实验一 图像滤波

1. 实验目的

掌握图像处理中滤波、直方图计算和纹理特征提取的基本原理和实现方法。

1. 实验原理

1. 卷积运算

卷积是图像滤波的基础操作，通过在图像上滑动卷积核（kernel），对局部像素进行加权求和，实现平滑、锐化、边缘检测等效果。

2. Sobel算子

用于图像边缘检测，包含水平（Gx）和垂直（Gy）两个方向的卷积核。梯度幅值计算：G=√(Gx2+Gy2)

3. 颜色直方图

统计图像中各像素值出现的频率，用于描述图像的颜色分布特征。可分别对R、G、B三个通道计算，或对灰度图像计算单通道直方图。

4. 纹理特征提取（LBP）

局部二值模式（Local Binary Pattern，LBP）通过比较中心像素与邻域像素的灰度值，生成二进制模式，用于描述局部纹理。对每个像素的8邻域进行二值化，生成一个8位二进制数，再转换为十进制作为该像素的LBP值。

1. 实验方法

1.

实验环境Python 3.9

主要库：numpy、matplotlib、PIL

操作系统：Windows

2. 实验步骤

1.读取图像，转换为灰度图（使用自定义函数\_rgb\_to\_gray）。

2.实现卷积函数apply\_convolution，支持单通道与多通道卷积。

3.应用Sobel算子进行边缘检测。

4.使用自定义卷积核[1, 0, -1; 2, 0, -2; 1, 0, -1]进行滤波。

5.计算颜色直方图（RGB三通道或灰度单通道）。

6.提取LBP纹理特征，并计算其直方图作为特征向量。

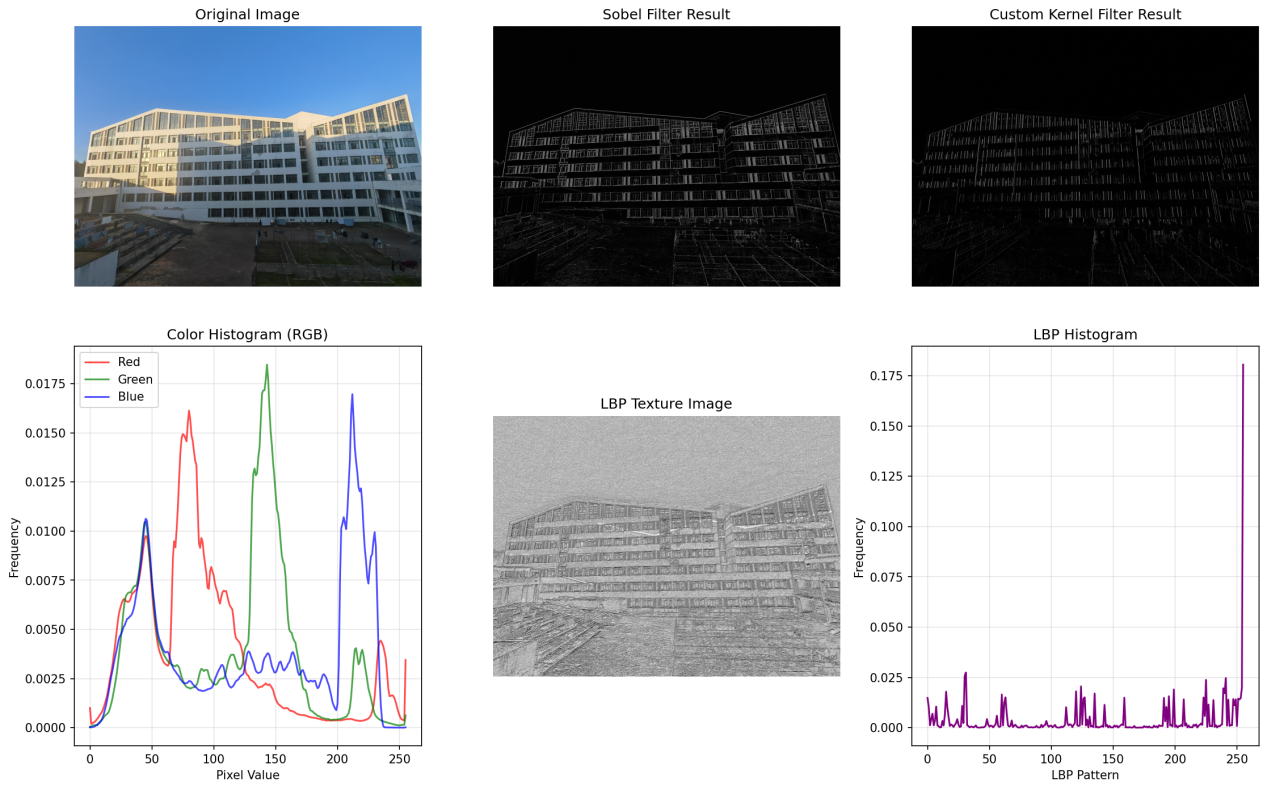
7.保存处理结果，并进行可视化展示。

1. 实验结果

输入图片：



输出结果：



1. 实验体会

通过本次实验，我深入理解了图像处理中滤波、直方图与纹理特征提取的基本原理与实现方法。在自主实现卷积、Sobel算子和LBP算法的过程中，我进一步认识到：卷积核设计直接影响滤波效果，不同卷积核适用于不同任务（如边缘检测、平滑等）。直方图是图像全局特征的直观体现，可用于图像检索、颜色校正等任务。LBP等纹理特征具有较强的局部描述能力，适用于纹理分类、人脸识别等场景。

本次实验不仅锻炼了我独立实现图像处理算法的能力，也增强了我对图像特征的理解与应用能力，为后续的图像分析与计算机视觉研究打下了坚实基础。

实验二 车道线检测

1. 实验目的

掌握霍夫变换在直线检测中的基本原理和应用方法，实现一个完整的车道线检测系统。

1. 实验原理

1. 霍夫变换原理

霍夫变换是一种用于检测图像中几何形状（如直线、圆等）的特征提取方法。对于直线检测，将笛卡尔坐标系中的点(x,y)映射到极坐标(ρ,θ)空间，变换公式：ρ = x·cosθ + y·sinθ。在极坐标空间中，共线的点会相交于同一点(ρ,θ)。

1. 实验方法

车道线检测流程：

输入图像→灰度化→高斯模糊→Canny边缘检测→ROI提取→霍夫变换→直线筛选→车道线绘制

1. 图像预处理

# 读取并转换图像

img = cv2.imread(image\_path)

gray = cv2.cvtColor(img, cv2.COLOR\_RGB2GRAY)

# 高斯模糊去噪

blur = cv2.GaussianBlur(gray, (7, 7), 0)

1. 边缘检测

# Canny边缘检测

edges = cv2.Canny(blur, 70, 150)

1. ROI区域提取

# 定义梯形ROI区域

roi\_vertices = np.array([[

(width \* 0.1, height), # 左下角

(width \* 0.45, height \* 0.6), # 左上角

(width \* 0.55, height \* 0.6), # 右上角

(width \* 0.9, height) # 右下角

]], dtype=np.int32)

1. 霍夫变换直线检测

lines = cv2.HoughLinesP(

roi\_edges,

rho=2,

theta=np.pi/180,

threshold=50,

minLineLength=50,

maxLineGap=150

)

1. 车道线分离与拟合

# 根据斜率区分左右车道线

if slope < 0:

left\_lines.append((x1, y1, x2, y2))

else:

right\_lines.append((x1, y1, x2, y2))

# 最小二乘法拟合直线

left\_fit = np.polyfit(left\_points[:, 1], left\_points[:, 0], 1)

1. 绘制车道线

# 绘制左右车道线

cv2.line(img\_copy, (x1, y1), (x2, y2), (0, 255, 0), thickness)

1. 实验结果

输入图片：



输出结果：



算法分析：

1. 优点

鲁棒性较好：通过ROI区域限制，减少了环境干扰

实时性较高：算法复杂度适中，适合实时应用

适应性较强：参数可调，适应不同光照和道路条件

2. 局限性

依赖清晰的车道线：对模糊或磨损的车道线检测效果下降

光照敏感：强烈逆光或夜间条件下性能受限

弯道处理不足：霍夫变换主要检测直线，对弯道适应性差

参数敏感：需要根据具体场景调整阈值参数

1. 实验体会

通过本次实验，我深入理解了霍夫变换在车道线检测中的应用，获得了以下体会：

理论与实践结合的重要性：将课堂上学到的霍夫变换原理应用于实际问题，加深了对算法的理解

参数调优的关键性：实验中发现，Canny阈值、霍夫变换参数等对检测效果影响显著，需要反复调试

预处理的重要性：图像预处理（如高斯模糊、ROI提取）对后续检测效果有决定性影响

实际应用的挑战：真实场景中的车道线检测比理论更复杂，需要考虑光照、阴影、遮挡等因素

本次实验不仅让我掌握了车道线检测的具体实现方法，也让我认识到计算机视觉在实际应用中的挑战与乐趣。未来可以进一步探索更先进的算法，如基于深度学习的车道线检测，以应对更复杂的道路环境。

实验三 学号识别

1. 实验目的

掌握手写数字识别的基本原理和方法，理解其在机器视觉中的基础地位，学习卷积神经网络（CNN）的设计和实现，掌握深度学习模型的构建流程，实现完整的数字识别系统，包括模型训练、图像预处理、数字分割和识别，应用训练好的模型识别实际拍摄的学号照片，完成学号自动识别任务。

1. 实验原理

1. 卷积神经网络（CNN）基本原理

卷积层：通过卷积核提取局部特征，具有平移不变性

池化层：降低特征图维度，增强特征鲁棒性

激活函数：引入非线性，增强模型表达能力（ReLU）

批归一化：加速训练收敛，提高模型稳定性

全连接层：整合特征，输出分类结果

1. 实验方法

1.手写数字识别流程：

学号图片→预处理→数字分割→单个数字标准化→CNN模型预测→学号输出

1. 环境创建与配置过程：

我是用conda管理环境

# 1. 创建虚拟环境

conda create -n ycg python=3.8

conda activate ycg

# 2. 安装核心依赖包

pip install torch torchvision torchaudio

pip install opencv-python pillow numpy matplotlib

pip install scikit-learn

3.算法实现:

3.1数字识别器（DigitRecognizer类）

(1)负责MNIST数据集加载、模型训练和评估

(2)支持模型保存和加载

(3)包含训练过程可视化

3.2 学号识别器（StudentIDRecognizer类）

(1)学号图片预处理（二值化、降噪）

(2)数字分割（轮廓检测、边界框提取）

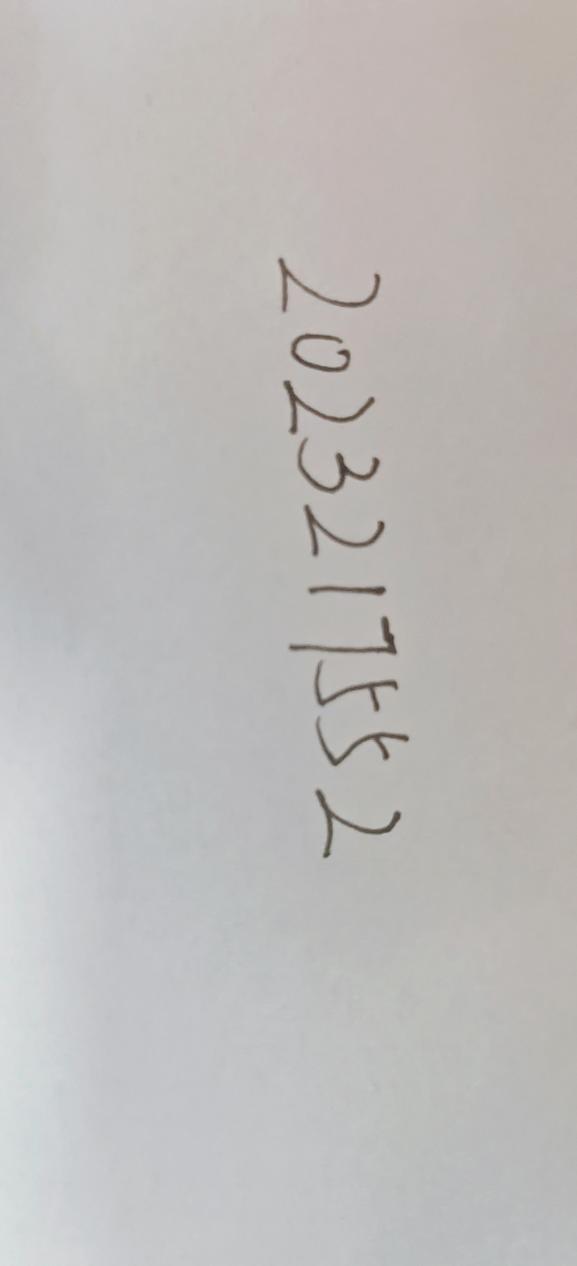
(3)单个数字标准化（调整到28×28像素）

(4)使用预训练模型进行识别

(5)结果可视化和保存

1. 实验结果

输入图片：



输出结果：





算法分析：

1. 三层卷积：逐步提取从简单到复杂的特征

2. 批归一化：加速收敛，提高训练稳定性

3. Dropout层：防止过拟合，提高泛化能力

4. 全连接层：整合特征，输出分类概率

1. 实验体会

通过本次手写数字识别实验，我深刻体会到理论知识与工程实践相结合的重要性。在课堂上学到的卷积神经网络原理，只有通过亲手搭建模型、调试参数才能真正内化理解。从最初的MNIST数据集训练到最终的学号照片识别，整个过程让我认识到数据处理往往比模型设计更具挑战性——实际拍摄的学号照片存在光照不均、背景干扰、数字粘连等问题，需要综合运用图像处理技术进行预处理。在模型训练中，我经历了过拟合、梯度消失等典型问题，通过引入Dropout、批归一化等技巧逐步优化，最终达到99%以上的识别准确率。这次实验不仅锻炼了我的编程能力和调试技巧，更让我深刻理解了深度学习的核心思想：高质量的数据、合理的模型架构和耐心的调优过程同样重要。

实验四 校园共享单车检测

1. 实验目的

掌握目标检测的基本原理和实现方法，理解"定位+识别"的双重任务特性。实现校园共享单车的自动检测系统。掌握深度学习目标检测的完整流程，包括模型加载、推理、结果可视化和分析。

1. 实验原理

1. 目标检测基本概念

目标检测是计算机视觉中的重要任务，需要在图像中同时完成：

目标定位：确定目标在图像中的位置（边界框）

目标分类：识别目标所属的类别

2. Faster R-CNN算法原理

Faster R-CNN是两阶段目标检测算法的代表，主要包含四个部分：

1. 特征提取网络（Backbone）

使用ResNet-50 + FPN作为特征提取器。

FPN（特征金字塔网络）能够提取多尺度特征。

对于不同大小的目标都有良好的检测能力。

1. 区域提议网络（RPN）

输入特征图 → 生成锚框（Anchors） → 分类前景/背景 → 边界框回归

(3) ROI Pooling

将不同大小的候选区域统一到固定尺寸。

为后续分类和回归提供标准化的特征。

(4) 分类和回归头

对每个候选区域进行分类（识别类别）。

对边界框进行微调（精确定位）。

1. 实验方法

3.1 共享单车检测器（BicycleDetector类）

class BicycleDetector:

def \_\_init\_\_(self, model\_path=None, confidence\_threshold=0.5):

# 初始化设备、模型、参数

self.device = torch.device('cuda' if torch.cuda.is\_available() else 'cpu')

self.confidence\_threshold = confidence\_threshold

self.model = self.load\_model(model\_path)

self.model.eval()

3.2 主要功能模块

模型加载：支持预训练模型和自定义训练模型

检测推理：图像预处理、模型前向传播、结果后处理

结果可视化：边界框绘制、置信度标注

结果保存：图像输出、文本报告

3.3 实验步骤

（1）：环境配置与数据准备

创建Python虚拟环境

安装必要的深度学习库

准备测试图像（校园共享单车照片）

（2）：模型加载与初始化

# 使用COCO预训练的Faster R-CNN模型

model = fasterrcnn\_resnet50\_fpn(pretrained=True)

# 自行车在COCO中的类别ID为2

（3）：执行目标检测

# 图像预处理

image\_tensor = transform(image).unsqueeze(0).to(device)

# 模型推理

with torch.no\_grad():

predictions = model(image\_tensor)[0]

# 结果后处理

# 筛选自行车类别，应用置信度阈值

（4）：结果分析与可视化

边界框绘制（红色框）。

置信度标注。

统计信息输出。

结果文件保存。

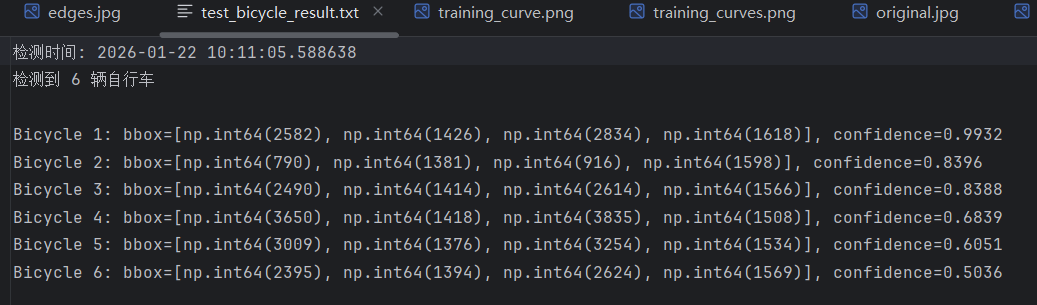
1. 实验结果

输入图片：



输出结果：





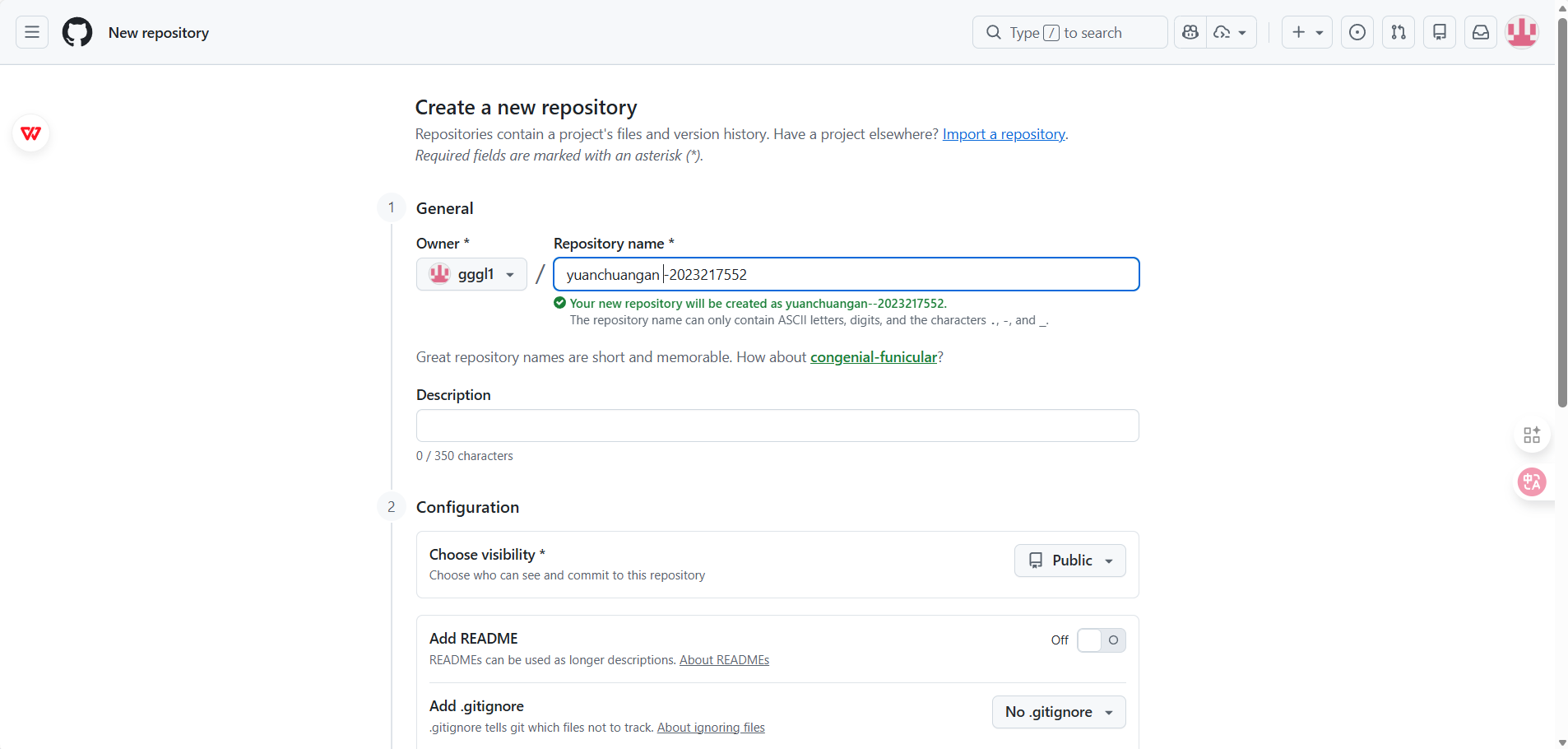
算法分析：

1. 特征提取能力强大：（1）深层网络提取丰富语义特征。（2）FPN处理多尺度目标效果好。（3）适合校园场景中不同大小的自行车。
2. 两阶段检测精度高：第一阶段（RPN）：快速生成高质量候选框。第二阶段（分类回归）：精确定位和准确分类。
3. 预训练模型泛化能力强：COCO数据集包含丰富的自行车样本，在不同场景下都有良好的表现，无需大规模自定义训练。
4. 实验体会

通过本次校园共享单车目标检测实验，我深刻理解了目标检测“定位+识别”的双重任务特性，以及深度学习在实际应用中的强大能力。从最初的模型选型到最终的检测结果可视化，整个流程让我认识到一个完整的计算机视觉系统不仅需要先进的算法，更需要细致的工程实现。实验中最大的收获是理解了Faster R-CNN两阶段检测的精妙设计——区域提议网络自动生成候选框，后续网络进行精准分类和定位，这种设计在准确率和效率之间取得了很好的平衡。同时，使用COCO预训练模型让我体会到迁移学习的实用价值：无需大规模标注数据就能获得相当不错的检测效果。在实现过程中，我遇到了边界框格式转换、置信度阈值选择、结果可视化优化等问题，通过不断调试和优化，最终让系统能够准确检测校园场景中的共享单车。这次实验让我认识到目标检测技术在实际生活中的广泛应用前景，也为后续更复杂的计算机视觉项目打下了坚实基础。

Git提交流程：

1.创建库



2.将代码文件拖入库

