

毕业设计

基于视觉识别的智能小车

手势控制系统设计

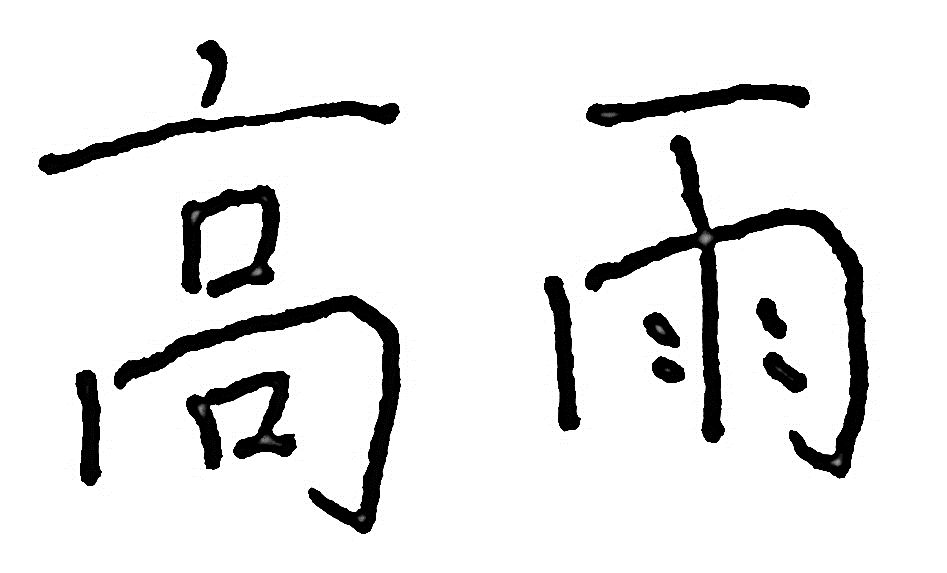
|  |  |
| --- | --- |
| 姓名 | 高雨 |
| 学号 | 182171108 |
| 学院 | 机械工程学院 |
| 专业 | 车辆工程 |
| 指导教师 | 孙永道（教授） |

二〇二二 年 五 月 二十七 日

学位论文原创性声明

本人所提交的学位论文《基于视觉识别的智能小车手势控制系统设计》，是在导师的指导下，独立进行研究工作所取得的原创性成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的研究成果。对本文的研究做出重要贡献的个人和集体，均已在文中标明。

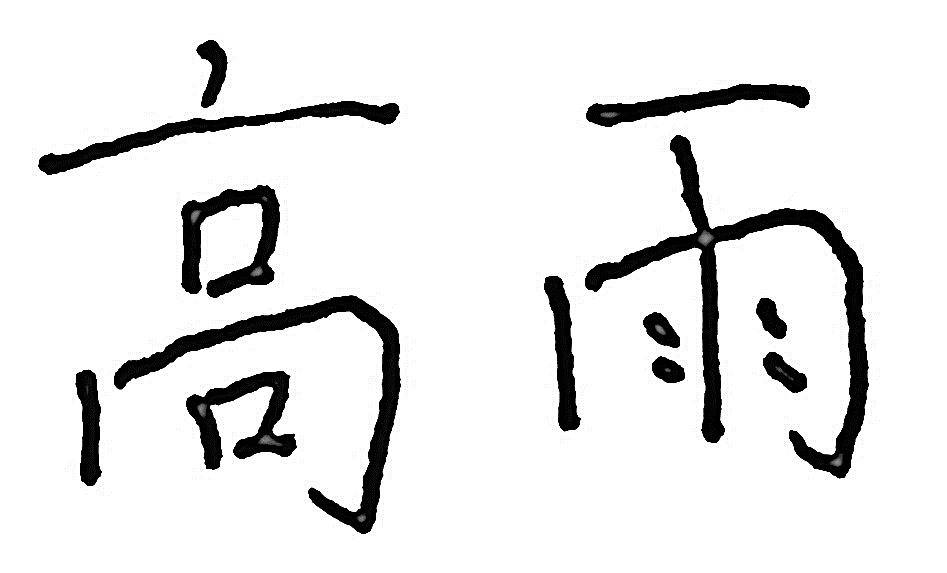
本声明的法律后果由本人承担。

论文作者（签名）： 指导教师确认（签名）：

2022年5月20日 2022年5月20日

学位论文版权使用授权书

本学位论文作者完全了解河北科技大学有权保留并向国家有关部门或机构送交学位论文的复印件和磁盘，允许论文被查阅和借阅。本人授权河北科技大学可以将学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或其它复制手段保存、汇编学位论文。

论文作者（签名）： 指导教师（签名）：

2022年5月20日 2022年5月20日

摘 要

手势控制小车，可以认为是智能小车人机交互的新发展趋势。手势控制操作简单，体验性好。随着物联网技术的不断发展，许多应用场景需求对传统人机交互提出了更智能、更便利的交互方式，智能小车控制交互体验也期望更智能交互。

本论文以手势识别为主要研究内容，通过加装在智能小车上的树莓派实现对智能小车的手势控制。本文首先设计了用于手势识别控制智能小车的基本硬件并进行组装；再利用Python编程语言对摄像头采集的实时手部图像，用基于肤色分割和轮廓提取方法进行了广泛深入的分析，包括图像滤波、图像形态学处理、颜色空间转换、轮廓等；接着对图像分析的结果利用。算法过程首先提取手部的关键特征，进而识别出具体的手势动作，最后，通过驱动电机改变小车的运行状态。通过实验验证，本文设计的手势识别控制的准确性比较高，有一定的实用价值。

关键词：手势识别；图像处理；树莓派

ABSTRACT

Gesture-controlled cars can be considered as a new development trend of human-computer interaction for smart cars. Gesture control is easy to operate and has a good experience. With the continuous development of Internet of Things technology, many application scenarios require more intelligent and convenient interaction methods for traditional human-computer interaction, and intelligent car control interaction experience also expects more intelligent interaction.

In this paper, gesture recognition is the main research content, and the gesture control of the smart car is realized through the Raspberry Pi installed on the smart car. This paper first designs and assembles the basic hardware for gesture recognition and control of the smart car; then uses the Python programming language to conduct extensive and in-depth analysis of the real-time hand images collected by the camera, using skin color segmentation and contour extraction methods, including image filtering. , image morphological processing, color space conversion, contours, etc.; then the results of the image analysis are utilized. The algorithm process first extracts the key features of the hand, and then recognizes the specific gestures, and finally changes the running state of the car by driving the motor. It is verified by experiments that the gesture recognition control designed in this paper has high accuracy and has certain practical value.

**Key words:** gesture recognition; image processing; raspberry Pi

目录

[摘 要 I](#_Toc104846877)

[1 绪论 1](#_Toc104846878)

[1.1 研究意义 1](#_Toc104846879)

[1.2 国内外研究现状 1](#_Toc104846880)

[1.2.1 人机交互发展概述 1](#_Toc104846881)

[1.2.2 手势识别研究现状 1](#_Toc104846882)

[1.3 论文主要目标 2](#_Toc104846883)

[2 视觉图像识别的关键技术 4](#_Toc104846884)

[2.1 图像处理引言（numpy） 4](#_Toc104846885)

[2.1.1 图像形成 4](#_Toc104846886)

[2.1.2 图像通道 4](#_Toc104846887)

[2.1.3 图像卷积运算 5](#_Toc104846888)

[2.2 预处理 6](#_Toc104846889)

[2.2.1 四种滤波 6](#_Toc104846890)

[2.2.2 图像形态学处理 9](#_Toc104846891)

[2.2.3 小结 10](#_Toc104846892)

[2.3 二值化 12](#_Toc104846893)

[2.3.1 灰度阈值分割 12](#_Toc104846894)

[2.3.2 HSV阈值颜色分割 15](#_Toc104846895)

[2.4 手部轮廓检测 16](#_Toc104846896)

[2.4.1 检测算法 17](#_Toc104846897)

[2.4.2 轮廓提取 19](#_Toc104846898)

[2.4.3 轮廓特征 20](#_Toc104846899)

[2.5 手势识别的实现 24](#_Toc104846900)

[2.5.1 数学原理 24](#_Toc104846901)

[2.5.2 识别的代码与注释 25](#_Toc104846902)

[2.6 小结 25](#_Toc104846903)

[3 智能车硬件设备驱动 26](#_Toc104846904)

[3.1 金属齿轮直流电机 26](#_Toc104846905)

[3.2 TB6612模块说明 26](#_Toc104846906)

[3.2.1 驱动模块原理 26](#_Toc104846907)

[3.2.2 电机驱动电路 27](#_Toc104846908)

[3.3 树莓派驱动电机程序设计 28](#_Toc104846909)

[3.3.1 树莓派GPIO引脚 28](#_Toc104846910)

[3.3.2 电机驱动代码 30](#_Toc104846911)

[4 总结与展望 33](#_Toc104846912)

[4.1 工作总结 33](#_Toc104846913)

[4.2 展望 33](#_Toc104846914)

[参考文献 34](#_Toc104846915)

[致谢 35](#_Toc104846916)

[附录 36](#_Toc104846917)

1. 绪论
   1. 研究意义

随着汽车与电子、计算机多领域融合，汽车以不再只是运载工具，大数据、智能传感器等技术的发展，使汽车正在与周围的物理环境及虚拟环境互联服务，车与车载设施的交互方式从按键到触摸屏，今天的语音控制使人与车的交互更加智能，而“隔空手控”更是科技感爆棚。以奥迪的MMI人机交互系统来说，这套系统可以连接蓝牙，插卡4G网络，有3D影像等等高科技，可以看出按照智能手机的蓝本，汽车已经越来越智能，随着计算机视觉的发展相信车载人机交互会迎来新的变革，取而代之的会是手势控制这种更高级的人机交互方式。

* 1. 国内外研究现状
     1. 人机交互发展概述

人机交互是指人与智能设备的沟通方式，能实现对设备的自然控制是人类对进入赛博朋克的追求。最初图形用户界面搭配鼠标或者按键极大的提高了人机交互的可操作性。进入智能人机交互阶段，网络通信和机器学习愈发智能，人与智能设备的互动沟通也愈发便利。现在人们可以通过语言、手势来完成人机交互。与语音识别，触摸屏识别相比，手势识别有更大空间。手势是人类通用的语言，它简单、直接，世界上的文明中手势有许多相似之处，比如，四指弯曲大拇指伸直表示很棒[1]。

* + 1. 手势识别研究现状

手势识别的实现方法大体可以分为两类，传感器式的数据手套识别和非接触式的视觉识别。两者基本热衷于锁定手部关节点。

通过佩戴带有传感器的数据手套，从传感器角度可以分为利用光学、应变感应及惯性三大类。光学数据手套要在关节点布置红外反光小球，由外置的多个红外摄像头捕捉定位，不难发现，一旦挡住了关节上的红外定位小球手部动作信息就会丢失，并且捕捉反射的红外光线需要分辨率极高的相机。应变感应器数据手套，每只手需要18或22个薄金属电阻片横/纵向交叉分布，利用手指弯曲引起的电阻变化，测量出的模拟信号由微处理器转换成数字信号，精度较高流畅稳定好，但价格昂贵。CyberGlove公司致力于生产高端VR数据手套，专注捕捉手部运动，哈工大高文教授和吴江琴教授用CyberGlove18型号的数据手套，结合人脸，手语等特征，采用DGMM（Dynamic Gaussian Mixture Model[动态高斯混合模型]）方法作为手势识别系统，将手语信号模拟化。传感器的发展在数据手套行业有着重要作用。康奈尔大学的研究人员日前发明了一种结合低成本发光二极管和染料的光纤传感器，名为“Stretchable Distributed Fiber-Optic Sensors（可伸缩分布式光纤传感器）”，能够检测诸如压力、弯曲和应变等变形，并精确它们的位置和大小。惯性数据手套成本低也没有遮挡问题，基本开箱即用。斯洛伐克布拉迪斯拉发科美纽斯大学的研究人员开发的 WaveGlove，就是在五个手指都连接上惯性传感器，用于测量加速度和角速度，使用的组件成本低可广泛使用。

非接触式手势识别可以通过微型雷达、颜色标记及视觉深度实现。三维手势识别包含深度信息，与二维手势识别技术相比要得到深度信息，目前有3种三维成像硬件原理。微软研究出Kinect体感游戏，该设备被作为XBOX的外围设备，用于实现游戏控制。Kinect一代依靠构造光技术，一个激光投射器，在穿过光栅成像时依据物体远近会得到不同位移，进而用算法计算出物体位置还原整个三维空间。一种基于TOF摄像模组的防疲劳驾驶方法[发明专利]，深度传感镜头TOF（Time of Flight，飞行时间）。这种技术的基本原理是依靠一个发光元件，发光元件发出的光子在碰到物体表面后会反射回来。根据光子飞行时间进而可以推算出光子飞行的距离，也就得到了物体的深度信息，这不需要任何计算机视觉方面的计算。美国Leap公司，推出可以在PC及Mac上使用的体感控制器Leap Motion，它以超过每秒 200 帧的速度追踪人的手部移动，双手犹如魔法棒。利用的多角成像（Multi-camera）技术，是根据几何原理来计算深度信息的。多角成像不需要任何额外的特殊设备，完全依赖计算机视觉算法来匹配两张图片里的相同目标。相比于结构光或者光飞时间这两种技术的成本高、功耗大的缺点，多角成像能提供“物美价廉”的三维手势识别效果。

Alex Krizhevsky等人凭借深度卷积神经网络Alex Net在Image Net大赛夺冠，随后深度学习研究领域得到广泛关注[2]。清华大学祝远新通过肤色信息与运动手势信息结合，成功在复杂环境下分割出手势，通过对获取图像的每一帧进行运动信息分割，结合人手的肤色与轮廓等信息进行识别。上海大学段洪伟用LS-SVM算法完成静态手势识别，并基于HMM模型实现了动态手势的识别。印度研究者meenaskshipanwar在视觉手势识别的基础上提出了一种基于结构特征的手势识别算法，通过背景去除、方向检测、拇指检测和手指数量检测，来最终识别手势。

* 1. 论文主要目标

首先，要能够分辨出手部，把图像处理的尽可能简单，减少树莓派的运算压力，也可以提高树莓派的运算速度。其次，找到合适的算法，利用数字图像信息识别手势，最后，完成手势识别与电机驱动的联动。

2. 视觉图像识别的关键技术
   1. 图像处理引言（numpy）
      1. 图像形成

数字图像的RGB颜色模式。在计算机中将三原色红、绿、蓝按照亮度分为0~255共256个等级，0表示亮度0%，255表示亮度100%，所以可以表示种颜色，255的含义是在计算机视觉领域缓存一个像素占8位，2^8=256，类型是uint8。

在OpenCV中，彩色图有三个通道R:G:B:，每一个像素点保存为一个三维数组。如图1所示，分别以R、G、B为轴建立来空间直角坐标系，千千万万的颜色就可以用三维数组表示了。图2左侧展示了坐标为的颜色，右侧展示了坐标为的颜色，按照数值越小亮度越低的原则，即数值越小对结果越没有影响，可以看到如图2所示，当关闭蓝色通道时间，紫色变成酒红色。

* + 1. 图像通道

计算机对像素的各种操作就是对这些数组进行运算。图像在计算机中的存储方式是一个矩阵，计算机可以分离三个通道，如图3所示，展示了将原图分离成了红、绿、蓝之后的样子，计算机可以对每个矩阵运算进而对图像进行各种操作。

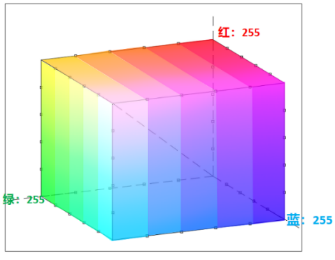


图1 RGB坐标图

我们可以认为原图，即彩色图是一个三维矩阵，而分离之后的是一个一维矩阵，这个一维矩阵的图像也叫做单通道图像。除了有红、绿、蓝这三色一维矩阵外，还有灰度图也是一个一维矩阵，它是由红、绿、蓝这三色一维矩阵按照一定权值计算得来的，在公式（2-5）中有展示[3]。接下来的滤波、图像形态学处理、灰度处理等操作都是作用于矩阵上。用计算机的眼光看图像，图像上的数值或深或浅，或者一马平川，或者波荡起伏。

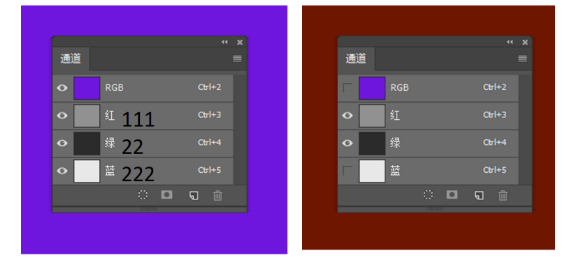


图2 颜色通道展示

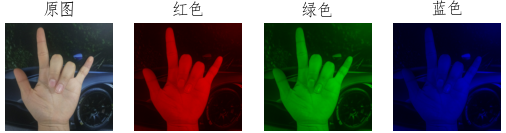


图3 220像素不同通道展示

这些图像预处理的目的就是要去除图像噪声，图像噪声是指图像中的不和谐像素，比如斑点和颗粒，这些额外的信息会干扰图像质量，经过图像滤波，（也称图像去噪）可以将这些突出了像素点去除，保留下有效的图像信息。这些不和谐的像素，往往是体现在图像灰度值的不连续变化。常见的图像滤波算法有均值滤波、高斯滤波、中值滤波、双边滤波等。

这一章将使用python语言来实现及优化，通过学习和应用OpenCV库中的相关函数来完成对图像的预处理、二值化以及轮廓检测。除了这些噪声，图还会受到光照的影响，在预处理时也应当注意融合颜色，不要让亮度过多的影响像素值的变化。

* + 1. 图像卷积运算

卷积的概念。在图像处理中，卷积操作是指使用一个卷积核对图像遍历。卷积核又称算子，是用来作图像处理的掩膜的，这个算子是一个由权值组成的矩阵。基本步骤就是在遍历像素过程中，将卷积核的中心像素值用核下原像素值与权值相乘并求和，得到新的像素值。可以想的当处理边缘像素时，卷积和有一部分超出了原图像，这就要是边缘效应，一般会补充像素值0或者复制边缘像素。卷积可以有两种结果，一种低通卷积，一种高通卷积。低通卷积是指的模糊、平滑图像这种操作，高通卷积则是对图像作锐化处理。

对应的概念还有低频信号和高频信号。高频信号就像突变点，比如图像边缘和噪点，这些像素点的值是不连续的，甚至差值很大。允许这些高频信号通过就会增强图像边缘，所以称为高通卷积。低通卷积则完全是相反的概念，只允许连续的图像通过，那么突变的噪点就会被摸出，进而实现对图像的模糊和平滑。另外就是灰度图（0-255）是对一个通道做卷积，彩色图是对三个通道分别卷积再相加，这其中的道理在上面已经解释过了。

* 1. 预处理
     1. 四种滤波

下面将详细讲述本次设计中使用到的四种滤波，介绍滤波的功能以及它对应的矩阵运算本质。

1. **均值滤波**

均值滤波的运算实质就是将某点像素，取以该点为中心的个像素点的平均值作为该点的新的像素值，而这个像素点称为这个均值滤波的卷积核。如图4所示，展示了使用的模板（卷积核）处理单通道图像的运算过程，其中左边为原图，右边为均值滤波的结果图，图上标注的数字为该点的灰度值。首先解释一般区域的卷积运算过程，也就是不临边的区域，计算核内平均值：，在计算机内部结果取整得88。最后是特殊的边框区域的卷积过程，这就需要先按离边框最近的像素值补全外圈边框，再计算其现在的核内平均值：，计算机内部取整得53。这里提到的补全外圈边框除了可以使用选择最近的像素点的像素值作为补充的值外，也有全部都补零的，或者随机补充的，还有的边框保持不变。



图4 均值滤波原理图

1. **中值滤波**

中值即中位数。用卷积核内像素值的中间值代替像素模板核心点的像素值，可以有效去除孤立斑点，尤其是0和255的像素点。适合去除椒盐噪声和斑点噪声。椒盐噪音形象的说椒是黑的，盐是白的，就是图像上随机出现的黑白像素[4]。如图5所示，取核内的像素值并按照大小排列，18 70 133 158 165 185 205 255，得到像素值排列位中的中间位置的像素值165，即为中值滤波的值。中值滤波的核必须是大于1的奇数。



图5 中值滤波原理图

在使用均值滤波时，往往图像会消除噪音的同时，将边界细节一同变得模糊。而中值滤波这种非线性的图像处理方法，就可以在去除噪声的同时保留下来重要的边缘细节信息。但是这种简单的局部卷积算法会使图片变得模糊，目前已经提出了可以保留边缘信息的平滑卷积算法。重要注意的是，如何选择卷积核的大小，卷积的的形状，卷积核的参数权值等。

总结一下，前面已经介绍了两种线性滤波，均值滤波只会使得图像整体变得模糊，对椒盐噪声没有作用，并且随着卷积核的逐渐增大，均值滤波会有明显的模糊现象，而中值滤波不会，中值滤波对孤立的噪点有非常好的消除效果。所以我认为，如果后期需要消除某些孤立的光电，使用中值滤波会有很好的效果。

1. **高斯滤波**

首先，高斯噪声是由拍摄时的光照引起的，噪点呈现出的概率密度是正态分布（高斯分布）。而高斯滤波与以上的滤波类似，都是对卷积核内像素值求加权和，只不过高斯滤波的权重是服从二维正态的。即靠进核心的权重最大，离核心越远权重越小，这称之为空间邻近度权值。其权重计算方法见公式(2-1)，标准差越大，权重分布越均匀，图像越模糊，需要注意的是其最终权值结果需要归一化。默认自动标准差的计算方法见公式(2-2)、公式(2-3)。

|  |  |
| --- | --- |
|  | 公式(2-) |
|  | 公式(2-) |
|  | 公式(2-) |

卷积核为的高斯加权模板：

卷积核为的高斯加权模板：

1. **双边滤波**

与前几种不同，双边滤波对边缘像素有不一样的处理，在对图像降噪的同时，对图像边缘较好的保留[5]。与上述几种滤波原理相同，双边滤波也是采用加权平均的方法，但处理图像的方法是两个高斯滤波的结合，双边滤波对高频率的波动信号起到平滑作用，同时保留大幅值的信号波动，进而实现对图像的边缘信息保留。一方面采用空间邻近度加权法，一方面是像素值相似度加权法。双边滤波的基本算法：在高斯滤波的权值基础上，将其优化为空间邻近度权值与像素相似度权值的乘积为最终权值。

|  |  |
| --- | --- |
|  | 公式(2-) |

为两像素点像素值之差的绝对值平方，当在像素值相差不大的平坦像素区域时，图像没有不和谐像素点，也没有强烈边缘信息差，临界像素值的差相差较小， 也就接近于1，相当于对此区域进行高斯滤波，达到了对图像的平滑效果。当邻域内图像像素值差距大变化剧烈时，就比如在边缘上的图像信息，接近于0，相乘后最终整个权值几乎为0，不会影响结果，从而保持了原始图像的边缘的细节信息。这样就既能平滑图片又能保持图像边缘。高斯滤波说到底是一种低通滤波，往往除了去噪也会使图像边缘信息被模糊掉，相比之下，双边滤波的图像处理更好。

某一个像素点的滤波结果的值由与该点领域内的像素点的像素值决定，领域内的像素值对该点的权值影响取决于两个像素点的距离和相似度。所以与其他滤波函数相比，双边滤波不仅仅要设置核的大小，还包括颜色空间的滤波标准值、空间坐标的滤波标准值。

总结一下就是，Sigmacolor越大，像素领域内就会有越多的颜色被混合到一起，会形成较大的颜色混合区域。Sigmaspace越大，说明距离较远的像素值对该点会由越大的影响，这就使得像素足够相近的像素点获得相同颜色的区域越大。

在图6中，分别展示了原图、高斯滤波结果和双边滤波结果。这两个参数设置的越大，图像变化越明显，会有卡通的效果，这种结果对后面的图像信息分析是极好的，是新的图像拥有大面积的相同颜色，并且边缘线条得到了保留。



图6 高斯滤波与双边滤波结果对比

* + 1. 图像形态学处理

形态学图像处理主要应用在图像的预处理和后处理中，起到细化和修剪毛刺的作用，基本思路是使用某个结构元素，在原图上经过相关运算遍历，对目标区域的边缘理想化。

形态学处理是对阈值处理后的二进制图像(0为黑，1为白)进行的膨胀和腐蚀操作的运算，这是高级形态学处理的基础，还有开运算和闭运算等，是由这两种基本运算合成的。形态学的操作也是卷积的实现，这里先要说明的kernel核叫做结构元素（StructElement），有矩形rect、椭圆ellipse、十字形cross，可以使用cv2.getStructuringElement()生成结构元素，图7中左侧结构元素就是使用cv2.getStructuringElement()函数生成的的十字形结构元素，或者如果只是想得到矩形结构元素可以直接生成矩形矩阵结构来代替。结构元素越大，被腐蚀或被膨胀增加的区域越大。

具体的图像处理过程建立在二进制图像信息上，腐蚀运算是消除小于结构元素的噪声点，有使得边缘向内收缩的效果，像素区域会减小，相连的部分区域可能会断开。膨胀运算可以填补图像空洞，有使得边缘向外部扩张的效果，像素区域会增加，不相连的部分可能会连起来[6]。接下来将展示以图7中的右侧这个简单的二进制图像为原图，使用图7左侧结构元素分别做腐蚀运算和膨胀运算的遍历过程。

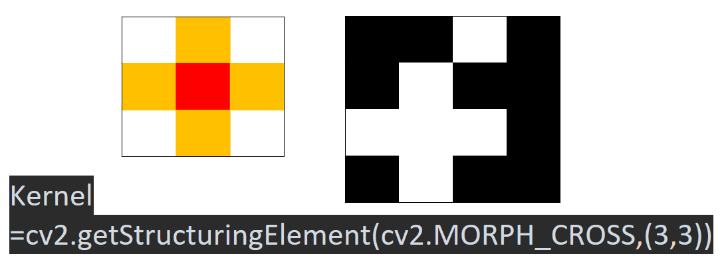


图7 形态学运算卷积核

在图8中，我们使用图7中左侧的结构元素遍历原图，红色是中心像素点，橙色代表二进制1，白色代表二进制0，原图中黑色代表二进制0，白色代表二进制1。我们只要记住结构元素与原图相“与”之后，腐蚀运算结果全为1保留像素，否则像素点置零；膨胀运算结果全为0时置零，否则保留像素。图8中省略了许多步，腐蚀和膨胀的最终结果分别展示在了图9中。

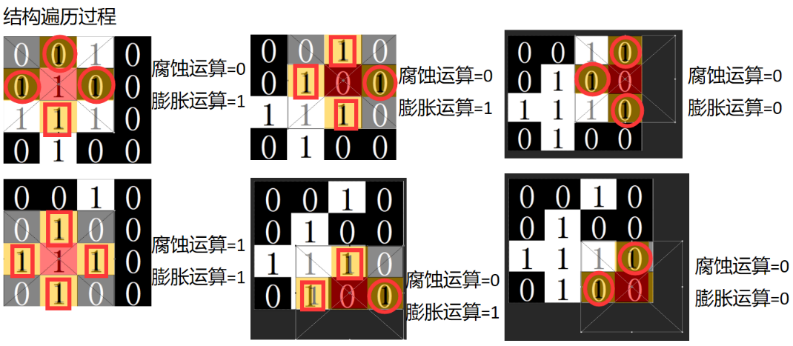


图8 形态学运算原理图

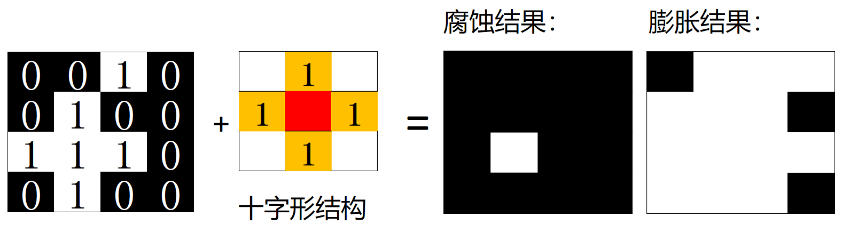


图9 形态学运算结果图

有了腐蚀和膨胀这两个运算基础，下面的就简单了。闭运算是先膨胀使白色部分扩张，再腐蚀运算，作用就是填充物体内细小的空洞，连接临近物体，平滑边界，同时不改变其面积。开运算反之，先腐蚀后膨胀，作用是去除孤立的噪点，消除小物体，平滑大物体边缘，同时不改变面积。下图图10，展示了将原图灰度化之后，使用阈值二值化之后，分别运行膨胀、腐蚀、开运算和闭运算的结果。我们可以看到在刚转化好的二进制图像中，有一些噪点，就是图10中红色圈出的部分。膨胀结果中白色被填满放大，腐蚀结果显得被图像分散支离；开运算去除了噪声，闭运算使手缝连接在一起。

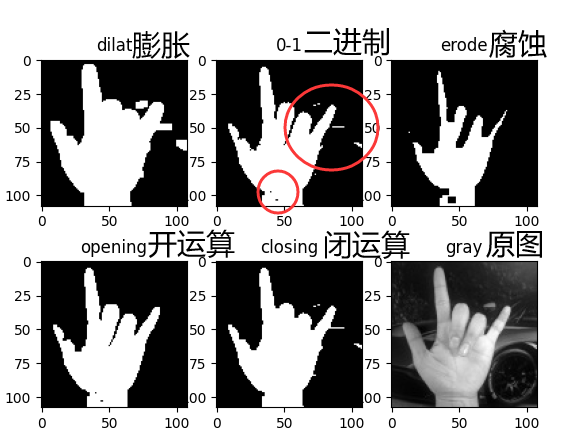
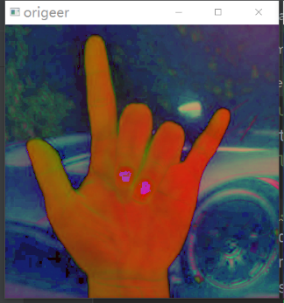


图10 220像素形态学处理结果图

* + 1. 小结

在均值滤波中随着卷积核大小的逐渐变大，图像会越来越模糊。如果设置卷积核为，那么将不会对图像有什么影响。分别对彩色图、HSV图和灰度图使用四种滤波。







在处理彩色图和HSV图时双边滤波效果最好，手部里面的颜色被模糊了，但手周围有明显的深色印记。在处理灰度图是，中值滤波对像素信息影响不大。从上面的结果仔细对比选择，最终本论文选择使用中值滤波对图像进行模糊处理，使图像达到一个肤色或者手部区域呈现一块色块区域的状态，这会大大提升图像分析的准确性。

* 1. 二值化

机器视觉包括预处理和物体检测两个步骤，而跨越两者的就是图像分割，最简单的图像分割就是二值化图像。上一节中，将手影显示在二进制图像中就是二值化，像素点的灰度值为0或255。二值化有两种方法，一种使用灰度图把大于某一灰度值的像素转化为255，进而以这一灰度值为界限，划分图像；另外一种是使用彩色图，设定肤色的hsv范围，从颜色角度分离出手与背景。下面将分两部分分别说明其原理。

* + 1. 灰度阈值分割

将一张灰度图转化成非黑即白的二值图像，那么如何转化成灰度图呢？在电脑中每个像素占的位不同可以看出位数越多能够表达的颜色越细腻。

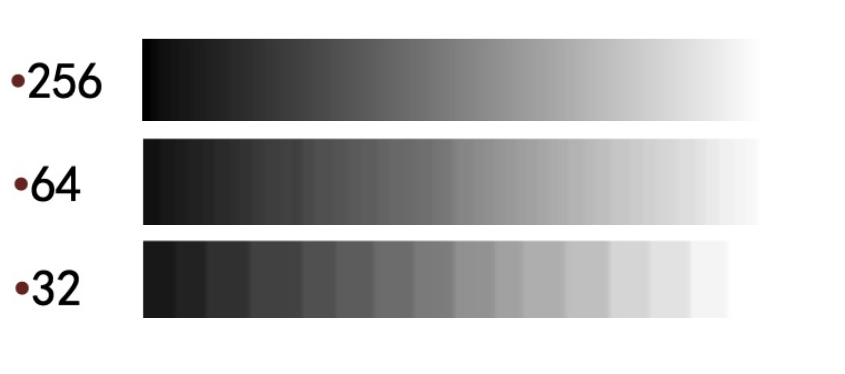


图11 灰度级图

我们在前面的引言中已经知道要表达一种颜色需要三个通道，它们是[R;G;B]，在图1中，我们可以想到要将一种颜色从一个三维空间映射到一维空间，过将像素坐标点作R=G=B直线的垂线，然后用0~255量化亮度值，得到对应的灰度值，其实映射到一维的方法有很多。但由于人眼对不同颜色光的波长的灵敏度不同，在计算灰度值时使用的公式是

|  |  |
| --- | --- |
|  | 公式(2-) |

我们把前面的图片压缩三十倍，得到一个的小图片，在图12中，左侧是RGB图像，右侧是其对应的灰度图。图13中分别展示了其三个通道的灰度值，灰度值越小，亮度越低，通道对彩色图的作用成分越少。图14是最终的灰度图的灰度值。图15展示了其对应的灰度值统计直方图，可以看到灰度值30左右的像素点最多，其次是灰度值为135左右的像素点，这其实就是图片的有效信息手部应为大于100灰度值的区域，背景则是较暗的区域，灰度值较小。



图12 14像素彩色图与灰色图

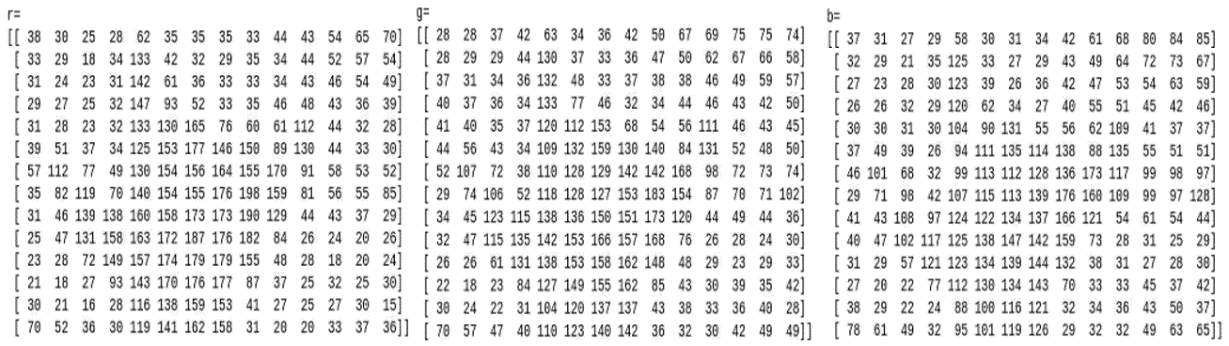


图13 14像素三通道灰度值

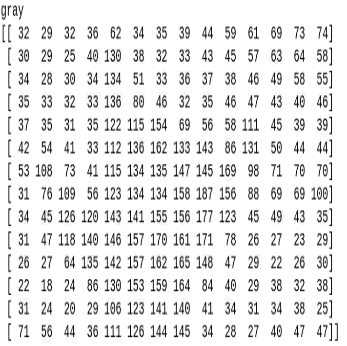


图14 14像素灰度图

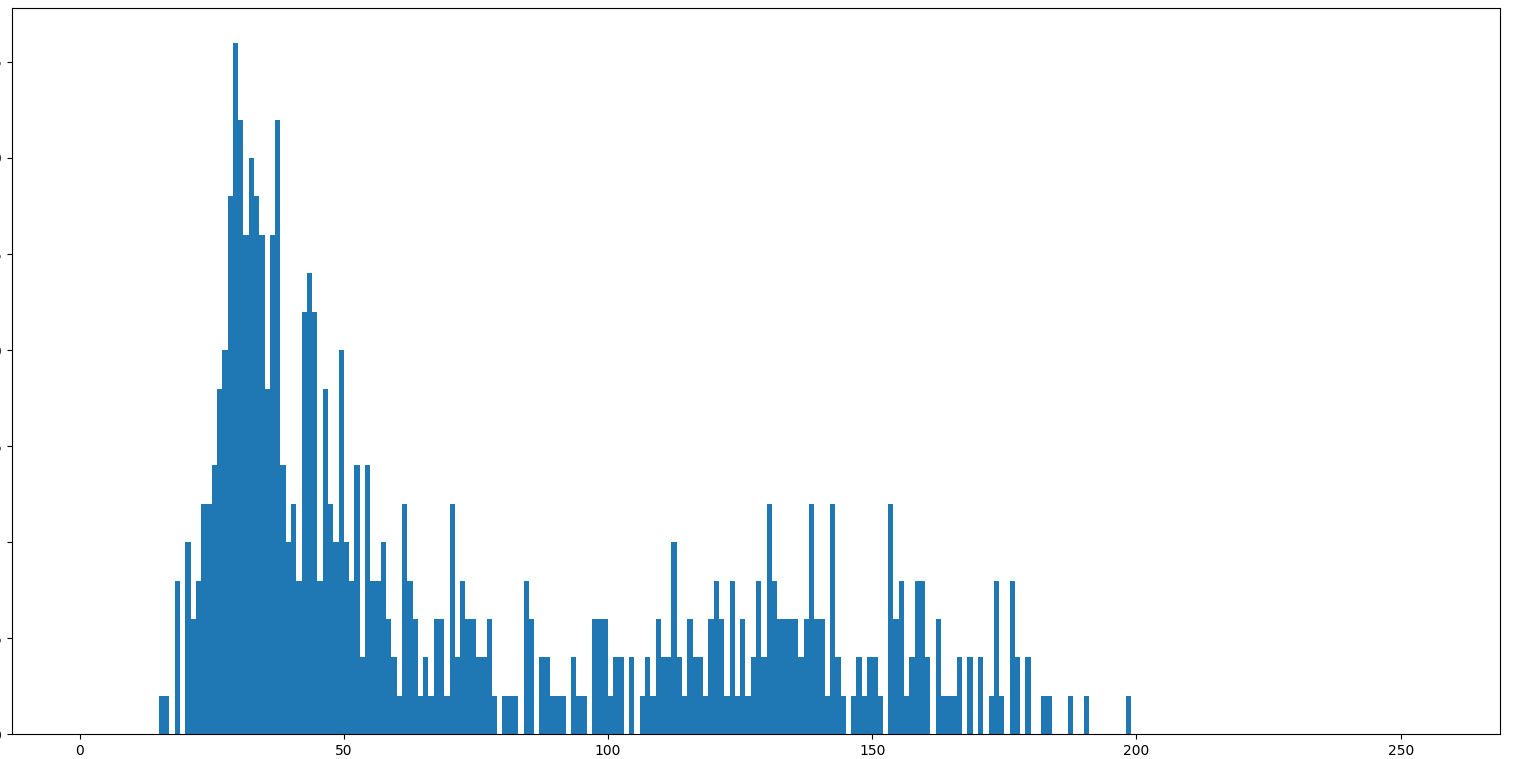


图15 14像素灰度直方图

我们就对这些灰度值开始做文章，我们选定某一数值，以此为节线将图片化为非黑即白的二值图像，使用dst=cv2.threshold(src,thresh,maxval,type)函数[5]。其有4个参数，src=原图，thresh=阈值，maxval=255（一般），type=模式。5种划分模式：

1. THRESH\_BINARY:
2. THRESH\_BINARY\_INV：
3. THRESH\_TOZERO：
4. THRESH\_TOZERO\_INV：
5. THRESH\_TRUNC：

原理在图16中展示，使用TRUNC模式，将大于100的变成100，其余不变。图17就是将图14中大于100的像素点全部置为255，将其他的置零的结果，也就是图18中展示的对图12的灰度图操作BINARY模式的结果。最后图19展示了对图6中的原图分别进行上述五种模式的不同结果。



图16 TRUNC模式操作原理图

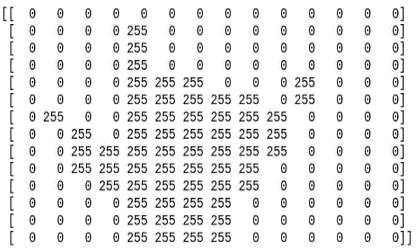


图17 14像素二值图

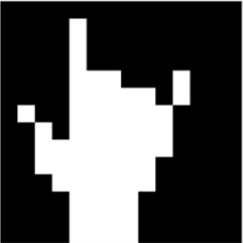


图18 14像素二值化结果图



图19 220像素灰度阈值图

* + 1. HSV阈值颜色分割

除了RGB空间，还有另外一种表达色彩的方式。在HSV空间中，H表示色彩，取值范围；S表示饱和度，取值范围，数值越大颜色越饱和；V表示亮度，取值范围，数值越大越亮。HSV空间中的颜色表达很直观，确定手的色彩H值在。

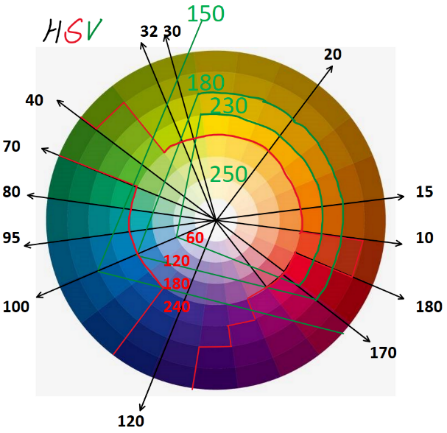


图20 HSV坐标图

我们设置手部的颜色范围，这个范围可能会使深红色和浅绿色也被计算机认为是手部，但不用担心一方面我们可以人为选择深色背景突出手部，另一方面我们还要对图像进行预处理和形态学处理，在最后一步之前尽量给计算机一张信息优质的图像可以提高计算机的识别。使用HSV颜色空间来识别手部对象，最重要的参数是H值。计算机的HSV图是如何得到的呢[7]？需要用的RGB颜色空间转换到HSV空间的公式：

|  |  |
| --- | --- |
|  | 公式(2-) |
|  | 公式(2-) |
|  | 公式(2-) |
|  | 公式(2-) |
|  | 公式(2-) |
|  | 公式(2-) |
|  | 公式(2-) |
|  | 公式(2-) |

之后像cv2.threshold()函数一样，我们要设置阈值，筛选出我们目标颜色区域，使用的是cv2.inRange()函数。cv2.inRange()函数有三个参数，sre、lower col、upper col，其中sre是原图，设置的lower\_col与upper\_col之间的像素值会变为255，其余为0。下图图21中，展示了设置参数值为时的手部分割图像。由于指甲呈现粉红色没有现实出来[5]。



图21 220像素肤色分割结果图

使用肤色来对手势进行分割，优点是对于手部的轮廓、角度、大小等不会受到太多干扰，转换到HSV空间后色彩参数H将成为主要阈值对象，并且允许更大的亮度和饱和度变化[1]。

* 1. 手部轮廓检测

图像边缘是图像最基本的特征，边缘是像素灰度值发生明显变化的地方，在图14中，可以看到手部灰度值明显高于其他地方，灰度值局部不连续。利用这一特征可以分割图像，边缘检测可以大幅减少数据量，只保留图像中重要的信息。我们已经清楚的感觉到，就是要利用这些数值之间的变化，利用数学来检测边缘，边缘检测算子可以分为一阶导数边缘算子、二阶导数边缘算子和其他边缘算子。一阶导数的边缘算子，通过卷积核在图像上遍历计算一阶导数，然后取适合的阈值提取图像边缘，这一类的代表算法是Sober算子和Scharr算子。二阶导数的边缘算子是依据二阶导数得零，这一类的代表算法是Laplacian算子，此类算子对噪声敏感。其他的算子有Canny检测算子，这一类不是通过微分算子检测的，是一种边缘检测最优化算子。前面我们提到卷积核，就是一个滤波模板，下面我们通常称之为检测算子，由不同的滤波模板衍生了不同的检测算子。

* + 1. 检测算法

1. **索贝尔(Sobel)检测算子**

检测每一个像素点的领域内的灰度变化情况，包括垂直方向检测和水平方向检查， Sobel算子是高斯平滑与微分操作的结合，基于一阶导数求边缘。求当前邻域内一阶导数计算公式

|  |  |
| --- | --- |
|  | 公式(2-) |

对图像点在水平和垂直方向求导，即取大小的卷积核，见公式(2-15)、公式(2-16)，得到水平变化垂直变化，结合得到最终结果，遍历统计最大值，得到图像边缘。一般公式(2-17)比较麻烦，可以简化计算直接使用公式(2-18)代替。当卷积核取时有明显误差，可使用Scharr算子计算，其结果更精确，它的卷积模板见公式(2-19)、公式(2-20)。在图22中，可以看到使用Scharr算子检测可以得到更细致的边缘，但为了简化图像，方便后期的进一步研究，我们可以使用Sobel算子检测。

|  |  |
| --- | --- |
|  | 公式(2-) |
|  | 公式(2-) |
|  | 公式(2-) |
|  | 公式(2-) |
|  | 公式(2-) |
|  | 公式(2-) |

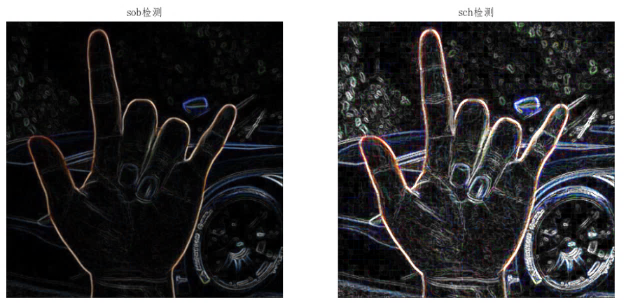


图22 220像素彩色图轮廓检测结果图对照

1. **Laplacian检测算子**

拉普拉斯算子利用二阶微分检测边缘。算法基本流程，举例使用的卷积核，如下所示：



我们观察这个神奇的卷积模板如果在图像中的灰度值较低的区域出现了一个较高的像素点，那么卷积结果会使这个较高灰度的像素点变得更高更亮，这就形成了拉普拉斯锐化。所有这无可避免也会增强图像噪点，所以使用该算子前要进行图像平滑。

1. **Canny边缘检测**

算法原理。第一步：使用高斯滤波去除噪声。第二步：图像梯度计算。利用Sobel算子计算水平和垂直方向导数和，得到边界梯度大小和方向，见公式(2-21)和公式(2-22)，如果是边缘，其梯度方向应与边缘垂直。梯度方向被归为四类：垂直，水平，两个对角线方向。第三步：非最大值抑制[1]。对每一像素检查，看这点的梯度是不是周围具有相同梯度方向的像素点的最大值，用以去除不构成边缘的无关像素点。用Sobel算子检测的边缘一般比较粗，非极大抑制就是起到瘦边的作用。第四步：滞后阈值。确定真正的边界，设定两个阈值minVal和maxVal，当梯度大于maxVal时被认为是边界，低于minVal时抛弃该点，介于阈值之间的，如果与确定边界相连则保留，否则抛弃。所以较小的minVal可以将间断的边缘连接起来，较大的maxVal可以检测出明显的边缘，抛弃更多无效信息。在图23中，A点梯度幅值大于maxVal，A点会被保留，认定为边界；B点低于设定的阈值，但由于它与A点相连，所有也保留下来；C点的梯度幅值已经低于maxVal，与A点不相连，所以该点被抛弃置零；D点低于minVal值，不是边界所以抛弃。图24中展示了参数minVal取9，maxVal取15时的检测结果[8]。

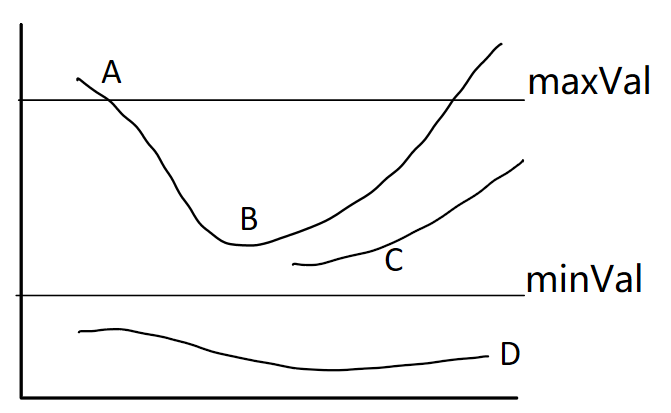


图23 canny滞后阈值

|  |  |
| --- | --- |
|  | 公式(2-) |
|  | 公式(2-) |

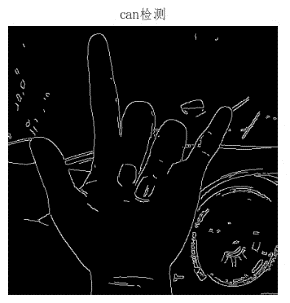


图24 220像素Canny边缘检测结果图

1. **基于形态学**

将二值图像膨胀后减去腐蚀结果，如下图25所示，使用cv2.morphologyEx(img,cv2.MORPH\_GRADIENT,kernel)函数。

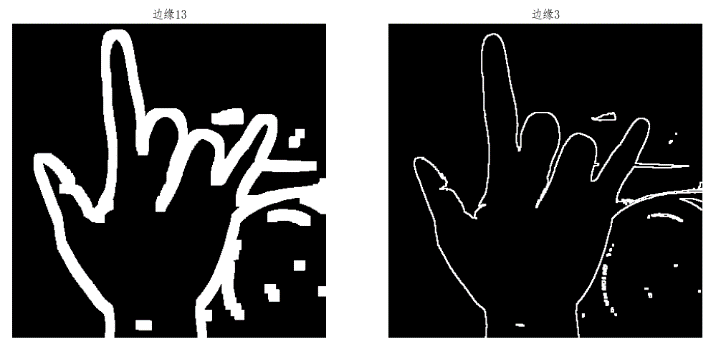


图25 220像素形态学轮廓检测结果图对照

* + 1. 轮廓提取

前面的所有工作下来，我们得到一张比较不错的二进制图像，也就是非黑即白的图像，下面我们需要知道连通的区域有哪些，连通的区域我们认为是一个物体。cv2.findContours()函数是轮廓提取函数，可以检索轮廓，检测的图像必须是二值图像。cv2返回2个值，cv3返回3个值。Countours返回找到的轮廓list，每个元素表示一组轮廓点集。hierarchy返回轮廓索引，每个轮廓有4个hierarchy元素,分别表示后一个轮廓，前一个轮廓，父轮廓，内嵌轮廓，当没有对应项时索引为负数。cv2.drawContours()函数用于绘制轮廓，输入参数包括索引以及轮廓样式[9]。对图18使用cv2.findContours()函数提取边缘得到下图图26，提取边缘得到轮廓点集，见图27。轮廓对应的索引号，见图28。

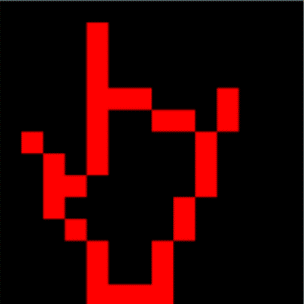


图26 14像素轮廓检测结构图

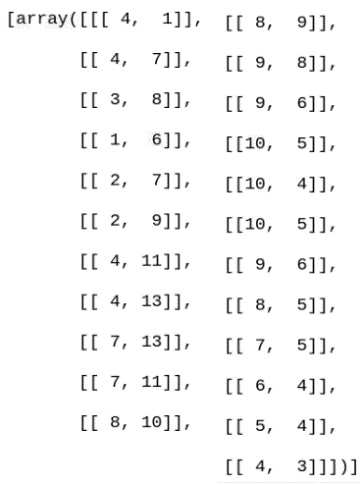


图27 轮廓的点坐标

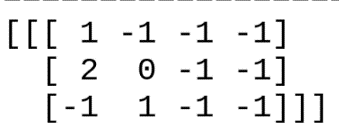


图28 轮廓索引号

* + 1. 轮廓特征

得到图像手部的轮廓以及各条轮廓的索引号，我们要对轮廓进行识别，那就要进一步的简化轮廓，在识别轮廓前需要进一步的了解轮廓的几何变化，比如长度、轮廓面积等，下面会介绍长度函数、面积函数、计数函数、近似多边形函数、凸包函数和凸性检测函数这六个函数。图29是检测到的手部轮廓的点集，图30是检测到的手部轮廓的闭合曲线图。



图29 轮廓检测点结果图

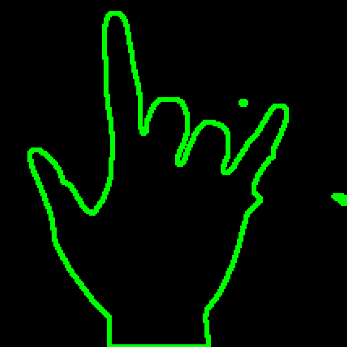


图30 轮廓检测线结果图

1. **轮廓数目函数**

使用函数len(contours)获取轮廓数目。

1. **面积函数**

使用函数cv2.contourArea(contours[i])获取轮廓面积。轮廓面积是指每个轮廓中所有的像素点的面积，单位是像素，函数返回值是double类型。

1. **长度函数**

使用函数cv2.arcLength(contours[i])获取轮廓长度。在图31中是图30中的三条轮廓的索引号、面积和长度。

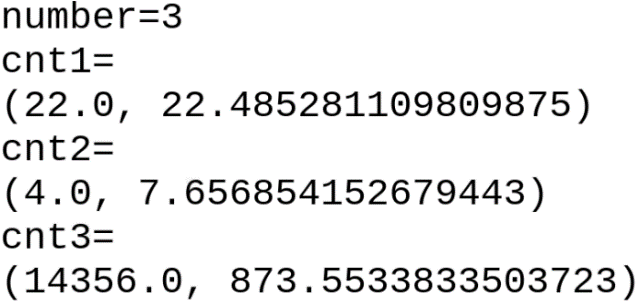


图31 三条轮廓的面积与长度

1. **轮廓线逼近函数**

OpenCV中对某一点集进行多边形逼近的函数-- cv2.approxPolyDP()函数，称为轮廓近似函数。在轮廓近似函数中影响多边形近似的重要参数是阈值d，其含义是指近似的折线到轮廓允许的最大距离值占轮廓线周长的百分比，可以通过这个参数控制逼近度。实现该函数的算法采用道格拉斯-普克算法（Douglas-Peucker）来实现，是将曲线近似表示为一系列点的过程，该算法以道格拉斯-普克算法和迭代终点拟合算法为名，该算法的原始类型分别由乌尔斯·拉默（Urs Ramer）于1972年以及大卫·道格拉斯（David Douglas）和托马斯·普克（Thomas Peucker）于1973年提出，出现后的数十年被其他学者不断完善。道格拉斯-普克算法原理：第一步，连接曲线首(A)尾(B)作曲线的弦。第二步，计算弦距离曲线上最远点(C)的距离d。第三步，比较与设置的阈值d的大小，不超过阈值认为这条曲线已经完成逼近。否则将曲线分为AC与BC两段，分别返回第一步重新逼近。第四步，直到所有曲线处理完毕，连接分段点为近似多边形。在图32中展示了示意图，经过三次近似逼近得到该图形的近似多边形，其中绿色的垂线段表示d。在图33中，分别显示设置d值为0.05和0.009的效果，意思就是越大，近似的越粗略[9]。

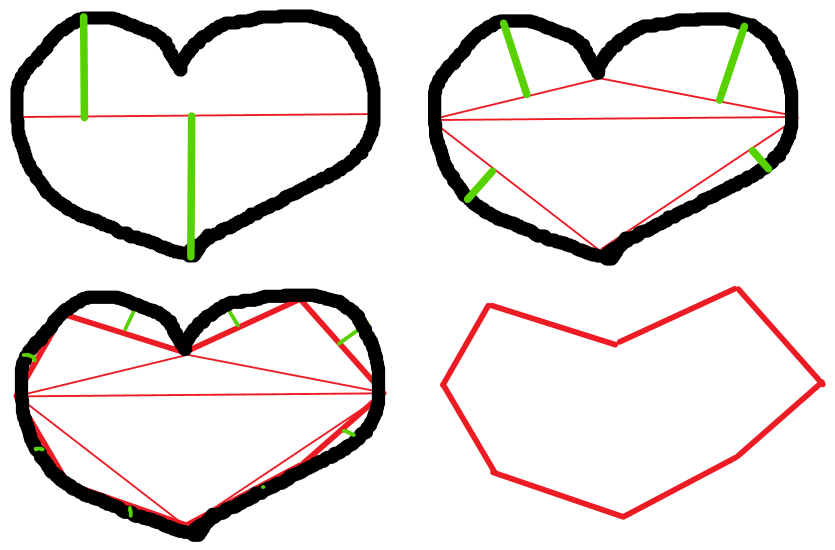


图32 曲线多变形状近似原理

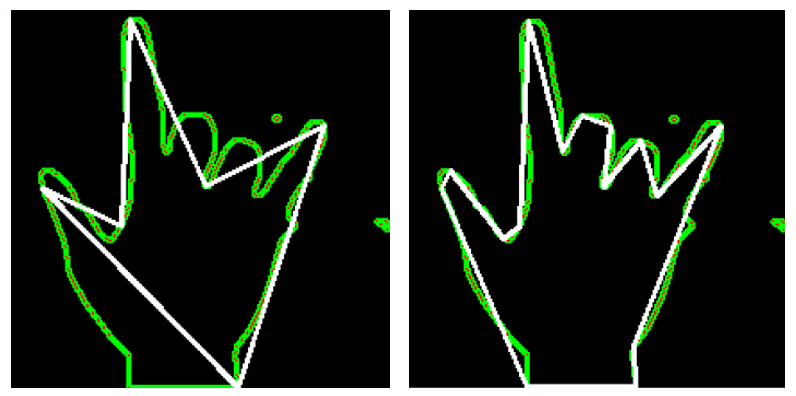


图33 两种不同的多边形近似效果图

1. **凸包**

凸包(convex hull)是计算几何中的概念，可以认为凸包是一条刚好包着所有点的橡皮圈，凸包是一个没有凹处的多边形，凸包上任意两点的连线都在凸包内，是包含所有点集的最小凸多边形。OpenCV中实现轮廓凸包检测的是函数cv2.convexHull()，输入二维点集，注意检测是有方向的，参数returnPoints=Ture时返回坐标，returnPoints=False时返回坐标索引。这里之所以要单独说明这个参数的选择，是因为如果我们想绘制凸包到图像上应该返回左边，但在下一步的凸性特征检测时我们需要的是坐标索引，因此因格外注意。确定凸包过程是这样的，现在我们已经得到近似多边形轮廓，并按照x从小到大分配了各自的索引，当x相等时，再考虑y的大小，遍历所有点坐标找到xMIN、yMIN、yMAX、xMAX，将轮廓初次分为四个区域。在图34中，以左上角区域举例说明，每次处理三个点，当时是直线，当时是凸包，同理当时不是凸包。对于右上角区域是从右边界到上边界的，也就是左上角按顺时针检测，而右上角是按逆时针检查。对于下半区域，左下角按逆时针检测，右下角按顺时针检测。图35展示的是，在经过筛选得到面积最大的轮廓后剩下的最大面积区域，即为真正的手部轮廓，使用凸包检测函数，绘制出能够包含所有像素点的多边形，其中的紫色线，白色线是近似过的近似多边形轮廓。以A、L、O、E四点除此划分区域，仔细说明A-E五点的确定情况：第一步，比较AB与BC线段的斜率，AB段斜率大于BC段斜率，确定B点为凸点；第二步，比较BC与CD线段斜率，BC段斜率小于CD段斜率，舍弃C点；第三步，比较CD与DE线段斜率，CD段斜率小于DE段斜率，舍弃D点。这里没有解释直线情况，在AB段内其实还包含很多点，但它们的斜率是相等的，就不做仔细说明了。

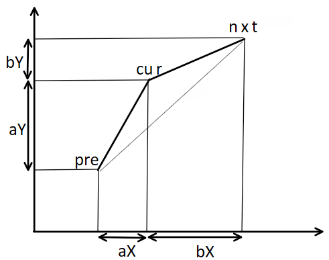


图34 凸包原理图

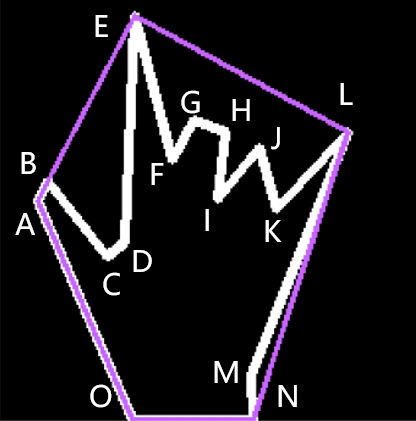


图35 凸包绘制图

最后是图像凸性检测函数cv2.convexityDefects()，其返回4个整形数据，第一个是start\_index表示缺陷在轮廓上的开始处的索引，第二个是end\_index表示缺陷结束处的索引，第三个是farthest\_pt\_index表示轮廓距离凸包最远的点的索引号，第四个是fixpt\_depth表示最远点到凸包的距离，以像素为单位。在图36和图37中，反映出三个凸性特征，首先我们看第一个重要信息，最远点到凸包的直线距离，分别是15739、12527和2082，我们舍弃2082这个特征，因为这个不是由手指指缝引起的距离，我们在程序代码编写中应该筛选出距离值过小的特征。之后我们把精力聚焦到剩下的特征三角形上。

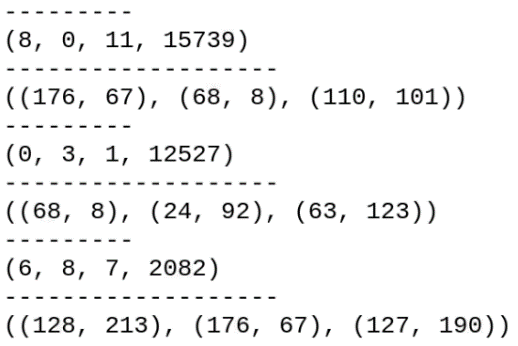


图36 凸性三角形索引号与顶点坐标

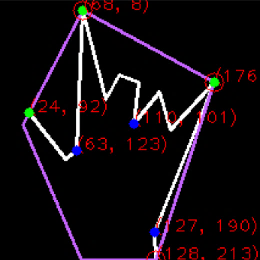


图37 凸性三角形sfe三点坐标

现在我们使用OpenCV已经得到一些凸包索引、一个距离以及一个三家形，下面会依据这些信息计算手势。

* 1. 手势识别的实现
     1. 数学原理

在图37中，我们指定start、far、end三点组成的三角形角度小于90°时认为是手指缝隙引起的夹角，图37中有，但这显然不是凸包引起来的，所以也要设置当d大于3000时认为是手指缝隙引起的缺陷，已经将其抛弃。三边长度用坐标差配合勾股定理计算得到，角度计算只需要使用余弦定理即可。我们仔细说明拇指与食指的几何关系确定，在图37中我们得到了三点的坐标分别是：(24，92)，(68，8)，(63，123)。第一条边长f计算：，第二条边长e计算：，第三条边长s计算：。

的角度计算：。这样手指指缝夹角小于90°且距离大于3000的几何特征，就被认定为是一个几何特征。图38中展示了三个三角形的距离和的角度。图39中是计算得来的三个三角形的三边长度。分析可以看出，手腕处产生的几何特征是不满足实际要求的，手指指缝夹角大于90°且距离小于3000。另外一个三角形特征分析可以看出，手指指缝夹角小于90°且距离大于3000，于是最终有两个符合要求的三角形几何特征，所以我们认为这张图有三个手指伸出。



图38 凸性三角形距离与夹角

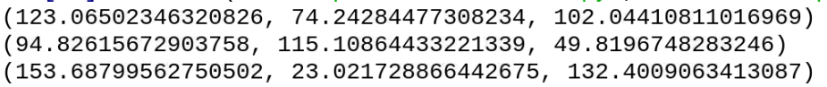


图39 凸性三角形三边长度

* + 1. 识别的代码与注释

#开始识别手势

l=0

#提取几何特征包括：起点坐标，终点坐标，最远点坐标

for i inrange(defects.shape[0]):

s,e,f,d=defects[i,0]

start=tuple(approx[s][0])

end=tuple(approx[e][0])

far=tuple(approx[f][0])

#求出三角形所有边的长度，到最远点的距离，夹角

a=math.sqrt((end[0]-start[0])\*\*2+(end[1]-start[1])\*\*2)

b=math.sqrt((far[0]-start[0])\*\*2+(far[1]-start[1])\*\*2)

c=math.sqrt((end[0]-far[0])\*\*2+(end[1]-far[1])\*\*2)

s=(a+b+c)/2

ar=math.sqrt(s\*(s-a)\*(s-b)\*(s-c))

d=(2\*ar)/a

angle=math.acos((b\*\*2+c\*\*2-a\*\*2)/(2\*b\*c))\*57

#忽略大于90度的角度并忽略非常靠近凸包的点（它们通常是由于噪声而来）

if angle<=90 and d>20:

l+=1

cv2.circle(draw,end,3,(255,0,0),-1)

cv2.circle(draw,start,2,(0,255,0),-1)

cv2.circle(draw,far,4,(255,255,255),-1)

#识别结束

l+=1

* 1. 小结

在论文的第二章中实现了论文的第一大任务，即完成对手势的识别。识别流程包括对图像的预处理、二值化分割、轮廓几何要素分析，在预处理的最后经过对各种滤波的比对，选择中值滤波作为模糊处理可以提高手势识别的准确性，二值化分割中选择使用HSV颜色空间下的肤色作为分割依据，可以提升逆光时的识别准确性，轮廓几何要素部分，影响识别结果的主要因素的在对轮廓多边形近似时的参数选择，这一步影响最终的几何要素点坐标。

2. 智能车硬件设备驱动
   1. 金属齿轮直流电机

本论文中使用到四个DC3V-6V直流减速电机和TB6612FNG的电机驱动模块。直流电机的工作原理简单的说就是处在磁场中的通电导线发生感应受力而产生旋转扭矩。主要解决控制电机正反转，控制电机速度，实现这两件事情需要两种输入线：电源输入线和控制信号线，实现调速的方法是使用PWM方波信号。直流电机只要两端供应不同的高低电，可以实现旋转，但实际应用中不能让小车一直保持全速，应具有可以改变速度的能力，PWM调速可以看作是一种不连续的供电，智能小车的电机驱动整体设计是，智能小车包括前后两组轮，每一组轮由一个电机驱动模块直接驱动，控制板由两块电机驱动模块它们的引脚是重复使用的，树莓派以微控制器的身份分三路控制电机模块，两个引脚配合控制电机正反转，另一引脚输出PWM方波控制电机转速。

* 1. TB6612模块说明

由于单片机IO口提供的电源能量太弱，想要驱动直流电机就需要一个功率放大器，树莓派自身没有电机驱动模块，因此要配合TB6612FNG电机驱动模块控制电机。TB6612FNG是一款直流电机驱动，具有大电流MOSFET-H结构，双通道电路输出，可以同时驱动2个电机；TB6612FNG每通道输出最高1A的连续驱动电流，有四种工作模式：正转、反转、制动、停止；PWM（Pulse Width Modulation）支持频率高达100KHz；相比L298N的高热耗性，其无需散热片；采用小型贴片封装[10]。

* + 1. 驱动模块原理

H桥驱动器。见图40所示，要使电机转动，需要导通对角线的电路，当2和4闭合，1和3断开时，记该旋转方向为正；则2和4断开，1和3闭合时为反转。当单1和4或单2和3闭合时，电路对电机旋转产生的电动势有阻碍作用，起到刹车的效果。在电路中其开关作用的可以是三极管、MOS管，MOS管的效率远远高于三极管。简单的开关只能控制电机正反转，引入PWM控制可以实现速度调节。PWM意思是脉冲宽度调制，脉冲是一种电压反复在高和低电平之间改变的电信号。脉宽代表高电平的持续时间。PWM接口是嵌入式设备常用接口之一，常用于电机驱动。调节占空比，占空比越大，平均电流越大，转速越高。

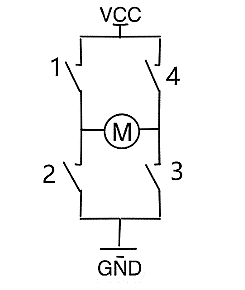


图40 H桥原理图

本论文中的驱动电路使用三路控制，使用MOS管代替原来的开关位置，并且接入使能接口，组成实际的H桥电路，见图42，其中使能ENA引脚就是PWM引脚，IN1和IN2就是普通的GPIO引脚，只控制电机正反转，ENA=1时，IN1和IN2引脚是可以上电的，调节使能ENA引脚的占空比，当PWM=0%时，电机停止转动；当PWM=100%时，电机转速达到最大。当ENA引脚接入高电平时，IN1和IN2引脚可以接通；当IN1=0，IN2=1时，M1和M2导通，电机两端的电压方向从右到左，电机正转；当IN1=1，IN2=0时，M3和M4导通，电机两端的电压方向从左到右，电机反转。当ENA=0时，电机停转。

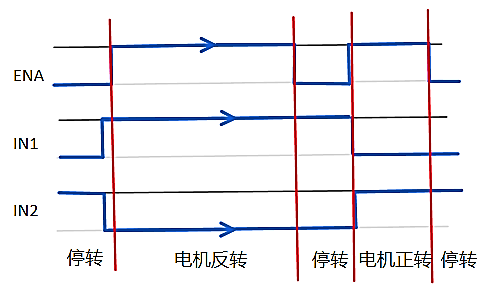


图41 电机控制示意图

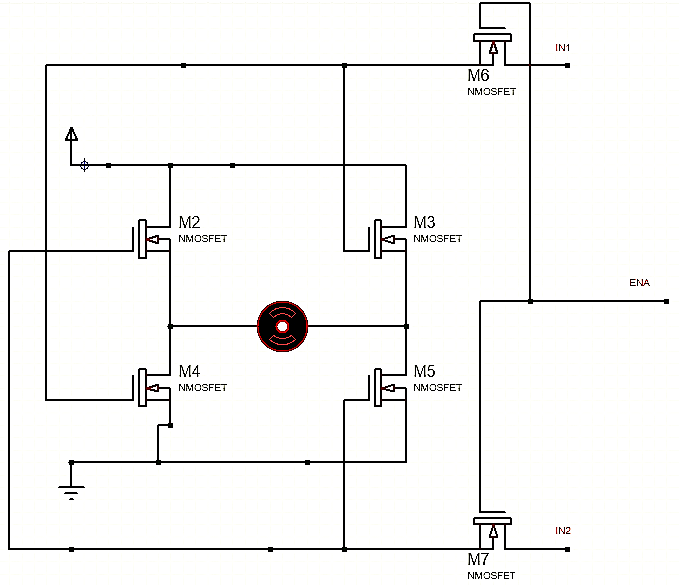


图42 三路H桥

* + 1. 电机驱动电路

图43中是TB6612FNG的主要引脚。控制信号输入端：AIN1/AIN2、BIN1/BIN2、APWM/BPWM。电机控制输出端：A01/A02、B01/B02。引脚13、14、24是VM引脚，为电机驱动电压输入端，接12V电源。VCC为逻辑电平输入端，接5V电源。19引脚STBY口，为正常工作/待机状态控制引脚，0电机全部停止，1允许通过AIN1/AIN2、BIN1/BIN2、逻辑信号控制电机正反转。驱动电路A：APWM接raspberry的PWM口，A01/A02接电机A组的两个脚。驱动电路B：BPWM接raspberry的PWM口，B01/B02接电机B组的两个脚。下表1是左右两路电机的真值表。

经过定时测速，获取到对应的准确的速度值与PWM占空比的比例关系，在占空比为百分之五十的情况下，计时10秒的直线行驶过程中，小车前进了205.2厘米，智能小车获得到的速度是20.52厘米每秒。在占空比为百分之六十的情况下，计时10秒的直线行驶过程中，小车前进了242.3厘米，智能小车获得到的速度是24.23厘米每秒。在占空比为百分之百的情况下，计时5秒的直线行驶过程中，小车前进了227.0厘米，智能小车获得到的速度是45.4厘米每秒。

表 1.1 两路电机真值表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| AIN1 | 0 | 1 | 0 | BIN1 | 0 | 1 | 0 |
| AIN2 | 0 | 0 | 1 | BIN2 | 0 | 0 | 1 |
|  | 停止 | 正转 | 反转 |  | 停止 | 正转 | 反转 |

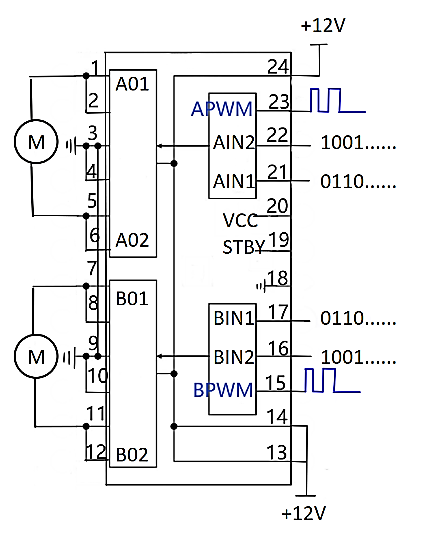


图43 电机驱动模块逻辑图

* 1. 树莓派驱动电机程序设计
     1. 树莓派GPIO引脚

树莓派有许多可编程的GPIO（通用输入输出引脚），全称 General Purpose Input Output，利用这些引脚控制外部硬件设备，或者从外部获取信息是与硬件交互的通道。树莓派GPIO有BCM、WiringPi、BOARD（板载编码）三种编码方式。在树莓派终端键入gpio readall，可以得到所有GPIO状态，共40个，图44所示的就是树莓派的终端窗口查阅的所有GPIO状态。其中，中间位置是板载编码，Name显示的是引脚的作用。我们可以在树莓派终端命令行(shell窗口)输入指令pinout来显示，图45展示的是树莓派的整体结构，Pi Model 4B V1.4是树莓派硬件版本。它的片上系统，集CPU、GPU、DSP和SDRAM为一体，使用SD卡作为内存硬盘，拥有以太网接口，有两个USB2.0和两个USB3.0接口可以连接键盘、鼠标、U盘这种外设，拥有视频、音频模拟输出以及HDMI高清输出，还具有GPIO、SPI、UART等接口，这些部件全部集成在一张信用卡大小的主板上，具有所有PC的基本功能，同时可以提供通用丰富的IO口，用于输入输出、通讯，用于物联网控制或传感器的互动[11]。

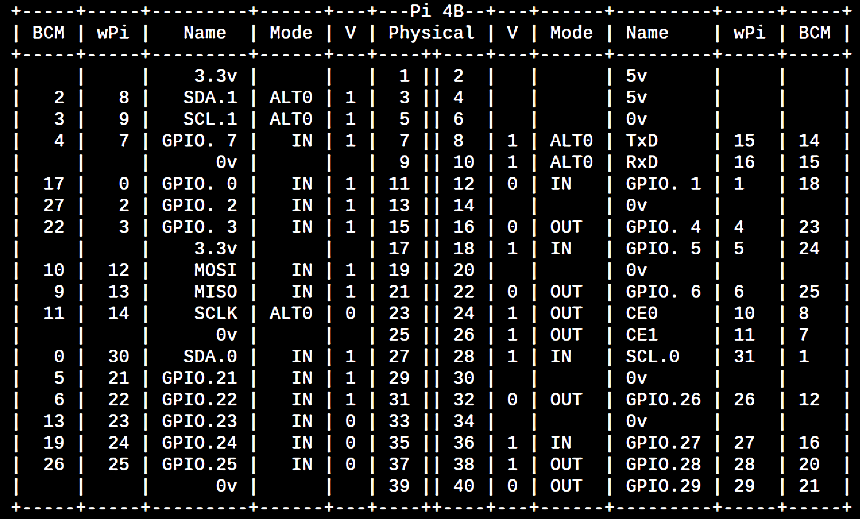


图44 树莓派详细引脚状态图

接下开的编码都是使用的BCM编码。板载编码是按照raspberry主板上的引脚排列顺序编号，使用时注意顺序。一般情况，Python使用的BCM编码，C语言使用的WiringPi编码。电源类引脚：5V、3.3V、GND。通讯引脚：SPI通讯、通讯、TXD/RXD串口通讯。最后还有普通的控制引脚[12]。

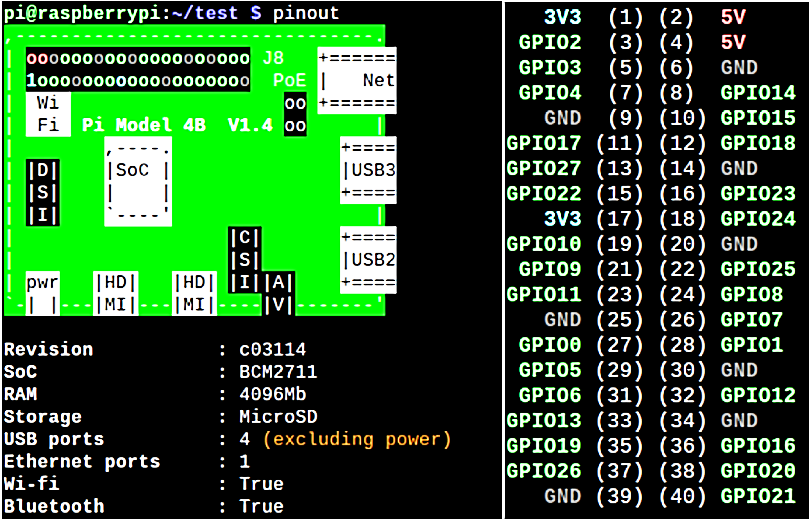


图45 树莓派结构图

* + 1. 电机驱动代码

首先设置编码形式，使用函数GPIO.setmode(GPIO.BCM)，即设置使用BCM编码形式。在设置引脚，设置某引脚为输入或输出，使用函数GOIO.setup(\*，GPIO.IN/GPIO.OUT)；设置某引脚为高电平或低电平，使用函数GPIO.output(\*，GPIO.HIGH/1/Ture GPIO.LOW/0/False)。代码最后应清空调用的GPIO口，使用函数GPIO.cleanup()。树莓派有专门的类叫pwm，首先设置PWM引脚实例，使用函数pwm1=GPIO.PWM(\*,freq)，频率参数freq设置赫兹数，比如100HZ，则一个周期10ms。设置使用pwm及其占空比，首先启动/清空PWM使用函数pwm1.start(dc)/pwm1.stop()，设置dc为初始占空比。改变占空比来控制速度，使用函数pwm1.ChangeDutyCycle(dc)，占空比0~100，0就是没有高电平，即零速；100即为全速[13, 14]。

#树莓派小车运动函数

deft\_up(speed,t\_time):#前进函数

L\_Motor.ChangeDutyCycle(speed)#控制左轮占空比，改变速度

GPIO.output(AIN2,False)#A：10；正转

GPIO.output(AIN1,True)

R\_Motor.ChangeDutyCycle(speed)#控制右轮占空比，改变速度

GPIO.output(BIN2,False)#B：10；正转

GPIO.output(BIN1,True)

time.sleep(t\_time)#前进时间

deft\_stop(t\_time):#停车函数

L\_Motor.ChangeDutyCycle(0)#控制左轮占空比，为0，即停止

GPIO.output(AIN2,False)#A：00；停止

GPIO.output(AIN1,False)

R\_Motor.ChangeDutyCycle(0)#控制右轮占空比，为0，即停止

GPIO.output(BIN2,False)#B：00；停止

GPIO.output(BIN1,False)

time.sleep(t\_time)#停止时间

deft\_down(speed,t\_time):#倒车函数

L\_Motor.ChangeDutyCycle(speed)#控制左轮占空比，改变速度

GPIO.output(AIN2,True)#A：01；反转

GPIO.output(AIN1,False)

R\_Motor.ChangeDutyCycle(speed)#控制右轮占空比，改变速度

GPIO.output(BIN2,True)#B：01；反转

GPIO.output(BIN1,False)

time.sleep(t\_time)#倒车时间

deft\_left(speed,t\_time):#左转函数

L\_Motor.ChangeDutyCycle(speed)#控制左轮占空比，改变速度

GPIO.output(AIN2,False)#A：10；正转

GPIO.output(AIN1,True)

speed+=35

R\_Motor.ChangeDutyCycle(speed)#控制右轮占空比，改变速度,要比左侧快

GPIO.output(BIN2,False)#B：10；正转

GPIO.output(BIN1,True)

time.sleep(t\_time)#左转时间

deft\_right(speed,t\_time):#右转函数

R\_Motor.ChangeDutyCycle(speed)#控制右轮占空比，改变速度

GPIO.output(BIN2,False)#B：10；正转

GPIO.output(BIN1,True)

speed+=35

L\_Motor.ChangeDutyCycle(speed)#控制左轮占空比，改变速度,要比右侧快

GPIO.output(AIN2,False)#A：10；正转

GPIO.output(AIN1,True)

time.sleep(t\_time)#右转时间

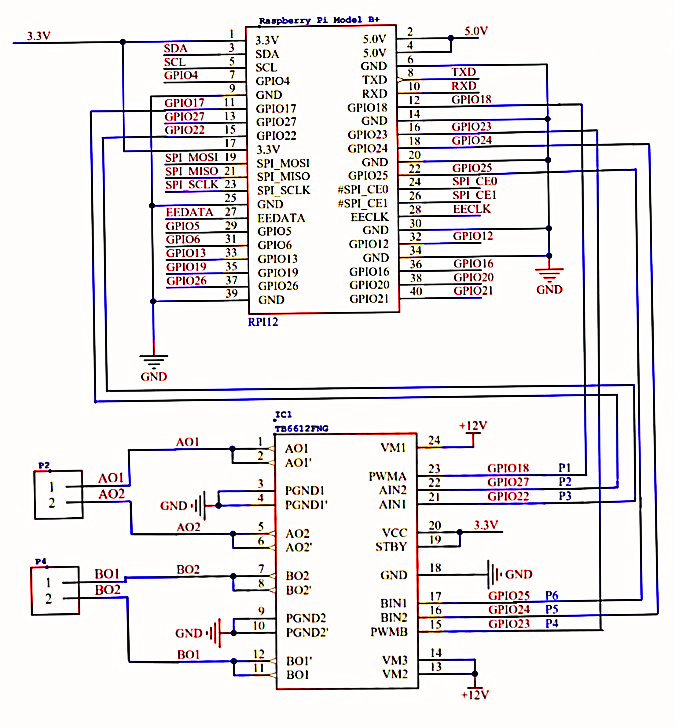


图46 电机电路连接图

2. 总结与展望
   1. 工作总结

人机交互是当前计算机研究领域的一个非常热门的研究方向，本论文设计了以树莓派为主板，运用OpenCV库的对手势进行识别，配合GPIO口应用对智能小车进行运动控制。本论文使用Python语言及丰富的OpenCV库函数对图像进行处理和识别操作，在图像处理方面OpenCV函数库使用方便，运行速度快。树莓派是学习计算机视觉处理的绝佳选择，能够运行图像处理的大量的分析算法。在这次毕业设计的学习过程中，深入学习了计算机视觉处理的本质、手势识别的算法、树莓派的数据传输。本次论文的设计在提升逆光识别能力方面任有很大提升，未来一段时间，计算机视觉会在各个领域得到广泛应用，随着嵌入式的计算能力的提升，嵌入式计算机视觉可以应用于无人驾驶和军事等领域。论文的主要内容和实现方法：

1. 图像的预处理技术，包括图像滤波和形态学处理。
2. 图像分割的二值化技术，包括基于灰度图和基于HSV空间的阈值方法。
3. 手势识别的原理，包括认识凸性特征和运用几何特征计算夹角。
4. 使用树莓派控制电机，包括Python语言的学习和BCM编码。
5. 由于摄像头的硬件限制，不能够快速的聚焦或者感光，对于一些逆光或者背景较复杂的情况不可以很好的识别。
   1. 展望

本篇论文中，主要实现了手势的识别，以及小车对识别结果的自动执行，应该将手势识别做进一步研究，可以有更多的手势，添加加速手势和减速手势，使智能小车的运动更加灵活。本篇论文是识别出了静态的六种手势，实现的代码还不足以克服复杂的背景，需要纯白色的或者稍微干净的背景，来方便图像的识别。后期的学习将主要针对可能复杂背景的识别实现。

参考文献

[1]何鑫. 购物小车的手势识别方法研究[D]. 哈尔滨理工大学, 2019.

[2]伍杰. 基于视觉的实时手势识别方法研究[D]. 大连理工大学, 2019.

[3]段佳雷. 基于ARM的跟踪小车的设计与实现[D]. 兰州理工大学, 2014.

[4]钱春妍. 基于OpenCV的实时手势识别与应用研究[D]. 重庆大学, 2015.

[5]孟国庆. 基于OpenCV的手势识别技术研究[D]. 西安科技大学, 2014.

[6]鲁一姝. 基于视觉的手势识别研究\_鲁一姝[Z]. 2016.

[7]贺航. 基于opencv的手势识别的设计改进与实现[D]. 华南理工大学, 2018.

[8]Ismail A P, Aziz F A A, Kasim N M, et al. Hand gesture recognition on python and opencv[J].

IOP conference series. Materials Science and Engineering, 2021,1045(1):12043.

[9]罗娜. 基于OpenCV的自然手势识别与交互系统研究[D]. 广东工业大学, 2012.

[10]陈茜茹, 李志为. 基于树莓派的自动跟随行李箱[J]. 电子技术与软件工程, 2019(13):99-101.

[11]Kölling M. Educational Programming on the Raspberry Pi[J]. Electronics (Basel), 2016,5(4):33.

[12]张策. 基于树莓派的牛舍巡检机器人系统的研究与设计[D]. 北方民族大学, 2021.

[13]戴文翔, 孙智勇. 基于树莓派的红外避障小车[J]. 数字技术与应用, 2018,36(01):89-90.

[14]Tiwari R. Vehicle Control Using Raspberrypi and Image Processing[Z]. 2017.

致谢

自2018年的夏天以来，我与这个学校、这里的老师和同学们，与这片土地上的所有人朝夕相处了近四年。我清晰记得第一教学楼敞亮的教室，清晰记得第二教学楼里生动的高等数学课，清晰记得第六教学楼大家一起上自习的场景。带着这些美好的回忆，我完成了我的学位论文，为自己的本科学习生活画上圆满的句号。

首先，我要感谢我的指导老师孙永道，孙老师学识前沿，从选题、开题到定稿，感谢老师带我上Python课，引导我深入学习机器语言，让我对自己的爱好有了更深的理解，对自己的未来规划也有了更为深刻的认识。非常欣赏孙老师做事的态度，对工作的认真和对知识的追求深深吸引了我，给我上了一趟非常深刻的人生追求课，是我未来学习的榜样和一直追求的偶像。在此，特意向我的指导老师致以真挚的感谢和崇高的敬意。

其次，无比感谢家人对我的理解，他们提供给我宽阔的空间，支持我完成我的毕业设计，帮我搭网线，给我提建议，在我的求学之路上一直支持我，在我迷茫时耐心开导我。在此，祝福我的所有家人身体健康，万事胜意。

再次，感谢我的汽车系，为我提供设备支持了我的毕业设计。感谢我的师兄张建熙，他学识丰富、思路创新，在我探索单片机的道路上与我一起探讨，相互学习。感谢我的舍友和同学，大学生活中我们互帮互助，互相监督学习。大学收获的不只是知识，还有亲切的同学和多姿多彩的回忆。我会带着回忆继续我的学业，继续追求实现我的理想抱负。此去经年，告别大学时代，我们依旧如初。

附录



