工

程

项

目

设

计

结

题

报

告

项目名称： 从数独图片中提取数字

项目负责人： 韩彬

项目成员： 陈希 周凯琳 陈桓

项目成员： 晏宇权 王英朝 朱雲泽

2021年12月制

摘 要

首先从数独的要求出发建立方程组,该方程组的解与原数独的解完全等价。然后由该方程组推导出一系列数学性质,包括删除候选数性质、唯一确定法性质、矛盾性质和不变性性质。并说明数独的人工推理规则包含在这些性质之中。最后由这些性质提出求解该方程组的算法,算法中用一个三维矩阵来表示待求解九宫数独的候选数矩阵,根据上述性质对候选数矩阵进行删减,直到能够解出此九宫数独。此算法能够求解出许多数独软件无法进行推理计算的数独难题,并用两个数独难题进行验证,说明了该算法的有效性。为了实现数独的快速求解,设计了计算机视觉求解数独系统。该计算机视觉求解数独系统主要由数独图像获取模块、数字矩阵获取模块、数独求解模块、数字矩阵输出模块构成。首先由OV2710摄像头获取矩阵图片的基本信息,之后通过opencv与python相结合对图片进行深入处理,识别数独题目并转化为数学问题,最终使用递归法求解,联组法验证。本系统将计算机视觉与数独求解相结合,具有运算速度快,易操作等优点,可实现对多种实物数独的求解推广。

**关键词：**提取数字，数据预处理，识别数字

ABSTRACT

Firstly, from the requirement of Sudoku, the equations are established, and the solutions of the equations are completely equivalent to those of the original Sudoku. Then, a series of mathematical properties are derived from the equations, including the properties of deleting candidate numbers, unique determination, contradiction and invariance. It also shows that the artificial reasoning rules of Sudoku are included in these properties. Finally, based on these properties, an algorithm for solving the equations is proposed. In the algorithm, a three-dimensional matrix is used to represent the candidate matrix to be solved in Jiugong Sudoku, and the candidate matrix is deleted according to the above properties until the Jiugong Sudoku can be solved. This algorithm can solve many Sudoku puzzles that Sudoku software can&apos;t perform reasoning calculation. Two Sudoku puzzles are used to verify the effectiveness of this algorithm. In order to solve Sudoku quickly, a computer vision system for solving Sudoku is designed. The computer vision Sudoku solution system mainly consists of Sudoku image acquisition module, digital matrix acquisition module, Sudoku solution module and digital matrix output module. First, the OV2710 camera gets the basic information of the matrix picture. Then, through the combination of opencv and python, the picture is deeply processed, the Sudoku problem is identified and transformed into a mathematical problem. Finally, the recursive method is used to solve the problem and the combined method is used to verify it. This system combines computer vision with Sudoku solution, which has the advantages of fast operation speed and easy operation, and can realize the solution and popularization of various kinds of physical Sudoku.

**Key Words:** Extract numbers, Data preprocessing, discriminating digit

目 录

[第1章 引言 1](#_Toc72071965)

[1.1 选题背景 1](#_Toc72071966)

[1.2 研究目标和意义 1](#_Toc72071967)

[1.3 研究思路 1](#_Toc72071968)

[第2章 研究的理论和基础 2](#_Toc72071969)

[2.1 研究方法 2](#_Toc72071970)

[第3章 数据处理 3](#_Toc72071971)

[3.1 提取数字 3](#_Toc72071972)

[3.2数据预处理 4](#_Toc72071973)

3.3训练KNN识别数字 [4](#_Toc72071974)

[第4章效果展示 5](#_Toc72071975)

[4.1代码示例 5](#_Toc72071976)

[4.2效果图展示 6](#_Toc72071977)

[第5章心得体会 6](#_Toc72071978)

[5.1 总结 6](#_Toc72071979)

[参考文献 7](#_Toc72071983)

[致谢 8](#_Toc72071984)

1. 引言
   1. 选题背景

数字图像处理(Digital Image Processing)是通过计算机对图像进行去除噪声、增强、复原、分割、提取特征等处理的方法和技术。本项目采用Python编程语言，结合OpenCV视觉库来解决数独问题。主要工作为用OpenCV的相关函数来获取图形上的数独矩阵，然后设计一定的算法求出数独的解，最后在图片上画出数独的解。数独源于18世纪的瑞士，又称九宫格，有九行、久列和九宫。玩家需要在九宫格中，根据已知的数字，利用逻辑和推理能力，填出所有的空格中应有的数字。填的时候要求每行、每列和每宫都要不重复地包含数字0-9。每行、每列和每宫中1-9都必须出现且只能出现一次，故称之为数独。数独游戏考察的是解题者的观察能力和逻辑推理能力，虽然规则很简单，但是数字的排列方式却是包含千变万化，是一种锻炼思维的绝佳方式。有时候数独不光有数字的变化，还有颜色的变化，更难但趣味也更多。

* 1. 研究目标和意义

由于图像的多义性和复杂性，许多分割的工作无法依靠计算机自动完成，而手工分割又存在工作量大，定位不准确的难题。因此，人们提出了一些人工交互和计算机自动定位相结合的方法，利用各自的优势，实现目标轮廓的快速定位。相信这些交互式方法的应用，必将推动图像目标分割与提取这一既具有广 阔的应用前景又具有重要的学术价值的课题的进一步研究，也必将成为一个更为独立和活跃的研究领域。目标：对图片上的数字进行提取和识别。

* 1. 研究思路

数字提取，就是在一张数独图片中提取出已知的数字。先提取方格，然后提取数字，从小方格中提取数字轮廓，其父轮廓有子轮廓，也即是说包含子轮廓的小方格里面就有数字，最后把检测到的数字画出来就可以得到。

**发展现状：**

快递面单识别：

使用数字识别技术，对快递面单、物流单据、外卖小票中的电话号码进行识别和提取，大幅度提升收货人信息的录入效率，方便进行收件通知，同时可识别纯数字形式的快递三段码，有效提升快件分拣速度。

仪表读数识别：

使用数字识别技术，对各类仪器仪表的读数进行识别和提取，可应用于对仪器仪表读数具有定时记录、数据统计、实时监控等需求的场景，有效降低人工录入成本，控制仪器使用风险

1. 研究的理论和基础
   1. 研究方法

在仔细研究了opencv轮廓提取函数findContours()之后，发现利用轮廓的层级结构会更加简单。

cv2.findContours(image, mode, method[, contours[, hierarchy[, offset] ] ]) → contours, hierarchy

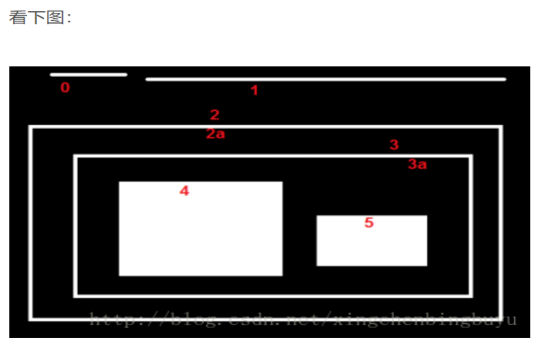
通过研究opencv轮廓提取函数findContours() ，发现利用轮廓的层级结构会更加简单。在Python中，findContours()接受如下参数并返回contours和hierarchy。1.image 源图像，一般为8为单通道图像，更具体来说，二值图像。其他情况暂且不论。

2.mode 轮廓检索模式，简要介绍几种：cv2.RETR\_EXTERNA只检测外轮廓。hierarchy[i][2]=hierarchy[i][3]=-1

cv2.RETR\_LIST 提取所有轮廓，并放置在list中，检测到的轮廓不建立等级关系。cv2.RETR\_TREE 提取所有轮廓，建立网状的轮廓结构。

3.method 轮廓的近似办法，是提取轮廓上所有像素点，还是只提取关键的一些点。比如一条线段是提取所有点还是只提取两个端点。

4.contours 检测到的轮廓，为组成轮廓的点集。

5.hierarchy表达的是不同轮廓之间的关系和联系。hierarchy 什么是层级结构呢？我们检测轮廓的时候，有时候可能会出现其中一个轮廓包含了另外一个轮廓，比如同心圆。这里我们认为外侧轮廓为父轮廓，内侧被包含的为子轮廓。同一级别的又有前一个轮廓和后一个轮廓。总的来说，hierarchy表达的是不同轮廓之间的 关系和联系。这样，每一个轮廓都会有[Next,Previous,First\_Child, Parent]上面说到，cv2.RETR\_EXTERNAL 只检测外轮廓。对所有轮廓设置hierarchy[i][2]=hierarchy[i][3]=-1。由于只检测最外围轮廓，所有检测到轮廓肯定没有父轮廓和子轮廓，所有层级结构的第三个和第四个元素都设置为-1。（如图所示）

如果只检测最外围轮廓，那么只会检测到轮廓0、1和2。

如果建立层级关系，以轮廓3为例，那么它的父轮廓是2a，子轮廓是3a，没有前一轮廓和后一轮廓，设为-1。所以它的hierarchy应该是[-1,-1,3a,2a]

如果是轮廓2，那么它的前一轮廓就是1，子轮廓是2a，没有后一轮廓和父轮廓。所以它的hierarchy应该是[-1,1,2a,-1]

第三章数据处理

3.1提取数字

**提取数字的方法：**

1.遍历文件夹下的原始数字图片；2.对每一张图片进行轮廓提取操作，只提取外围轮廓；

3.求轮廓外包矩形，并根据矩形大小信息筛选出所有的数字轮廓；4.然后根据位置信息对数字框排序，显然第一排依次是12345，第二排依次是67890；5.提取出每一个数字所在的矩形框，作为ROI取出。

**提取数字什么样的轮廓包含数字？**

一般来说经过前面的阈值分割得到二值图像，然后从二值图像中提取的轮廓是这样的。这是处理的比较好的情况下：

显然最最外面的那个包围所有的就是0号轮廓，里面的九九八十一个小方格就是0号轮廓的子轮廓。而每一个已知数字的轮廓都是对应方格的子轮廓。

所以我们的办法就是先提取方格，然后提取数字。由于八十一个小方格父轮廓都是0号轮廓所以：boxes = []for i in range(len(hierarchy[0])):

if hierarchy[0][i][3] == 0:

boxes.append(hierarchy[0][i])

然后从小方格中提取数字轮廓。其父轮廓有子轮廓，也即是说包含子轮廓的小方格里面就有数字。所以：for j in range(len(boxes)):

if boxes[j][2] != -1：

x,y,w,h = cv2.boundingRect(contours[boxes[j][2]])

number\_boxes.append([x,y,w,h])最后把检测到的数字画出来就可以得到这幅图了。

3.2数据预处理（把每一张数字图片ROI转换为二值图像。）

1.把每一张ROI大小统一变换为40 x 20。2.阈值分割。3.把二值图像转换为0-1二值图像。4.把处理完的数字图片保存到对应数字的文件夹中。

5.把处理完的二值图像展开成一行。6.最后把展开成的一行行样本保存起来作为训练用的数据。7.对应的，把数字标签按照数字的保存顺序对应保存成训练用的数据

3.3训练kNN识别数字

（用opencv自带的knn算法实现）1.加载上面保存的样本和标签数据；2.分别用80个作为训练数据，20个作为测试数据；3.用opencv自带的knn训练模型；4.用训练好的模型识别测试数据中的数字；5.输出预测值和实际标签值。

第四章效果展示

**4.1代码示例**

import cv2

img = cv2.imread("C:/Users/86138/Pictures/12.jpeg")

gray = cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY)

## 阈值分割

ret,thresh = cv2.threshold(gray,200,255,1)

## 对二值图像执行膨胀操作

kernel = cv2.getStructuringElement(cv2.MORPH\_CROSS,(5, 5))

dilated = cv2.dilate(thresh,kernel)

## 轮廓提取，cv2.RETR\_TREE表示建立层级结构

image,contours,hierarchy= cv2.findContours(dilated,cv2.RETR\_TREE,cv2.CHAIN\_APPROX\_SIMPLE)

## 提取小方格，其父轮廓都为0号轮廓

boxes = []

for i in range(len(hierarchy[0])):

if hierarchy[0][i][3] == 0:

boxes.append(hierarchy[0][i])

## 提取数字，其父轮廓都存在子轮廓

number\_boxes = []

for j in range(len(boxes)):

if boxes[j][2] != -1:

#number\_boxes.append(boxes[j])

x,y,w,h = cv2.boundingRect(contours[boxes[j][2]])

number\_boxes.append([x,y,w,h])

img = cv2.rectangle(img,(x-1,y-1),(x+w+1,y+h+1),(0,0,255),2)

cv2.namedWindow("img", cv2.WINDOW\_NORMAL);

cv2.imshow("img", img)

cv2.waitKey(0)

img\_path = gb.glob("numbers\\\*")

k = 0

labels = []

samples = []

for path in img\_path:

img = cv2.imread(path)

gray = cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY)

blur = cv2.GaussianBlur(gray,(5,5),0)

thresh = cv2.adaptiveThreshold(blur,255,1,1,11,2)

image, contours, hierarchy = cv2.findContours(thresh,cv2.RETR\_EXTERNAL,cv2.CHAIN\_APPROX\_SIMPLE)

height,width = img.shape[:2]

w = width/5

rect\_list = []

list1 = []

list2 = []

for cnt in contours:

#if cv2.contourArea(cnt)>100:

[x,y,w,h] = cv2.boundingRect(cnt)

if w>30 and h > (height/4):

if y < (height/2):

list1.append([x,y,w,h])

else:

list2.append([x,y,w,h])

list1\_sorted = sorted(list1,key = lambda t : t[0])

list2\_sorted = sorted(list2,key = lambda t : t[0])

for i in range(5):

[x1,y1,w1,h1] = list1\_sorted[i]

[x2,y2,w2,h2] = list2\_sorted[i]

number\_roi1 = gray[y1:y1+h1, x1:x1+w1] #Cut the frame to size

number\_roi2 = gray[y2:y2+h2, x2:x2+w2] #Cut the frame to size

resized\_roi1=cv2.resize(number\_roi1,(20,40))

thresh1 = cv2.adaptiveThreshold(resized\_roi1,255,1,1,11,2)

resized\_roi2=cv2.resize(number\_roi2,(20,40))

thresh2 = cv2.adaptiveThreshold(resized\_roi2,255,1,1,11,2)

number\_path1 = "number\\%s\\%d" % (str(i+1),k) + '.jpg'

j = i+6

if j ==10:

j = 0

number\_path2 = "number\\%s\\%d" % (str(j),k) + '.jpg'

k+=1

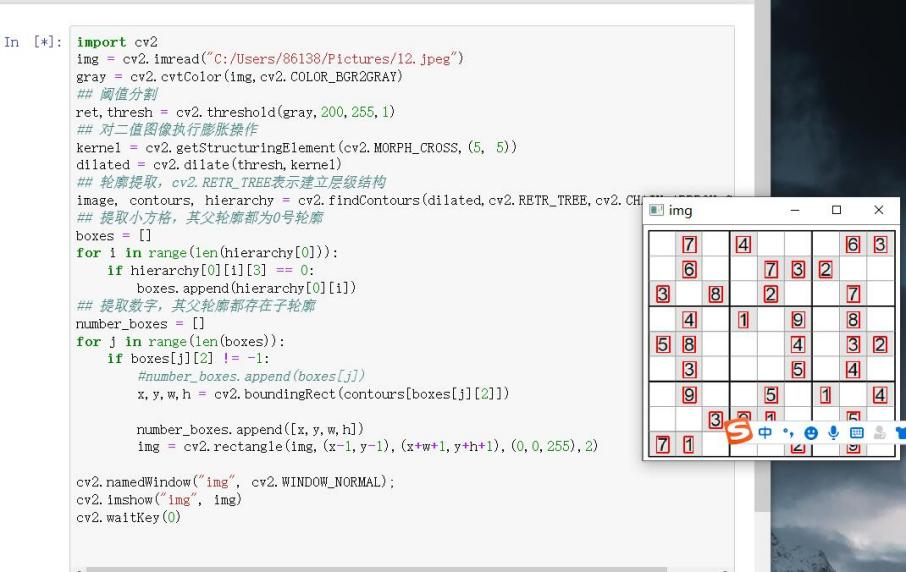
normalized\_roi1 = thresh1/255.

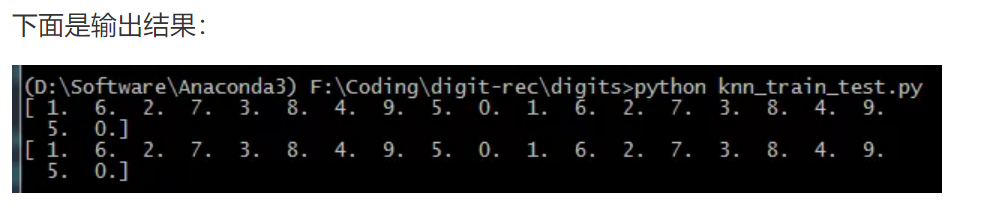
normalized\_roi2 = thresh2/255.

cv2.imwrite(number\_path1,thresh1)

cv2.imwrite(number\_path2,thresh2)

**4.2效果图展示**





1. **心得体会**

**5.1总结**

通过本次实验了解了从数独图片中提取数字实验的意义和目的，在课程中我们了解到数字图像处理里面许多的知识也对opencv有了初步的了解，图像处理是指对图像信息进行加工，从而满足人类的心理、视觉或者应用的需求的一种行为。通过小组的共同学习与交流对本次实验让我们了解到如何从数图中提取数字对什么样的轮廓包含数字的了解中不断学习与改进完成了对本次实验的工作。也让我们在本次的小组学习中每个人都得到对知识的学习理解和成长。

参考文献

[1]孙晨曦,许谨,王海明,张延普.基于计算机视觉求解数独的系统设计[J].中国科技信息,2020(02):76-77.

[2]汤然. 基于图像区块的大容量图像信息隐藏方案研究[D].安徽大学,2018.

[3]郑慧敏. 基于傅里叶描述子和数独的数字图像水印算法研究[D].深圳大学,2016.

[4]晏细兰,谢景明,熊茂华.基于DWT和SUDOKU的多功能数字图像水印算法[J].网络安全技术与应用,2015(01):48+50.

[5]黄祖贤.数独游戏的问题生成及求解算法优化[J].安徽工业大学学报(自然科学版),2015,32(02):187-191.

[6]肖波.数独高效随机生成算法的研究与实现[J].电脑知识与技术,2019,15(09):86-87+89.DOI:10.14004/j.cnki.ckt.2019.0869.

致谢

我要首先感谢小组中每个成员的共同努力，在大家的共同努力学习完成此次任务。感谢大家在过程中的不段努力，让每个人都得道许多的成长。正是由于大家的帮助和支持，我们才能克服一个一个的困难和疑惑，直至本文的顺利完成。也感谢大家提供了许多有价值的信息，让我们吸取许多的经验来改进自己提高自己。经历了这几个月快乐的学习生活，感谢伙伴们的一路陪伴，时间流逝，大家一起变的更努力也让自己不断成长提高，感谢大家在学习过程中相互鼓励相互支持最终得到更好的改变和成长。