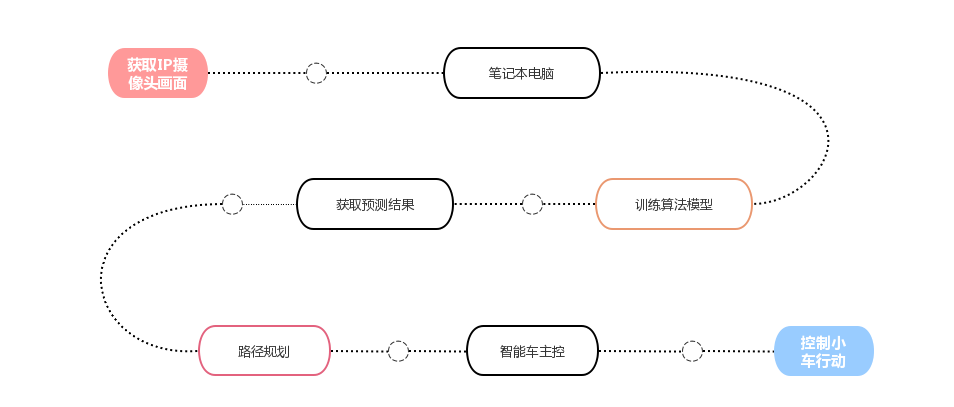
1. 通过键盘实现小车行走
2. 小车自动沿着道路行走（无人驾驶）

整体思路：



1. 获取摄像头画面

通过WiFi热点获取摄像头画面并把它呈现在网页上。

1. 训练算法模型

此部分可见小组两份第一部分作业：

https://aistudio.baidu.com/aistudio/projectdetail/4366888

https://aistudio.baidu.com/aistudio/projectdetail/4355010

1. 训练模型  
   和第1部分的作业一样,我们可以很简单的使用下面一句话进行模型的训练，其中模型的相关参数解释如下:

python train.py -h usage: train.py [-h] [--epochs E] [--batch-size B] [--learning-rate LR] [--load LOAD] [--scale SCALE] [--validation VAL] [--amp]

Train the UNet on images and target masks

optional arguments:

-h, --help show this help message and exit

--epochs E, -e E Number of epochs

--batch-size B, -b B Batch size

--learning-rate LR, -l LR Learning rate

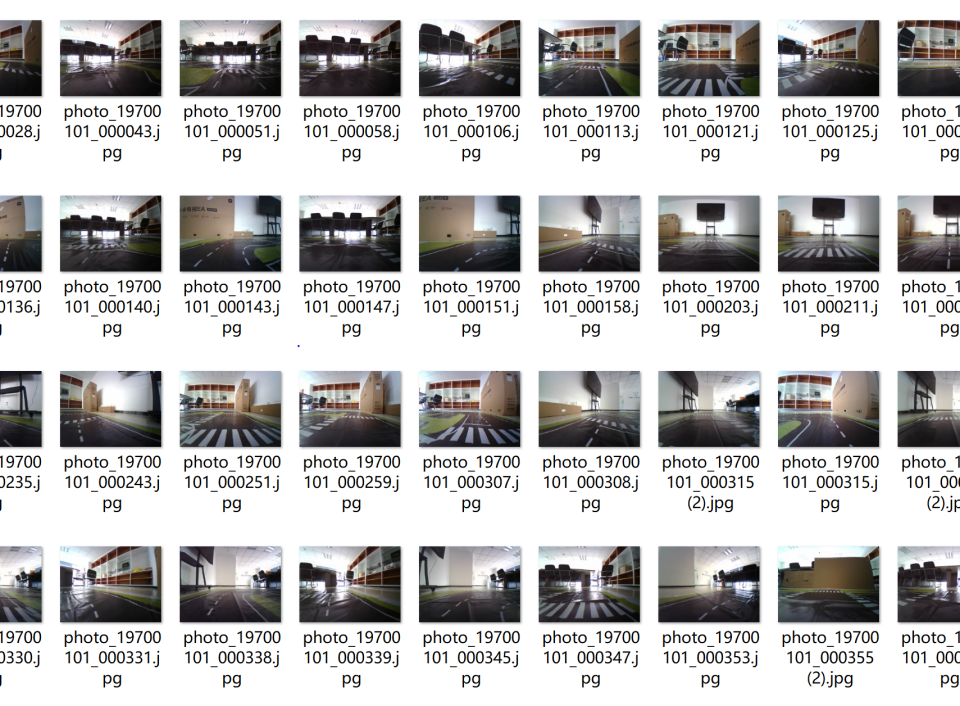
--load LOAD, -f LOAD Load model from a .pth file

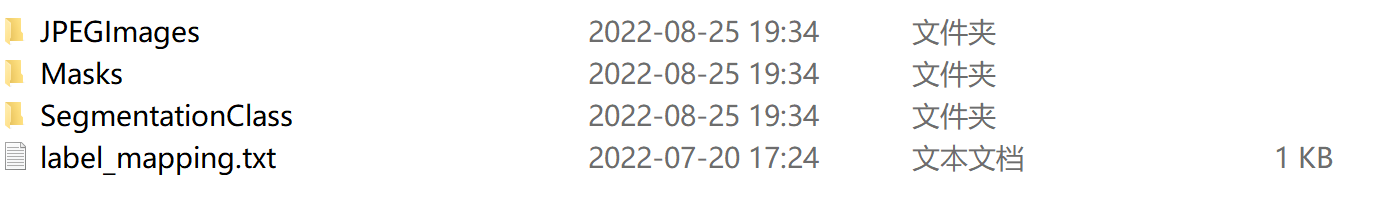
--scale SCALE, -s SCALE Downscaling factor of the images

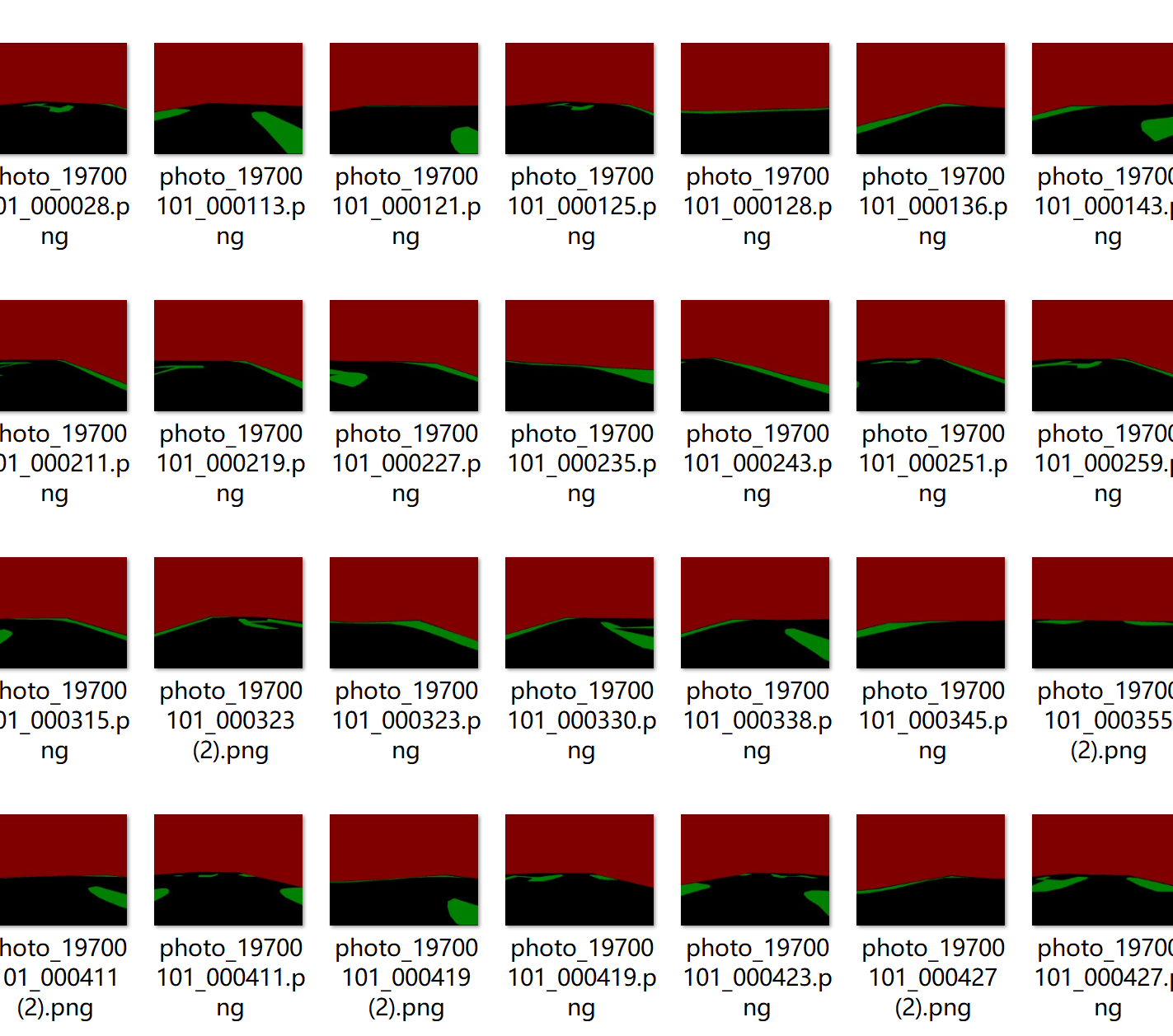
--validation VAL, -v VAL Percent of the data that is used as validation (0-100)

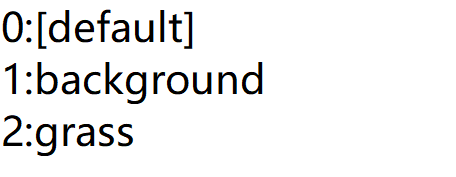
--amp Use mixed precision

1. 使用小车跑道数据训练  
   对于所需要做的无人驾驶部分,我们需要重新获取数据集，我们对跑道进行拍摄和标注，如下，对这些数据集进行处理，转化为灰度图像进行训练，得到所需要的训练结果的配置文件，从而为后续预测提供配置文件。









（3）得到训练完成的文件

1. 获取预测结果

通过request函数爬取网页上的内容获得关于小车摄像头的照片,再通过上述训练得到的模型，获取预测结果，即小车下一步应该向何处运动。

1. 路径规划

智能车系统是一个各部分协调运作的控制系统，系统要完成传感器接收，控制器判断计算到执行的整个过程。对路径的迅速判断，相应的转向控制和舵机的控制要紧密结合在一起。对于车辆的控制首先定义电机的占空比、获取舵机的角度和实测速度等函数，获取车辆的状态类，首先我们要我们根据前式距离计算下一个目标点，然后选择一个舵机的转向角度进行控制，可以设置三种不同的小车运动类型，分别是直线型，正弦函数型和L弯型。把照片从网格预测的大小放射到拍摄大小并矫正图片，进行鸟瞰图的转换，补全车前的盲区，选取终点并且把路径设为平滑，通过request的函数获取网页上关于照片以及车辆速度车辆角度等相关信息，使用a\*算法进行路径规划。当车轮一定与轨迹相切圆弧切线的垂线一定指向圆心，根据前后轮的朝向就能确定瞬时的旋转中心，前左轮和前右轮协同转动，瞬时旋转中心重合。

相关代码见\_photo\_to\_pursuit.py文件  
5.智能车主控

通过测速计和IMU获取小车速度和角度相关并呈现在网页上，从而实现更好的对小车的控制。

1. 测速计（控制速度）



（测速器图源网络，因为我的测速计已经安装好了）

工作原理：码盘随着小车轮子的运动转动时，码盘（光栅）会不断遮挡光敏元件发出的光波，这时候编码器就会根据光栅的遮挡不断的产生方波信号，方波信号会从“OUT”引脚输出，只需不断检测“OUT”引脚的输出，根据方波信号的周期简介计算出小车运行的速度。电机同轴的码盘上有很多开孔（光栅），编码器相当于光敏元件。

相关代码见test\_http\_request.py

（2）IMU（控制角度）

IMU是测量物体三轴姿态角（或角速率）以及加速度的装置。测量物体在三维空间中的角速度和加速度，并以此结算出物体的姿态。

6.控制小车行动

通过获取许多IP摄像头的照片，传入我们的笔记本电脑，使用我们的道路分割模型，对这些新的数据集进行训练，得到训练的配置文件，再通过request函数获取网页上实时拍摄的摄像头画面进行预测。结合路径规划算法和智能车的imu以及测速器，舵机等相关控制元件，综合运动学调节角度和速度之间的关系等来实现控制小车的行动。通过循环设置走多少次流程，使小车一直走下去。

相关代码见final.py

1. 小车识别交通标志并发出蜂鸣声

详情可见小组第二部分作业：https://aistudio.baidu.com/aistudio/projectdetail/4383161

和二类似，不过模型不需要重新训练，可以直接使用第二部分作业训练得到的配置文件，按照二的方法通过获取实时IP摄像头的照片，传入我们的笔记本电脑，使用我们的交通信号识别与检测模型，通过request函数获取网页上实时拍摄的摄像头画面进行预测。预测结果会有三种，使其对应不同的蜂鸣器声音（改变延时和发音次数即可）。

交通标志识别的模块，我们用机器学习算法的卷积神经网络实现。

首先，导入模块numpy、matplotlib.pyplot、cv2、pickle、os、pandas、random、Sequential、Dense、Adam、to\_categorical、Dropout、Flatten、Conv2D、MaxPooling2D、train\_test\_split、ImageDataGenerator，设置基本参数，包括数据集路径、迭代次数、整个训练集训练次数，测试集占比、验证集占比等。

然后引入图像加载图像与标签，即把不同文件夹里的图像编号整理并以数列形式保存，分割测试集和验证集，检查图像数量是否与每个数据集的标签数量匹配并输出结果。

因颜色不影响交通标志识别，为使后面的算法运行更快，我们把图片转为灰度图，抛弃冗余的数据，然后进行直方图均衡化，进一步精简信息，数据预处理完毕。

为数据集增加一维，然后经过第一个卷积核两次卷积一次池化，第二个卷积核两次卷积一次池化，输出后编译，数据处理结束。

用imagedatagenerator进一步处理，平移、缩放、错切、旋转一个微小值，经过迭代，训练模型，评估后保持模型，交通标志识别的训练网络部分完成。

预测模型，设置参数，导入训练好的模型，定义每种标志含义。

图像预处理后，用predict、predict\_classes命令预测 ，预测得到的概率与阈值0.75比较，大于阈值0.75判断有效检测，返回标志种类，否则返回“no”，即不能识别。