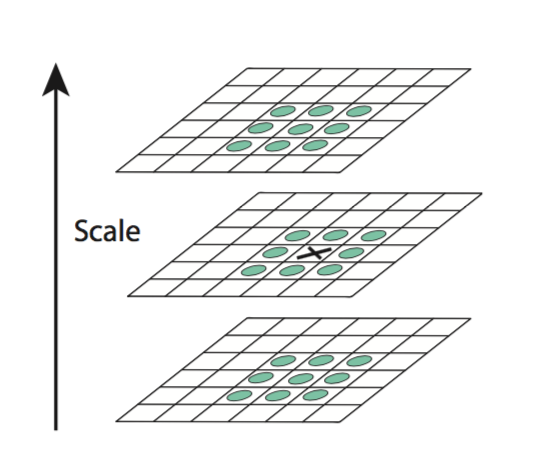
如图x所示。



图ｘ构建高斯差分尺度空间（DOG scale-space）

关键点定位

如果一个像素在DOG尺度空间本层以及上下两层的领域中具有最大值或最小值时，则认为该像素是图像在该尺度下的一个关键点。如图x所示。



图x DoG尺度空间关键点定位

中间的检测点和它同尺度的8个相邻点和上下相邻尺度对应的9×2个相邻点共26个相邻点进行比较，以确保在尺度空间和二维图像空间都检测到极值点。

关键点方向确定

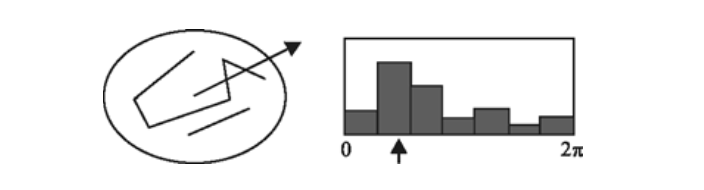
基于关键点邻域像素的梯度分布特性，分配给每个关键点一个或多个方向参数。后续的对图像的操作均相对于关键点的方向、尺度和位置进行变换，从而提供对于这些变换的不变性。



公式x、x分别为处梯度的模值和方向公式。其中所用的尺度为每个关键点各自所在的尺度。

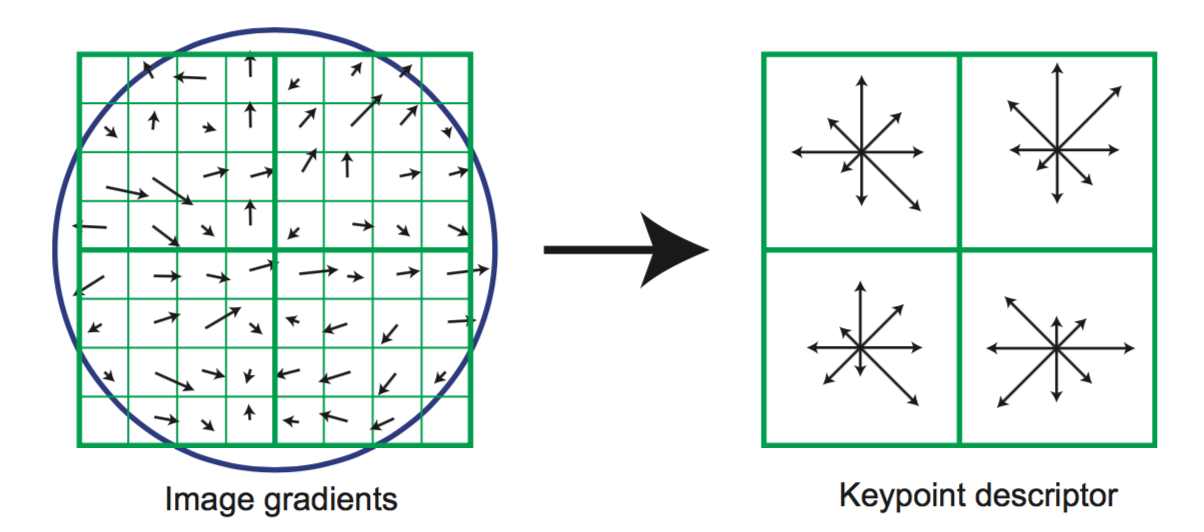
在实际计算时，我们在以关键点为中心的邻域窗口内采样，并用直方图统计邻域像素的梯度方向。梯度直方图的范围是0～360度，其中每45度一个柱，总共8个柱, 或者每10度一个柱，总共36个柱。Lowe在论文中建议使用高斯函数对直方图进行平滑，减少突变的影响。直方图的峰值代表该关键点处邻域梯度的主方向，用作该关键点的方向。图x是采用7个柱来统计邻域像素梯度方向并确定主方向的示例。



图x确定关键点方向

关键点描述子的生成

首先将坐标轴旋转为关键点的方向，以确保旋转不变性。以关键点为中心取8×8的窗口。



图ｘ关键点描述子的生成

图左部分的中央为当前关键点的位置，每个小格代表关键点邻域所在尺度空间的一个像素，利用公式x求得每个像素的梯度幅值，利用公式ｘ求得每个像素的梯度方向，箭头方向代表该像素的梯度方向，箭头长度代表该像素的梯度模值，然后用高斯窗口对其进行加权运算。图中圈代表高斯加权的范围（越靠近关键点的像素梯度方向贡献越大）。然后在每4×4的小块上计算8个方向的梯度方向直方图，绘制每个梯度方向的累加值，即可形成一个种子点，如图右部分示。此图中一个关键点由2×2共4个种子点组成，每个种子点有8个方向信息。这种邻域方向性信息联合的思想增强了算法抗噪声的能力，同时对于含有定位误差的特征匹配也提供了较好的容错性。

计算关键点周围的16×16的窗口中每一个像素的梯度，并使用高斯下降函数降低远离中心的权重。这样就可以对每个关键点形成一个4×4×8=128维的描述子。将这一向量归一化之后，可进一步去除了光照的影响。

## 2.5 视觉词袋模型

词袋（Bag of words）[x]是文档分析领域统计关键词出现频率的一项技术，被广泛应用于信息检索和文本分类。视觉词袋模型是词袋模型在计算机视觉领域的应用。它将图像映射为视觉单词的分布直方图，既保存了图像的局部特征又压缩了图像的描述。

视觉词袋实现过程如下：

步骤一：特征提取。从图像集的图片中提取出所有局部特征。

步骤二：聚类生成视觉词典。将步骤一种得到的特征集利用K-means进行聚类。设是一组观测值序列，其中，每个观察值都是一个d维向量。利用K-means将这n个观察值划分到k个序列中，其中是的均值。



步骤三：将图像用词典表示。将图像中的局部特征映射到距离最近的视觉词上，统计图像中各个视觉词的频率，得到分布直方图，最终用直方图K维向量表示图像。其中，K为词典中包含的视觉词数量。局部特征之间的相似度用欧式距离度量。

## 2.6 图像分割技术

图像分割是图像工程中目标检测、特征提取和参数测量的基础，是图像分析、模式识别、计算机视觉领域的关键问题之一，它使得高层次的图像理解成为可能。

利用基于能量最小化框架的图割理论进行图像分割已成为近年来的一个研究热点。它的优势包括它的全局最优求解能力以及结合了多种知识的统一图像分割框架。在此基础上，针对不同应用场景，人们提出了多种变种分割方法[1-6]。

在Interactive Graph Cuts算法[x]基础上，Rother等人提出了GrabCut算法。改进的内容包括：（1）采用高斯混合模型（Gaussian Mixture Model，GMM）替代灰度直方图，支持彩色图像分割；（2）在GMM参数估计过程中，采用多次迭代算法替代一次最小估计；（3）算法采用非完全标号（incomplete labelling）的方式，降低了用户的交互工作量。

图割理论

图像分割即把像素标为前景/背景，是典型的二元标号问题。首先构造一个能量函数，用于计算像素的标号值。之后借助网络流理论，把标号问题转化为最大流/最小割问题解决。

设为一无向图，是定义在边集上的容量函数：，则无向图及其边集上的容量函数构成一个网络，记作，其中，是网络的源点，是网络的汇点。借助最小化能量函数把顶点集划分为两个顶点集，，分别与源点和汇点相连，(,)。



其中，为的一个标号，；为数据项，用来衡量和所观察到的数据的不一致性；为光滑项，用来衡量非分片光滑的程度；为相互作用的相邻顶点对。

网络中的顶点对应图像中的像素，网络边上的容量对应像素特征之间的差异或者相似度，即用网络表示一个图像。图像分割能量函数的最小值对应网络的最小割。根据最大流/最小割定理，网络的最大流与最小割是等价的。最终，能量函数的最小化问题转化为网络的最大流问题。

GrabCut算法

GrabCut算法将图像分割问题定义为：对图像的每个像素点，求其标号值，其中，1代表前景；0代表背景。代表前景/背景的概率密度模型。借助图割理论，图像分割问题可表示为



GrabCut算法基本步骤包括：首先，由用户在前景周围画一个矩形，通过“非完全标号”的方式来标定图像的背景区域（Trimap Background）和未知区域（Trimap Unknown），矩形框外为背景区域，矩形框内为未知区域。将未知区域内的像素标注为1，背景区域内的像素标注为0，目标区域（Trimap Foreground）设为空。利用用户标定的背景区域和未知区域分别初始化前景/背景GMM。然后将把未知区域划分为前景/背景两类，对新划分的前景/背景像素进行切割，更新GMM。在新的GMM参数下继续对未知区域进行划分，迭代直至满足收敛条件，最终确定GMM参数。利用最终的GMM对未知区域进行一次切割，最终得到前景图像。

# Android平台相关技术

## 3.1 Android平台的发展

Android是一个以Linux为基础的开放源代码移动操作系统，主要用于智能手机和平板电脑，由Google成立的Open Handset Alliance（OHA，开放手持设备联盟）持续领导与开发中。目前Google发布Android的最新正式版本为Android 6.0“Marshmallow”。 Android系统最初由安迪·鲁宾（Andy Rubin）等人开发制作，最初开发这个系统的目的是创建一个数码相机的先进操作系统；但是后来发现市场需求不够大，加上智能手机市场快速成长，于是Android被改造为一款面向智能手机的操作系统。于2005年7月11日被美国科技企业Google收购 。2007年11月，Google与84家硬件制造商、软件开发商及电信营运商成立开放手持设备联盟来共同研发改良Android系统，随后，Google以Apache免费开放源代码许可证的授权方式，发布了Android的源代码，让生产商推出搭载Android的智能手机，Android操作系统后来更逐渐拓展到平板电脑及其他领域上。 2010年末数据显示，仅正式推出两年的Android操作系统在市场占有率上已经超越称霸逾十年的诺基亚Symbian系统，成为全球第一大智能手机操作系统。 在2014年Google I/O开发者大会上Google宣布过去30天里有10亿台活跃的安卓设备，相较于2013年6月则是5.38亿。

## 3.2 Android平台的组成

Android系统大致可以分为五层，由下到上依次为Linux内核层、Android运行层、函数库层、应用程序框架层和应用程序层。各层各司其职，使用下层提供的服务，屏蔽本层及下层的差异，为上层提供统一的服务，具有高内聚、低耦合的优点。

### 3.2.1 Linux内核

Linux核心层是Android的最底层。Andriod基于Linux2.6提供包括进程管理、内存管理、网络、硬件驱动等核心系统服务。同时，Linux核心层也是硬件和软件之间的抽象层，它屏蔽硬件差异为上层提供统一服务。

### 3.2.2 系统运行库

Android运行层的核心是Dalvik虚拟机。在Android系统中，每一个Android应用程序对应一个Dalvik虚拟机。Dalvik虚拟机在设计上也使得一台移动设备可以高效的运行多个它的实例。Dalvik虚拟机可执行文件格式是dex，dex格式是专为Dalvik虚拟机设计的压缩格式，适合内存和处理器速度有限的移动设备。开发Android应用时，Java程序通过编译生成.class文件，还需要通过SDK中提供的dx工具转化为.dex文件才能在虚拟机上执行。一个.dex文件通常会包含多个class。Dalvik虚拟机依赖下层的Linux内核来实现线程和内存管理等功能。

函数库层主要是C/C++函数库，主要有：

* 系统C库——标准C系统库（libc）的BSD衍生，为基于嵌入式Linux设备调整
* 媒体库——基于PacketVideo的OpenCORE。这些库支持播放和录制许多流行的音频和视频格式，以及静态图像文件，包括MPEG4、 H.264、 MP3、 AAC、 AMR、JPG、 PNG
* 界面管理——管理访问显示子系统和无缝组合多个应用程序的二维和三维图形层
* LibWebCore——新式的Web浏览器引擎,驱动Android 浏览器和内嵌的web视图
* SGL——基本的2D图形引擎
* 3D库——基于OpenGL ES 1.0 APIs的实现。库使用硬件3D加速或包含高度优化的3D软件光栅
* FreeType ——位图和矢量字体渲染
* SQLite ——所有应用程序都可以使用的强大而轻量级的关系数据库引擎

### 3.2.3应用程序框架

应用程序框架层是与Android开发者联系最紧密的一层，包括一组服务和组件，主要有：

* 视图（View）——丰富的、可扩展的视图集合，可用于构建一个应用程序。包括列表、网格、文本框、按钮，甚至是内嵌的网页浏览器
* 内容提供者（Content Providers）——使应用程序能访问其他应用程序（如通讯录）的数据，或共享自己的数据
* 资源管理器（Resource Manager）——提供访问非代码资源，如本地化字符串、图形和布局文件
* 通知管理器（Notification Manager）——使所有的应用程序能够在状态栏显示自定义警告
* 活动管理器（Activity Manager）——管理应用程序生命周期,提供通用的导航回退功能

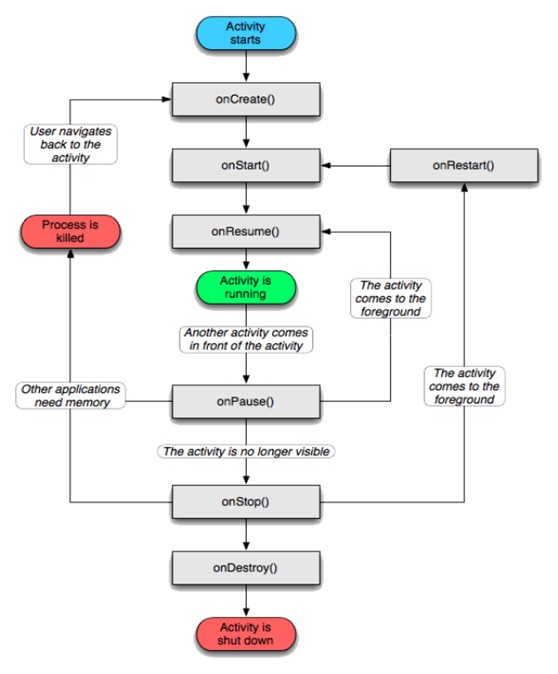
### 3.2.4应用程序

应用程序层在Android系统的最上层，面对用户提供服务，我们平常使用的拨号、消息、计算器等应用均位于此层。

## 3.3 Android开发技术

### 3.3.1 Activity

Activity称为活动,是应用程序的表示层,每个程序包括一个或多个Activity。应用程序的每个屏幕显示都通过继承和扩展Activity基类实现。应用程序通过startActivity或startActivityForResult方法从一个活动转到另一个活动。若新的被激活的activity组件属于另一个应用程序，则那个activity组件会运行在那个应用程序的进程中，但是从用户的角度来看，好像就是属于本应用程序一样。Android是通过将之前的activity组件和新被激活的activity组件放入同一个任务栈来实现这个功能的。从用户的角度看，一个任务栈就代表了“一个应用程序”。它实际上是一个栈，里面放着一组被排列好的相关的activity组件。位于栈底的activity（根activity）就是开启这个任务栈的activity组件，一般情况下，就是应用程序的主界面。而位于栈顶的activity组件即代表当前被激活的activity组件（可接收用户行为的activity）。任务栈中包含了activity组件的对象，且任务栈中可以包含有某一个activity组件类型的多个实例对象。在任务栈中的activity组件不能被重排序，只能被压栈和弹栈。任务栈不是某个类型，也不是某一个元素，它是一组activity组件的组织形式。所以没有办法在不影响任务栈中的activity组件的情况下，单独设置任务栈的参数。根activity的参数既是整个任务栈的参数，它会影响任务栈中的所有activity组件。当某个应用程序在前后台切换的时候，实际上就是代表这个应用程序的一个任务栈在前后台切换。Activity的状态从创建到销毁可能经历多种状态，在不同状态之间切换时,将调用不同的回调方法（CallBack），如图一所示。



Activity生命周期中包含四种状态，分别是活动状态、暂停状态、停止状态和非活动状态。

* 活动（Active/Running）状态

当Activity运行在屏幕前台（处于当前任务活动栈的最上面），此时它获取了焦点，可以响应用户的操作，属于运行状态。同一个时刻只会有一个Activity 处于活动（Active/Running）状态。

* 暂停（Paused）状态

当Activity失去焦点但仍对用户可见（如在它之上有另一个透明的Activity或Toast、AlertDialog等弹出窗口时）它处于暂停状态。暂停的Activity仍然是存活状态（它保留着所有的状态和成员信息并保持和窗口管理器（WindowsManager）的连接）。但是当系统内存极度紧张时可能被系统杀掉。

* 停止（Stopped）状态

当Activity完全被另一个Activity遮挡时处于停止状态，它仍然保留着所有的状态和成员信息。只是对用户不可见,当其他地方需要内存时它往往被系统杀掉。

* 非活动（Dead）状态

当Activity尚未被启动、已经被手动终止、已经被系统回收时处于非活动的状态，要手动终止Activity，可以在程序中调用"finish"方法。

### 3.3.2 Service

Service跟Activity的级别差不多，区别是Service只能运行在后台，不提供用户界面，并且可以和其他组件进行交互。一个Service是一个可以长期运行在后台的应用程序组件，不提供用户界面。另一个应用程序组件可以启动一个服务，它将继续在后台运行，即使用户切换到另一个应用程序。此外，一个组件可以绑定到一个服务与之交互，甚至执行进程间通信(IPC)。例如，一个服务可以在后台处理网络交易、播放音乐、执行文件I / O，或与内容提供者交互等。

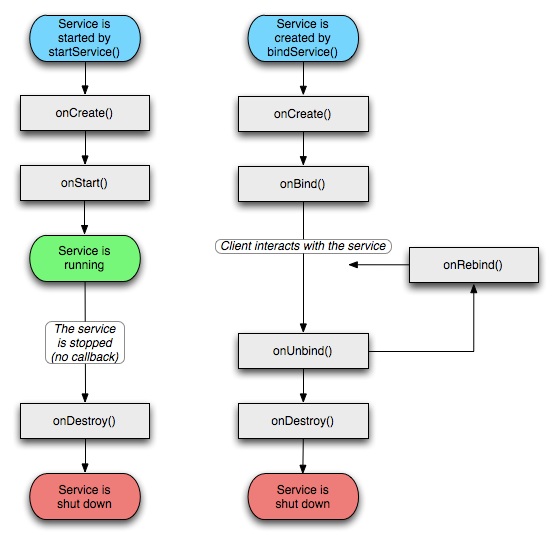
一个服务可以基本上有两种形式：

* Started

一个服务将被开启，当一个应用程序组件（例如，一个Activity）启动它，调用startService()。一个服务一旦启动，可以无限期地在后台运行，即使启动它的组件被摧毁。通常情况下，开始服务执行一个操作，不向调用者返回一个结果。 例如，它可能通过网络下载或上传文件。当操作完成，服务应该自动销毁。

* Bound

一个服务被“绑定”当一个应用程序组件绑定到它通过调用 bindService（）。一个绑定服务提供了一个客户端 - 服务器接口，允许组件与服务交互，发送请求，得到的结果，甚至这样做跨进程与进程间通信（IPC）。一个service可以同时和多个客户绑定，当多个客 户都解除绑定之后，系统会销毁service。

需要注意的是，一个服务运行在其宿主进程，该服务不会创建自己的线程，并不在一个单独的进程中运行的主线程（除非另行指定）。这意味着，如果你的服务是打算进行任何耗时操作（例如MP3播放或网络下载等）， 你应该在该服务中创建一个新的线程来执行该工作。通过使用一个单独的线程，你会减少应用程序的不响应（ANR）错误和应用程序的UI主线程可以继续致力于为响应用户的操作。

### 3.3.3 Broadcast

Broadcast使得应用可以对感兴趣的外部事件(如电话呼入、数据网络变为可用)进行接收并做出响应。广播接收器没有用户界面，但是可以启动一个Activity或Serice 来响应它们收到的信息，也可以用NotificationManager 来通知用户。通知一般来说是在状态栏上放一个持久的图标，用户可以点击打开它并获取消息。

广播类型：

（1）普通广播，通过Context.sendBroadcast(Intent myIntent)发送。

（2）有序广播，通过Context.sendOrderedBroadcast(intent, receiverPermission)发送。该方法第2个参数决定该广播的级别，级别数值是在 -1000 到 1000 之间 , 值越大 , 发送的优先级越高。广播接收者接收广播的优先级可通过intentfilter中的priority进行设置。priority设为2147483647表示优先级最高。同级别接收器的接受顺序是随机的。高级别的或同级别的接收器先接收到广播的可以通过abortBroadcast()方法截断广播，使其他的接收者无法收到该广播。

（3）异步广播，通过Context.sendStickyBroadcast(Intent myIntent)发送的。还有sendStickyOrderedBroadcast(intent, resultReceiver, scheduler, initialCode, initialData, initialExtras)方法，该方法具有有序广播的特性也有异步广播的特性；发送异步广播需要<uses-permission android:name="android.permission.BROADCAST\_STICKY" />权限，接收并处理完Intent后，广播依然存在，直到你调用removeStickyBroadcast(intent)主动把它去掉。

监听广播的步骤如下：

（1）写一个继承BroadCastReceiver的类，重写onReceive()方法。广播接收器仅在它执行这个方法时处于活跃状态。当onReceive()返回后，它即为失活状态。

（2）注册该广播接收者。注册有两种方法，分别是程序动态注册和在AndroidManifest文件中进行静态注册（可理解为系统中注册）如下：

静态注册，在AndroidManifest文件中进行声明。

<receiver android:name=".SMSBroadcastReceiver" >

　　<intent-filter android:priority = "2147483647" >

　　　　<action android:name="android.provider.Telephony.SMS\_RECEIVED" />

　　</intent-filter>

</receiver >

动态注册，一般在Activity可交互时（onResume()内）注册BroadcastReceiver。

使用Broadcast需要注意两点。

（1）BroadCastReceiver生命周期只有十秒左右，如果在onReceive() 内做超过十秒内的事情，就会报ANR(Application No Response) 程序无响应的错误信息。如果需要完成一项比较耗时的工作，应该通过发送 Intent 给 Service，由Service 来完成。这里不能使用子线程来解决，因为 BroadcastReceiver 的生命周期很短 ，子线程可能还没有结束BroadcastReceiver 就先结束了。BroadcastReceiver 一旦结束， 此时 BroadcastReceiver 的所在进程很容易在系统需要内存时被优先杀死 ，因为它属于空进程（没有任何活动组件的进程）。如果它的宿主进程被杀死 ，那么正在工作的子线程也会被杀死。所以采用子线程来解决是不可靠的。

（2）动态注册广播接收器还有一个特点，就是当用来注册的Activity关闭后，广播也就失效了。静态注册无需考虑广播接收器是否被关闭，只要设备是开启状态，广播接收器就是有效的。也就是说即便APP本身未启动，APP订阅的广播在触发时也会对它起作用。

### 3.3.4 ContentProvider

当应用继承ContentProvider类，并重写该类用于提供数据和存储数据的方法，就可以向其他应用共享其数据。ContentProvider为存储和获取数据提供了统一的接口。虽然使用其他方法也可以对外共享数据，但数据访问方式会因数据存储的方式而不同，如采用文件方式对外共享数据，需要进行文件操作读写数据；采用sharedpreferences共享数据，需要使用sharedpreferences API读写数据。而使用ContentProvider共享数据的好处是统一了数据访问方式。无论应用程序是否启动，其它应用程序都可以通过接口来操作该应用程序的内部数据，包括增加数据、删除数据、修改数据、查询数据等。

一般来说ContentProvider是单例模式的，当多个应用程序通过ContentResolver操作ContentProvider的数据时，ContentResolver调用的数据将会委托给同一个ContentProvider处理。

# 4 基于Android平台的服饰图像搜索系统实现

本章将详细介绍上基于内容的图像检索系统的架构及各个模块功能。在中将介绍本图像检索系统的模块划分。中将展示本系统的设计架构。

本系统共包含搜索图像输入、图片预处理、抽取图像特征值、特征匹配计算、搜索结果展示共个模块，这些模块的具体实现过程将在到小节中进行详细描述。

本论文实现的图像检索系统共包含以下这几个模块：

搜索图像输入模块：基于内容的图像检索需要用户输入检索图片，而不是输入关键词。本论文实现的图像检索系统支持两种输入图片的方法。一种是直接利用手机进行拍照，调用系统的拍照方法。用户拍照后将照片以临时文件的形式存储在卡当中，当做输入图像。第二种方式是直接读入手机当中的图片进行检索。在本系统中搭建了一个文件选择界面，用户可以选择自己想要检索的图片，输入的文件支持、图片类型。在这个模块的用户界面中，提供接供使用者设置特征类型及匹配方式。可以从非压缩型和压缩型中选择一种作为特征提取方式，从谷本系数、曼哈顿距离、余弦相似度中选择一种作为特征匹配方式。

## 系统架构

本文系统由三个部分组成。一是Android客户端。二是服饰图像数据库及特征池。三是服务器端。



2.2 开发平台

Android客户端开发使用Android Studio集成开发环境。服饰图像数据库及特征池的建立以及服务器端开发使用Pycharm集成开发环境。

JSON简介

JSON(JavaScript Object Notation) 是一种轻量级的数据交换格式，易于人阅读和编写，同时也易于机器解析和生成。它是基于JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999的一个子集。JSON采用完全独立于语言的文本格式，但是也使用了类似于C语言家族的习惯（包括C, C++, C#, Java, JavaScript, Perl, Python等）。这些特性使JSON成为理想的数据交换语言。

JSON建构于两种结构：

一是“名称/值”对的集合（A collection of name/value pairs）。不同的语言中，它被理解为对象（object），纪录（record），结构（struct），字典（dictionary），哈希表（hash table），有键列表（keyed list），或者关联数组 （associative array）。

二是值的有序列表（An ordered list of values）。在大部分语言中，它被理解为数组（array）。

这些都是常见的数据结构。事实上大部分现代计算机语言都以某种形式支持它们。这使得一种数据格式在同样基于这些结构的编程语言之间交换成为可能。

JSON具有以下这些形式：

对象是一个无序的“‘名称/值’对”集合。一个对象以“{”（左括号）开始，“}”（右括号）结束。每个“名称”后跟一个“:”（冒号）；“‘名称/值’ 对”之间使用“,”（逗号）分隔。

数组是值（value）的有序集合。一个数组以“[”（左中括号）开始，“]”（右中括号）结束。值之间使用“,”（逗号）分隔。

值（value）可以是双引号括起来的字符串（string）、数值(number)、true、false、 null、对象（object）或者数组（array）。这些结构可以嵌套。

字符串（string）是由双引号包围的任意数量Unicode字符的集合，使用反斜线转义。一个字符（character）即一个单独的字符串（character string）。

字符串（string）与C或者Java的字符串非常相似。

数值（number）也与C或者Java的数值非常相似。除去未曾使用的八进制与十六进制格式。除去一些编码细节。

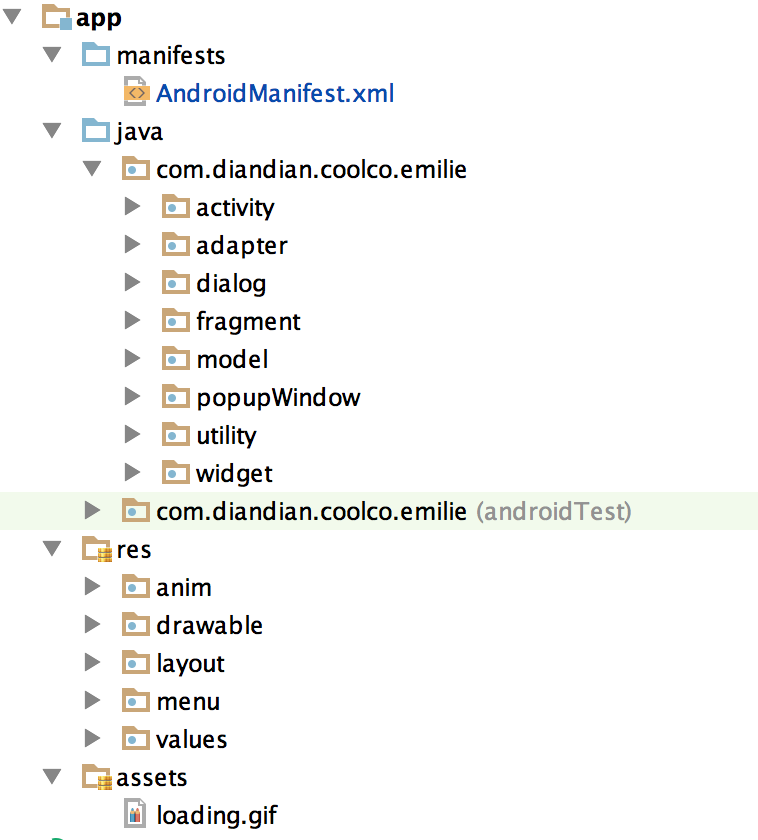
空白可以加入到任何符号之间。

## 4.2 Android客户端实现

Android客户端主要由查询图像的获取和搜索结果的展示的两个主要模块组成。查询图像的获取有两种方式，一是通过摄像头拍照得到，二是通过手机图库选择得到。搜索结果有两种展示模式，一是概要展示模式，将结果图片缩略图以瀑布流的形式展示给用户；二是详情展示模式，将结果图片原图以翻页的形式展示给用户，在详情展示模式下，支持查看服饰品牌、价格、材质等相关信息，并提供购买入口。除查询图像的获取和搜索结果的展示的两大主要模块之外，本文的Android客户端还包括收藏、分享、意见反馈、版本升级等辅助功能。功能结构如下图所示。



硬件要求：摄像头；系统要求：Android 4.0 以上；权限要求：访问摄像头权限（CAMERA）、写外部存储权限（WRITE\_EXTERNAL\_STORAGE）、访问网络状态权限（ACCESS\_NETWORK\_STATE）、访问网络权限（INTERNET）、挂载外部文件系统权限（MOUNT\_UNMOUNT\_FILESYSTEM）



本文项目名为Amelie（爱美丽），出自法国著名电影《天使爱美丽》，主打服饰图像搜索，目标人群锁定为都市时尚女性。

打开项目的Android移动客户端，首先看到的是启动页面（Splash）。启动页面播放动画的同时，客户端后台做初始化工作，比如检查更新、检查网络状态等。启动页面也包含了我们项目的名称和logo，有利于APP的推广。

【splash截屏】

### 4.2.1 逻辑架构

在逻辑上，本文将项目划分为四个层级：界面层、核心层、接口层、模型层。界面层负责UI展示；核心层负责处理业务逻辑；接口层负责与服务器进行交互；模型层定义所有的数据模型。如图所示，界面层依赖核心层、模型层，核心层依赖接口层、模型层，接口层依赖模型层，模型层不依赖任何层级。在构建项目，依照模型层、接口层、核心层、界面层的顺序依次构建。



模型层横跨所有层级，定义模型类，封装项目用到的数据。在本文项目中，主要定义了Image类，封装服饰图像的图像地址、购买地址、品牌、价格、材质等信息。

接口层的主要的功能是封装网络请求。接口层首先将请求封装好发送给服务器，然后将网络访问得到的数据转化为模型层定义的数据类型返回给上层的核心层。在本文项目中，客户端和服务器以json格式交互数据。发送请求使用HttpURLConnection系统类。请求结果处理上，本文利用Gson库将服务器返回的json格式服饰图像信息直接映射成为Image类型。

核心层介于接口层和界面层之间，主要处理业务逻辑。向上，向层界面层提供待展示的数据。向下，调用接口层获取服务器的数据。由于网络访问是异步的，此层暴露给上层界面层的API都有一个CallBack参数。界面层向核心层请求数据的时候，核心层会立即返回，在后台调用接口层向网络层请求数据，在得到数据并处理后通过CallBack返回给界面层。CallBack模式的引入可以避免网络请求阻塞UI线程造成的ANR(Application Not Responding)错误。

界面层处于最上层，负责UI的展示。根据不同类型划分，主要分为以下几个包：activity、adapter、fragment、util、view。其中activity、adapter、fragment各自都有一个基类，做统一的处理，比如定义了一些共用的常量、对象和方法等。

### 4.2.2 图像输入模块

启动页面之后，客户端将跳转到搜索主界面，如图xxx所示。

客户端支持拍照搜索，从本机图片中选取图片搜索两种搜索方式。点击拍照搜索将跳转到自定义的拍照页面进行拍照，点击选图搜索将跳转到自定义图片选择页面进行选图。尽管图像来源不同，拍照和选图都将返回一个图片的绝对路径，供后续搜索模块使用。

#### 自定义拍照页面

拍照功能可以调用系统服务完成。本文中的客户端采用自定义页面进行拍照的原因主要有两个。一是项目需要在拍照预览界面加矩形框，提醒用户将搜索目标置入矩形框内，进而提高搜索准确度。而系统服务并不支持界面的定制。二是不同版本的Android系统拍照界面也不同，一些版本的系统拍照界面十分简陋。。考虑到了用户体验的一致性以及界面风格的统一，本文项目的拍照功能由自己实现，而非调用系统服务。拍照页面如图xxx和xxx所示。

【有拍照按钮的页面】

页面背景为摄像头的实时预览（preview）。在实时预览上层有一个矩形框，框内透明，框外蒙着一层灰色，和矩形框上部的文字一起提醒用户将搜索目标置入矩形框内然后再拍照。矩形框和蒙层由一个自定义视图（Custom View）实现，这个自定义视图不借助Android SDK中的控件，显示效果完全由自己绘制。关键代码如下：

shadowPaint = new Paint();  
shadowPaint.setColor(Color.parseColor("#80000000"));  
clearPaint = new Paint();  
clearPaint.setXfermode(new PorterDuffXfermode(PorterDuff.Mode.CLEAR));

@Override  
 protected void onDraw(Canvas canvas) {  
// super.onDraw(canvas);  
 canvas.drawRect(previewRect, shadowPaint);  
 canvas.drawRect(boxRect, clearPaint);  
}

页面底部为拍照按钮。点击拍照按钮进行拍照。然后播放转场动画（Transition Animation）——拍照按钮向下坠落然后消失，取消、确定、重拍按钮向上升起然后出现。转场动画的存在让客户端的页面跳转更加友好，没有动画的过渡，用户会对界面的变化感到困惑（come from nowhere）。拍照后将出现取消、确定、重拍三个按钮。如果对拍照结果满意，点击确定按钮，跳转到搜索结果展示页面。如果对拍照结果不满意，点击重拍按钮。点击取消，跳转到搜索主界面。

如果点击了确定按钮，客户端将立即跳转到搜索结果展示页面，在此页面展示进度条（Progress Bar）提醒用户等待，避免出现应用程序无响应错误（ANR，Application Not Responding）。点击了确定按钮，拍照页面将把拍照所得图像做旋转处理并存储为临时图像文件并得到临时图像文件的路径的工作交给整个应用程序共用的线程池去执行，执行结束后通过EventBus异步的通知搜索结果展示页面。数据处理和页面跳转相互分离。之所以先跳转到搜索结果展示页面再通过广播把临时图像文件路径传递给搜索结果展示页面是为了减少一次进度条的展示。直接的做法是在拍照页面展示一次进度条，让用户等待APP存储临时图像文件，得到临时图像文件的路径后，夹带路径信息，跳转到搜索结果展示页面，展示一次进度条，让用户等待服务器返回搜索结果。但是这样处理，将连续展示两次进度条，而且两次之间用户没有任何操作，严重影响用户体验。通过上述的方法，可以很好的合并进度条，提升用户体验。关键代码如下：

private void rotateCropSavePicInBg(final byte[] data, final File pictureFile) {  
 ((MyApplication) getActivity().getApplication()).getAsyncExecutor().execute(new AsyncExecutor.RunnableEx() {  
 @Override  
 public void run() throws Exception {  
 byte[] rotatedBitmapData = rotateCropBitmap(data);  
 saveBitmap(rotatedBitmapData, pictureFile);  
 EventBus.getDefault().post(new Event.PictureOperationCompletedEvent());  
 }  
 });  
 }

可以画个流程图

#### 自定义图像选择器

图像选择也可以通过调用系统服务实现。之所以项目中选择自己实现图像选择器主要原因有两个。一是，Android系统图像选择服务以文件夹的形式组织图片，选中一张图片至少需要两步（选中相册、选中图片），并且选择过程中涉及界面的跳转，不够简单便捷。项目中的自定义图像选择器把图像文件夹选择和图像选择置于同一页面中，图像文件夹的选择以底部弹窗（Popup Window）的形式出现。自定义图像选择器还增加了所有图像这一逻辑文件夹，并按时间倒序排列图像，所有图像这一逻辑文件夹也是图像选择器的默认文件夹，所以，用户打开自定义的图像选择器就可以看到最近拍摄、保存的所有照片，点击即可选择目标图片。

图片的展示采用GridView布局。Android设备内存有限，在展示大量图像的时候容易导致OOM（Out of Memory）错误。造成OOM的主要原因是图像文件在展示之前需要解压（decode）为位图（Bitmap），而位图非常占用内存，在展示大量图片的时候容易导致OOM。针对这一问题，本文项目做了如下优化：

1. 使用缩略图代替原图。在图像选择器页面中，每张图片只占一格，其显示尺寸远远小于其实际尺寸。所以在解压图片的时候，可以一定的采样率去解压，然后进行伸缩变换，得到的位图的尺寸将与显示的尺寸的匹配，而占用内存远远小于将图片直接解压做占用的内存。关键代码如下所示：

options.inSampleSize = getPowerOfTwoForSampleRatio(ratio);

Bitmap sampledBitmap = BitmapFactory.decodeFile(srcImgPath, options);

Bitmap scaledBitmap = null;  
if ((float) options.outWidth / width > (float) options.outHeight / height) {  
 scaledBitmap = Bitmap.createScaledBitmap(sampledBitmap, width, ((int) (((float) width) \* options.outHeight / options.outWidth)), false);  
} else {  
 scaledBitmap = Bitmap.createScaledBitmap(sampledBitmap, ((int) (((float) height) \* options.outWidth / options.outHeight)), height, false);  
}  
if (sampledBitmap != scaledBitmap) {  
 sampledBitmap.recycle();  
}

1. 缓存缩略图。用户在选择图片的时候，经常会上下滑动。如果采取图片被滑出屏幕时被回收（recycle），滑入屏幕时重新解压（decode）的策略，有两个弊端。一是，解压是高计算量操作，频繁解压将大量消耗计算资源。二是，用户在上下滑动的时候，大量的图片被滑出屏幕和滑入屏幕，大量的位图将被创建和回收，导致频繁的GC（垃圾回收），将造成内存抖动，进而导致界面卡顿。在GingerBread（Android 版本2.3）之前，Dalvik虚拟使用的垃圾收集机制有以下特点：1. Stop-the-world，也就是垃圾收集线程在执行的时候，其它的线程都停止；2. Full heap collection，也就是一次收集完全部的垃圾；3. 一次垃圾收集造成的程序中止时间通常都大于100ms。 在GingerBread以及更高的版本中，Dalvik虚拟使用的垃圾收集机制得到了改进，如下所示：1. Cocurrent，也就是大多数情况下，垃圾收集线程与其它线程是并发执行的；2. Partial collection，也就是一次可能只收集一部分垃圾；3. 一次垃圾收集造成的程序中止时间通常都小于5ms。本文项目中，利用LRU策略缓存图片缩率位图，在现实的图片时，将首先到缓存查找对应的位图。如果命中，直接显示。如果未命中，从原图片文件中采样解压、缩放得到与显示尺寸匹配的位图，显示并加入缓存。图片在刚刚被滑出屏幕又被滑入屏幕时，不会被回收，也不需要重新从原图片文件中解压得到，很好的解决了上文描述的两个问题。关键代码如下所示：

private static LruCache<String, Bitmap> mMemoryCache;

private static ImageLoader mImageLoader;

private ImageLoader() {

int maxMemory = (int) Runtime.getRuntime().maxMemory();

int cacheSize = maxMemory / 8;

mMemoryCache = new LruCache<String, Bitmap>(cacheSize) {

protected int sizeOf(String key, Bitmap bitmap) {

return bitmap.getByteCount();

}

};

}

public void addBitmapToMemoryCache(String key, Bitmap bitmap) {

if (getBitmapFromMemoryCache(key) == null) {

mMemoryCache.put(key, bitmap);

}

}

public Bitmap getBitmapFromMemoryCache(String key) {

return mMemoryCache.get(key);

}

### 4.2.3 搜索结果展示模块

#### 搜索结果列表展示

搜索结果列表展示页主要功能是让用户上下滑动快速浏览系统搜索结果，并选择感兴趣的图片点击跳转到搜索结果详细展示页。此外，页面还提供继续裁剪待搜索图片、搜索结果数量统计、搜索耗时统计等辅助功能。搜索结果以瀑布流的形式分两列展示给用户，从上到下相似度依次降低。搜索结果图片的加载使用开源的图像加载组件Fresco。Fresco 是一个强大的图片加载组件。

1.Fresco 中设计有一个叫做 image pipeline 的模块。它负责从网络，从本地文件系统，本地资源加载图片。为了最大限度节省空间和CPU时间，它含有3级缓存设计（2级内存，1级文件）。

2.Fresco 中设计有一个叫做 Drawees 模块，方便地显示loading图，当图片不再显示在屏幕上时，及时地释放内存和空间占用。

3.Fresco 支持 Android2.3(API level 9) 及其以上系统。

Fresco在内存管理页表现突出。

解压后的图片，即Android中的Bitmap，占用大量的内存。大的内存占用势必引发更加频繁的GC。在5.0以下，GC将会显著地引发界面卡顿。在5.0以下系统，Fresco将图片放到一个特别的内存区域。当然，在图片不显示的时候，占用的内存会自动被释放。这会使得APP更加流畅，减少因图片内存占用而引发的OOM。关键代码如下：

public void setImageUrl(String imageUrl) {

this.imageUrl = imageUrl;

GenericDraweeHierarchyBuilder builder = new GenericDraweeHierarchyBuilder(getResources());

GenericDraweeHierarchy hierarchy = builder

.setProgressBarImage(new ProgressBarDrawable() {

@Override

protected boolean onLevelChange(int level) {

setLoadingProgress(level);

return true;

}

@Override

public void draw(Canvas canvas) {

}

})

.setPlaceholderImage(new ColorDrawable(getResources().getColor(R.color.drawee\_view\_place\_holder)))

.setActualImageScaleType(ScalingUtils.ScaleType.FIT\_CENTER)

.build();

draweeView.setHierarchy(hierarchy);

draweeView.setImageURI(Uri.parse(imageUrl));

}

Fresco的设计遵循MVC模式。DraweeView继承自View，负责图片的显示，相当于MVC中的V（View）。DraweeHierarchy用于组织和维护最终绘制和呈现的图片，比如上文代码中占位图片和按下蒙层图片的设置均通过DraweeHierarchy实现，相当于MVC中的M（Model）。DraweeController负责和image loader交互（默认是Fresco中的image pipeline），实现对所要加载的图片的更多控制，比如，图片加载失败情况下单击重新加载。相当于MVC中的C（Controller）。服务器返回的相似图片URL交给Fresco去加载，用户点击感兴趣的图片跳转到图片详情页面。

#### 搜索结果详细展示模块

图片详情页面全屏展示图片。为了给用户沉浸式体验，默认情况下，顶部ActionBar（工具条）不可见。单击图片可显示ActionBar。ActionBar的更多（overflow）菜单中包括收藏、分享、保存到相册、去购买、查看服饰信息等菜单项，如图xxx所示。

收藏操作将把当前图片URL及对应的服饰信息保存到本地数据库中，之后可在收藏页面查看。具体实现见下文收藏模块。

保存到相册功能是把服饰图片保存到本地相册。

去购买功能给用户提供了购买的渠道。购买地址是在使用网络爬虫抓取服饰图片时一并抓取并保存的。用户点击去购买菜单，APP将打开相应的购买网页，如图xxx所示。打开网页操作在APP内嵌的浏览器完成，并没有调用系统浏览器或第三方浏览器。内嵌浏览器保证了APP在各个Android版本的手机上使用体验一致。本文项目中实现的内嵌浏览器还提供了复制链接、用浏览器打开链接和分享三个辅助功能。其中，分享的内容并不仅限于服饰的购买地址，还包括APP相关信息。内嵌浏览器基于Android SDK中的标准组件WebView实现。Android从4.4起提供基于Chromium实现的WebView。此前WebView基于WebKit实现。WebKit提供网页解析、布局和绘制以及JS运行等基础功能。Chromium在WebKit基础上为WebView提供进程、线程和渲染等基础构架。因此基于Chromium实现的WebView更好地提供了网页浏览功能。

点击查看服饰信息菜单将弹出对话框（Dialog）用于展示服饰信息，如图xxx所示。展示的服饰信息包括服饰品牌、服饰价格等信息。

分享操作的实现的基础是系统的分享功能。Android有着完善的应用间通信机制，其与Unix的开发式工具链哲学一脉相承，是一个完整的体系。在 Android 中，所有的应用都向系统注册其所持有的组件和每个组件能够处理的任务。这样系统对于每个应用的每个组件能完成何种任务都了如指掌。当一个应用需要启动其它组件完成任务时，只需向系统发送一个消息，即可调用系统中可以完成该项任务的组件，而应用本身无需关心具体哪个应用的哪个组件最终处理了这个消息。这样跨应用的内容分享变得很方便。在本文项目中，要分享的是纯文本，其中包含了图片的URL以及服饰相关信息。关键代码如下：

Intent intent = new Intent(android.content.Intent.ACTION\_SEND);

intent.setType("text/plain");

intent.putExtra(android.content.Intent.EXTRA\_SUBJECT, subject);

intent.putExtra(android.content.Intent.EXTRA\_TEXT, text);

activity.startActivity(Intent.createChooser(intent, chooserTitle));

收藏、分享、查看服饰信息等命令除了通过点击相应菜单项触发以外，还可以由手势（gesture）触发。收藏通过下拉触发，（与常见的下拉刷新操作类似），如图xxx所示。如果已收藏，下拉则取消收藏，添加收藏与取消收藏均有文字提示。分享通过上拉触发，如图xxx所示。查看服饰信息通过双击触发，如图xxx所示。双击显示服饰信息的过程为3D翻页动画，感觉上服饰信息就在图片背后。翻页后，显示服饰信息，图片在背面。再次双击，再次翻页，图片回到正面，服饰信息被隐藏在背面。这种交互方式源于生活经验，也易于理解。手势操作的优势是全局触发，相比菜单项必须点击某一矩形区域而言，手势更加方便。除此之外，众所周知移动设备屏幕尺寸有限，使用手势代替菜单可以使界面简洁，释放大量视觉空间。并且，手势操作被用户接受并熟悉后，会变成用户的直觉化操作，高效而便捷。手势操作的实现有两个关键部分。

一是头部视图（Header View）和底部视图（Footer View）的隐藏与显示。Android中视图显示分为三个主要步骤，依次是测量（measure）、布局（layout）和绘制（draw）。所谓隐藏，其实是将头部视图显示在屏幕以上，底部视图显示在屏幕以下。具体实现方法是，自定义线性布局（LinearLayout），从上至下添加头部视图、内容视图（Content View）和底部视图。重写View中的onLayout方法，改写头部视图的布局参数，使得头部视图的左下角与父视图（自定义线性布局）的左上角重合，把头部视图显示在屏幕以上，用户不可见。同理，改写底部视图的布局参数，使得底部视图的左上角与父视图（自定义线性布局）的左下角重合，把底部视图显示在屏幕以下，用户不可见。头部视图和底部视图的显示通过函数scrollTo函数改变mScrollY实现。关键代码如下：

@Override

protected void onLayout(boolean changed, int l, int t, int r, int b) {

int width = r - l;

headerView.layout(0, t - headerView.getMeasuredHeight(), width, t);

contentView.layout(0, t, width, b);

footerView.layout(0, b, width, b + footerView.getMeasuredHeight());

}

@Override

protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec) {

super.onMeasure(widthMeasureSpec, heightMeasureSpec);

contentView.measure(MeasureSpec.makeMeasureSpec(getMeasuredWidth(), MeasureSpec.EXACTLY), MeasureSpec.makeMeasureSpec(getMeasuredHeight(), MeasureSpec.EXACTLY));

}

二是对触摸事件（Touch Event）的响应和对手势的响应。在Android中，一个手势以一个Down Event开始，中间有若干个Move Event，以一个Up Event或Cancel Event结束。三者要分别处理。在本文项目中，通过setOnTouchEvent截获TouchEvent。如果是Move Event，滚动整个视图，露出头部视图或者底部视图。如果是Up Event或Cancel Event，复原到初始状态。关键代码如下：

private void handleTouchEvent(MotionEvent event) {

if (downY == -1) {

downY = event.getY();

}

if (lastY == -1) {

lastY = event.getY();

}

float dy = event.getY() - lastY;

lastY = event.getY();

switch (event.getAction()) {

case MotionEvent.ACTION\_DOWN:

break;

case MotionEvent.ACTION\_MOVE:

if (scrolling) {

float calibratedTotalDy = (event.getY() - downY) \* SCROLL\_RATIO;

scrollTo(0, (int) -calibratedTotalDy);

} else {

float totalDy = event.getY() - downY;

if (Math.abs(totalDy) > touchSlop) {

scrolling = true;

float calibratedTotalDy = totalDy \* SCROLL\_RATIO;

scrollTo(0, (int) -calibratedTotalDy);

}

}

updateHeaderFooterView();

break;

case MotionEvent.ACTION\_UP:

case MotionEvent.ACTION\_CANCEL:

if (getScrollY() >= footerView.getMeasuredHeight()) {

if (pullListener != null) {

pullListener.onPullUp();

}

} else if (getScrollY() <= -headerView.getMeasuredHeight()) {

if (pullListener != null) {

pullListener.onPullDown();

}

}

smoothScrollTo(0);

lastY = -1;

downY = -1;

break;

}

}

在图片详情页面，用户可左右滑动来查看上一张图片和下一张图片。左右滑动浏览大图的实现使用的是Android SDK中的标准组件ViewPager。

### 4.2.4 其它辅助模块

#### 新手引导模块

上文中提到，本项目客户端中较多的使用了手势命令，为此本项目客户端也配备了新手引导模块。在用户第一次打开相关页面的时候，利用动画引导用户相关手势操作。引导动画相比文字、静态图片更加形象，最大程度的降低了用户的学习成本。在第一次打开相关页面的时候引导该页面可以使用的手势操作，可以把操作学习分散到各个实际场景，更有利于用户接受和学习。

引导动画的实现使用Android中的属性动画（ValueAnimator）。用SharedPrefrence记录用户是不是第一次打开此页面。

float handMoveDistance = getResources().getDimensionPixelOffset(R.dimen.show\_case\_hand\_move\_distance);

ValueAnimator animator = ObjectAnimator.ofFloat(pullUpHintView, "translationY", 0, -handMoveDistance);

animator.setDuration(2000);

animator.setRepeatCount(ValueAnimator.INFINITE);

animator.setRepeatMode(ValueAnimator.RESTART);

animator.start();

#### 收藏模块

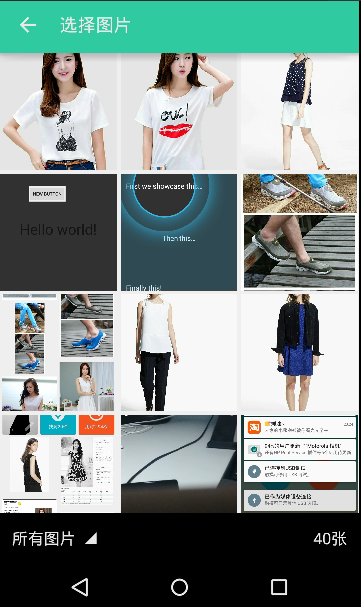
本文项目客户端支持用户收藏感兴趣的服饰。用户收藏服饰时，APP将服饰信息存入本地数据库。用户浏览自己的收藏时，APP从本地数据库读取数据并显示。本地数据库使用SQLite。SQLite是一个进程内的库，实现了自给自足的、无服务器的、零配置的、事务性的 SQL 数据库引擎。与其他数据库一样，SQLite 引擎不是一个独立的进程，可以按应用程序需求进行静态或动态连接。SQLite 直接访问其存储文件。客户端使用ORMLite简化数据库操作。ORMLite是一个轻量级的Java对象关系映射持久层框架。支持包括 MySQL、Postgres、Microsoft SQL Server、H2、Derby、HSQLDB和SQLite等在内的数据库。提供灵活的QueryBuilder来构建复杂的数据查询以及强大的抽象DAO（Database Access Object）类。

#### 用户模块

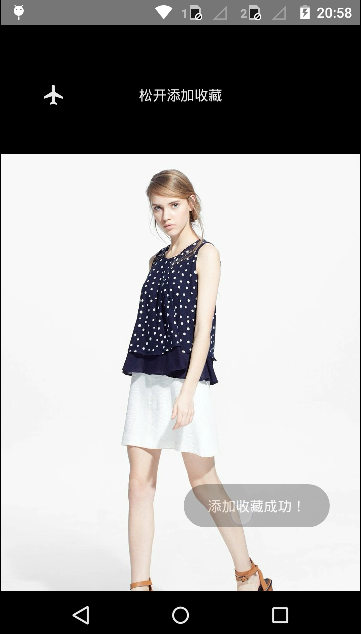
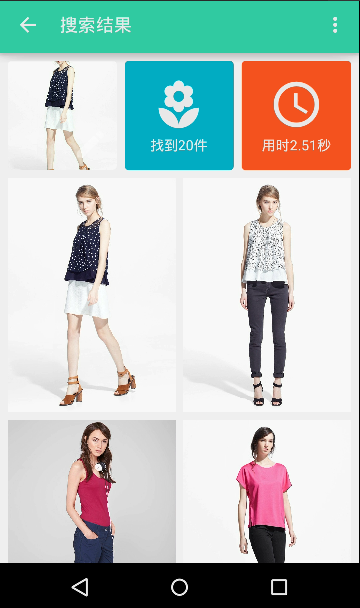
#### 通用模块

通用模块包括“欢迎页”、“用户协议”、“常见问题”、“告诉朋友”、“意见反馈”、“版本升级”、“给我们评分”、“开源许可”、“关于我们”。通用模块的存在使得APP功能更加完整。

* + 1. **图像获取模块界面实现**



* + 1. **搜索结果展示模块**



选中一幅搜索的服饰照片，下拉进行快速收藏，如图7所示，上拉进行快速分享如图8所示。

图7 下拉快速收藏服饰照片

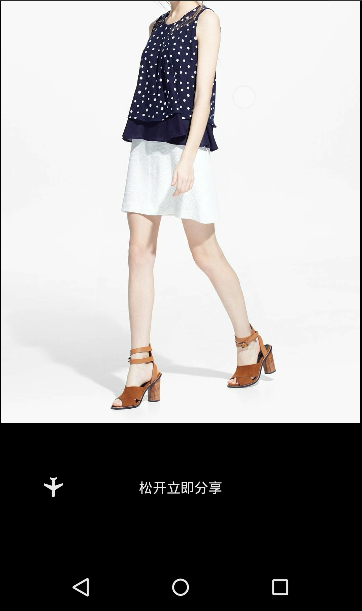


图8 上拉进行快速分享

双击选中图片可以快速查看服饰详情，如图9所示。单击选中的服饰照片，会出现结果详情标题栏，收藏、分享和查看服饰详情都可以通过结果详情右上角的设置按钮中进行操作，如图10所示。图9 双击快速查看服饰详情



图10 结果详情设置界面

点击图10右上角“保存到相册”服饰照片就保存在手机图库中，如图11所示。图11 将服饰照片保存到相册

点击图10右上角“去购买”跳转到购买网页，如图12所示。

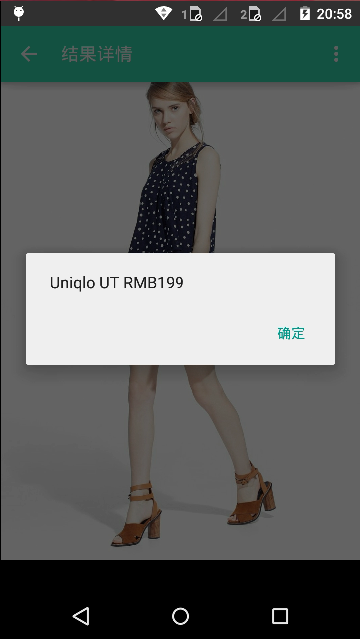


图12 服饰购买网页

点击图10右上角“查看服饰信息”进行服饰信息查看，如图13所示。

图13 查看服饰信息

* + 1. **其它辅助功能模块**

1. App主菜单如图16所示。
2. 
3. 图16 App主菜单
4. 点击主菜单上“搜索”进行拍照搜索或选图搜索，点击“我的收藏”查看收藏的服饰照片，如图17所示。
5. 点击主菜单上“用户”查看用户信息，如图18所示。
6. 
7. 图18 用户信息
8. 点击主菜单上的“设置”进入设置界面进行App端的设置，如图19所示。
9. 主菜单上的“历史记录”、“猜你喜欢”和“潮流前线”为待完善功能。

## 4.3 图像集及特征池的建立

服饰图像集的建立和服饰图像特征池的建立是离线的。

### 4.3.1 图像集的建立

本文系统采用的服饰图像来自国内主流电商网站，通过网络爬虫程序得到。网络爬虫不仅爬取服饰图像本身，也爬取服饰相关信息，比如品牌、价格、材质、货号。

网络爬虫程序使用Python实现，借助pyquery库。pyquery是类似jquery的Python库。pyquery允许我们在xml文件上做jquery查询，API和jquery非常相似。pyquery利用lxml实现快速的xml和html操作。相比正则表达式，借助pyquery实现网络爬虫更加简洁高效。

网络爬虫程序流程图



|  |  |
| --- | --- |
| 图像数量 | 36960 |
| 图像尺寸 | 1100 × 1390 |
| 图像集大小 | 5941MB |

### 4.3.2 特征池的建立

服饰图像特征池具体是指存储服饰图像特征的数据库，本文系统使用MySQL数据库。特征提取使用Python实现，借助OpenCV库。OpenCV通过BSD协议开源。OpenCV旨在高效计算，并专注于实时应用。底层基于C/C++，OpenCV可以利用多核进行并行计算。如果启用OpenCL，OpenCV可以借助各式各样的计算平台的硬件加速。

## 4.4 服务器端实现

服务器端主要功能是特征距离的计算与排序。

服务器用Python语言实现，并部署在Django架构之上。

### 4.4.1 Python简介

Python是一种面向对象、直译式的计算机程序语言，具有近二十年的发展历史。它包含了一组功能完备的标准库，能够轻松完成很多常见的任务。它的语法简单，与其它大多数程序设计语言使用大括号不一样，它使用缩进来定义语句块。

与Scheme、Ruby、Perl、Tcl等动态语言一样，Python具备垃圾回收功能，能够自动管理内存使用。它经常被当作脚本语言用于处理系统管理任务和网络程序编写，然而它也非常适合完成各种高级任务。Python虚拟机本身几乎可以在所有的作业系统中运行。使用一些诸如py2exe、PyPy、PyInstaller之类的工具可以将Python源代码转换成可以脱离Python解释器运行的程序。

Python的官方解释器是CPython，该解释器用C语言编写，是一个由社区驱动的自由软件，目前由Python软件基金会管理。

Python是完全面向对象的语言。函数、模块、数字、字符串都是对象。并且完全支持继承、重载、派生、多重继承，有益于增强源代码的复用性。Python支持重载运算符，因此Python也支持泛型设计。相对于Lisp这种传统的函数式编程语言，Python对函数式设计只提供了有限的支持。有两个标准库（functools, itertools）提供了与Haskell和Standard ML中类似的函数式程序设计工具。

虽然Python可能被粗略地分类为“脚本语言”（script language），但实际上一些大规模软件开发项目例如Zope、Mnet及BitTorrent，Google也广泛地使用它。Python的支持者较喜欢称它为一种高级动态编程语言，原因是“脚本语言”泛指仅作简单程序设计任务的语言，如shell script、VBScript等只能处理简单任务的编程语言，并不能与Python相提并论。

Python本身被设计为可扩充的。并非所有的特性和功能都集成到语言核心。Python提供了丰富的API和工具，以便程序员能够轻松地使用C、C++、Cython来编写扩充模块。Python编译器本身也可以被集成到其它需要脚本语言的程序内。因此，有很多人把Python作为一种“胶水语言”（glue language）使用。

Python支持命令式程序设计、面向对象程序设计、函数式编程、面向侧面的程序设计、泛型编程多种编程范式。

Python属于动态类型语言，动态类型语言是在运行期间检查数据的类型，不得不保持描述变量值的实际类型标记，程序在每次操作变量时，需要执行数据依赖分支，而静态类型语言相对于动态类型语言，在声明变量时已经指定了数据类型和表示方法，根据这一原理导致Python相对于C、Visual Basic等静态类型语言来说运行速度较慢。不过，根据二八定律，大多数程序对速度要求不高。在某些对运行速度要求很高的情况，Python设计师倾向于使用JIT技术，或者用使用C/C++语言改写这部分程序。目前可用的JIT技术是PyPy。

Python经常被用于Web开发。比如，通过mod\_wsgi模块，Apache可以运行用Python编写的Web程序。使用Python语言编写的Gunicorn作为Web服务器，也能够运行Python语言编写的Web程序。Python定义了WSGI标准应用接口来协调Http服务器与基于Python的Web程序之间的沟通。一些Web框架，如Django、Pyramid、TurboGears、Tornado、web2py、Zope、Flask等，可以让程序员轻松地开发和管理复杂的Web程序。

Python对于各种网络协议的支持很完善，因此经常被用于编写服务器软件、网络蠕虫。第三方库Twisted支持异步在线编写程序和多数标准的网络协议（包含客户端和服务器），并且提供了多种工具，被广泛用于编写高性能的服务器软件。另有gevent这个流行的第三方库，同样能够支持高性能高并发的网络开发。

Python采用动态类型系统。在编译的时候，Python不会检查对象是否拥有被调用的方法或者属性，而是直至运行时，才做出检查。所以操作对象时可能会抛出异常。不过，虽然Python采用动态类型系统，它同时也是强类型的。Python禁止没有明确定义的操作，比如数字加字符串。

Python拥有一个强大的标准库[9]。Python语言的核心只包含数字、字符串、列表、字典、文件等常见类型和函数，而由Python标准库提供了系统管理、网络通信、文本处理、数据库接口、图形系统、XML处理等额外的功能。

Python标准库的主要功能有：

文本处理，包含文本格式化、正则表达式匹配、文本差异计算与合并、Unicode支持，二进制数据处理等功能

文件处理，包含文件操作、创建临时文件、文件压缩与归档、操作配置文件等功能

操作系统功能，包含线程与进程支持、IO复用、日期与时间处理、调用系统函数、日志（logging）等功能

网络通信，包含网络套接字，SSL加密通信、异步网络通信等功能

网络协议，支持HTTP，FTP，SMTP，POP，IMAP，NNTP，XMLRPC等多种网络协议，并提供了编写网络服务器的框架

W3C格式支持，包含HTML，SGML，XML的处理。

其它功能，包括国际化支持、数学运算、HASH、Tkinter等

Python社区提供了大量的第三方模块，使用方式与标准库类似。它们的功能覆盖科学计算、Web开发、数据库接口、图形系统多个领域。第三方模块可以使用Python或者C语言编写。SWIG,SIP常用于将C语言编写的程序库转化为Python模块。Boost C++ Libraries包含了一组库，Boost.Python，使得以Python或C++编写的程序能互相调用。Python常被用做其他语言与工具之间的“胶水”语言。

### 4.4.2 Django简介

本文系统服务器采用Python Django架构。Django是基于Python的开源Web框架。它有部署快速、安全性高、伸缩性强三个主要特点。

Django以MTV（Model-Template-View）模型架构，其中Model为数据模型，与数据库操作相关；Template为模板系统，用于数据的格式化显示；View为控制器，负责调用Model、Template进行业务逻辑的处理，是MTV模型的调度中枢。

Django作为Web架构提供的功能有：

* Application的可插入（Plug-in）管理
* ORM（对象关系映射）

使用Django开发Web应用，只需用Python定义数据模型即可，数据库将由类映射得到，数据库操作也都有相应的API，不必自己写SQL语句（如果需要，也可以自己写SQL语句）。

* URL分发

在Django中，URL的分发通过正则表达式实现，简单高效。

* 模板系统

模板系统很好的分隔了数据与显示。Django的模板系统是可扩展的，能很好的满足个性化需求。

* Cache系统

Django为开发者提供了各种粒度的缓存策略，方便开发者快速开发。

Python应用程序的基本单元是Module（模块），一个Python应用程序由一系列的Module组成。Module是一个Python文件，包含Python对象定义和Python语句。每个Module（模块）有自己命名空间，并可以被别的Module（模块）导入并使用。Module（模块）实现了Python语言的解耦和复用。

服务端核心Module有Grabcut Module、颜色特征提取Module、纹理特征（Uniform LBP）提取Module、SIFT特征提取Module、视觉词Module、距离计算Module、排序Module。

### 4.4.3 GrabCut模块

GrabCut是为了提高搜索精确度而进行的预处理，利用openCV中的方法对图像集中的图像进行图像分割，将处理后的图像保存。后续的特征提取均在处理后的图像上进行。源图像和处理后的图像一一对应，返回给客户端的是源图像。关键代码如下：

for filename in os.listdir(workDir):

img = cv2.imread(os.path.join(workDir, filename))

if img is None:

continue

cv2.grabCut(img, mask, rect, bgdModel, fgdModel, 1, cv2.GC\_INIT\_WITH\_RECT)

mask2 = np.where((mask == 2) | (mask == 0), 0, 1).astype('uint8')

img = img \* mask2[:, :, np.newaxis]

cv2.imwrite(os.path.join(destDir, filename), img)



1. 系**统测试与实验**

测试环境

|  |  |  |  |
| --- | --- | --- | --- |
| Android手机型号 | Android系统版本 | 分辨率 | 测试结果 |
| 三星GalaxyS3 | Android 4.3 | 1280\*720 | 拍照、图库搜索较好地展示搜索结果 |
| 华为荣耀3C | Android 4.4 | 1280\*720 | 拍照、图库搜索较好地展示搜索结果 |
| 索尼Xperia C6603 | Android 4.1 | 1920\*1080 | 拍照、图库搜索较好地展示搜索结果 |
| 摩托罗拉  Moto G | Android 5.0 | 1280\*720 | 拍照、图库搜索较好地展示搜索结果 |

### 4.4.4 颜色特征提取模块

本文项目在进行颜色特征提取时使用了第三方库PIL和标准库colorsys。PIL：Python Imaging Library，已经是Python平台事实上的图像处理标准库了。PIL功能非常强大，但API却非常简单易用。colorsys的主要功能是进行颜色空间的转换，在本项目中被用于把数字图像的RGB颜色空间转换为HSV颜色空间。依据上文理论，对h、s、v分别进行量化（compress），量化后H[0,7]、S[0,2]、V[0,2]。按照量化级数，把各个颜色分量合成为一维特征矢量，公式为。一维特征矢量W的范围是[0,71]，统计其分布直方图，然后进行归一化，即可得到图像的72维颜色特征向量。关键代码如下：

wl = [0]\*72

for count, (r, g, b) in img.getcolors(img.size[0]\*img.size[1]):

h, s, v = colorsys.rgb\_to\_hsv(r/255.0, g/255.0, b/255.0)

h = int(h\*360)

H, S, V = compress\_hsv(h, s, v)

W = 9\*H + 3\*S + V

assert W < 72

wl[W] = wl[W] + count

wl = normalize(wl)

### 4.4.5 纹理特征提取模块

本文项目的纹理特征采用R=1、P=8的Uniform LBP特征。特征提取算法如下所示：

1. 定义
2. 读取图像并转化为灰度图
3. 遍历每个像素点，针对每个像素点执行3~x，遍历完成后跳到y继续执行
4. 将关键点邻域内得到八维二值向量pattern[]关键代码如下：

for x in range(0, len(img)):

for y in range(0, len(img[0])):

center = img[x,y]

pixels = []

for point in range(1, P + 1):

r = x + R \* math.cos(2 \* math.pi \* point / P)

c = y - R \* math.sin(2 \* math.pi \* point / P)

if r < 0 or c < 0:

pixels.append(0)

continue

if int(r) == r:

if int(c) != c:

c1 = int(c)

c2 = math.ceil(c)

w1 = (c2 - c) / (c2 - c1)

w2 = (c - c1) / (c2 - c1)

pixels.append(int((w1 \* get\_pixel\_else\_0(img, int(r), int(c)) + \

w2 \* get\_pixel\_else\_0(img, int(r), math.ceil(c))) / (w1 + w2)))

else:

pixels.append(get\_pixel\_else\_0(img, int(r), int(c)))

elif int(c) == c:

r1 = int(r)

r2 = math.ceil(r)

w1 = (r2 - r) / (r2 - r1)

w2 = (r - r1) / (r2 - r1)

pixels.append((w1 \* get\_pixel\_else\_0(img, int(r), int(c)) + \

w2 \* get\_pixel\_else\_0(img, math.ceil(r), int(c))) / (w1 + w2))

else:

pixels.append(bilinear\_interpolation(r, c, img))

values = thresholded(center, pixels)

variations = find\_variations(values)

transformed\_img.itemset((x,y), 255)

if variations <= 2:

res = 0

variating\_blocks += 1

for a in range(0, len(values)):

res += values[a] \* 2 \*\* a

transformed\_img.itemset((x,y), res)

pixel\_values.add(res)

else:

unassigned.append((x,y))

unassigned\_value = len(pixel\_values)

pixel\_values = sorted(pixel\_values)

no\_of\_pixel\_values = len(pixel\_values)

trans\_p1\_u2 = {}

for p in range(0, len(pixel\_values)):

trans\_p1\_u2[pixel\_values[p]] = p

### 4.4.6 SIFT特征提取模块

本文项目中的SIFT特征提取使用openCV库中方法。关键代码如下：

import cv2

...

img = cv2.imread(img\_path)

sift = cv2.SIFT()

kp, des = sift.detectAndCompute(img, None)



### 4.4.7 视觉词模块

本文首先对图像集中的所有图片提取SIFT特征并保存。然后利用KMeans算法对所有SIFT特征进行聚类。聚类操作依赖于openCV中的方法。满足迭代100次或者距离小于10时退出聚类操作。关键代码如下：

k = 100

attempts = 100

eps = 10.0

criteria = (cv2.TERM\_CRITERIA\_EPS + cv2.TERM\_CRITERIA\_MAX\_ITER, attempts, eps)

ret, label, center = cv2.kmeans(des, k, criteria, attempts, cv2.KMEANS\_RANDOM\_CENTERS)

聚类中心点即为视觉词。然后统计图像中特征点label的分布得到直方图，即可实现用视觉词表示图像的目的，归一化即可作为图像的SIFT特征向量。关键代码如下：

def getVectorFromLabelAndKpCount(file, label):

res = []

s = open(file, 'r').read()

kpCounts = json.loads(s)

base = 0

for kpCount in kpCounts:

vector = [0] \* 100

for i in xrange(kpCount):

labelIndex = base + i

belongClass = label[labelIndex]

vector[belongClass] = vector[belongClass]+1

res.append(vector)

base = base + kpCount

return res

### 4.4.8 排序模块

本文项目中的排序问题是典型的Top K问题，给定N（N>>10000）个特征间距离，求出其中前K个最小的距离。由于输入为大量数据，而只需要求得前K个最小值，因此对整个输入数据进行保存并排序是不必要的。

本文项目利用大顶堆数据结构来处理该问题。大顶堆如图所示，每个非叶子节点的数值，一定不小于其孩子节点的数值。这样可以使用含有K个节点的大顶堆来保存K个目前的最小值（当前根节点是其中的最大值）。每次有数据输入时，先于根节点比较，如果不小于根节点，则舍弃该数据；如果小于根节点，用该数据替代根节点，并进行大顶堆调整。数据输入完时，大顶堆保存的即是N个数据中前K个最小值。然后使用快速排序（qsort）对这K个数据进行排序得到从小到大有序的前K个最小距离。关键代码如下：

heap = images[:k]

heapq.heapify(heap)

for i in xrange(k, len(images)):

# if the dist is small than the current biggest dist, replace the biggest dist with this one

if images[i].dist < heap[0].dist:

biggest = heapq.heapreplace(heap, images[i])

top\_k\_images = sorted(heap, key=lambda x:x.dist)

# 致 谢

值此论文完稿之际，感激之情油然而生。

本文是在导师赵春霞教授的悉心指导下完成的，论文的完成过程中倾注了导师的谆谆教诲，在此向赵老师表示衷心的感谢。同时，我要感谢我的母校南京理工大学，感谢给我授课的各位老师，他们的努力，让我学到了更多的知识，使得我今天能够完成这篇论文！

此外我还要感谢和我一起走完研究生生活的同窗好友们，因为有你们，我的研究生生活才过得这么丰富多彩，这么充实。

最后，还要特别感谢我的父亲、母亲对我学习的全力支持！他们期望和关心是我人生永远向前的动力！

# 参考文献

1. Psaraftis H Interactive graph cuts for optimal boundary&region segmentation of objects in ND images Computer Vision 14 2001
2. Michael R.Swihart, Jason D, Papastavrou Exploiting Smart-Phone USB Connectivity For Fun And Profit European Journal of Operational Research 114 2010 447-464
3. Teodorovic D, Pavkovic G Privilege escalation attacks on Android Transportation Planning and Technology 16 2011 261-273
4. Minkoff A A Stealthy and Context-Aware Sound Trojan for Smartphones Operations Research 8 2010 1279-1295
5. 曹健 图像目标的表示与识别 机械工业出版社 2011
6. 吴小珍，李表奎 Android 手机的轻量级访问控制 计算机应用研究 5 2014 176-178
7. 郭耀煌，谢秉磊 Linux 内核完全注释 机械工业出版社 4 2004 114-115
8. 陈壁峰，陆昊娟，黄樟灿 Linux 内核的分析及应用 西安邮电学院学报 3 2002 46-48
9. 蒋腾飞 深入理解 Linux 内核 东南大学出版社 2013
10. 郑海虹 一种基于图的交互式目标分割算法 计算机工程与应用 4 2006
11. 李晓东，王东 人体图像中周边物品检测分类技术研究 南京邮电大学 1 2014 172-175
12. 李军，郭耀煌 面向购物搜索的目标提取算法研究及系统实现 西南交通大学 2001
13. 常朝稳，徐江科 终端行为可信评估及其访问控制方法研究 小型微型计算机系统 3 2014 493-499
14. 兆凤久 交互式服饰图像检索研究及系统实现 西南交通大学 2 2013 1
15. 章权 云计算及安全分析 华南理工大学学报 8 2011 109-112
16. Van Hemert J I, La Poutre J A Parallel Problem Solvers Nature VIII 2004 690-699
17. Parfaits H Compressed histogram of gradients a low bit rate feature descfiptor Transportation Science 17 1983 351-360
18. 李宁，邹彤，孙德宝 基于内容的图像检索综述 基于内容的图像检索综述 4 1999 130-135
19. Bodin L D, Sexton T R Distinctive image features from scale—invariant keypoints Working Paper MS/S 1982
20. 陆琳 基于文本的与基于内容的图像检索技术比较研究 南京航空航天大学 2009
21. 郭建红 基于内容图像检索中的对象提取与检索算法研究 北京交通大学 2013
22. THANGIAH S, NYGARDK, JUELL P G Towards optimal bag—of-features for object categorization and semantic video retrieval Proceedings of the seventh Conference on Artificial Intelligence Applications 1991 322-325
23. JOE L, ROGER L Query by image and video content: The QBIC system Proceedings of Fifth International Conference on Genetic Algorithms 1993 452-459
24. Martin Desrochers, Jacques Desrosiers, Marius Solomon Towards optimal bag—of-features for object categorization and semantic video retrieval Operations Research 40(2) 1992 342-354
25. Diaz A Through the Google goggles:Sociopolitical bias in search engine design WebSearch.SpringerBerlinHeidelberg 40 1992 342-355
26. Van der Bruggen, Lenstra L J , Schuur P C Variable-depth search for the single vehicle pickup and delivery problem with time windows Transporation Science 27 1993 298-311
27. 霍住震，张磊 基于 CIFS 协议的存储加密代理设计与实现 第 24 次全国计算机安全学术交流会 5 2009 64-66
28. 叶耀华，朱晓梅，陈霖 理解 Android 上的安全性 ：利用沙箱、应用程序签名和权限增强应用程序安全性 系统工程理论与实践 3 2009 110-112
29. 施朝春，王旭 隐蔽信道研究 计算机工程与应用 34 2009 21-24
30. 马华伟 用 Android 开发手机应用 程序员 2009
31. R. Feris and L. Davis. Image ranking and retrieval based on multi-attribute queries. In CVPR, 2011. 6
32. Z. L. Guangcan Liu and Y. Y. Robust subspace segmentation by low-rank rep- resentation. In ICML, 2010. 6
33. B. Hasan and D. Hogg. Segmentation using deformable spatial priors with application to clothing. In BMVC, 2010. 2
34. C. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In CVPR, 2009. 3
35. Z. Lin, M. Chen, L. Wu, and Y. Ma. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. Arxiv preprint, 2010. 6
36. S. Pan and Q. Yang. A survey on transfer learning. TKDE, 2010. 3
37. Z. Song, M. Wang, X. Hua, and S. Yan. Predicting occupation via human clothing and contexts. In ICCV, 2011. 3, 5
38. N. Wang and H. Ai. Who blocks who: Simultaneous clothing segmentation forgrouping images. In ICCV, 2011. 2
39. X. Wang and T. Zhang. Clothes search in consumer photos via color matching and attribute learning. In ACM MM, 2011. 3
40. J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma. Robust face recognition via sparse representation. TPAMI, 2009. 5
41. M. Yang and K. Yu. Real-time clothing recognition in surveillance videos. In ICIP, 2011. 2, 3
42. Y. Yang and D. Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In CVPR, 2011. 2, 3, 4