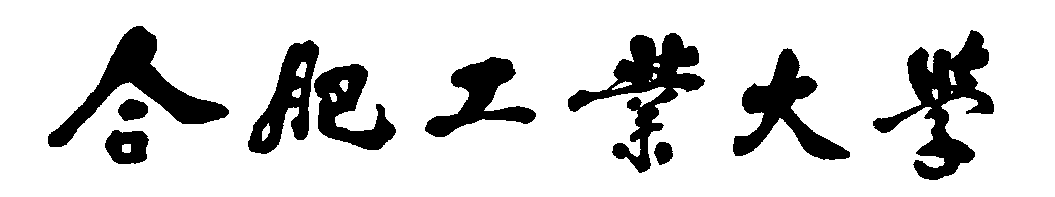
****

机器视觉

实验报告

|  |  |
| --- | --- |
| 专 业 班 级 | 计科23-2班 |
| 学生姓名及学号 | 2023215332-李奕霖 |
| 任 课 教 师 | 洪日昌、吴晶晶 |
| 实验指导教师 | 洪日昌、吴晶晶、金明 |
| 实验地点 | 翡翠湖校区科教楼C座机房 |
| 2025～2026学年第 一 学期 | |

**实验序号及名称：实验 二**  车道线检测

|  |
| --- |
| **预习内容** |
| **一、实验目的和要求∶**   1. 理解车道线检测在自动驾驶中的重要作用，掌握其基本处理流程。 2. 掌握图像预处理方法，包括灰度化、高斯滤波、Canny 边缘检测等操作。 3. 掌握 ROI感兴趣区域提取方法，能够聚焦道路区域并减少干扰。 4. 掌握 霍夫变换检测直线的方法，并能提取图像中的车道线候选直线段。 5. 学会对霍夫检测出的多条直线进行分类与拟合，得到稳定的左右车道线结果。 6. 实现车道线检测完整算法，输出最终车道线可视化结果图像，并能够对结果进行分析总结。 |
| **二、实验任务∶**   输入：自己拍摄的校园道路图像（包含车道线）。本实验代码使用输入图像：road.png。   使用霍夫变换实现车道线检测，输出图像中车道线的位置（效果类似示例图）。   对检测流程进行完整实现，包括：  边缘检测  ROI 区域裁剪  霍夫直线检测  左右车道线拟合与绘制   保存关键步骤结果图像，提交算法源代码、实验结果与实验分析。 |
| **三、实验准备方案，包括以下内容：**  **1. 实验原理**  车道线检测常见流程为： **图像预处理 → 边缘检测 → ROI 提取 → 霍夫变换直线检测 → 车道线拟合与绘制**  **（1）灰度化与高斯滤波**  为了降低计算量并减少噪声影响，先将彩色图转为灰度图，并使用高斯滤波平滑图像：  灰度化：将 BGR 图像转换为单通道灰度图。  高斯滤波：使用 5×5 高斯核对图像平滑，抑制噪声点，使边缘更加连续。  **（2）Canny 边缘检测**  Canny 算法通过梯度检测提取图像中的边缘信息。本实验使用阈值：  threshold1 = 50  threshold2 = 150  边缘结果通常包含车道线、车辆轮廓、路边护栏等多种边缘信息，需要结合 ROI 进行筛选。  **（3）ROI 感兴趣区域提取**  由于车道线一般出现在图像下半部分，本实验设置梯形 ROI 区域，仅保留道路区域边缘信息，从而减少背景（天空、建筑、车辆等）带来的干扰。  本实验 ROI 顶点按图像宽高比例设置为梯形：  左下：  右下：  右上：  左上：  **（4）霍夫变换直线检测**  利用概率霍夫变换 cv2.HoughLinesP() 提取图像中的直线段，参数设置：  rho=1  theta=π/180  threshold=40  minLineLength=60  maxLineGap=80  霍夫检测会产生多条直线段，因此需要进行筛选与拟合。  **（5）左右车道线分类与拟合**  对检测出的直线段计算斜率 slope：  slope < -0.5：判定为左车道线候选（负斜率）  slope > 0.5：判定为右车道线候选（正斜率）  对左右车道线分别求平均 slope 与 intercept，得到稳定车道线，并通过两点式计算绘制直线端点，实现最终输出。  **2. 实验环境与工具**  操作系统：Ubuntu22.04  开发语言：Python  使用库：  OpenCV（cv2）：图像处理、边缘检测、霍夫变换、绘图显示  NumPy：数组与矩阵计算  **3. 设计方案**  **车道线检测算法流程：**   1. 读入道路图像（road.png） 2. 转换灰度图 3. 高斯滤波降噪 4. Canny 边缘检测 5. ROI 区域截取（梯形区域） 6. 霍夫变换检测直线段 7. 斜率分类（左/右车道线） 8. 对左右车道线求平均拟合 9. 绘制最终车道线并输出保存结果   **4. 程序清单（主要函数说明）**   1. region\_of\_interest(img)   输入边缘图  输出 ROI mask 后的边缘图 + ROI 顶点坐标   1. average\_lines(img, lines)   对霍夫直线结果按斜率分类  平均拟合左右车道线   1. detect\_lane(image\_path)   主流程：读图、预处理、霍夫检测、车道线绘制、结果保存 |

|  |
| --- |
| **实验内容** |
| **一、实验用仪器、设备：**   计算机一台，i7核心   Python 运行环境（Python 3.x）   OpenCV、NumPy 图像处理库   输入测试图像：road.png |
| **二、实验内容与步骤（过程及数据记录）：**  本实验基于 OpenCV 实现车道线检测，主要流程包括：图像读取、灰度化、高斯滤波、Canny 边缘提取、ROI 区域筛选、霍夫直线检测以及左右车道线拟合与绘制。实验输入为道路图像 road.png，输出为最终车道线检测结果图像 lane\_result.png。  **步骤1：读取道路图像并显示原图**   1. 使用 cv2.imread(image\_path) 读取输入图像 road.png； 2. 若读取失败则输出提示信息并结束程序； 3. 使用 cv2.imshow("1 - Original", img) 显示原图。   **输入：** road.png（彩色图像） **输出：** 原图显示窗口 1 - Original    **步骤2：灰度化处理**  为了降低计算量，并为边缘检测提供单通道输入，将原图转换为灰度图：  gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)  **输入：** 原图 img（BGR） **输出：** gray（灰度图）    **步骤3：高斯滤波降噪**  为了减少噪声对边缘检测的影响，对灰度图进行高斯模糊处理，核大小设置为 5×5：  blur = cv2.GaussianBlur(gray, (5, 5), 0)  高斯滤波能平滑图像，使边缘更加连续。  **输入：** gray **输出：** blur（平滑后的灰度图）    **步骤4：Canny 边缘检测**  对模糊后的图像进行 Canny 边缘提取，阈值参数设置为 50 和 150：  edges = cv2.Canny(blur, 50, 150)  cv2.imshow("2 - Canny Edges", edges)  该步骤得到道路中的边缘信息，包括车道线边缘、车辆边缘和路旁边界等。  **输入：** blur **输出：** edges（边缘图），显示窗口 2 - Canny Edges    **步骤5：ROI 感兴趣区域提取（梯形区域）**  由于车道线一般集中在图像中下方区域，实验中对边缘图进行 ROI（Region of Interest）裁剪，仅保留道路区域内的边缘信息。   1. 在 region\_of\_interest() 中根据图像尺寸构造梯形 ROI 顶点：   左下点：  右下点：  右上点：  左上点：   1. 使用 cv2.fillPoly() 构造 ROI mask，并与边缘图进行按位与操作：   cropped\_edges, roi\_points = region\_of\_interest(edges)  cv2.imshow("3 - ROI Masked", cropped\_edges)   1. 为了可视化 ROI 区域是否合理，将 ROI 梯形画到原图上：   cv2.polylines(roi\_visual, [roi\_points], True, (0, 255, 0), 3)  cv2.imshow("4 - ROI Visualization", roi\_visual)  cv2.imwrite("roi\_visualize.png", roi\_visual)  **输入：** edges **输出：** cropped\_edges（ROI 区域边缘图） **保存文件：** roi\_visualize.png  **步骤6：霍夫变换检测直线段**  对 ROI 裁剪后的边缘图进行概率霍夫变换，检测直线段：  lines = cv2.HoughLinesP(  cropped\_edges,  rho=1,  theta=np.pi / 180,  threshold=40,  minLineLength=60,  maxLineGap=80  )  为了观察霍夫检测得到的原始直线段，将检测结果绘制在原图上（红色线段）：  hough\_img = img.copy()  if lines is not None:  for l in lines:  x1, y1, x2, y2 = l[0]  cv2.line(hough\_img, (x1, y1), (x2, y2), (0, 0, 255), 2)  cv2.imshow("5 - Hough Raw Lines", hough\_img)  cv2.imwrite("hough\_raw.png", hough\_img)  **输入：** cropped\_edges **输出：** lines（霍夫检测直线段集合） **保存文件：** hough\_raw.png  **步骤7：左右车道线筛选、拟合与绘制**  霍夫检测会产生多条直线段，为获得稳定的车道线，实验中使用 average\_lines() 对直线段进行筛选与拟合：   1. 计算直线斜率： 2. 根据斜率符号进行分类：   slope < -0.5：判定为左车道线  slope > 0.5：判定为右车道线  过滤 slope 过小或过大的干扰线段   1. 对左右车道线分别求 slope 与 intercept 的平均值，得到拟合后的左右车道线参数； 2. 设定绘制范围：   下端点取图像底部  上端点取  再通过直线方程反求 坐标，生成最终线段端点；   1. 将最终左右车道线绘制到原图中，线宽设置为 8，颜色为绿色：   averaged = average\_lines(img, lines)  output = img.copy()  if averaged:  for x1, y1, x2, y2 in averaged:  cv2.line(output, (x1, y1), (x2, y2), (0, 255, 0), 8)  **步骤8：输出最终车道线检测结果并保存**  将最终绘制车道线后的结果图像显示并保存：  cv2.imshow("6 - Final Lane", output)  cv2.imwrite("lane\_result.png", output)  **输出：** 最终车道线检测结果图 **保存文件：** lane\_result.png  **步骤9：等待退出并关闭窗口**  程序通过 cv2.waitKey(0) 等待按键输入，之后关闭所有窗口：  cv2.waitKey(0)  cv2.destroyAllWindows() |

|  |
| --- |
|  |
| **三、实验结果分析∶**  **1. 原始图像结果分析**  本实验输入为道路场景图像，如图1所示。该图像具有典型的车道线检测特征：车道线清晰且呈现向远处收敛的透视效果，图像下方车道线边缘明显，适合进行基于边缘与直线检测的车道线提取。与此同时，图像中还包含车辆、路旁护栏、树木、电线杆等干扰因素，这些干扰结构在后续边缘提取时可能产生误检，因此需要进一步的 ROI 筛选与直线拟合。  图1 原始道路图像（Original）  **2. Canny 边缘检测结果分析**  对原始图像进行灰度化与高斯模糊后，采用 Canny 算子提取边缘，结果如图2所示。从图中可以观察到：   1. 车道线在边缘图中呈现出清晰且连续的亮边，说明 Canny 能够较好地提取车道线的梯度信息； 2. 除车道线外，车辆轮廓、远处道路边界、树木轮廓、电线杆等也被检测为边缘，这些属于非车道线边缘干扰； 3. 整体边缘检测结果较干净，噪声点较少，说明高斯滤波在一定程度上抑制了噪声，使边缘更平滑连续。   因此，Canny 边缘检测能够有效获取车道线候选边缘，但需要进一步通过 ROI 区域截取来减少无关边缘的影响。  图2 Canny 边缘检测结果图（Canny Edges）  **3. ROI 提取结果分析**  由于车道线主要位于图像中下方区域，本实验对 Canny 边缘图进行了梯形 ROI（感兴趣区域）提取，结果如图3所示。可以看到，ROI 提取后：   1. 图像中上部天空、远处山体、树木等区域的边缘被大量过滤； 2. 保留下来的边缘主要集中在道路中间的车道线区域，车道线边缘更加突出； 3. ROI 处理显著减少了干扰边缘数量，提高了后续霍夫变换检测直线的准确性与效率。   因此，ROI 的引入能够有效聚焦道路区域，是车道线检测中非常关键的一步。  图3 ROI 区域提取后的边缘结果图（ROI Masked）  **4. ROI 可视化效果分析**  为了验证 ROI 区域是否选取合理，同时便于调参，实验将 ROI 梯形区域绘制在原始图像上，如图4所示。从图4中可以看出：   1. ROI 梯形覆盖了道路主体区域，并包含左右车道线主要分布范围； 2. ROI 的上边界设置在图像中部附近，使得检测重点集中在车道线最明显且最具代表性的区域； 3. ROI 左右边界与道路透视方向一致，有助于过滤无关区域并保留有效车道线信息。   整体来看，该 ROI 的形状与位置能够满足车道线检测需求，为后续霍夫直线检测提供了较好的输入范围。  图4 ROI 区域可视化结果图（ROI Visualization）  **5. 霍夫变换检测原始直线结果分析**  在 ROI 边缘图基础上采用概率霍夫变换检测直线段，并将检测到的原始直线绘制到原图中，结果如图5所示。可以观察到：   1. 霍夫检测得到的红色直线段大部分与车道线方向一致，说明车道线边缘在霍夫空间中具有较强的直线特征； 2. 检测结果呈现多条分散的线段，这是因为车道线在边缘图中可能存在断裂、噪声或局部不连续； 3. 原始霍夫线段包含车道线的主要结构，但仍可能存在部分冗余线段，需要进一步拟合融合为更稳定的左右车道线。   因此，霍夫变换能够有效检测车道线候选直线，但需要后续通过斜率分类与平均拟合来得到最终稳定车道线。  图5 霍夫变换检测原始直线结果图（Hough Raw Lines）  **6. 最终车道线检测结果分析**  实验最后对霍夫检测出的直线段按斜率进行分类（左车道线斜率为负、右车道线斜率为正），并分别求平均得到左右两条车道线，最终绘制结果如图6所示。可以看到：   1. 最终输出的绿色车道线清晰标注出左右车道边界，能够较准确地反映车道线位置； 2. 相比霍夫原始直线结果，最终结果更加平滑、稳定，减少了杂乱线段的影响； 3. 车道线线条粗细适中（代码中设置 thickness=8），便于观察最终检测效果； 4. 在该道路场景下，本算法能够成功完成车道线检测任务，满足实验要求。   综上，本实验实现的“Canny + ROI + 霍夫变换 + 斜率拟合”车道线检测方法能够在较为清晰的道路场景中有效检测车道线位置，并生成稳定可靠的输出结果。  图6 最终车道线检测结果图（Final Lane）  至此，实验二完成。 |
| **四、感想、体会、建议∶**  通过本次车道线检测实验，我对自动驾驶中车道线检测的基本流程有了更直观的认识。从输入道路图像开始，经过灰度化、高斯滤波、Canny 边缘检测，再到 ROI 感兴趣区域筛选和霍夫变换直线检测，整个过程体现了机器视觉“先提取特征、再筛选目标”的典型思路。尤其是在最终拟合左右车道线时，我能明显感受到将零散的直线段融合成稳定车道线的重要性，也更加理解了为什么车道线检测需要多步处理才能得到可靠输出。  在实验过程中，我最大的体会是：参数和ROI区域的设置对结果影响非常大。例如 Canny 的阈值大小会直接影响边缘是否完整，ROI 梯形区域如果选取不合理就可能导致车道线检测不全或者引入过多干扰边缘，而霍夫变换中的阈值、最小线段长度和最大间隔等参数也会影响检测出的直线数量和质量。通过观察中间结果图，我发现一步一步检查处理效果非常必要，这样才能及时调整参数，让最终车道线检测结果更加准确和稳定。  总体来说，本实验让我把课堂中学到的边缘检测、ROI裁剪、霍夫变换等知识真正应用到了实际图像中，并且通过完整的实验流程输出了较清晰的车道线检测结果。我也意识到传统方法在光照变化、车道线模糊或遮挡情况下可能会存在局限性，后续如果要进一步提升鲁棒性，可以尝试颜色空间阈值分割、透视变换（鸟瞰图）、滑窗拟合甚至深度学习方法等。通过这次实验，我对机器视觉在自动驾驶领域的应用更加感兴趣，也为之后更复杂的视觉任务打下了基础。 |
| **实验成绩∶**  **指导教师签名：**  年 月 日 |