

# 实验报告

姓名：杨君涵

## 一、 实验内容：

- a) Tusimple 车道线检测。
- b) 编程语言：C 或者 C++，源代码见 lane.cpp
- c) 除了图像的读入和输出外，中间过程不能调用任何库。

## 二、 实验原理：

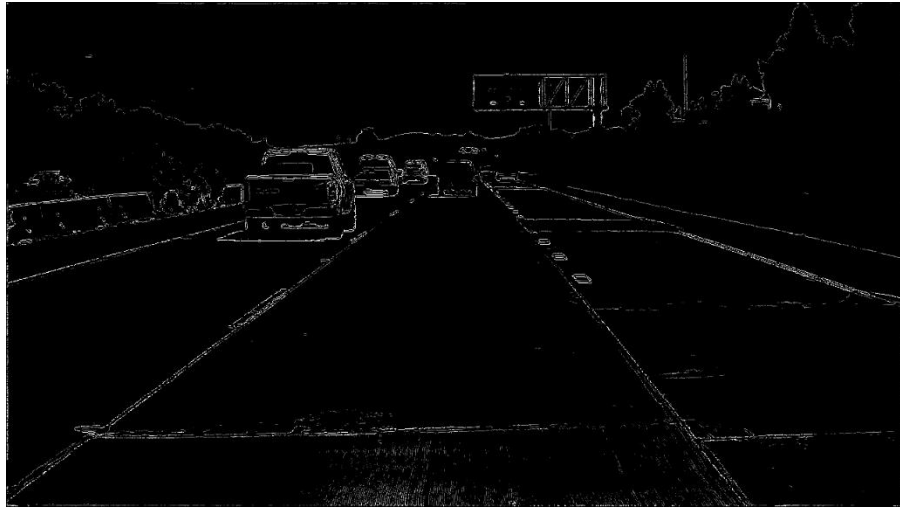
### a) 步骤：

- i. Canny 边缘检测算法处理，获得图像轮廓
- ii. 霍夫变换直线检测算法，投票筛选可能直线
- iii. 使用 k-means 算法对霍夫变换确定的点对进行聚类
- iv. 以 tusimple 格式将结果写为 json

### b) Canny 边缘检测：

- i. Canny 边缘检测算法是目前最为优秀的边缘检测算法，被广泛地使用在各种需要进行边缘检测的场合。在本实验中，将采用 Canny 边缘检测算法识别车道线轮廓，滤除图像中的无关细节。
- ii. 算法介绍：
  - 1. 使用高斯滤波器处理灰度图，滤除噪声
  - 2. 用 sobel 算子得到图像的梯度幅值和方向
  - 3. 从梯度方向上对图像的梯度幅值进行非极大值抑制
  - 4. 用双阈值算法检测和连接边缘

iii. 效果示例：

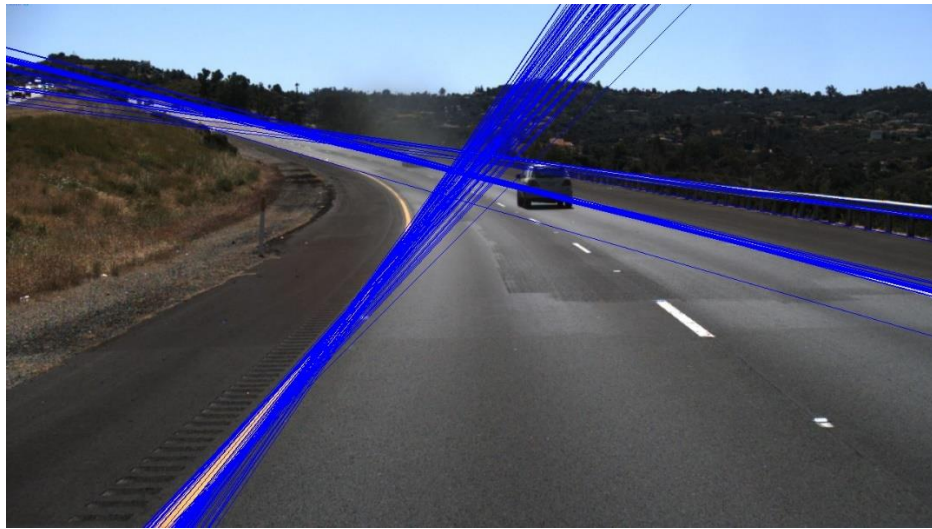


c) 霍夫直线检测

- i. 霍夫变换是由 Paul Hough 提出的，图像处理领域检测几何形状的一种方法，通过参数空间变换，将几何形状检测问题转化为峰值统计问题。
- ii. 霍夫变换直线检测：
  1. 直角坐标系中的一条直线  $y=kx+b$ ，可以表示为参数空间的一个点对  $(r, \theta)$
  2. 同时，直角坐标系中的一个点对  $(x, y)$ ，可以表示为参数空间中的一条  $r=x\cos\theta+y\sin\theta$
  3. 将参数空间的  $r$  和  $\theta$  若干等分，可以将参数空间分割为大小相同的离散空间
  4. 遍历图像中的所有边缘点，将边缘点转换为参数空间中的曲线，每当曲线经过一个离散空间，就对该离散空间的计数器进行一次加一操作

5. 选择计数器超过设定阈值的离散空间，代表该离散空间的点对既是原图像空间中的一条可能直线；这里采用
6. 通过形态学和规则筛除一部分直线

iii. 效果示例：



d) K-means 聚类

- i. K-means 聚类
- ii. 使用 K-means 聚类对霍夫变换点对进行聚类
  1. 将霍夫变换点对集合按 $\theta$ 大小进行排序
  2. 均匀选取 6 个初始聚类中心
  3. 距离计算公式为  $\text{dist} = \sqrt{(r/\text{size})^2 + (\theta/180)^2}$ ，其中  $\text{size}$  为图像对角线大小；也可以通过调整  $\text{size}$  大小来改变  $r$  和  $\theta$  在距离计算中所占权重
  4. 将所有霍夫变换点对分配到各聚类中
  5. 计算新的聚类中心（均值）
  6. 重复 4、5 两步直到各聚类中心不再改变

7. 注：如果循环过程中某聚类中的点对个数为 0，则直接删除此聚类

iii. 效果示例:



### 三、实验结果:

a) 输出结果: 见 pred.json

输出示例:

[illegible]

## b) 准确率：

经 tusimple 官方提供的 lane\_demo.ipynb 检测，本程序对乐学中选取的 100 个用例检测的准确率为 46.875%



The screenshot shows a Jupyter Notebook interface with two code cells. The first cell (index 6) contains Python code to load ground truth and predicted lane data from JSON files. The second cell (index 7) contains code to shuffle the predicted lanes and evaluate the performance using LaneEval. The output of the second cell shows the evaluation results: (0.46875, 1.0, 1.0).

```
[6] 1 pred, gt = json_pred[0], json_gt[0]
    2 pred_lanes = pred['lanes']
    3 #run_time = pred['run_time']
    4 gt_lanes = gt['lanes']
    5 y_samples = gt['h_samples']
    6 raw_file = gt['raw_file']

运行时长: 4毫秒 结束时间: 2021-02-14 15:33:53

[7] 1 np.random.shuffle(pred_lanes)
    2 # Overall Accuracy, False Positive Rate, False Negative Rate
    3 print(LaneEval.bench(pred_lanes, gt_lanes, y_samples,0))

运行时长: 8毫秒 结束时间: 2021-02-14 15:33:53
(0.46875, 1.0, 1.0)
```

## c) 结果分析：

### i. 准确率不高

### ii. 可能原因：

1. 使用霍夫变换检测直线的方式来判定车道线，会将所有类似直线的边缘判定为一条可能的车道线
2. 部分车道线存在弧度，不一定是直线
3. 部分车道线不连续，并且存在被车辆遮挡的情况
4. K-means 聚类不能很好地聚合点对

### iii. 改进方法：

1. 对于霍夫变换确定的直线，用一定规则逐像素判定
2. 改进聚类的方式

#### 四、 总结：

- a) 本实验使用的纯数字图像方法检测车道线很难令人满意：
  - i. 容易将类似车道线的直线物体检测为车道线
  - ii. 车辆等中途阻断车道线的物体会严重影响车道线检测
  - iii. 不连续的、有弧度的、不同视角的车道线难以检测