计算机视觉

—车道线检测

说明文档

参赛学校 温州大学

队伍名称

指导老师 任明

参赛学生：董相铭 王雪奇 杨京东 闫世为 谭祖春

二零二二年

目录

一.项目介绍……………………………………………………………3

二.性能分析……………………………………………………………4

三.创新说明……………………………………………………………5

四.未来优化……………………………………………………………5

五.总结…………………………………………………………………5

六.参考文献……………………………………………………………6

一.项目介绍

1.1、基于python-opencv方法的车道线识别

**基于python-opencv方法的车道线识别**依赖于手工提取的特征来识别，如颜色的特征、结构张量、轮廓等，这些特征还可能与霍夫变换、各种算子或卡尔曼滤波器相结合。在识别车道线之后，采用后处理技术来过滤错误检测并将其分组在一起以形成最终车道。

1.2、实现步骤：

Canny边缘检测

对特定区域遮罩，提高检测准确度

对车道线轮廓检测并进行上色

计算中心点并输出中心点坐标

计算视频帧数并显示

1.3、环境搭建

PyCharm 2021

python3.8

1.4、准备工作

在默认环境中安装后续代码依赖的第三方包

OpenCV: pip install opencv-contrib-python -i <https://pypi.tuna.tsinghua.edu.cn/simple/>

Matplotlib: pip install matplotlib、

Sklearn: pip install sklearn、

Labelme: pip install labelme、

Imutils: pip install imutils

1.5、代码结构和各部分功能如下：

Opencv-python

├── canny# 边缘检测

├── gray = cv.cvtColor#颜色空间转换

├── GaussianBlur#高斯模糊降噪

├─segment#图像分割

├── polygons = np.array#创建数组构建多边形选定感兴趣区

├── mask = np.zeros\_like #构建一个同维度的数组，并初始化所有变量为零。

├── cv.bitwise\_and(frame, mask) #RGB图像选取掩膜（mask）选定的区域

├── do\_color(img)#对车道线进行上色

├── cv.drawContours #轮廓绘制

├── do\_center(image)#处理中心点

├── cv.threshold#阈值处理

├── cv.findContours#查找检测物体的轮廓。

├── moments#图像矩

├── circl#根据给定的圆心和半径等画圆

├── image\_resize#将图片改为（，）大小

├── start\_time = time.time()#计算代码运行时间

├── fps = cap.get(cv.CAP\_PROP\_FPS) #视频平均帧率

├─ ret, frame = cap.read()#视频读取

├─ counter += 1#计算帧数

├─cv.imshow#展示图像

├─ cap.release()#需要放置在while循环体外。发布软件资源，释放硬件资源

二.性能分析

fps 平均帧数达到了50帧/秒与原视频帧数几乎相等

三.创新说明

3.1由于霍夫直线检测方法准确但不能做弯道检测，拟合方法可以检测弯道但不稳定，透视变换操作会对相机有一些具体的要求，在变换前需要调正图像，而且摄像机的安装和道路本身的倾斜都会影响变换效果。所以我们对边缘检测和区域遮罩后只含有车道线的图像进行轮廓识别处理上色，计算中心点坐标，排除外在影响

3.2对于不同性能的使用要求，我们可以通过调节视频的播放倍速来改变视频帧数的多少，从而达到多种检测需求

3.3对区域遮罩后的图像再进行高斯滤波减噪和阈值处理，检测车道线轮廓，计算检测到的车道线轮廓的中心点坐标，在一个新的图像中绘制出检测到的中心点，并在终端中输出每帧的中心点坐标数组

3.4通过resize函数来更改输出的图像大小，以便于更好的对比观察识别出的车道线图像

四.未来优化

4.1将提取出的中心点轮廓连接后与原车道线融合

4.2将上色后的图像叠加到原图像中

五.总结

这次的计算机视觉-车道线检测我们采用的是基于传统方法的车道线检测，使用opencv-python代码来实现车道线检测，完成对轮廓的识别处理，计算中心点坐标并输出坐标数组，计算视频平均帧数并显示。

优点：基于传统方法的车道线检测方法简单，速度较快。不需要构建神经网络，硬件成本低。

缺点：①易受光线、环境、道路车辆等因素的影响，使得结果有偏差；

②需要调整的参数较多

效果视频地址如下【【智能车航天物流组】opencv车道线中点识别并提取坐标点-哔哩哔哩】 https://b23.tv/Hthvchw

六.参考文献

1、[Python] OpenCV+霍夫直线检测的车道线识别

https://www.52pojie.cn/thread-1473150-1-1.html

2、cv2.imshow调整窗口大小

https://blog.csdn.net/weixin\_42929622/article/details/115273454

3、传统车道线检测-canny边缘检测-霍夫变换-完整代码（python）

https://blog.csdn.net/m0\_46988935/article/details/109234900

4、OpenCV 图像缩放：cv.resize() 函数详解

https://blog.csdn.net/hysterisis/article/details/112381220

5、cv2.findContours() 轮廓检测

https://blog.csdn.net/Easen\_Yu/article/details/89365497

6、python用opencv实现直线检测和计算交点

https://blog.csdn.net/qq\_33004317/article/details/100079230