**电子科技大学成都学院**

**实验报告**

课程名称 基于opencv的银行卡卡号识别

分 院 计算机学院

年 级 2020

专　　业 人工智能

班 级 人工智能3班

任课教师 陈东祥

学生姓名 何磊 林超煌 罗永乐 刘粱兵

2021年12月制

摘 要

在平常生活中，我们在与金融有关的APP中看到添加银行卡，扫描得到银行卡号的快捷输入方式。比如，在支付宝中就能看到，其实这个需求现在已经有一些成熟的第三方库，收费或者不收费。学习Opencv一段时间后，我们也可以拥戴末实现记录下来，方便如果以后遇到，更好的使用或者有条件的优化别人的第三方库。

本论文设计的银行卡卡号程序，可以完成银行卡卡号的自动识别，具有较高的识别速率和准确率，程序根据银行卡号，字符分割和字符识别的位置执行识别银行卡号的过程。在图像预处理的基础上，图像灰度化，二值化，找到所有轮廓并将轮廓画出，对找到的轮廓排序,得到轮廓集合，再将每个数字制成一个模板。最后，用模板匹配识别字符以获得银行卡号。

**关键词：**银行卡卡号识别、图像处理、模式识别、轮廓处理、二值化

ABSTRACT

In ordinary life, we see the added bank card in the financial-related APP, and scan the shortcut input mode of the bank card number.For example, in Alipay can see, in fact, in fact, this demand now has some mature third-party library, charging or no charge.After learning Opencv for a period of time, we can also support the final implementation to record, convenient if encountered in the future, better use or conditionally optimize others's third-party library.

The bank card number program designed in this paper can complete the automatic identification of bank card number, with a high identification rate and accuracy. The program performs the process of identifying bank card number according to the position of bank card number, character division and character identification.Based on image preprocessing, the image is grayscale, binarized, finds all profiles and draws the outline, sorts the found profiles, gets the set of profiles, and makes each number into a template.Finally, the recognition characters were matched with a template to obtain the bank card number.

**Key Words:** Bank card number identification,image processing, pattern recognition, contour processing、binarization

目 录

[第1章 引言 1](#_Toc23892)

[1.1 选题背景 1](#_Toc27035)

[1.2 研究目标和意义 1](#_Toc30349)

[1.3 研究思路 1](#_Toc16406)

[第2章 研究的理论和基础 3](#_Toc16719)

[2.1 图像灰度化 3](#_Toc22361)

[2.2 图像的二值化 3](#_Toc28297)

[2.3 寻找轮廓并将模块轮廓划分和排序 4](#_Toc27078)

[2.4 定义卷积核和顶帽操作 5](#_Toc3988)

[2.5 索贝尔算子 7](#_Toc2503)

[2.6 进行闭运算【先膨后腐】 8](#_Toc459)

[2.7 用threshold进行二值化处理并重复一次闭操作 9](#_Toc10646)

[2.8 用cv.findCountours检测轮廓 10](#_Toc30622)

[2.9 遍历轮廓并预处理 11](#_Toc7810)

[2.10 轮廓排序 12](#_Toc4944)

[2.11 模板匹配 12](#_Toc25382)

[第3章 银行卡卡号识别现状 15](#_Toc26838)

[3.1 现状概括 15](#_Toc14432)

[第4章 结论 16](#_Toc10087)

[4.1 实验结论 16](#_Toc788)

[4.2 心得体会 16](#_Toc28854)

[4.3 优化及改进方法 16](#_Toc29952)

[参考文献 17](#_Toc16682)

[致谢 18](#_Toc17455)

1. 引言
   1. 选题背景

随着互联网金融的兴起，人们需要在各种终端中录入银行卡号，并绑定银行卡从而进行资金交易，在卡号的录入过程中，需要对拍摄或者预先保存的银行卡图像进行卡号识别，从而将识别结果实现录入。银行卡号识别方法首先将银行卡图像进行灰度化、二值化等预处理，再定位银行卡号，最后对银行卡号进行数字识别。

* 1. 研究目标和意义

通过基于opencv设计的程序识别银行卡上的卡号，并且该程序具有较高的识别速率和准确率，让人们不用手动输入银行卡号就能进行自动识别。本文根据银行卡号的特点，设计了一个基于模版匹配的银行卡号识别程序，经过实验发现，这个程序可以比较快速而准确地识别银行卡号，并且基本上可以保证银行卡号识别功能要求。通过对图像处理与模式识别中已有算法的分析研究，设计并实现了基于模板识别的银行卡号识别系统。不仅可以识银行卡号，其相关技术也可应用到车牌号码、护照号、身份证号或其他印刷体字符的识别。

* 1. 研究思路

一 、准备模板，能够与卡号数字样式进行模型匹配

二、处理模板图片：将其分割成为单个数字的图片，和所对应的数字相匹配存储可迭代的数据类型中。

①读入模板图片，并对模板图片进行灰度化，最后转换为二值图像

②计算模板的轮廓，给轮廓排序（按照在银行卡上的位置），这里选择检索外轮廓，压缩水平的，垂直的和斜的部分

③按照数字的顺序将每个数字的小图片存入列表

三、如何给轮廓排序

①先将轮廓的外接矩形画出

②外接矩形会返回左上角点的坐标以及长和宽，利用横坐标进行排序

四、处理输出图片

①读入图像，进行灰度化、二值化处理，调整图片的大小。

②为了突出比原轮廓亮的部分，使用顶帽操作处理图像，求图片的梯度，并进行求绝对值，归一化处理。

③需要找到包含银行卡号的小矩形，需要提取银行卡号所在小块的轮廓。

其中包括：闭运算（使文字，数字等融合成一块一块的）。二值转换（自适应阈值）。闭运算（消除白色块中的杂质）。求轮廓，在图像中画出轮廓，画出每个轮廓的外接矩形，返回左上角的坐标以及长与。根据长宽比与坐标值，筛选出我们需要的矩形框。

五、将所有小块中的每个数字取出，与模板进行匹配，得到银行卡号

①将得到的4个小块（对应四块银行卡号）分别遍历，用每个块的位置信息得到原图像的灰度图中的相关区域图像（稍微扩大一些 ）

②对这些区域进行二值化处理，计算轮廓，并进行排序，得到每一个数字的图像

③将单个数字与模板中的每一个数字进行匹配，得到最相似的模板图像所对应的索引就是该数字所对应的数字。

1. 研究的理论和基础
   1. 图像灰度化

将彩色图像转化成为灰度图像的过程称为图像的灰度化处理。灰度化，在RGB模型中，如果R=G=B时，则彩色表示一种灰度颜色，其中R=G=B的值叫灰度值，因此，灰度图像每个像素只需一个字节存放灰度值（又称强度值、亮度值），灰度范围为0-255。彩色图像中的每个像素的颜色有R、G、B三个分量决定，而每个分量有255个值可取，这样一个像素点可以有1600多万的颜色的变化范围。而灰度图像一个像素点的变化范围为255种，所以在数字图像处理种一般先将各种格式的图像转变成灰度图像以使后续的图像的计算量变得少一些。灰度图像的描述与彩色图像一样仍然反映了整幅图像的整体和局部的色度和亮度等级的分布和特征。对于灰度化的方式有分量法、最大值法、平均值法，加权平均法等。

下面用代码进行灰度处理，代码运行如图2-1-1，结果如图2-1-2：

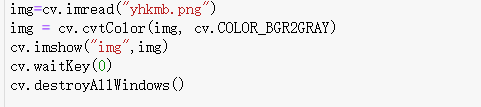


图2-1-1 图像灰度化代码运行图

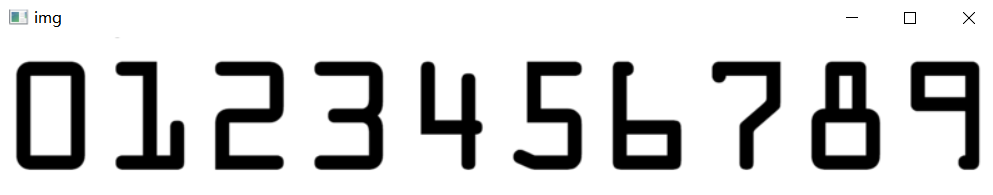


图2-1-2 灰度处理后的结果

* 1. 图像的二值化

通过以上对彩色图片进行灰度化以后，把获取到的灰度图像进行二值化处理。对于二值化，其目的是将目标用户背景分类，为后续车道的识别做准备。灰度图像二值化最常用的方法是阈值法，他利用图像中目标与背景的差异，把图像分别设置为两个不同的级别，选取一个合适的阈值，以确定某像素是目标还是背景，从而获得二值化的图像。图像二值化就是将灰度图像上的像素值设置为0或255，也就是将整个图像呈现出明显的黑白效果的过程。自定义阈值为10,反二进制阈值。把亮的处理成黑色，暗的处理成白色，也表示阈值的二值化翻转操作，大于阈值的使用0表示，小于阈值的使用最大值（255）表示，大于10的是黑色，小于的是白色，ret的值为True或False。下面用代码进行图像二值化处理，代码运行如图2-2-1，结果如图2-2-2：

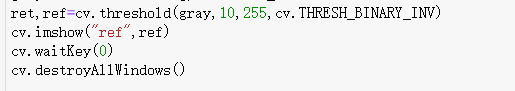


图2-2-1 图像二值化运行代码图



图2-2-2 二值化处理结果

* 1. 寻找轮廓并将模块轮廓划分和排序

使用cv.findContours函数检测轮廓，refCnts为我们得到轮廓（要用copy原始数据图像后的文件），这里选择参数cv.RETR\_EXTERNAL只检测外轮廓，cv.CHAIN\_APPROX\_SIMPLE只保留终点坐标，返回的list中每个元素都是图像中的一个轮廓。使用cv2.drawContours函数绘制轮廓(选择参数-1，代表绘制出所有轮廓，选择参数,即用红色绘制轮廓)，使用contours.sort\_contours排序函数（从左到右,从上到下)，排序枚举遍历每一个轮廓，使用cv2.boundingRect(c)函数计算外接矩形参数,并通过cv2.resize函数构成最合适的尺寸后，循环写入digits[i]元组。其中refCnts = contours.sort\_contours(refCnts, method="left-to-right")[0]将数字模板每个数字的轮廓计算出来，并且排序，refCnts可以看做轮廓合集。用for (i,c) in enumerate (refCnts)遍历每个轮廓，得到模块。enumerate() 函数用于将一个可遍历的数据对象(如列表、元组或字符串)组合为一个索引序列，同时列出数据和数据下标，一般用在 for 循环当中。(x,y,w,h)=cv.boundingrect(c)表示获得轮廓的外接矩形，返回元组矩形左上坐标和矩阵长宽。roi=ref[y:y+h,x:x+w]表示获得roi感兴趣部分。roi=cv.resize(roi,(57,88))表示转化大小适用于模板匹配。运行代码如图2-3-1，处理结果如图2-3-2所示：

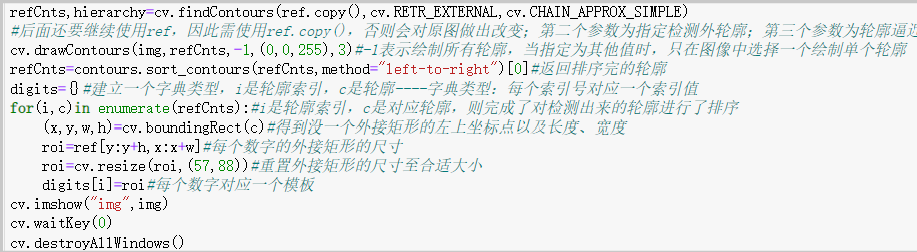


图2-3-1 轮廓处理运行代码



图2-3-2 轮廓处理结果

* 1. 定义卷积核和顶帽操作

初始化卷积核，读取输入图像,预处理（调用cv.resize函数调整大小,cv.cvtColor函数转化为灰度图)，调用cv.morphologyEx函数进行顶帽操作,突出更明亮的区域，增大图片对比度（参数根据需要提取信息的范围大小来选取,这里选用初始化卷积核rectKernel)。顶帽运算(img) = 原始图像(img) - 开运算(img)，图像开运算主要使用的函数morphologyEx，它是形态学扩展的一组函数，其参数cv2.MORPH\_TOPHAT对应开运算。其原型如下：dst = cv2.morphologyEx(src, cv2.MORPH\_TOPHAT, kernel)，参数dst表示处理的结果，src表示原图像，cv2.MORPH\_TOPHAT表示顶帽运算，kernel表示卷积核。运行代码如图2-4-1，输出结果如图2-4-2与图2-4-3所示：

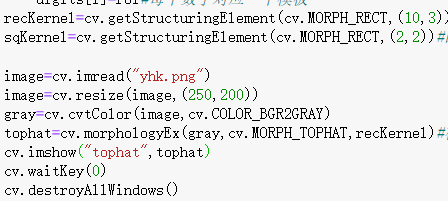


图2-4-1 运行代码图

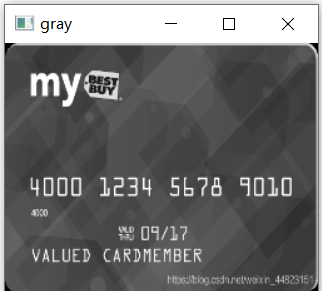


图2-4-2 定义卷积核处理结果

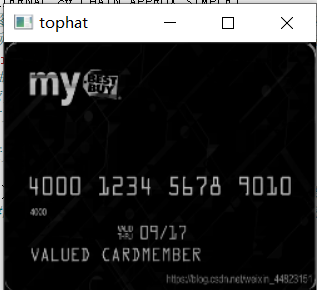


图2-4-3 顶帽操作处理结果

* 1. 索贝尔算子

索贝尔算子（Sobeloperator）主要用于获得数字图像的一阶梯度，是一种离散性差分算子。它是prewitt算子的改进形式，改进之处在于sobel算子认为，邻域的像素对当前像素产生的影响不是等价的，所以距离不同的像素具有不同的权值，对算子结果产生的影响也不同。一般来说，距离越远，产生的影响越小。

图像处理中认为，灰度值变化剧烈的地方就是边缘。 sobel算子就是对一副图像的输入到输出边缘信息的整个处理过程。

sobel算子的思想，Sobel算子认为，邻域的像素对当前像素产生的影响不是等价的，所以距离不同的像素具有不同的权值，对算子结果产生的影响也不同。一般来说，距离越远，产生的影响越小。

sobel算子的原理，对传进来的图像像素做卷积，卷积的实质是在求梯度值，或者说给了一个加权平均，其中权值就是所谓的卷积核；然后对生成的新像素灰度值做阈值运算，以此来确定边缘信息。

sobel算子作用，Sobel算子根据像素点上下、左右邻点灰度加权差，在边缘处达到极值这一现象检测边缘。对噪声具有平滑作用，提供较为精确的边缘方向信息，边缘定位精度不够高。当对精度要求不是很高时，是一种较为常用的边缘检测方法。调用cv.Sobel算法进行图像梯度处理(因为这里只计算gradX效果比较好，gxadX参数为dx=1,dy=0，ksize-1相当于用3\*3的卷积核)。运行代码如图2-5-1，输出结果如图2-5-2所示：

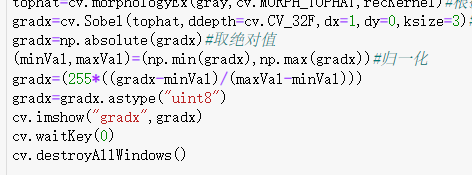


图2-5-1 索贝尔算子运行代码

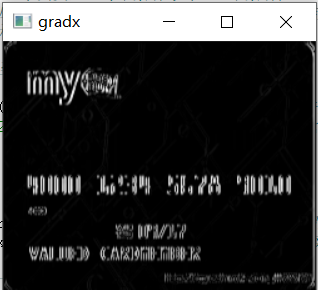


图5-2-2 索贝尔算子处理结果图

* 1. 进行闭运算【先膨后腐】

函数cv,morphologyEx()进行形态学其他操作。闭运算，先膨胀，后腐蚀。将数字连在一起。它有助于关闭前景物体内部的小孔，或物体上的小黑点。因为卡号信息为四个数字为一组，所以调用cv.morphologyEx函数进行闭操作，将数字连在一起。使图像中的孔隙填充（先对图像膨胀后腐蚀 作用:用来填充物体内的小空洞,连接邻近的物体,连接断开的轮廓线,平滑其边界的同时不改变面积），闭运算运行代码如图2-6-1，输出结果如图2-6-2所示



图2-6-1 闭运算运行代码

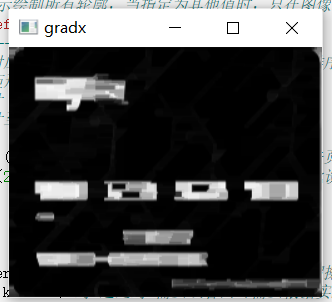


图2-6-2 闭运算处理结果

* 1. 用threshold进行二值化处理并重复一次闭操作

调用cv.threshold进行二值化处理,使得我们所需信息更加清晰明确（参数选择THRESH\_OTSU，会自动寻找合适的阈值,适合双峰,此时需把阙值参数设置为0）。因为前面操作结果显示,需要提取的卡号信息中,有部分数字连接不够紧密,中间仍有空白空缺,所以重复一次闭操作,使得后面轮廓检测能够更加准确。

thresh=cv.threshold(gradX,0,255,cv2.THRESH\_BINARY |cv.THRESH\_OTSU)[1]

参数说明， src表示输入的图片， thresh表示阈值， maxval表示最大值， type表示阈值的类型 type的类型

1.cv2.THRESH\_BINARY 表示阈值的二值化操作，大于阈值使用maxval表示，小于阈值使用0表示

2. cv2.THRESH\_BINARY\_INV 表示阈值的二值化翻转操作，大于阈值的使用0表示，小于阈值的使用最大值表示

3. cv2.THRESH\_TRUNC 表示进行截断操作，大于阈值的使用阈值表示，小于阈值的不变

4. cv2.THRESH\_TOZERO 表示进行化零操作，大于阈值的不变，小于阈值的使用0表示

5. cv2.THRESH\_TOZERO\_INV 表示进行化零操作的翻转，大于阈值的使用0表示，小于阈值的不变

运行代码如图2-7-1，输出结果如图2-7-2所示：

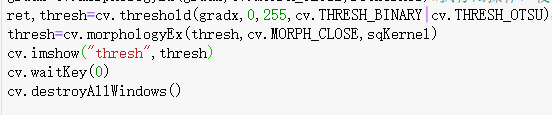


图2-7-1 二值化处理并进行闭操作

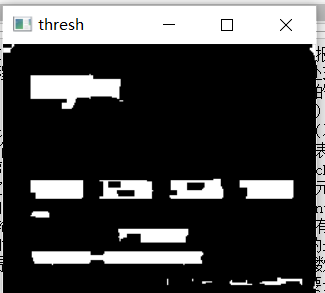


图2-7-2 进行二值化处理与闭操作后的输出结果

* 1. 用cv.findCountours检测轮廓

使用cv2.findContours函数检测轮廓,threshCnts为我们得到的轮廓（需使用copy原数据图像后的文件,这里选择参数cv2. RETR EXTERNAL只检测外轮廓,cV2. CHAIN\_APPROX\_SIMPLE只保留终点坐标)，并用红色绘制出所有轮廓。

检测轮廓运行代码如图2-8-1，输出结果如图2-8-2

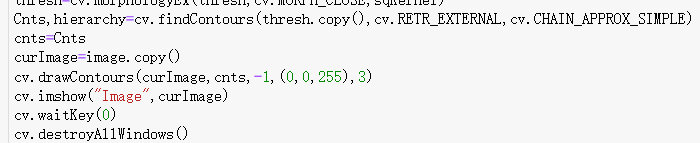


图2-8-1 检测轮廓运行代码

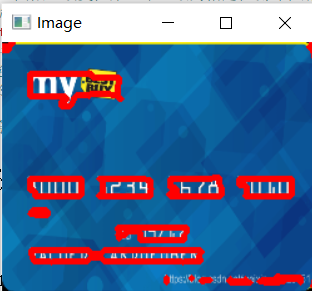


图2-8-2 检测轮廓输出结果

* 1. 遍历轮廓并预处理

遍历轮廓,筛查条件大致设置为我们所需的四组卡号轮廓的长宽比，从而得到我们需要的四组卡号轮廓,写入locs列表,将符合的轮廓从左到右排序。遍历得到的四个小组轮廓,根据坐标依次提取每一个小组轮廓。进行预处理（调用cv.threshold函数进行二值化)。遍历轮廓如图2-9-1，其输出结果如图2-9-2所示。

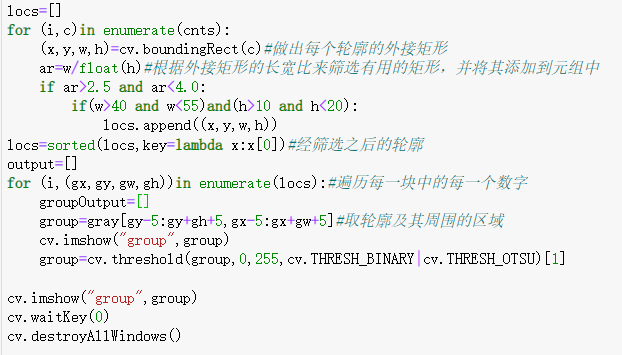


图2-9-1 遍历轮廓运行代码



图2-9-2 遍历轮廓输出结果

* 1. 轮廓排序

调用cv2.findContours函数检测轮廓,调用contours.sort\_contours函数从左到右进行排序,依次找到该组当前的每一个数值的轮廓,resize成跟之前模板处理的大小一致,接着遍历下一组，直到得到全部四组的每一位数值。hierarchy返回值：该函数还可返回一个可选的hiararchy结果，这是一个ndarray，其中的元素个数和轮廓个数相同，每个轮廓contours[i]对应4个hierarchy元素hierarchy[i][0] ~hierarchy[i][3]，分别表示后一个轮廓、前一个轮廓、父轮廓、内嵌轮廓的索引编号，如果没有对应项，则该值为负数。轮廓排序运行代码如图2-10-1，输出结果如图2-10-2所示。

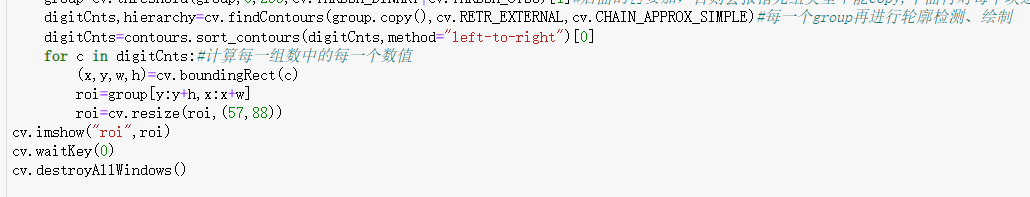


图2-10-1 轮廓排序运行代码

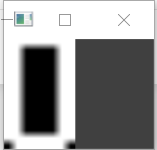


图2-10-2 轮廓排序处理结果

* 1. 模板匹配

最后是核心部分，进行模板匹配，创建一个scores []列表，存放匹配得分，接着需要调用cv2.matchTemplate函数,将银行卡号里每一位数字，与之前模板处理后得到的十个数字，进行依次逐个匹配，从而得到十个不同的得分,(因为这里参数使用的是cv2.TM\_CCOEFF，表示相关性系数，即 dot(x-x', y-y'),x’和y’表示的是x的均值和y的均值，相关性系数越大，模板与主图部分的匹配度越高)，所以使用cv2.minMaxLoc找到最高的得分，将它写入scores[ ]列表中,接着通过np.argmax (scores)查找该分数对应的数值，从而得出正确的数值后写入group0utput[]列表中，当该组四位卡号都成功读取后，则循环进行下一组同样的操作，直到得到全部四组的每一位数值，即完整的卡号信息后则输出识别到的完整。cv.putText(img, str(i), (123,456), font, 2, (0,255,0), 3)

各参数依次是：图片、添加的文字，左上角的坐标、字体、字体大小、颜色、字体粗细。

string.join() 具体作用如下： join()： 连接字符串数组。将字符串、元组、列表中的元素以指定的字符(分隔符)连接生成一个新的字符串.对字典进行连接，只能将字典的键连接起来。

.extend()与.append()都是给列表中添加元素，二者的区别：

list.append(a)添加的是a这个整体，而list.extend(a)会把a中的各个元素分开，a中的内容不再是一个整体。

cv.rectangle(image,(gx-5,gy-5),(gx+gw+5,gy+gh+5),(0,0,255),1)通过对角线上的两个顶点绘制简单、指定粗细或者带填充的矩形。分别表示图像，矩形的一个顶点，矩形对角线上的另一个顶点，线条颜色 (RGB) 或亮度，组成矩形的线条的粗细程度。取负值时（如CV\_FILLED）函数绘制填充了色彩的矩形。

模板匹配运行代码如图2-11-1，输出结果如图2-11-2所示。

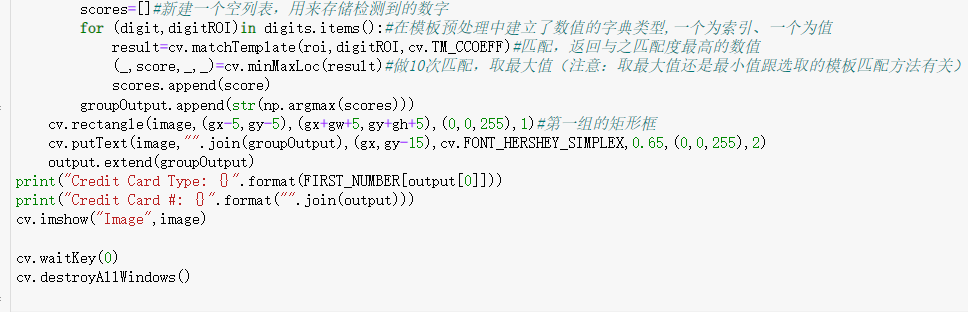


图2-11-1 模板匹配运行代码

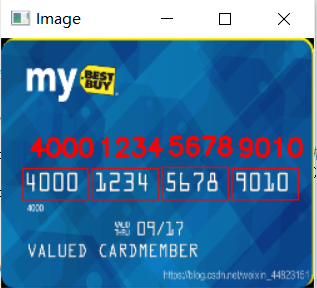
 

图2-11-2 模板匹配处理结果

1. 银行卡卡号识别现状
   1. 现状概括

在图像字符识别方面，现在大多采用的是神经网络、模板匹配等技术。神经网络具有推广能力、并行处理和自组织性等特性，因此它适合用于图像识别。特别在图像进行特征提取和识别时，神经网络明显优于传统的识别方法。目前使用最为广泛的神经网络是BP网，它对样本的识别率虽然很高。但网络训练比较麻烦，需要选择大量的样本才能保证最终的识别结果。同时，在实现过程中，由于受参数初始化的影响，它时常会无法收敛，或陷入局部极值。差别不等式需要每一张纸币的特征差别点和选取合适的阈值，这些需要专家根据经验来选择，因此具有较大的主观性。

1. 结论
   1. 实验结论

本实验通过对图像处理中已有算法的分析研究，借鉴了已有的一些程序，实现了银行卡卡号的识别。此程序不仅可以识别银行卡卡号，其相关技术也可应用到车牌号码、身份证号等其他图像识别。顶帽操作有突出轮廓较亮部分的作用。闭运算可以将文字（或数字）与背景不同颜色的东西融合，也可以消除白色块（背景是黑色）中的杂质。二值转换并不是所有的阈值都是127，有时候要进行自适应的阈值设定。

* 1. 心得体会

在设计这个程序过程中，也遇到了一些问题。在一开始，我使用opencv的轮廓查找函数来对银行卡卡号进行定位。这时遇到了一个问题，在opencv里用cvFindContours查找轮廓时，有时候无法对一个轮廓完整的进行查找，查找轮廓得到的边界往往不包含整个轮廓。但最终在同学的帮助下得到了解决。

* 1. 优化及改进方法

在以后的研究中，我认为可以从以下几个方面对该系统进行改进。首先，可以加入图像倾斜校正模块，使得样本图像倾斜时可以校正。其次，应该适当扩大数字模板库，使得程序可以识别更多的银行卡类别。最后，通过适当方式增加函数处理图像，使得程序可以识别背景比较复杂的银行卡。

在对不同银行卡进行识别时，由于不同银行卡的数字模版不同，我需要使用不同模版对得到的字符图像进行对比。原来将得到的字符与各种银行卡模版同时使用，最相似的模版对应的数字就是结果。

参考文献

1. 刘振飞.基于OpenCV的银行卡号识别系统[J].电子技术与软件工程,2019(11):63.

[2]姚妮,刘传博,高政源.基于深度学习的银行卡号识别系统[J].信息技术与信息化,2020(10):100-103.

[3]徐毓凯,杨国平.基于Sobel算子的银行卡号识别研究[J].计算机与数字工程,2021,49(08):1672-1675.

[4]邸平.基于卷积神经网络的银行卡数字识别研究[J].电脑与信息技术,2021,29(05):7-10.DOI:10.19414/j.cnki.1005-1228.2021.05.002.

[5]贾树林,郭磊,马双宝,董玉婕,林巍.基于模式匹配的银行卡卡号定位与识别算法[J].武汉纺织大学学报,2021,34(06):60-65.

[6]向德辉.数字图像处理课程的一种自顶向下教学方法[J].数字通信世界,2021(08):255-257+262.

致谢

首先我要感谢陈老师，在他的对图像处理一些操作的讲解后，我受益匪浅，学会了二值化、闭运算等操作。然后我要感谢我们组的所有成员。在设计这个程序过程中，在对不同银行卡进行识别时，由于不同银行卡的数字模版不同，我需要使用不同模版对得到的字符图像进行对比。直到他们一起发现了问题，并在网上查阅了许多资料，让我知道将得到的字符与各种银行卡模版同时使用，最相似的模版对应的数字就是结果，解决了这个问题。