上海交通大學

工程实践与科技创新 III -F

结题报告

项目名称：　基于图像处理的树莓派智能小车

姓　　名：　陈思哲

学　　号：　516021910038

班　　级：　F1603203

同组同学：　张沛东，牟天昊

联系电话：　13262292070

电子邮箱：　729020210@qq.com

2018-2019 学年第 2 学期

2019 年 6 月 15 日

目录

研究内容综述

研究任务

在设计智能小车控制算法，控制树莓派小车在场地上巡线行驶，并在巡线过程中通过识别到的标志执行相应的动作。

实验要求

小车在静止状态下识别不同标志，准确率大于 90%。

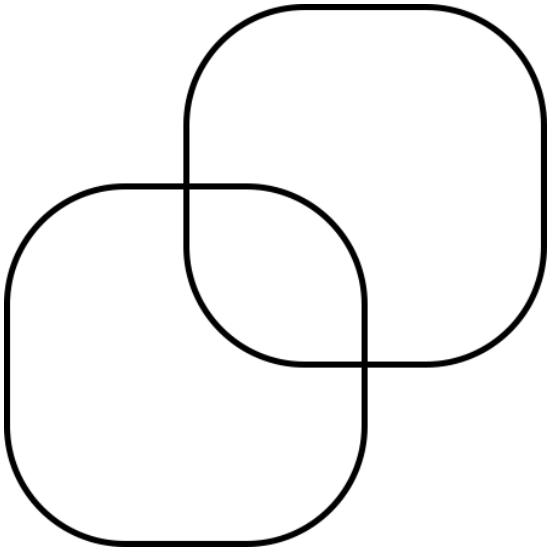
在布置有标志牌的场地上行驶，小车根据直行、左转、右转的标志牌分别执行相应的动作，最终到达终点。

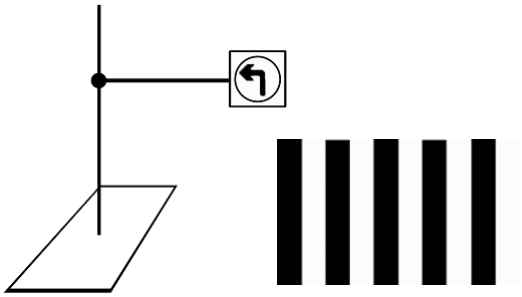
巡线完成时间在 60s 以内。

能够识别斑马线（暂定），并识别后在路口停车 3 秒让行。

设备说明

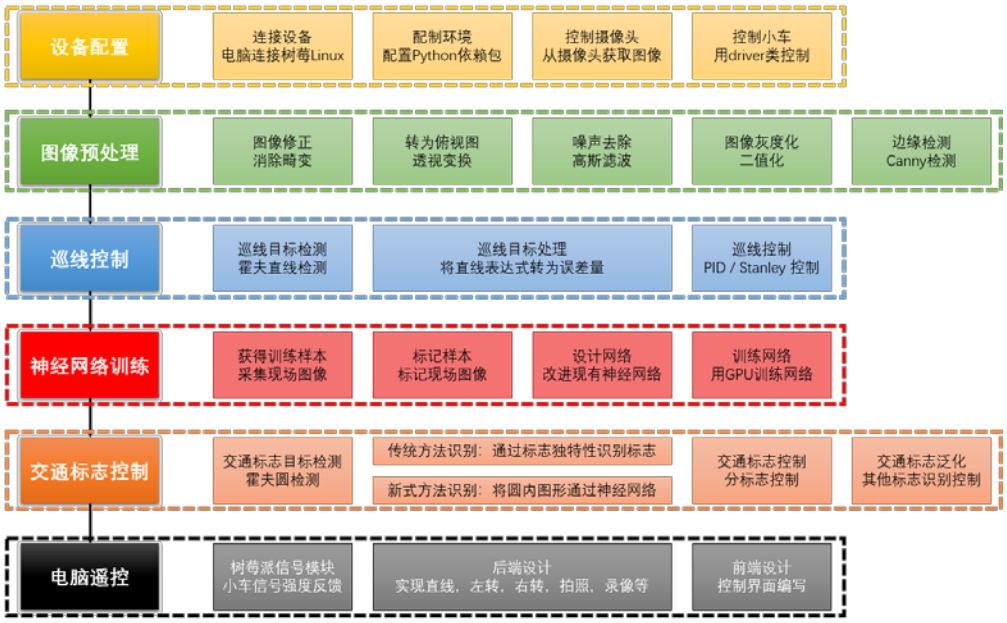
巡迹场地以实地为准，主要由直线、弯道以及道路交叉组成。标志识别的巡迹场地上有两个交叉路口，交叉路口放置有标志牌指示转向方向。其中圆弧半径为 50cm，线宽 2cm。直行、左转、右转标志牌大小为 5cm*5cm。标志牌的布置：标志牌粘贴于透明亚克力板上，亚克力板固定于铁架台上，标志牌高度暂定于 20cm，放置于交叉路口指示转向。斑马线的布置：斑马线尺寸为 40cm*30cm，采用间隔 3cm，线宽为 3cm 的黑线，直接印刷或粘贴在巡线场地上。



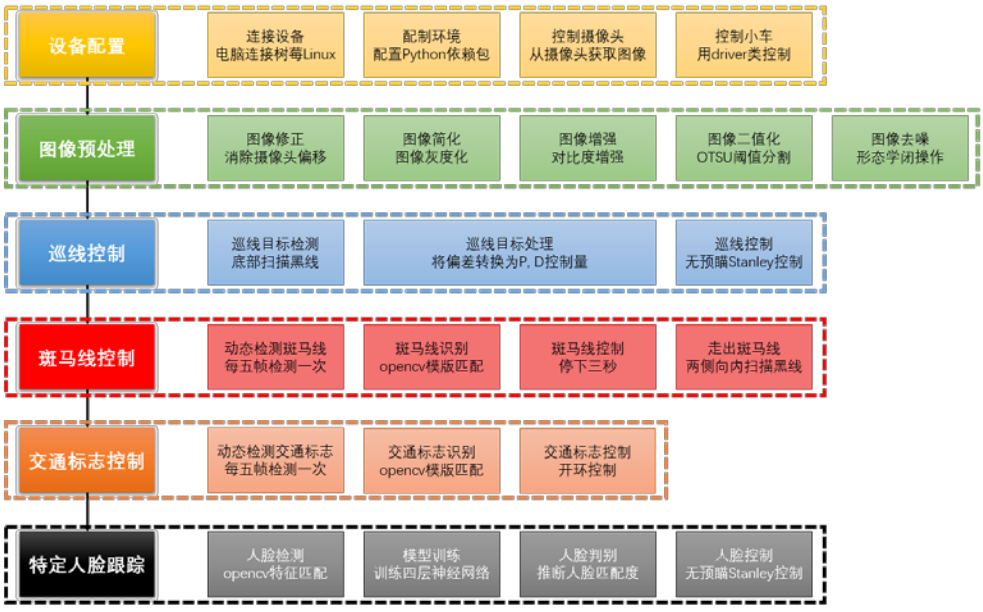


项目架构

在开题报告中，我们计划通过以下的技术流程实现最终的项目及自选项目，当时我们认为我们可以在检测出霍夫圆后，并不是使用传统方法判断，而是将圆内图像通过训练好的神经网络，以判定其所属分类，从而快速高效完成本项目中难度最高的信号检测技术点。同时我们认为自选项目可以采取获取键盘中断从而操作树莓派小车的形式以实现陌生环境下的拟 SLAM 建模。



最终我们完成了所有的任务（具体效果见研究成果），但途径和架构都略有不同（如下图），因为原先的架构部分效果不佳，而部分囿于硬件条件实现起来无法体现工作量。



研究状况综述

随着电子技术和嵌入式开发的不断发展，树莓派凭借其小体量、高算力逐渐成为开发实践项目的新宠儿。对于我组的树莓派小车，我们综合考量了老师给与的教程、长春工程学院的开发课程[1]和孙智勇等人的简单实训项目[2]，对树莓派小车的整个硬件结构和相关电力电子理论基础有了一个比较清晰的把握。

在本次 3F 实践项目 C 题中，我们的树莓派小车首先需要解决的是巡线问题。针对实际问题中可能存在的道路路面结构不均匀、光照变化、阴影遮挡以及其他车辆的存在等使得道路图像变复杂,车道线识别度低的情况，冯庆等人给出了一种基于灰度优化的方法[3]。至于在图像识别中普遍存在的畸变问题，鲁通通等人[4]提出了一种基于神经网络的校正技术。而对于路线识别后的控制方法及优化，徐爱昆等人[5]采用了一种画面的自适应切割以寻找道路理论中间值的方法，从而得到控制误差以适应双轮循迹平衡车的 PID 控制。

其次需要解决的是对交通标志的识别问题。该问题可以分解为图像采集、图像处理、反馈交互三个线程。对于一张未经任何处理的数字图像，算法很难得到精确的关注点以实现对标志的进一步处理。基于此，楼怡杭等人[6]在车牌识别处理项目中采用 HIS 模型和抹布处理法实现了对车牌相对准确的定位；而刘鹏程等人[7]则在人像识别项目中证明了 Faster R-CNN 的 ZF 模型对于人物位置的标定有极强的效用。在一定的标定和预处理之后的样本将被送入下一层网络。

接下来是更为重要的图像识别技术。大批研究者如孙云云等[8]和曹振军等[9]证明了深度学习在各领域图像处理上的成功运用及将其应用至树莓派上的可行性。此外，李凯等人[10]提出了一个新型的的生成对抗网络，通过对抗方式提高了原始网络对不清晰图片的识别准确率；马治楠等人[11]则提出了一种对深层卷积网络的剪枝优化方法，为将复杂网络应用至嵌入式开发提供了更好的技术手段。对于我组的自选拓展题目，杨志勇等人[12]提供了一种基于移动智能终端平台，借助于 4G 通信技术实现树莓派小车与用户交互的开发方案。而罗佳伟等人[13]则在他们的项目中尝试同时应用深度学习技术和智能遥控技术以进行控制方案的补充。最后，刘燕娜等人[14]在板球走迷宫系统中提出了一个富有创新意识的方案，算法将通过图像的采集和处理尝试复现整个环境，以梯度下降方式获取最优解决办法，从而进一步强化控制系统。

在本次 3F 自选项目中，我们首先借鉴了朱琳[15]等人的研究成果，他们在文中详细描述了 Linux 系统下基于 Python 建立通用串行总线与摄像头模组通信,调用 OpenCV、Face++相关 API,从而来辨别实验室成员和陌生人。对系统进行模拟测试,实验室、仓库、家庭等地的识别成功率达到 99.7%。在复杂环境下,人脸检测会因为很多外部因素影响检测的准确性，如人脸姿态变化、光照改变和遮挡等。将人脸检测应用于实际,需要解决人脸拍摄角度、光照变化以及低分辨率图像的问题。余飞等人[16]提出了一种级联深度卷积神经网络,级联结构的设计能使人脸框图定位更准确,深度卷积神经网络能提取更多有效特征,使人脸检测结果更准确。

其次树莓派需要在识别出的人脸中找到特定人脸，秦超[17]等人实现了基于树莓派的人脸识别门禁系统通过在教室、实验室等场景门口处安装来实现对出入人员身份的识别。相关神经网络的构建对我们十分具有借鉴意义。而张满青[18]等人基于树莓派开源硬件,通过传感器及调用百度云的语音合成接口引导用户与系统交互,在此基础上利用 face++人脸识别云服务完成学生身份识别。

研究方案

设备配置

我们首先用网线获取 IP 地址，然后打开手机热点，连接了小车和电脑。利用 VNCviewer 可视化界面进行文件传输，通过 putty 命令行操纵 linux 系统。

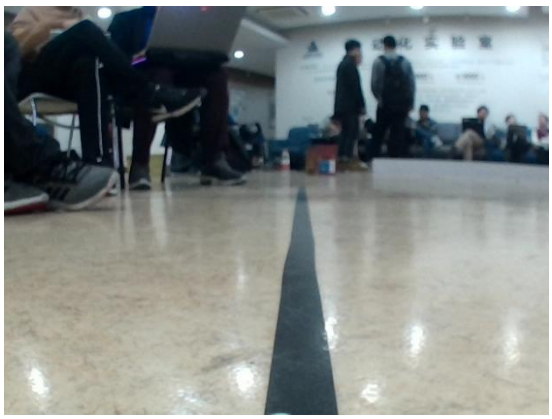
我们在树莓派上配置好环境，主要是 python3，opencv，keras，tensorflow。在安装库时，热点下载速度很慢。希望修改网络配置，通过网线来下载，但是这样做导致环境崩溃，重刷了系统。

driver 类控制中，一条指令有效 3s 左右。若不持续发送指令，小车将会停下。如果程序停止时，没有删除小车类对象，小车将持续运行，需要按键重置下位机。另外，未关闭 VNCviewer 时不可以关闭小车。

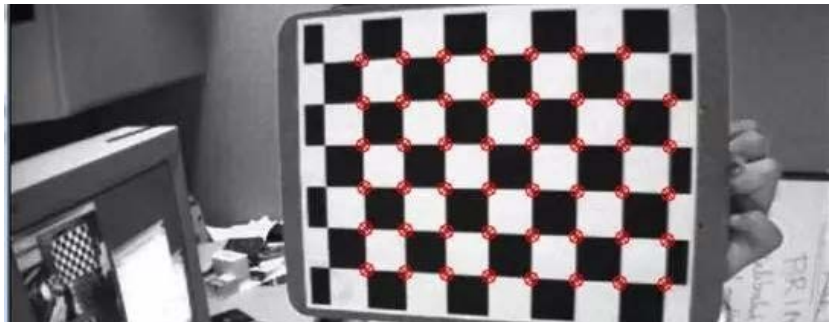
现实环境中，我们在小车前加装手电筒，以增加画面底部的图像对比度。实践中证明，我们的小车在没有手电筒时也能完成任务，但在斑马线识别时，必须要手电筒。

图像预处理

首先分析之前方法的问题，以下图为例，畸变消除的效果如下。



畸变消除的原理是，将一张图的每一个像素，作一个仿射变换，映射到一个新的位置上。具体的映射关系使用正方形棋盘标定。



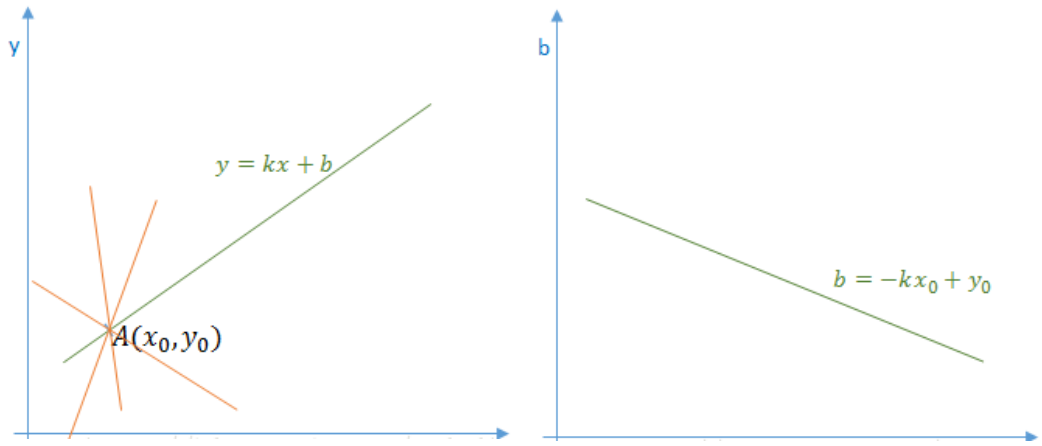
标定系数如下

对于实时系统，做这样一个复杂的逐像素映射，是非常慢的，大概需要 0.5s。而且正如展示的图片所示，这样做的效果并不显著。或许对于精准慢速停车是十分必要的，但是对于快速巡线，就显得很多余了。同理，透视变换也是使用相似的方法，我们发现这一个时间问题后，就没有尝试做透视变换的标定工作。

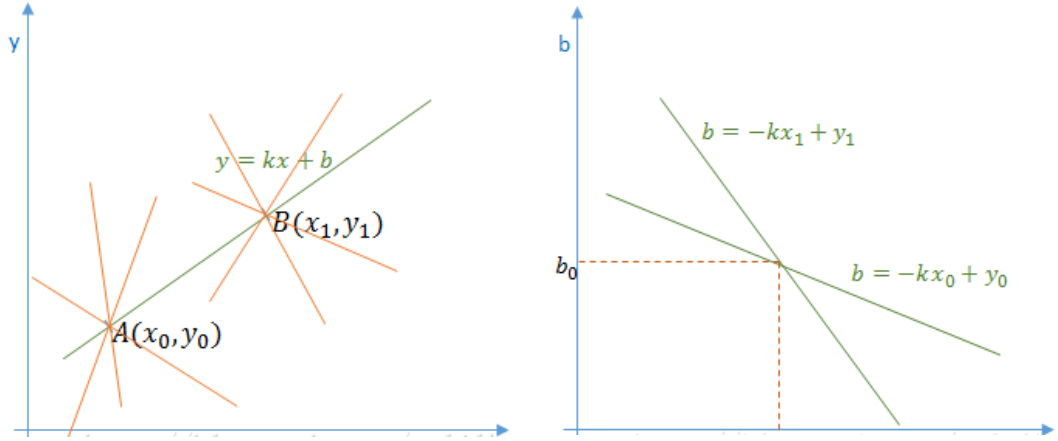
在截取图像时，我们发现把小车放在线上，视野里的黑线并不在中间。因为不知道摄像头可以左右移动，我们通过截取图像来消除了这 60 像素的误差。

另外，我们希望通过 canny 检测+霍夫直线检测提取直线信息。霍夫变换(Hough Transform)是图像处理中的一种特征提取技术，它通过一种投票算法检测具有特定形状的物体。该过程在一个参数空间中通过计算累计结果的局部最大值得到一个符合该特定形状的集合作为霍夫变换结果。霍夫变换于 1962 年由 Paul Hough 首次提出，后于 1972 年由 Richard Duda 和 Peter Hart 推广使用，经典霍夫变换用来检测图像中的直线，后来霍夫变换扩展到任意形状物体的识别，多为圆和椭圆.

首先，我们通过实例来解释一下对偶性的意义。图像空间中的点与参数空间中的直线一一对应 在图像空间 x-y 中一条直线在直角坐标系下可以表示为: $y = kx + b$. 过某一点 A$(x0, y0)$的所有直线的参数均满足方程$y_0 = kx_0 + b$, 即点 A(x_0, y_0)确定了一族直线。如果我们将方程改写为: $b = -kx_0 + y_0$, 那么该方程在参数空间 k-b 中就对应了一条直线。



图像空间中的直线与参数空间中的点一一对应。我们在直线y = kx + b上再增加一个点 B(x_1，y_1)，那么点 B(x_1，y_1)在参数空间同样对应了一条直线。



可以看到，图像空间 x-y 中的点 A 和点 B 在参数空间 k-b 中对应的直线相交于一点，这也就是说 AB 所确定的直线，在参数空间中对应着唯一一个点，这个点的坐标值（k_0，b_0）也就是直线 AB 的参数。这个性质也为我们解决直线检测任务提供了方法，也就是把图像空间中的直线对应到参数空间中的点，最后通过统计特性来拟合直线。

Canny 检测和霍夫直线检测的效果如下。

可以看到，canny 检测得到了太多噪声信息，霍夫直线检测中，我们难以获取真正的直线。

联想到工科创 4D 时的 Stanley 无预瞄算法，我们明白直接检测最底部的黑线位置即可。那么，只需要从中央向两侧搜索黑色，找到即为黑色的位置，思路非常简单。但是黑色的阈值随环境的变化极大，有时是 30 左右，有时是 100 左右，两周时间，我们调好一版参数，下一次来就完全跑不了了。这时候我们明白，阈值分割非常必要，黑线是视野内最黑的部分。

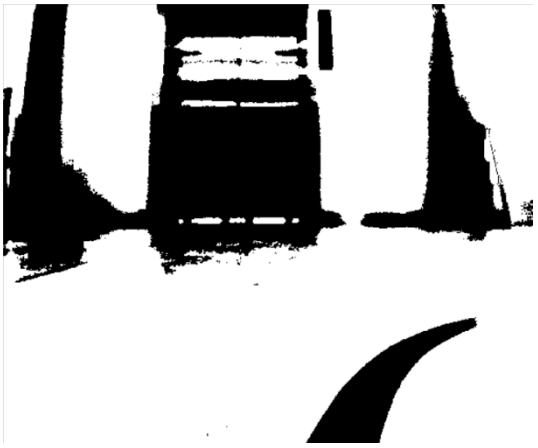
阈值分割法的原理是按照灰度级，对像素集合进行一个划分，得到的每个子集形成一个与现实景物相对应的区域，各个区域内部具有一致的属性，而相邻区域不具有这种一致属性。在本次作业中，我们采用简单阈值法，OTSU 阈值分割和自适应阈值分割。

简单阈值法通过人为设置阈值来把像素级分成若干类，从而实现目标与背景的分离。这种方法的好处是算法简单、直观。坏处是对阈值设置依赖性大，不同的阈值可能得到不同的分割结果。而对复杂图像来说，往往不存在一个全局阈值能够把目标和背景分开，故此方法经验性较强，泛化能力较差

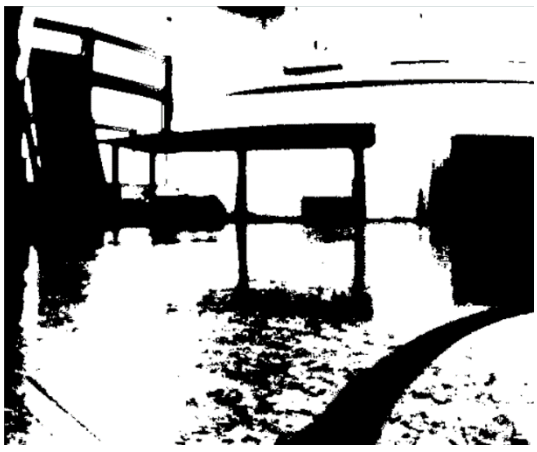
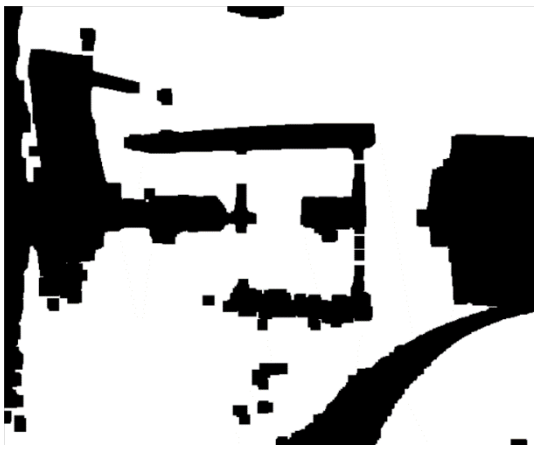
OTSU 阈值分割使用最大类间方差自动确定阈值，将图像分为前景和背景两个部分，最佳阈值分割下两部分之间的差别应该是最大的。前景和背景之间的类间方差如果越大，就说明构成图像的两个部分之间的差别越大，当所取阈值的分割使类间方差最大时就意味着错分概率最小。此算法的阈值是自动选择的，并且在选择标准上更加合理。

自适应阈值可以看成一种局部性的阈值，通过设定一个区域大小，比较这个点与区域大小里面像素点的平均值的大小关系确定这个像素点的情况。自适应阈值分可以处理更为复杂的图象，不要求图象灰度分布为双峰的。算法对邻域大小十分敏感，邻域越大，条纹越粗，图像越连续。

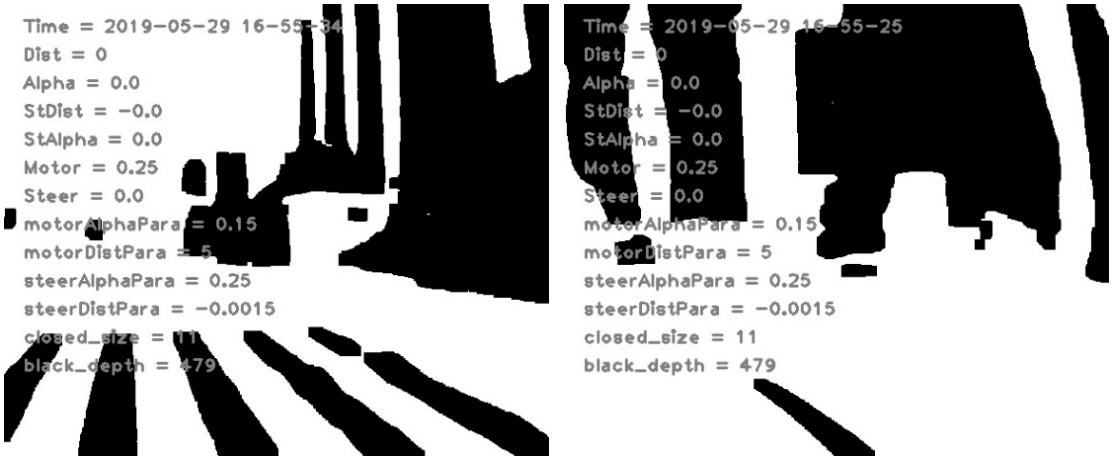
可以看到 OTSU 阈值分割的效果最佳，具有自适应性，也对邻域不敏感，故我们选取此方法。我们进一步使用更多图片进行测试。

可以看到，在这一张对比度不足的图片中，OTSU 阈值分割失效了。为了增强图像，我们将图像的对比度增强，使得黑线更加突出。具体而言，我们使用 opencv 自带的对比度增强函数，然后再将灰度做线性变化。识别出的效果如下左图，已经能得到黑线了，但是地板上的噪声会影响直线检测，于是我们使用形态学闭操作去除这样的噪声（因为二值化图像中黑色部分有效）。闭操作填谷使图像的轮廓变得光滑，但与开运算相反，其作用是使断开部位和细长的沟熔合，填补轮廓上的间隙，消除小孔洞。经过闭操作，视野里的黑线就孤零零地凸显出来了，如下右图所示。

我们在测试时，常会发现汽车走偏。尽管可以输出控制量，但是控制量与实际场景，以及小车看到的图像，往往对不上。于是我编写了一个自动化保存图片的测试函数。在主函数中设置 dosave=True，则自动建立文件夹，保存经过图像处理的图像，并把重要的参数和控制量作为水印加在图片上，如下图所示。



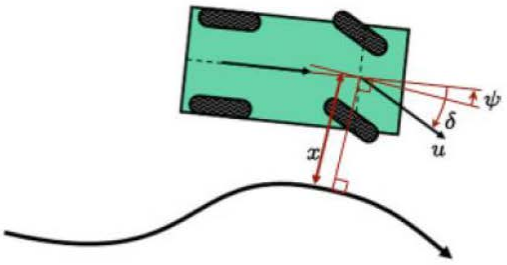
巡线控制

如上所述，因为采取无预瞄的算法，我们仅需要找到底部的黑线位置。我们最初的方案是，选取图像最底部和略上一些的两个高度，开始从中央往两侧搜索黑线。最底部的偏差为x，两预瞄点的斜率为φ，δ为转向的 steer。但是这样做的效果不好，因为预瞄了前方的点，误差较大。同时由于近大远小，这样计算出的φ并非真实的φ，因为黑线可能本身就是弯的。

于是我们改用 PD 控制的思想获取误差。仅在最底部搜索黑线，记录上一时刻的偏差x_0，与这一时刻的偏差x_1的差$x_1 - x_0$即为φ，这一时刻的偏差x_1为x。

从最底部开始搜索，若最底部没有搜索到黑线，则逐步向上搜索，直到一个位置若仍未搜索到黑线，则认为没有黑线，保持上一步控制策略。在搜索黑线时，最开始时步长为 1，保证误差较小时精度较高，步长逐渐增加为 5，保证误差较大时搜索速度较快。

Stanley 算法的公式如下，每一项前面乘以一个系数。$u(t)$为此刻的 motor 值，与 PD 两个误差量均成反比例关系。



$$\delta(t) = \psi(t) + \arctan\frac{kx(t)}{u(t)}$$

我们在使用 Stanley 算法时发现，Stanley 本质是一个横向偏差x的 PD 控制，只不过在数学上优化了公式表达，使之更符合实际场景。相比于有预瞄的其他算

法，我们仅在最底部搜索，误差更小，而且最终控制效果也很好。

我们针对拐弯提出了一种优化的方法，就是在屏幕中央搜索黑线，如果发现即将入弯，那么车辆巡的标准线向入弯的地方偏移一些，走内线能增强鲁棒性，防止车辆跑出弯。实践中我们发现，相比于屏幕底部，屏幕中央的噪声较多，难以准确搜索到拐弯的趋势，调整偏移反而会导致车辆轨迹振荡，D 控制量不准确的问题。

斑马线检测

斑马线检测和交通标志检测本质上是一样的，都是在原始图中匹配特定模版图的过程。我们以斑马线检测为例，阐述我们研究探索的思路与过程。

我们最初的想法是，从原图中提取和模版图一样大小的区域，与原图进行灰度上的均方误差 MSE 计算。但是这种直观的方法并不奏效，准确率召回率均很低。在苦恼之际，我们发现了 opencv 的模版匹配函数 matchTemplate。此函数返回一个和原图一样大的二维数组 p，每个点的值代表以此点为左上角，与模版等大的区域内，在 MSE 意义上匹配成功的概率。

我们对此矩阵较小的值置零，认为此像素匹配无效，为噪声。对于剩下的匹配成功的有效值，我们对其求和，得到最终的置信度。若置信度超过一定阈值，则认为匹配成功，执行响应的控制指令。

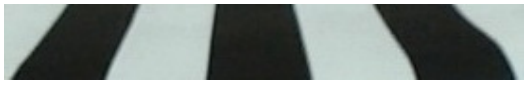
此函数的问题在于，在整个区域内进行逐像素匹配，速度较慢。而且对于尺寸/角度不一样的目标，无法匹配。我们通过缩小范围来加快速度，而第二个性质客观上让我们在识别出模版时，与其距离是个定值。但是，这样一帧的时间从 0.004s 增加到 0.01s，严重影响了正常巡线。我们的解决方法是，每 5 帧检测一次斑马线和交通标志，这样在 80%的时间内，控制速度和巡线无异，保证了精度，延迟 0.02s 再检测也不影响检测精度。同时降低巡线最高速度，保证鲁棒性。

在控制上，识别到斑马线并停下并不难。但是再启动后，无法正常巡线离开。因为视野里有 5 条线，若车身不正，则会慢慢跟随到最外面的线。我们解决方法是在一定区域内，从两侧向内搜索黑线，左右黑线中心点为最终要巡的线。这样可能不会巡最中间的线，但是可以巡一条固定的线，再走一定距离，就是斑马线出口。

交通标志检测

交通标志的识别原理与斑马线识别相同，每 5 帧检测一次，这里不再赘述。识别策略不同的是，我们需要同时匹配三个模版，选择置信度最高返回，若其置信度大于某个阈值，则执行对应决策的控制算法。

斑马线识别的区域在视野下方，交通标志识别的区域在视野上方，模版如下。



我们最初希望使用神经网络，将整张图通过神经网络，输出图像的决策信息为向左，直走，向右，或者没有交通标志。经过查找相关文献，我们了解到整张图的特征过于复杂，神经网络无法提取有效的信息。我们进而希望通过霍夫圆检测，将检测出的图片 resize 后通过神经网络判别。

霍夫圆变换的基本思路是认为图像上每一个非零像素点都有可能是一个潜在的圆上的一点，跟霍夫线变换一样，也是通过投票，生成累积坐标平面，设置一个累积权重来定位圆。在笛卡尔坐标系中圆的方程为：

$$(x-a)^2+(y-b)^2=r^2$$

其中（a,b）是圆心，r 是半径，也可以表述为：

$$x=a+r\cos\theta \quad a=x-r\cos\theta$$

$$y=b+r\sin\theta \quad b=y-r\sin\theta$$

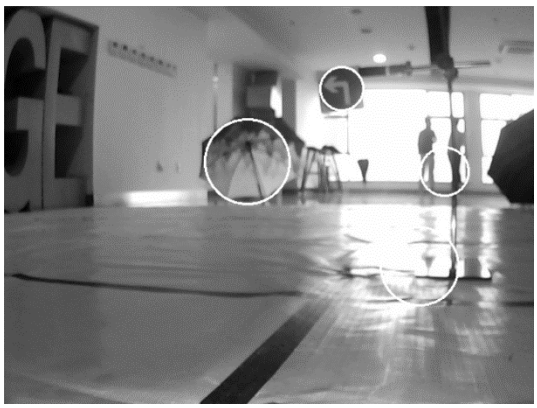
所以在 abr 组成的三维坐标系中，一个点可以唯一确定一个圆。而在笛卡尔的 xy 坐标系中经过某一点的所有圆映射到 abr 坐标系中就是一条三维的曲线。过 xy 坐标系中所有的非零像素点的所有圆就构成了 abr 坐标系中很多条三维的曲线。在 xy 坐标系中同一个圆上的所有点的圆方程是一样的，它们映射到 abr 坐标系中的是同一个点，所以在 abr 坐标系中该点就应该有圆的总像素 N0 个曲线相交。通过判断 abr 中每一点的相交（累积）数量，大于一定阈值的点就认为是圆。

以上是标准霍夫圆变换实现算法，问题是它的累加一个三维的空间，意味着比霍夫线变换需要更多的计算消耗。Opencv 霍夫圆变换对标准霍夫圆变换做了运算上的优化。它采用的是“霍夫梯度法”。它的检测思路是去遍历累加所有非零点对应的圆心，对圆心进行考量。如何定位圆心呢？圆心一定是在圆上的每个点的模向量上，即在垂直于该点并且经过该点的切线的垂直线上，这些圆上的模向量的交点就是圆心。霍夫梯度法就是要去查找这些圆心，根据该“圆心”上模向量相交数量的多少，根据阈值进行最终的判断。

霍夫圆检测的参数繁多，有累加面与原始图像相比的分辨率的反比参数，两个圆心之间的最小距离，Canny 边缘检测的高阈值（低阈值被自动置为高阈值的一半），累加平面对是否是圆的判定阈值，检测到的圆的半径的最大值和最小值。我们通过大量的尝试，调试出一版最佳参数，平衡准确率与召回率，在现场采集的图像中测试效果如下。但我们面临的问题是霍夫圆检测的准确率不高，实地获取素材耗时极高，神经网络的大模型加载，极大地提高了调试的时间成本。我们认为，这样简单的任务并不需要庞大的神经网络来解决。

我们进而尝试另一种方法，将霍夫圆检测出来的圆与标准图计算 MSE 误差。

但是这样匹配的效果，与实际图片中交通标志的灯光情况有很大相关性。我们对模版和识别出的圆进行阈值分割，这样匹配的特征较为明显。但是画面中大面积的黑色，导致全黑图匹配成功的概率甚至比真实交通标志还高，召回率高但准确率极低。结合霍夫圆检测的不稳定性，导致灾难性的效果。所以最终，我们采用了第一段描述的模版匹配方法。




选用了模版匹配的方法后，有两种检测的模式。第一种是不停检测十字路口，检测到十字路口，再检测交通标志，第二种是不停检测交通标志。我们首先选用的第一种方法，判别整行为黑线，识别为十字路口。此方法效果不佳，鲁棒性极差。于是采用模版匹配十字路口，准确率同样低，甚至存在大量的误判情况，因为十字路口的特征，在二值化后的图像中，大量存在。而且，即使检测到十字路口，若没有检测到交通标志，情况将会十分复杂。于是我们选择第二种方法。不停检测交通标志的缺点是要匹配三个模版，时间较长。但是 5 帧一次的速度是可以接受的，效果较好。

在识别出交通标志后，我们不能巡线，而要直接转弯。我们以固定速度固定角度转弯固定时间，如果出弯后仍没有检测到黑线，则保持这样的控制策略。由于检测到交通标志的距离固定，故开环控制的效果较好。只不过由于路线图不对称，我们需要摆放 4 个交通标志，并将其放置在固定的位置。

自选部分研究方案一

按照我们的开题报告所述，我们拟打算实现一个通过键盘直接操控树莓派小车行动并实时获取当前视频界面的 app，以期能够实现复杂陌生环境下动态建图目的。这个项目主要需要解决的技术难点是如何利用树莓派小车获取键盘事件和搭建相应的 app 界面。

在实际操作中我们发现，键盘事件的获取已经有了比较成熟的中断对应表，我们只需要通过简单的 if...else 语句结合 opencv 的 cv.waitKey 函数即可实现相应的控制决策，因此技术上只需通过 appinventor 搭建一个比较美观的 app 界面图即可完成该自选方案。由此我们尝试在 appinvertor 上初步实现了前端工作。

但是，考虑到项目目的是希望能够实现复杂陌生环境下，尤其是希望能够在一些人类不能直接进入的环境下的建图，但是由于我们的小车仅配备了 2D 摄像头，缺乏激光雷达或者深度摄像机等能够反馈回来深度信息的元件，我们认为我们能够在该项目中取得的进展不会很大。综合讨论下来我们认为这个项目并不能完全体现我们的工作量，同时能取得的实际意义也不大，因此我们放弃了这个方向，相关的后端代码也就没有进一步实现下去了。

自选部分研究方案二

我们的第二种想法是尝试实现复杂环境下的特定操作员跟随，这个项目主要需要敲定的是我们的树莓派小车应该识别操作员的哪一个部分，在敲定之后，这个项目主要需要解决的技术难点就转化成了：

1. 如何在一张细节丰富内容复杂的 2D 画面中准确找到所有我们需要识别的对象；
2. 如何在这一堆筛选出来的对象中准确识别我们需要的特定特征；
3. 如何将特定对象在 2D 图像中的信息传递到下游的控制策略。

在这样的 pipeline 指导下我们首先考虑到由于小车摄像头整体的海拔高度偏低，因此在实际情况中更有可能摄像头看到的都是人群的鞋跟和裤腿位置，无疑这是极难进行准确识别的，但我们依旧尝试进行搭建并训练了自己的分类器，想要进行第一步：在一张图像中识别出所有裤腿+鞋跟的对象。

经过简单的神经网络模型搭建和训练，我们初步实现了如下效果：

但好景不长，我很快发现这个模型是不鲁棒的，首先由于网络层数并不深，模型实际获取的语义信息并不是我们所期望的对象信息，而是一种不具有可移植性和衍生性的灰度分布信息，更为致命的是我们发现，当我穿着同样的裤子和鞋子走到另一个距离的时候模型并不能产生作用。这样的问题可以说是直接宣告了我们方案二的死亡。

自选部分最终研究方案

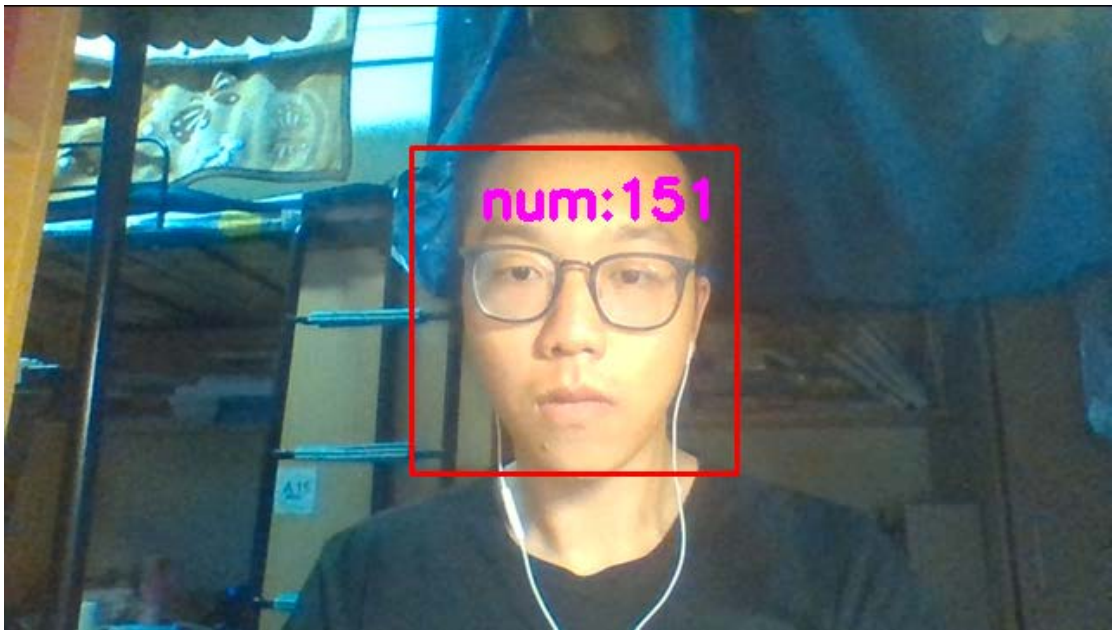
方案三基本上是方案二的改版，区别在于我们开始考虑牺牲一定的实用性转而识别人群中一些更为显著的特征，譬如说人脸，但实际上我们的 pipeline 依然是沿用了方案二中的三个步骤。但在这个定版的研究方案中，我们重新添加了一些功能，或者说，我们主要是想模拟实现服务型机器人的一个重要挑战：如何在复杂人群环境中准确且安全地进行特定操作员的跟随。这里由于我们的方案改组成了去识别人脸，这意味着我们可以使用许多诸如 opencv 这样的比较成熟的依赖包函数去完成我们的第一步：在一张细节丰富内容复杂的 2D 画面中准确找到所有我们需要识别的对象。

对于第二步我们选择了搭建自己的神经网络去识别某张特定的面孔，而对于

第三步识别信息向控制策略的传导和动态避障功能，我们则主要沿用了小车巡线中的思想，采用了一种伪 Stanley 的算法进行小车的方向和电机控制，相关具体的实现思想和方法我将在接下来的段落中进行叙述。

区别于传统的人脸识别与目标跟随，自建神经网络的引入让树莓派小车跟随特定操作员，排除复杂人群的干扰，再加上动态避障的规划让自选项目有了一定的实际意义，理想情况下可以实现服务型机器人的复杂环境下的特定操作员跟随。我横向比较了助教提供的前些年学长们的视频展示，并认为这不过是一个粗浅的 opencv 的应用，且下游控制算法无法实现有效快速反应，远无法实现我们的构思，因此我们进行了大幅的改进。

对于复杂环境下数目不定的人脸检测和特定操作员识别，我们采用 opencv 从当前画面中找到所有人脸，并用自适应框图将人脸截取下来，把这些图片按时间顺序进行编号从而得到了神经网络训练的样本集，在吸取了方案二中失败的经验之后，我们在此次样本的采集过程中要求对象在视频中不断改变自己的远近距离和面向摄像头的角度。相关自适应采集样本的效果如下所示：



其中方框中所示的 num 可以实时显示当前已经采集到了多少样本。

由于此时截出的人脸图大小不一，我们发现如果此时将图片直接输送进入模型，numpy 自带的 reshape 和 resize 只能对矩阵进行操作，也就是只能进行简单的丢弃或者补零，因而这需要我们自己编写 resize 函数对图像进行放缩也就是相应的上/下采样修补，这里我选择的是最简单的均值补齐和平均丢弃，最终在每一次视频流采样得到的画面里返回一批相同规格的人脸图像，效果如下所示。

未经过 resize 操作的原样本

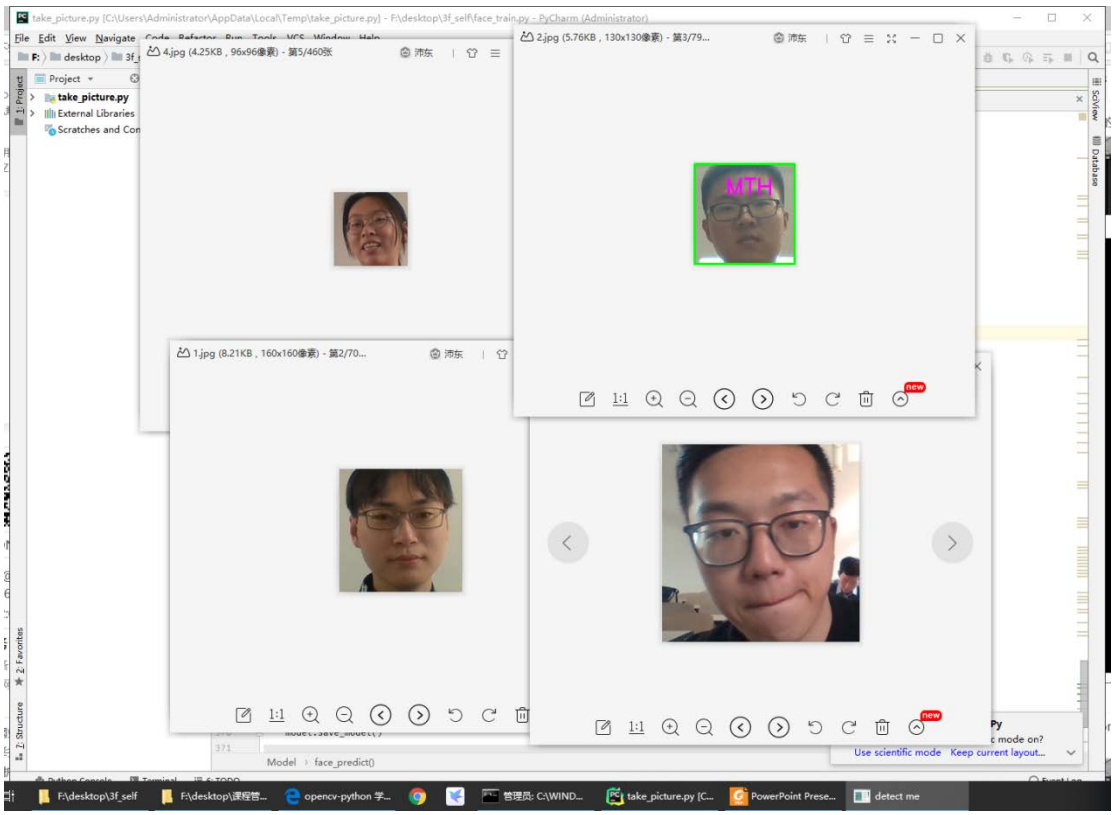
经过均值上采样得到标准大小的样本（示例）

此外，经过方案二失败案例的反思和思考，我们觉得不能完全相信一个环境下采集到的样本，因此我们在代码中加入了对样本集的噪声变换，实现了样本的扩充和模型鲁棒性的增强，以下是打印出了一些简单噪声效果。

此前我们自己用 keras 搭建了一个含四层卷积的简单神经网络，并使用实时视频流人脸截图得到了约 7000 张图像样本(其中包含 1000 个正样本)，事实上，再经过数据提升、样本清洗和样本生成等技术手段之后，样本量的大小实际达到了 20000 余张。采用 categorical crossentropy 作为损失函数，SGD 算法梯度下

降，设置学习率α为 0.01，衰减率 decay 为 1e-6，向量值 momentum 为 0.9，经过 200 个 epoch 训练最终模型得到测试精度 95%。

至此我们基本实现了如何在一张包含丰富细节多个人脸的图像中准确识别特定操作员的人脸，我们以某个同学为例，当一张图中同时出现多个人脸时模型能够准确识别并标注它所“认识”的那张脸，得到的效果如下所示：



每次动态视频流得到的截图经 opencv+神经网络得到目标操作员在图中的坐标，由于前期消除了摄像头畸变的影响我们可以认为该坐标在树莓派小车的相对坐标系下是无误差的。由于本次目标是跟随系统，只需要从图像中提取误差量，传入公式。我们的跟随思想是将得到的操作员坐标与图像中刻线坐标的差值作为方向 steer 参数的正比系数，实际 steer 值为该差值除以当前速度值取正切值再乘以坐标偏差系数，图像的大小与标准阈值的差值作为电机 motor 参数的正比系数。

对于动态避障系统，我们经过论文检索和实践分析发现想要通过单纯 2D 图像的灰度分布还原出类似 RGB 深度图效果实在难以实现，但为了整个自选项目的完整性，我们考虑到当前方出现动态障碍物时我们的小车必然是难以继续检测操作员的，因此当目标丢失（即被障碍物遮挡）时，我们将停下等待动态障碍物离开以实现避障功能。

我们的整体控制采取快速反应策略，每次检测操作员返回的决策是一个相对比较小的控制量，这保证了我们能够实时跟随且面对障碍即目标丢失的情况也能快速停止，有利于整体项目的完整度。

实际参数

图像处理

摄像头偏移：左边多出 60 像素
对比度增强：img = 1.5*img + 100，若灰度溢出，则保留最大值
闭运算模版大小：11*11

巡线控制

```
def constrain(value, threshold_max, threshold_min): return min(max(value, threshold_min), threshold_max)
if alpha == 0 and dist == 0: motor = motorMax
if alpha == 0 and dist != 0: motor = constrain(abs(motorDistPara / dist), motorMax, motorMin)
if alpha != 0 and dist == 0: motor = constrain(abs(motorAlphaPara / alpha), motorMax, motorMin)
if alpha != 0 and dist != 0: motor = constrain(abs(motorAlphaPara / alpha) + abs(motorDistPara / dist), motorMax, motorMin)
steer = constrain(alpha * steerAlphaPara + arctan(steerDistPara * dist / motor), steerMax, -steerMax)
```

motorAlphaPara：0.15
motorDistPara：5
motorMax：0.4
motorMin：0.15
steerAlphaPara：0.25
steerDistPara：-0.0015
steerMax：1

模版匹配

交通标志大小：50*50
交通标志区域：(0, 200, 80, 500)　# up down left right
交通标志概率置零阈值：0.5
交通标志概率求和置信度阈值：30
斑马线大小：40*275
斑马线区域：(360, 480, 0, 580)　# up down left right
斑马线概率置零阈值：0.6
斑马线概率求和置信度阈值：10

人脸跟踪

人脸大小范围：50-200
Motor 线性负相关控制系数：0.333
误差阈值：0.3
steerDistPara：-0.0005

神经网络

神经网络学习率α：0.01
优化器衰减率 decay：1e-6
优化器 momentum：0.9
训练代数：200
精度：95%
网络结构：如下

```
self.model = Sequential()

self.model.add(Convolution2D(32, 3, 3, border_mode='same',input_shape = dataset.input_shape))
self.model.add(Activation('relu'))  # 2 激活函数层

self.model.add(Convolution2D(32, 3, 3))  # 3 卷积层
self.model.add(Activation('relu'))  # 4 激活函数层
self.model.add(MaxPooling2D(pool_size=(2, 2)))  # 5 池化层
self.model.add(Dropout(0.25))  # 6 Dropout

self.model.add(Convolution2D(64, 3, 3, border_mode='same'))  # 7  卷积层
self.model.add(Activation('relu'))  # 8  激活函数层

self.model.add(Convolution2D(64, 3, 3))  # 9  卷积层
self.model.add(Activation('relu'))  # 10 激活函数层
self.model.add(MaxPooling2D(pool_size=(2, 2)))  # 11 池化层
self.model.add(Dropout(0.25))  # 12 Dropout

self.model.add(Flatten())  # 13 Flatten
self.model.add(Dense(512))  # 14 Dense
self.model.add(Activation('relu'))  # 15 激活函数层
self.model.add(Dropout(0.5))  # 16 Dropout
self.model.add(Dense(nb_classes))  # 17 Dense
self.model.add(Activation('softmax'))  # 18 分类
```

研究成果

小车在有黑线的场地上巡线，在 40s 内到达终点。

在布置有标志牌的场地上行驶，小车识别直行、左转、右转、斑马线的标志，识别准确率大于 95%，并同时执行相应的控制指令，在 90s 内完成全程。识别到交通标志时，成功转移到固定的赛道；识别到斑马线时，停车 3s 让行，并能顺利走出斑马线区域，继续任务。

小车在静止状态下识别出数量不定的人脸，准确率大于 95%。小车在多个人脸的视频流中能准确框出预存入的特定人脸，准确率大于 85%。小车能够实现特定人脸的跟随并停止在安全范围内。小车能够实时识别障碍物，在丢失目标时能及时停下。

研究总结

收获

在本次项目中，我深刻体会到了软件仿真与实地测试本质上的不同，恰如控制科所不断强调的那样，在实地测试中干扰的来源、方式和造成的效果是千差万别且很难提前预知的，并且由于我们往往无法判断是什么样的干扰造成了失误，导致我们经常把时间浪费在了错误参数的调试和整定之上，而随之积累的经验大概也让我们成为了专家整定法的重要参与者。

相比于车速，树莓派处理器速度很慢，要权衡算法复杂度与算法效果，简单的算法可能由于速度快而获得奇效。巡线算法单帧 0.005s 左右；交通标志检测速率相同，但每 5 帧有 1 帧为 0.01s，检测交通标志和斑马线；人脸跟踪一帧要 1.5s 以上。我们通过增加搜索黑线的步长，减少交通标志的识别频率，减小交通标志有效的范围，简化神经网络等来加速。

其次，在我们组寄予厚望的自选项目环节上，我们主要是想模拟实现服务型机器人的一个重要挑战：如何在复杂人群环境中准确且安全地进行特定操作员的跟随。我横向比较了助教提供的前些年学长们的视频展示，并认为这不过是一个粗浅的 opencv 的应用，且下游控制算法无法实现有效快速反应，远无法实现我们的构思。因此我们进行了大幅的改进，相关说明已在前文有所体现。我们的感悟是：对于一个复杂需求，它的指标是多维度多方向的，如果在一个维度上精益求精实际上很容易牺牲其他方向上的精度。而对于需求来说，能够完整实现它才是最重要的，因此我们整体采取快速反应策略，保证了项目的完整度和安全性。

在编程实践中，我们也面临了很大的挑战。这是一个很庞大的项目，代码行数超过五百行，参数繁多，接口复杂，如何清晰明了地模块化编程，对测试和调参极为重要。我们看到一个往届代码的失败案例，将 500 余行代码写在一整个 main 函数里，不给人任何研究的兴趣。我们曾经尝试过，将一整个 task 封装到类，所有参数为类内私有变量，这样大量的 self.让画面极为混乱，而且各个接口不明确，所有函数都可能操作变量，无法模块化调试。

我们采取的方法是，将每个小功能集成为函数，不超过 30 行，所有参数都外接为接口。而参数定义为全局变量，在 main 函数循环体内传入函数。最精妙的是，写交通标志识别时，可以从巡线脚本里直接导入所有的函数和参数，仅需要在金字塔上添砖加瓦。如果需要修改或增加参数，同样在全局变量里面操纵。

我们认为，模块化编程中，功能的集成分为，集成到函数，集成到类，集成到文件，集成到文件夹。集成的方法只是途径，目的是为了让程序结构清晰，易于修改，需要频繁修改的部分（如参数）和不需要频繁修改的部分分离。根据人的接受能力，函数不超过 30 行，类与文件不超过 100 行，文件夹内文件不超过 10 个，相对合适。

经过这次项目实践，我们发现实际中运动控制的难度远超我们想象，我们对环境复杂度的估计往往偏低。我们得到了宝贵的现实场景图像处理的锻炼，熟练使用 opencv 库，提高了 python 代码能力。最重要的，我们寻找问题与解决问题的能力得到大幅提升，这在以后科研以及生活中极为重要。

困难

本次研究存在的最大挑战在于，项目的交付需要现场测评，而不同于以往只用选取跑得最好的一次尝试，这意味着我们将关注的重心从如何跑得更漂亮转移到如何跑得更鲁棒。因此我主要提出了一个重要的基于提升鲁棒性的思想：丢失巡线轨迹的处理。由于我们对小车进给量尽可能快速的反应，配合 Stanley 无预瞄的特点，事实上是很容易当小车在转弯时丢失轨迹的，而我们引入 keep 的思想，丢失轨迹时保持上一秒状态直到再次看到线才回归正常巡线状态。这样的思想也在看到信号灯后的开环转弯中取得极大突破，这意味着我们的开环只需转出十字路口到丢失巡线，接下来便转化为闭环问题，大大提升了鲁棒性。

帧率是此次运动控制的核心问题之一，更快的处理速度意味着更高的控制精度，但也意味着算法不能有太高的计算量。最终的单帧速度巡线 0.004s，检测 0.01s，人脸识别 1.5s. 在最初的测试中，我们发现巡线一帧居然需要 2s，不合逻辑。经过排查发现，我们每一帧都重新定义了摄像头，而摄像头的初始化需要很长时间。所以应该先定义摄像头，再从中读取图像。另外，当我们保存每一帧的图片，用于详细观察运动状况时，单帧速度下降一倍，甚至会影响到巡线效果，导致巡线失败。

另外还有，场地限制，我们必须在现场调，回来后只能离线写代码，发现问题与解决问题的闭环延时太长。而且我们要与其他组共用场地，现场没有椅子。场地很迟才搭建起来，为此我们甚至自己买胶带，铁架台，打印交通标志搭建场地测试。

而且小车调试的时间成本很高，一次电脑修改，通过手机热点，传输到小车上运行，需要一分钟。每次开机要重连，热点有些地方不稳定，导致平时电脑上一键解决的问题，被放大了几十倍。且上位机下位机有可能会崩，总之试错成本很高，不得不花的低效时间很多。

致谢

在此，必须特别感谢老师和助教，在课程设计中，给我们良好的设备，以及极大的自由度，没有限制我们的自由发挥，同时布置的琐碎任务也很少，真正打造了一门纯粹而有深度的实践课，让我们受益良多。同时，项目的展示，考核灵活而多元，锻炼了我们从环境配置，架构设计，技术操作，现场测试，汇报展示，书写报告的全方位能力。我们诚挚地建议，在以后的课程中，发完设备就搭建好所有的场地，并详细规定场地物品尺寸以及验收标准。另外，我们认为，课程报告一组一份，大家共同书写打磨，就已足够。

参考文献

[1]边蓓蓓,许琳,秦钟.嵌入式程序设计实训教学探索[J].电脑知识与技术,2019,15(01):122-123.
[2]孙智勇,戴文翔,程文龙.基于树莓派的超声波避障小车[J].电脑知识与技术,2018,14(30):206-207.
[3]冯庆,陈开燕,张冬梅,李佳阳,唐宇朋.基于灰度优化的车道线识别[J].科学技术创新,2019(01):72-73.
[4]鲁通通. 基于神经网络的非线性畸变图像的校正和识别技术的研究[D].中国计量学院,2013.
[5]徐爱昆,康怡琳,冀子超,段律,刘科.基于树莓派摄像头的双轮循迹平衡车设计[J].单片机与嵌入式系统应用,2018,18(12):74-77.
[6]楼怡杭.基于数字图像处理的车牌识别技术[J].电子制作,2019(Z1):72-75.
[7]刘鹏程.基于树莓派的行人检测小车设计[J].软件导刊,2018,17(02):114-116.
[8]孙云云,江朝晖,董伟,张立平,饶元,李绍稳.基于卷积神经网络和小样本的茶树病害图像识别[J/OL].江苏农业学报,2019(01):48-55.
[9]曹振军,景军锋,苏泽斌,张缓缓.基于树莓派的深度学习色织物疵点检测研究[J].棉纺织技术,2019,47(01):11-15.
[10]李凯,彭亦功.基于生成对抗网络的图像识别改进方法[J].计算机工程与设计,2019,40(02):492-495+532.
[11]马治楠,韩云杰,彭琳钰,周进凡,林付春,刘宇红.基于深层卷积神经网络的剪枝优化[J].电子技术应用,2018,44(12):119-122+126.
[12]杨志勇,黄文锋,刘灿.基于树莓派的远程控制智能拍照小车[J/OL].现代电子技术:1-5.
[13]罗佳伟,孙建梅,徐国旭.基于树莓派和深度学习技术的智能遥控车的设计与实现[J].计算机产品与流通,2018(01):185.
[14]刘燕娜,齐迹,林佳涛.基于机器视觉的板球走迷宫系统设计[J].科学技术创新,2018(27):66-70.
[15]朱琳,张凤,申泽轩,夏远满.树莓派 3B 系列的人脸识别实验室门禁系统[J].单片机与嵌入式系统应用,2019,19(04):69-71+76.
[16]余飞. 级联深度卷积神经网络人脸检测应用研究[D].五邑大学,2018.
[17]]秦超,刘正强,刘林,先杨,黎艳,朱倩钰,蒋玲.基于树莓派的人脸识别校园门禁管理系统[J].物联网技术,2019,9(02):13-14.
[18]张满青,唐辉,张慧,张嫚嫚.树莓派和人脸识别的云端智能课堂考勤系统[J].单片机与嵌入式系统应用,2019,19(01):35-37.