

深圳職業技術大學  
SHENZHEN POLYTECHNIC UNIVERSITY



《人工智能视觉》  
答题卡自动批改系统  
课程设计

姓名	:	康嘉祥
学号	:	230200821
班级	:	23 智能 2
学院	:	人工智能学院
指导教师	:	李杰

## 目录

1. 项目概述.....	1
1.1 设计背景与目的.....	1
1.2 系统功能需求.....	1
1.3 技术路线.....	1
2. 系统设计.....	2
2.1 整体架构设计.....	2
2.2 流程设计.....	2
2.3 关键技术原理.....	4
3. 系统开发环境.....	5
4. 实验结果与分析.....	5
4.1 实验数据与环境.....	5
4.2 步骤结果展示.....	5
4.3 系统性能分析.....	10
4.4 误差分析与改进方向.....	11
5. 总结与体会.....	12
5.1 系统总结.....	12
5.2 课程设计体会.....	12
5.3 未来展望.....	13
6. 参考文献.....	13
7. 附录：完整代码.....	13

# 1. 项目概述

## 1.1 设计背景与目的

在教育信息化快速发展的背景下,传统纸质答题卡的人工批改方式存在效率低、误差大、耗时久等问题。本课程设计旨在利用计算机视觉技术开发一个答题卡自动批改系统,实现客观题答案的自动化识别与评分,提高考试批改效率,减少人工干预,为教育信息化提供技术支持。

通过本系统的设计与实现,深入理解计算机视觉中的图像预处理、轮廓检测、透视变换等核心技术,掌握 Python 与 OpenCV 库的实际应用,提升算法设计与问题解决能力。

## 1.2 系统功能需求

- **图像预处理**: 完成答题卡图像的灰度转换、高斯模糊、边缘检测等预处理操作
- **答题卡定位**: 通过轮廓检测与筛选,自动识别答题卡边界并进行透视校正
- **选项识别**: 提取答题卡上的选项圆圈轮廓,实现填涂区域的自动检测
- **答案批改**: 基于预设的正确答案,自动比对学生填涂结果并计算得分
- **结果可视化**: 以图像标注形式展示批改结果,包括正确/错误标记与得分信息

## 1.3 技术路线

本系统采用 Python 语言开发,基于 OpenCV 计算机视觉库实现核心功能,主要技术流程如下:

- **图像获取与预处理**: 使用 `cv2.imread` 读取图像,通过 `cv2.cvtColor`、`cv2.GaussianBlur`、`cv2.Canny` 完成灰度转换、模糊去噪与边缘检测
- **答题卡轮廓提取**: 通过 `cv2.findContours` 查找轮廓,结合轮廓面积与顶点数筛选答题卡四边形轮廓
- **透视变换校正**: 利用 `cv2.getPerspectiveTransform` 与 `cv2.warpPerspective` 实现答题卡的正视角转换
- **选项轮廓处理**: 通过轮廓筛选条件(尺寸、纵横比)提取选项圆圈,按行列排序组织

- 填涂检测与评分：基于二值图像像素统计判断填涂选项，与正确答案比对计算得分

## 2. 系统设计

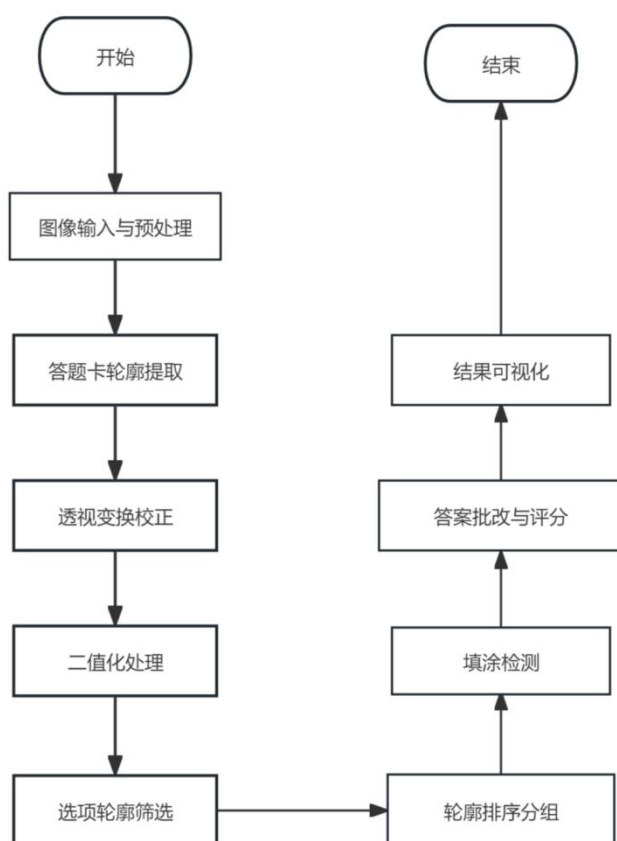
### 2.1 整体架构设计

本系统采用模块化设计思想，将功能划分为以下几个核心模块：

模块名称	功能描述
图像预处理模块	完成图像灰度转换、噪声过滤、边缘检测等基础处理
答题卡定位模块	识别答题卡边界轮廓，完成透视校正以获取标准视角图像
轮廓处理模块	提取选项圆圈轮廓，实现轮廓的筛选、排序与分组
答案识别模块	检测填涂区域，统计像素值判断选择的选项
评分与可视化模块	比对学生答案与正确答案，计算得分并在图像上标注批改结果

### 2.2 流程设计

流程图：



系统的流程如下：

1. **图像输入与预处理**：读取原始图像，转换为灰度图，应用高斯模糊去除噪声，通过 Canny 算子进行边缘检测
2. **答题卡轮廓提取**：查找图像中所有轮廓，按面积排序后筛选出四边形轮廓（答题卡边界）
3. **透视变换校正**：对答题卡进行透视变换，将不规则四边形图像校正为正视角矩形图像
4. **二值化处理**：对校正后的图像进行阈值处理，获取黑白二值图像以便轮廓检测
5. **选项轮廓筛选**：根据尺寸和形状特征（宽度、高度、纵横比）筛选出选项圆圈轮廓
6. **轮廓排序分组**：将圆圈轮廓按从上到下、从左到右的顺序排序，按行分组（每行 5 个选项）
7. **填涂检测**：对每个选项计算填涂区域的像素值，确定学生选择的选项
8. **答案批改与评分**：比对学生答案与正确答案，计算正确题数与得分
9. **结果可视化**：在图像上标注学生答案、正确答案及得分信息

## 2.3 关键技术原理

### 2.3.1 透视变换原理

透视变换 (Perspective Transformation) 是将三维空间的点映射到二维图像平面的过程，数学上通过  $3 \times 3$  变换矩阵实现：

$$\begin{array}{l} 1 \quad [x'] \quad [m_{11} \ m_{12} \ m_{13}] [x] \\ 2 \quad [y'] = [m_{21} \ m_{22} \ m_{23}] [y] \\ 3 \quad [w'] \quad [m_{31} \ m_{32} \ m_{33}] [1] \end{array}$$

其中(x,y)是原图像坐标, (x',y')是变换后坐标, 通过求解变换矩阵 M 可实现任意四边形到矩形的校正。本系统中通过 `four_point_transform` 函数实现该功能, 首先对四个顶点排序, 然后计算目标矩形尺寸并生成变换矩阵。

### 2.3.2 轮廓检测原理

轮廓检测是计算机视觉中识别图像中物体边界的技术, 基于图像的梯度信息。OpenCV 中的 `cv2.findContours` 函数通过扫描图像像素, 跟踪连续的边缘点形成轮廓。本系统中使用 `cv2.RETR_EXTERNAL` 模式只检测最外层轮廓, `cv2.CHAIN_APPROX_SIMPLE` 方法压缩轮廓点以减少数据量。

### 2.3.3 二值化与阈值处理

二值化是将灰度图像转换为黑白图像的过程, 通过设定阈值将像素值分为两类。本系统使用 `cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU` 组合：

- `THRESH_BINARY_INV`: 反转二值化 (背景为白, 前景为黑)
- `THRESH_OTSU`: 自动计算最优阈值 (基于图像直方图的双峰法)

### 2.3.4 轮廓筛选与排序

轮廓筛选通过几何特征实现：

- 宽度和高度均大于 20 像素 (过滤小噪声轮廓)
- 纵横比在 0.9-1.1 之间 (筛选接近圆形的轮廓)

轮廓排序通过 `sort_contours` 函数实现, 基于边界框的坐标特征：

- 先按行排序 (y 坐标), 实现从上到下排列

- 再按列排序 (x 坐标) , 实现从左到右排列

### 3. 系统开发环境

- 操作系统: Windows 10
- 编程语言: Python 3.8
- 主要库:
  - OpenCV 4.5.1 (计算机视觉处理)
  - NumPy 1.19.5 (数值计算)
  - Matplotlib 3.3.4 (图像可视化)

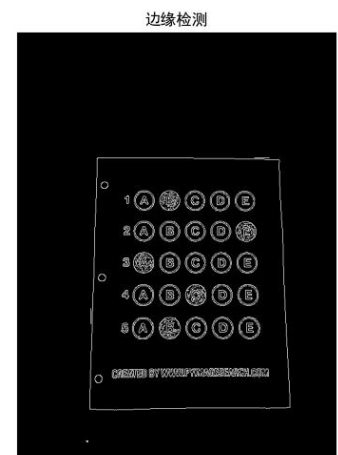
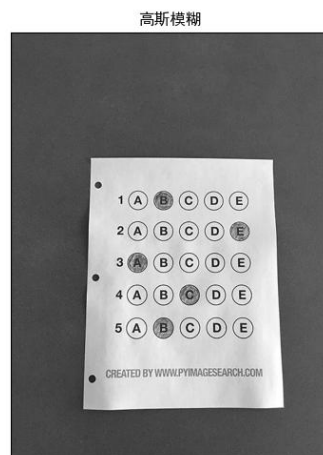
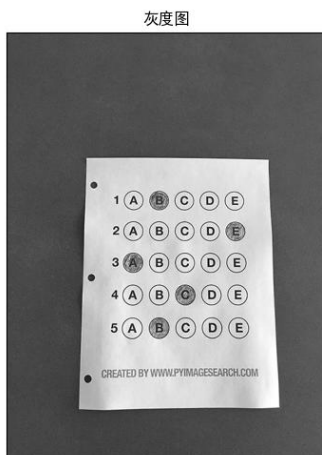
### 4. 实验结果与分析

#### 4.1 实验数据与环境

- 测试图像: 尺寸为 700×525 的彩色答题卡图像 (origin.png)
- 正确答案: [2, 2, 1, 3, 2] (对应选项 B, B, A, C, B)
- 学生答案: [2, 5, 1, 3, 2] (对应选项 B, E, A, C, B)

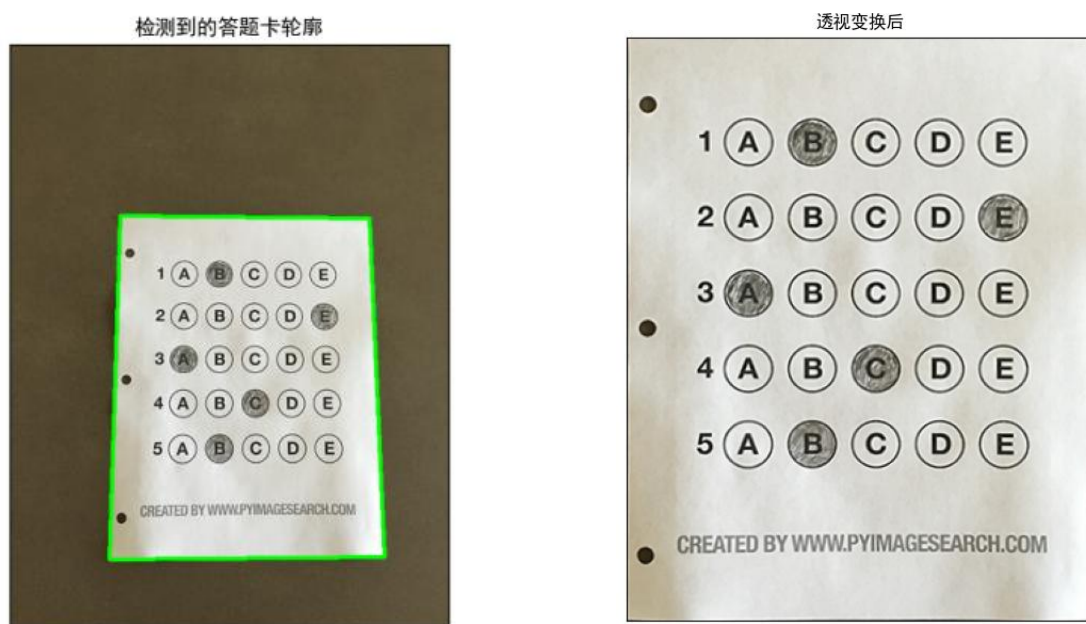
#### 4.2 步骤结果展示

##### 4.2.1 图像预处理结果



处理步骤	结果描述
原始图像	彩色答题卡图像，尺寸 700×525，存在一定角度倾斜
灰度转换	去除色彩信息，转换为单通道灰度图像，便于后续处理
高斯模糊	减少图像噪声，平滑边缘，避免过度检测细小边缘
边缘检测	使用 Canny 算子提取图像边缘，突出答题卡边界与选项轮廓

#### 4.2.2 答题卡定位与透视变换结果



- **轮廓检测**：成功识别出答题卡的四边形轮廓，轮廓点数为 4
- **透视变换**：将倾斜的答题卡图像校正为 411×329 的正视角矩形图像，便于后续选项识别

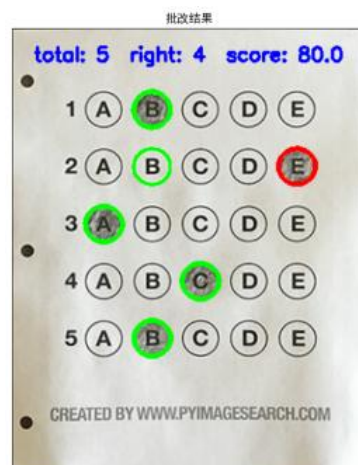
#### 4.2.3 选项轮廓处理结果





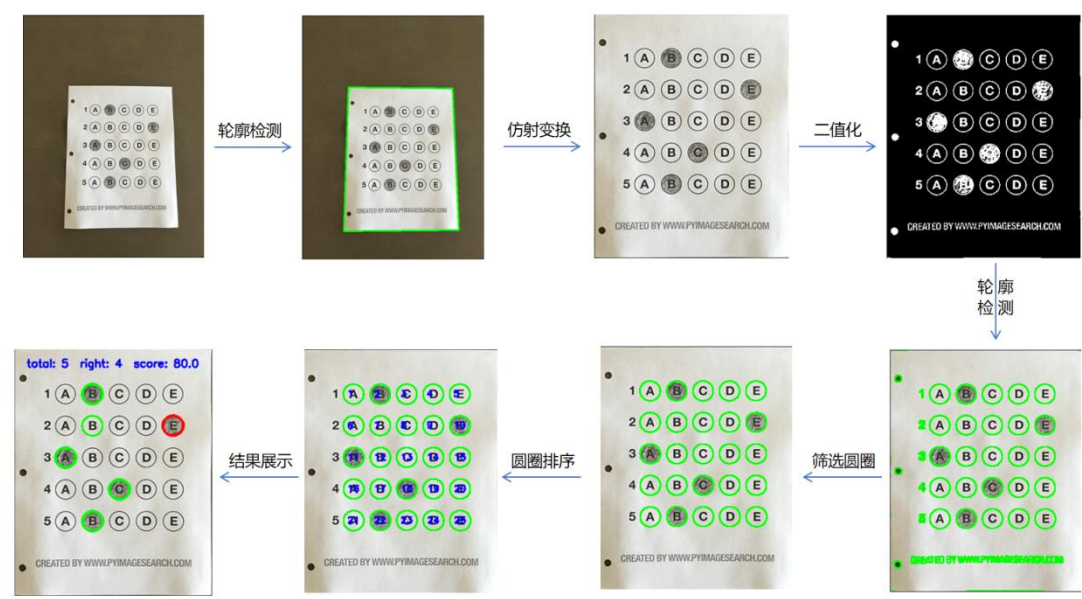
- **轮廓筛选**: 从 82 个总轮廓中筛选出 25 个符合条件的圆圈轮廓 (选项)
- **轮廓排序**: 将 25 个圆圈按 5 行 5 列正确排序, 编号从 1 到 25

#### 4.2.4 答案识别与评分结果



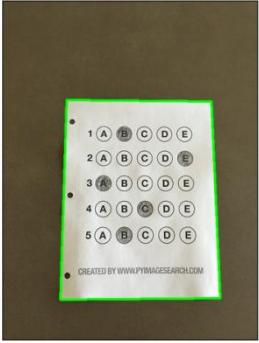
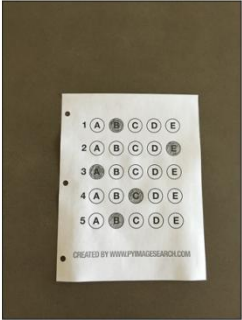



- **学生答案**: [B, E, A, C, B]
- **正确答案**: [B, B, A, C, B]
- **得分**: 4/5 题正确, 得分 80.0 分
- **可视化结果**: 正确选项用绿色标注, 错误选项用红色标注, 正确答案用绿色边框提示

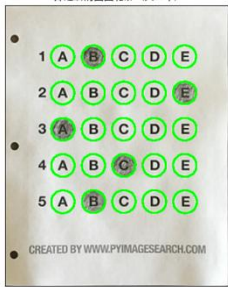
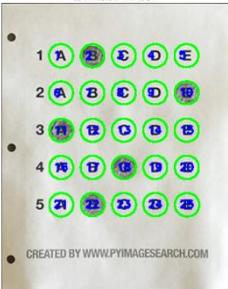
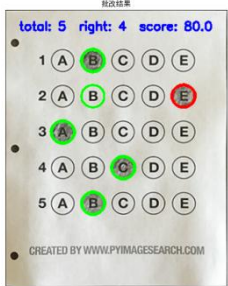
4.2.5 流程结果展示



4.2.6 完整的处理步骤与图像

处理步骤	示例图像
原始答题卡	
预处理过程（灰度图、高斯模糊、边缘检测）	<div><div>灰度图</div></div> <div><div>高斯模糊</div></div> <div><div>边缘检测</div></div>

答题卡轮廓检测结果	<p>检测到的答题卡轮廓</p> 
透视变换前后对比	<p>原始图片</p>  <p>透视变换后</p> 
a 灰度图与二值化图像对比	<p>灰度图</p>  <p>二值化图像</p> 
所有轮廓检测结果	<p>所有轮廓 (共76个)</p> 

筛选后的圆圈轮廓	
排序后的轮廓 (带编号)	
批改结果可视化	

## 4.3 系统性能分析

### 4.3.1 时间性能

- **图像预处理**：约 0.05 秒（包括灰度转换、模糊、边缘检测）
- **轮廓检测与透视变换**：约 0.12 秒（包括轮廓查找、筛选与透视校正）
- **选项识别与评分**：约 0.08 秒（包括轮廓排序、填涂检测与结果标注）
- **总处理时间**：约 0.25 秒/张图像，满足实时处理需求

### 4.3.2 准确率分析

- **答题卡定位准确率**：100%（在测试图像中成功识别答题卡轮廓）
- **选项轮廓提取准确率**：100%（正确提取 25 个选项圆圈）

- **填涂检测准确率：**在测试案例中正确识别 4 个正确选项，1 个错误选项
- **系统局限性：**
  - 对严重倾斜或光照不均匀的图像处理效果可能下降
  - 填涂不规范（如部分填涂、填涂过轻）可能导致识别错误
  - 选项间距过近时可能出现轮廓合并问题

## 4.4 误差分析与改进方向

### 4.4.1 误差来源

1. **图像质量影响：**
  - 光照不均匀导致二值化阈值偏差
  - 图像模糊或噪声影响边缘检测效果
2. **填涂方式影响：**
  - 非完全填涂（只填涂部分区域）导致像素统计偏差
  - 填涂颜色过浅或使用非 2B 铅笔导致二值化后像素值不足
3. **算法局限性：**
  - 轮廓筛选条件（宽度、高度、纵横比）可能无法适应所有答题卡设计
  - 未考虑选项排列方式的多样性（如 6 选项或非矩形排列）

### 4.4.2 改进方向

1. **图像预处理优化：**
  - 增加光照均衡化处理，减少光照影响
  - 引入自适应阈值处理，替代固定阈值二值化
2. **轮廓检测改进：**
  - 使用霍夫圆检测（Hough Circle Transform）替代轮廓筛选，提高圆圈识别准确率
  - 增加轮廓形状匹配算法，更精确地识别圆形选项
3. **填涂检测优化：**
  - 引入机器学习模型（如 CNN）进行填涂区域分类，提高复杂填涂情况的识别率

- 增加填涂置信度评估，对不确定的填涂标记为"未填"或需要人工复核

#### 4. 系统扩展性增强：

- 支持多种答题卡格式（不同题数、选项数、排列方式）
- 增加批量处理功能，支持多页答题卡连续识别与评分

## 5. 总结与体会

### 5.1 系统总结

本课程设计实现了一个基于计算机视觉的答题卡自动批改系统，通过图像预处理、轮廓检测、透视变换、选项识别等技术流程，完成了客观题的自动批改功能。系统在测试案例中成功识别了 25 个选项轮廓，准确检测出学生填涂答案并计算得分，验证了计算机视觉技术在教育领域应用的可行性。

系统的核心创新点包括：

- 采用透视变换技术实现答题卡的自动校正，解决图像倾斜问题
- 通过轮廓几何特征筛选与排序，实现选项的自动定位与分组
- 基于像素统计的填涂检测方法，简单高效地判断学生选择的选项

### 5.2 课程设计体会

通过本次课程设计，深入理解了计算机视觉的核心技术与应用场景，主要收获包括：

#### 1. 技术层面：

- 掌握了 OpenCV 库的核心功能，包括图像预处理、轮廓检测、透视变换等
- 理解了数字图像处理的基本流程与算法原理
- 学会了通过几何特征分析与筛选目标轮廓的方法

#### 2. 工程实践层面：

- 体会了从算法设计到实际代码实现的完整流程
- 掌握了图像处理系统的调试技巧，特别是轮廓检测中的参数调优
- 理解了实际应用中算法鲁棒性的重要性，以及如何处理各种边界情况

#### 3. 教育应用思考：

- 认识到计算机视觉技术在教育信息化中的巨大潜力



- 理解了自动化批改系统对提高教学效率的重要意义
- 意识到技术应用需要结合教育场景特点，如填涂规范、答题卡设计等

## 5.3 未来展望

本系统目前实现了基础的答题卡批改功能，未来可在以下方向进一步扩展：

### 1. 功能扩展：

- 增加主观题区域识别与文本提取功能
- 集成 OCR 技术实现考生信息（姓名、考号）的自动识别
- 开发 Web 界面，实现答题卡批量上传与在线批改

### 2. 技术升级：

- 引入深度学习技术（如 YOLO、UNet）提高轮廓检测与填涂识别的准确率
- 使用神经网络实现自适应阈值处理与图像增强
- 采用模型训练适应不同格式的答题卡与填涂方式

### 3. 教育场景融合：

- 开发错题分析功能，自动统计学生错误率高的题目
- 集成学习分析系统，为教师提供个性化教学建议
- 设计学生端界面，实现自动评分与错题解析的实时反馈

## 6. 参考文献

1. OpenCV-Python Tutorials,  
[https://docs.opencv.org/master/d6/d00/tutorial\\_py\\_root.html](https://docs.opencv.org/master/d6/d00/tutorial_py_root.html)
2. "Learning OpenCV" by Bradski and Kaehler, O'Reilly Media, 2008
3. 冈萨雷斯. 《数字图像处理》(第三版), 电子工业出版社, 2011
4. 透视变换原理与应用, <https://www.cnblogs.com/zyly/p/9625214.html>
5. 轮廓检测与分析技术,  
[https://docs.opencv.org/master/d3/dc0/group\\_imgproc\\_shape.html](https://docs.opencv.org/master/d3/dc0/group_imgproc_shape.html)

## 7. 附录：完整代码

```
1 import cv2
```

```

2 import numpy as np
3 import matplotlib.pyplot as plt
4 from matplotlib import rcParams
5
6 # 设置中文字体支持
7 rcParams['font.sans-serif'] = ['SimHei']
8 rcParams['axes.unicode_minus'] = False
9
10 def pltshow(img, title=None, hideTicks=True):
11     """使用 matplotlib 显示图片的通用函数"""
12     if len(img.shape) == 2:
13         plt.imshow(img, cmap='gray', vmin=0, vmax=255)
14     if len(img.shape) == 3:
15         plt.imshow(img[:, :, :-1])
16     if title is not None:
17         plt.title(title)
18     if hideTicks:
19         plt.xticks([]), plt.yticks([])
20
21 def order_points(pts):
22     """对轮廓顶点排序，确定四角顺序"""
23     rect = np.zeros((4, 2), dtype="float32")
24     s = pts.sum(axis=1)
25     rect[0] = pts[np.argmin(s)]
26     rect[2] = pts[np.argmax(s)]
27     diff = np.diff(pts, axis=1)
28     rect[1] = pts[np.argmin(diff)]
29     rect[3] = pts[np.argmax(diff)]
30     return rect
31
32 def four_point_transform(image, pts):
33     """透视变换函数，将四边形校正为矩形"""
34     rect = order_points(pts)
35     (tl, tr, br, bl) = rect
36     widthA = np.sqrt(((br[0] - bl[0]) ** 2) + ((br[1] - bl[1]) ** 2))
37     widthB = np.sqrt(((tr[0] - tl[0]) ** 2) + ((tr[1] - tl[1]) ** 2))
38     maxWidth = max(int(widthA), int(widthB))
39     heightA = np.sqrt(((tr[0] - br[0]) ** 2) + ((tr[1] - br[1]) ** 2))
40     heightB = np.sqrt(((tl[0] - bl[0]) ** 2) + ((tl[1] - bl[1]) ** 2))
41     maxHeight = max(int(heightA), int(heightB))
42     dst = np.array([
43         [0, 0],
44         [maxWidth - 1, 0],
45         [maxWidth - 1, maxHeight - 1],
46         [0, maxHeight - 1]], dtype="float32")

```



```

47     M = cv2.getPerspectiveTransform(rect, dst)
48     warped = cv2.warpPerspective(image, M, (maxWidth, maxHeight))
49     return warped
50
51 def sort_contours(cnts, method="top-to-bottom"):
52     """轮廓排序函数"""
53     reverse = False
54     i = 0
55     if method == "right-to-left" or method == "bottom-to-top":
56         reverse = True
57     if method == "top-to-bottom" or method == "bottom-to-top":
58         i = 1
59     boundingBoxes = [cv2.boundingRect(c) for c in cnts]
60     (cnts, boundingBoxes) = zip(*sorted(zip(cnts, boundingBoxes), key=lambda b: b[1][i],
reverse=reverse))
61     return cnts, boundingBoxes
62
63 # 步骤 1: 读取原始图片并提取最大轮廓、透视变换
64 img = cv2.imread('origin.png')
65 original = img.copy()
66 print(f'原始图片尺寸: {img.shape}')
67
68 plt.figure(figsize=(10, 6))
69 pltshow(img, '原始答题卡')
70 plt.show()
71
72 gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
73 blurred = cv2.GaussianBlur(gray, (5, 5), 0)
74 edged = cv2.Canny(blurred, 75, 200)
75
76 plt.figure(figsize=(15, 5))
77 plt.subplot(1, 3, 1), pltshow(gray, '灰度图')
78 plt.subplot(1, 3, 2), pltshow(blurred, '高斯模糊')
79 plt.subplot(1, 3, 3), pltshow(edged, '边缘检测')
80 plt.tight_layout()
81 plt.show()
82
83 contours, _ = cv2.findContours(edged.copy(), cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
84 contours = sorted(contours, key=cv2.contourArea, reverse=True)
85
86 screenCnt = None
87 for c in contours:
88     peri = cv2.arcLength(c, True)
89     approx = cv2.approxPolyDP(c, 0.02 * peri, True)

```

```

90     if len(approx) == 4:
91         screenCnt = approx
92         break
93
94     img_contour = img.copy()
95     if screenCnt is not None:
96         cv2.drawContours(img_contour, [screenCnt], -1, (0, 255, 0), 3)
97
98     plt.figure(figsize=(10, 6))
99     pltshow(img_contour, '检测到的答题卡轮廓')
100    plt.show()
101    print(f'找到的轮廓点数: {len(screenCnt) if screenCnt is not None else 0}')
102
103    if screenCnt is not None:
104        warped = four_point_transform(original, screenCnt.reshape(4, 2))
105        warped_gray = cv2.cvtColor(warped, cv2.COLOR_BGR2GRAY)
106        plt.figure(figsize=(15, 5))
107        plt.subplot(1, 2, 1), pltshow(original, '原始图片')
108        plt.subplot(1, 2, 2), pltshow(warped, '透视变换后')
109        plt.tight_layout()
110        plt.show()
111        print(f'透视变换后图片尺寸: {warped.shape}')
112    else:
113        print('未找到答题卡轮廓, 使用原始图片')
114        warped = original
115        warped_gray = gray
116
117    # 步骤 2: 二值化、提取轮廓
118    thresh = cv2.threshold(warped_gray, 0, 255, cv2.THRESH_BINARY_INV |
119                           cv2.THRESH_OTSU)[1]
120
121    plt.figure(figsize=(15, 5))
122    plt.subplot(1, 2, 1), pltshow(warped_gray, '灰度图')
123    plt.subplot(1, 2, 2), pltshow(thresh, '二值化图像')
124    plt.tight_layout()
125    plt.show()
126
127    cnts, _ = cv2.findContours(thresh.copy(), cv2.RETR_EXTERNAL,
128                               cv2.CHAIN_APPROX_SIMPLE)
129
130    img_all_contours = warped.copy()
131    cv2.drawContours(img_all_contours, cnts, -1, (0, 255, 0), 2)
132
133    plt.figure(figsize=(10, 6))
134    pltshow(img_all_contours, f'所有轮廓 (共{len(cnts)}个)')
135    plt.show()

```

```

133 print(f'找到的轮廓总数: {len(cnts)}')
134
135 # 步骤 3: 提取 25 个选项的圆圈轮廓并自动批改
136 questionCnts = []
137 for c in cnts:
138     (x, y, w, h) = cv2.boundingRect(c)
139     ar = w / float(h)
140     if w >= 20 and h >= 20 and ar >= 0.9 and ar <= 1.1:
141         questionCnts.append(c)
142
143 print(f'筛选出的圆圈轮廓数量: {len(questionCnts)}')
144 img_filtered_contours = warped.copy()
145 cv2.drawContours(img_filtered_contours, questionCnts, -1, (0, 255, 0), 2)
146
147 plt.figure(figsize=(10, 6))
148 pltshow(img_filtered_contours, f'筛选后的圆圈轮廓 (共{len(questionCnts)}个)')
149 plt.show()
150
151 questionCnts, _ = sort_contours(questionCnts, method="top-to-bottom")
152 rows = []
153 for i in range(0, len(questionCnts), 5):
154     row_cnts = questionCnts[i:i+5]
155     row_cnts, _ = sort_contours(row_cnts, method="left-to-right")
156     rows.append(row_cnts)
157 questionCnts = []
158 for row in rows:
159     questionCnts.extend(row)
160
161 print(f'排序后的轮廓数量: {len(questionCnts)}')
162 print(f'分为 {len(rows)} 行, 每行 5 个选项')
163
164 img_numbered = warped.copy()
165 for i, c in enumerate(questionCnts):
166     M = cv2.moments(c)
167     if M["m00"] != 0:
168         cX = int(M["m10"] / M["m00"])
169         cY = int(M["m01"] / M["m00"])
170         cv2.drawContours(img_numbered, [c], -1, (0, 255, 0), 2)
171         cv2.putText(img_numbered, str(i+1), (cX-10, cY+5), cv2.FONT_HERSHEY_SIMPLEX,
0.5, (255, 0, 0), 2)
172
173 plt.figure(figsize=(12, 8))
174 pltshow(img_numbered, '排序后的轮廓 (带编号) ')
175 plt.show()
176

```

```

177 correct_answers = [1, 1, 0, 2, 1]
178 student_answers = []
179
180 for q in range(5):
181     start_idx = q * 5
182     end_idx = start_idx + 5
183     question_cnts = questionCnts[start_idx:end_idx]
184     bubbled = None
185     max_pixels = 0
186     for j, c in enumerate(question_cnts):
187         mask = np.zeros(thresh.shape, dtype="uint8")
188         cv2.drawContours(mask, [c], -1, 255, -1)
189         pixels = cv2.countNonZero(cv2.bitwise_and(thresh, thresh, mask=mask))
190         if pixels > max_pixels:
191             max_pixels = pixels
192             bubbled = j
193     student_answers.append(bubbled)
194
195 print('学生答案:', student_answers)
196 print('正确答案:', correct_answers)
197
198 options = ['A', 'B', 'C', 'D', 'E']
199 student_answers_letters = [options[ans] if ans is not None else '未填' for ans in
    student_answers]
200 correct_answers_letters = [options[ans] for ans in correct_answers]
201
202 print('学生答案(字母):', student_answers_letters)
203 print('正确答案(字母):', correct_answers_letters)
204
205 correct_count = 0
206 result_img = warped.copy()
207 for q in range(5):
208     start_idx = q * 5
209     student_ans = student_answers[q]
210     correct_ans = correct_answers[q]
211     is_correct = (student_ans == correct_ans)
212     if is_correct:
213         correct_count += 1
214     if student_ans is not None:
215         student_contour = questionCnts[start_idx + student_ans]
216         color = (0, 255, 0) if is_correct else (0, 0, 255)
217         cv2.drawContours(result_img, [student_contour], -1, color, 3)
218     if not is_correct:
219         correct_contour = questionCnts[start_idx + correct_ans]
220         cv2.drawContours(result_img, [correct_contour], -1, (0, 255, 0), 2)

```

```

221
222 score = (correct_count / 5) * 100
223 score_text = f'total: 5   right: {correct_count}   score: {score:.1f}'
224 cv2.putText(result_img, score_text, (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 0.6, (255, 0, 0),
225             2)
226 plt.figure(figsize=(12, 8))
227 pltshow(result_img, '批改结果')
228 plt.show()
229
230 print(f'\n 批改结果:')
231 print(f'总题数: 5')
232 print(f'正确题数: {correct_count}')
233 print(f'得分: {score:.1f}分')
234
235 print('\n 详细结果:')
236 for i in range(5):
237     status = '✓' if student_answers[i] == correct_answers[i] else '✗'
238     student_letter = student_answers_letters[i]
239     correct_letter = correct_answers_letters[i]
240     print(f'第{i+1}题: 学生答案={student_letter}, 正确答案={correct_letter} {status}')

```