第二次实验报告

王诗俊 2015201951　廖钰蕾 2015201953　孟妍廷 2015202009　刘笑 2015201925

2017 年 11 月 4 日

一　目标:

首先替换上一阶段损坏的电池，利用手机蓝牙控制小车行驶。进一步，我们实现了在小车上固定了一个开启着摄像头的手机，手机通过 WiFi 将摄像头所拍画面实时传至电脑端，
电脑端每隔 2s 截一张图，根据自己训练出的模型选择合适的转弯方向，并对图像进行处理，训练小车能够识别红绿色，“红灯停，绿灯行”。

二　收集数据:

1. 数据要求

小车在实际行驶过程中以不同角度靠近墙壁的图片

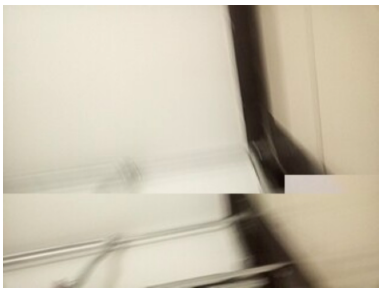
2. 数据获取

⑴ 电脑端安装 DroidCam 软件，手机端（固定在小车上）安装对应 APP，连在同一局域网
⑵ 手机端打开摄像头，电脑端能看到手机拍摄的实时画面（WiFi 状况良好的情况下）
⑶ 电脑端安装 Python 的 opencv 包，编写 py 程序调用此包，程序功能是每隔 2s 截一张图片并保存
⑷ 人为控制：
· 手动控制小车撞墙的角度，保证角度多样化
· 在小车撞墙后，手动将小车与墙壁分离
· 在小车翻车后，手动将小车扶正
· 在小车零件被震翻、线路被震断时，手动恢复

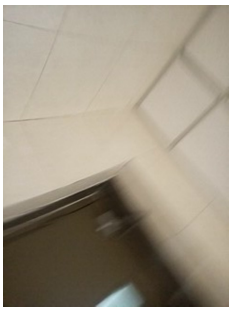
3. 时刻关注电脑端的截图、WiFi 连接状况和截图完成情况

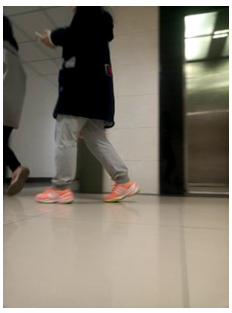
4. 数据筛选

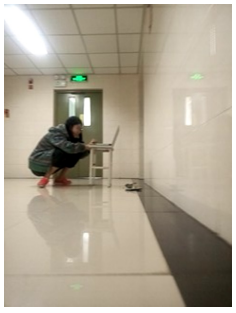
⑴ 剔除了各种“人类无法识别”的图片 (WiFi 刚连接的前几张图片为纯色的、WiFi 连接不稳定时的图片会错位)



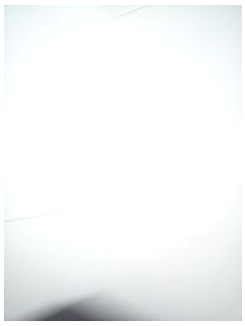
⑵ 剔除了各种“非撞墙”的图片（天花板、楼道里来来往往的人、地板上的小虫子、蹲在电脑前监控拍摄进展的队友）

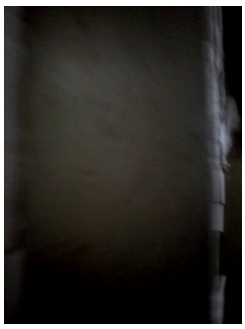






⑶ 剔除了各种“无法标注”的图片（离墙太远（无需转弯）、离墙太近（整个图片都是白花花/灰惨惨/黑漆漆的墙壁））





5. 数据量
原始数据量：2600 张图片（26 个字母、每个字母 100 张）
筛选后数据量：1787 张图片

三　**代码逻辑**:

——训练模型

1. 图片预处理
对图片灰度处理，使用高斯平滑处理图片降噪，再使用 Canny 边检测器检测物体边际

2. 特征提取
用 SIFT(尺度不变特征变换) 来提取特征点，得到 nx128 的特征向量 (n 为特征点个数)

3. 向量量化
⑴ 利用词袋模型对向量进行聚类，将每一簇向量看成一个“词”，将一幅画看成一个“袋”。
⑵ 利用 k-means 算法提取所有图片的 SIFT 特征，若共有 n 个向量，将这 n 个向量划分成 k 类，对每一维 SIFT 特征，均映射到一维上。
⑶ 计算每个图片中的每一类向量出现的频率，做归一化处理。

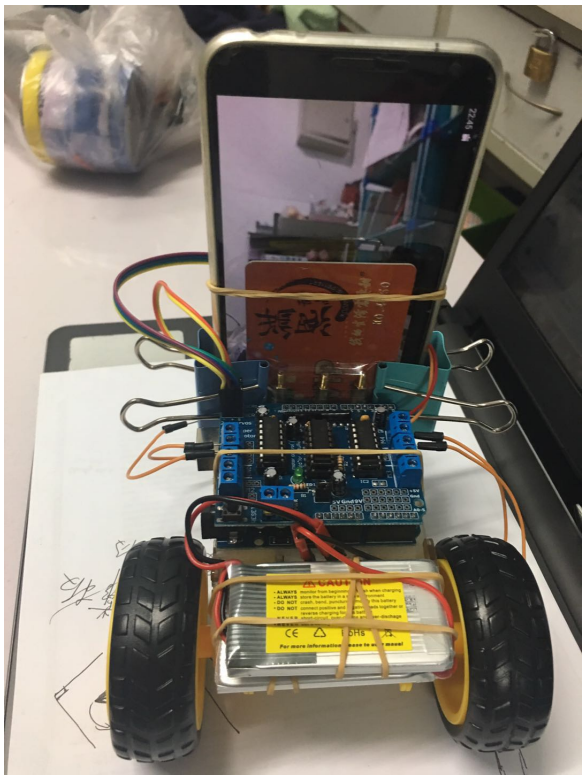
4. 模型调优
采取 SVM 模型, 采用 GridSearchCV 函数在参数范围内自动调优, 寻找最优超参数, 范围如下:
$parameter_grid = [$
$\{'kernel': ['linear'], 'class_weight': ['balanced'], 'gamma': [0.01, 0.001], 'C': [0.5, 1, 10, 50]\},$
$\{'kernel': ['poly'], 'class_weight': ['balanced'], 'gamma': [0.01, 0.001], 'degree': [2, 3]\},$

$'kernel' : ['rbf'], 'class_weight' : ['balanced'], 'gamma' : [0.01, 0.001], 'C' : [0.5, 1, 10, 50],$
]
最优模型保存在“svm.model”中

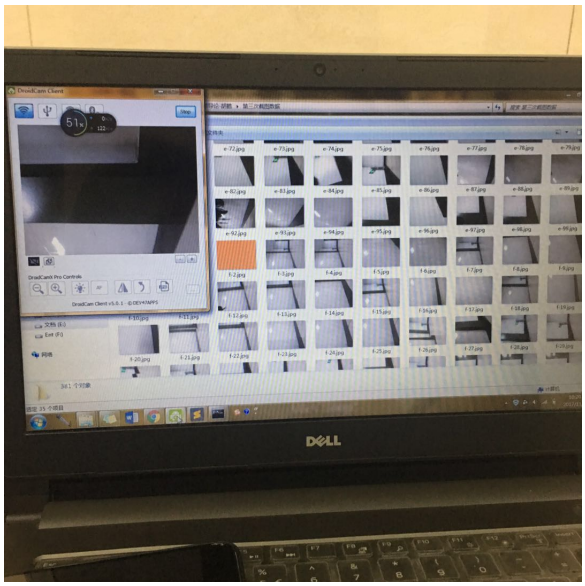
——运行过程
1. 读取模型
2. 实时截取图片，并做红绿判断—将 RGB 图像转化为 HSV 图像，遍历图像判断红绿色彩
3. 发送信号获取传感器距离信息
4. 离墙较近时，对图片进行预处理，特征提取，向量量化等，并使用 SVM 进行分类，决定转弯方向，返回给小车

四　遇到的问题:
1.. 收集数据的效率较低，只能靠人工慢慢控制，数据量较少。
2. 筛选数据时每位成员的标准不同，导致筛选效率不够高
3. 由于 OpenCV 的版本不同，成员在检查代码的时候出现了目前不知道如何调试的错误

五　成果:
1. 小车



2. 收集数据



3. 训练结果

```
终端 文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
#### Searching optimal hyperparameters ####

Highest scoring parameter set:
{'C': 10, 'gamma': 0.01, 'class_weight': 'balanced', 'kernel': 'linear'}

################################

Classifier performance on training dataset

             precision    recall  f1-score   support

    Class-0       0.70      0.62      0.66       283
    Class-1       0.64      0.72      0.68       266

avg / total       0.67      0.67      0.67       549

################################

################################

Classifier performance on testing dataset

             precision    recall  f1-score   support

    Class-0       0.71      0.66      0.68        96
    Class-1       0.65      0.70      0.68        88

avg / total       0.68      0.68      0.68       184

################################
lyl@lyl-Lenovo-B40-45:~/aicar$
```

4. 不同 SVM 得到的结果表

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA
	mean_fit_t	mean_score	mean_test_	mean_train	param_C	param_clas	param_degr	param_gamm	param_kern	params	rank_test_	split0_tes	split0_tra	split1_tes	split1_tra	split2_tes	split2_tra	split3_tes	split3_tra	split4_tes	split4_tra	std_fit_ti	std_score_	std_test_s	std_train_score	
0	0.015465	0.003059	0.653916	0.67349	0.5	balanced		0.01	linear	{'C': 0.5,	6	0.711712	0.666667	0.663636	0.665148	0.654545	0.678915	0.633028	0.670455	0.605505	0.686364	0.000188	0.000324	0.035303	0.007993	
1	0.015462	0.002919	0.653916	0.67349	0.5	balanced		0.001	linear	{'C': 0.5,	6	0.711712	0.666667	0.663636	0.665148	0.654545	0.678915	0.633028	0.670455	0.605505	0.686364	0.000109	4.97E-05	0.035303	0.007993	
2	0.015499	0.002906	0.653916	0.666665	1	balanced		0.01	linear	{'C': 1, '	6	0.693694	0.671233	0.654545	0.665148	0.654545	0.656036	0.66055	0.661364	0.605505	0.679545	0.000227	0.00014	0.02818	0.008126	
3	0.015351	0.002815	0.653916	0.666665	1	balanced		0.001	linear	{'C': 1, '	6	0.693694	0.671233	0.654545	0.665148	0.654545	0.656036	0.66055	0.661364	0.605505	0.679545	0.00019	6.08E-05	0.02818	0.008126	
4	0.019905	0.002712	0.664845	0.680327	10	balanced		0.01	linear	{'C': 10, '	1	0.693694	0.689498	0.672727	0.669704	0.663636	0.669704	0.66055	0.672727	0.633028	0.7	0.00067	9.55E-05	0.019638	0.012285	
5	0.019878	0.002721	0.664845	0.680327	10	balanced		0.001	linear	{'C': 10, '	1	0.693694	0.689498	0.672727	0.669704	0.663636	0.669704	0.66055	0.672727	0.633028	0.7	0.000451	3.87E-05	0.019638	0.012285	
6	0.030547	0.002686	0.661202	0.679869	50	balanced		0.01	linear	{'C': 50, '	3	0.702703	0.687215	0.672727	0.676538	0.636364	0.660592	0.651376	0.668182	0.642202	0.706818	0.002123	4.12E-05	0.024275	0.016121	
7	0.030509	0.002682	0.661202	0.679869	50	balanced		0.001	linear	{'C': 50, '	3	0.702703	0.687215	0.672727	0.676538	0.636364	0.660592	0.651376	0.668182	0.642202	0.706818	0.002119	7.42E-05	0.024275	0.016121	
8	0.018938	0.003648	0.489982	0.49091		balanced	2	0.01	poly	{'degree':	14	0.513514	0.515982	0.481818	0.485194	0.481818	0.485194	0.486239	0.484091	0.486239	0.484091	0.000171	5.71E-05	0.01201	0.012545	
9	0.018853	0.003592	0.489982	0.49091		balanced	2	0.001	poly	{'degree':	14	0.513514	0.515982	0.481818	0.485194	0.481818	0.485194	0.486239	0.484091	0.486239	0.484091	0.000273	6.79E-05	0.01201	0.012545	
10	0.018258	0.003384	0.489982	0.49091		balanced	3	0.01	poly	{'degree':	14	0.513514	0.515982	0.481818	0.485194	0.481818	0.485194	0.486239	0.484091	0.486239	0.484091	0.000186	2.31E-05	0.01201	0.012545	
11	0.018525	0.003653	0.489982	0.49091		balanced	3	0.001	poly	{'degree':	14	0.513514	0.515982	0.481818	0.485194	0.481818	0.485194	0.486239	0.484091	0.486239	0.484091	0.000167	7.09E-05	0.01201	0.012545	
12	0.029025	0.005784	0.489982	0.49091	0.5	balanced		0.01	rbf	{'C': 0.5,	14	0.513514	0.515982	0.481818	0.485194	0.481818	0.485194	0.486239	0.484091	0.486239	0.484091	0.000198	0.000131	0.01201	0.012545	
13	0.029003	0.005824	0.489982	0.49091	0.5	balanced		0.001	rbf	{'C': 0.5,	14	0.513514	0.515982	0.481818	0.485194	0.481818	0.485194	0.486239	0.484091	0.486239	0.484091	0.000313	0.000138	0.01201	0.012545	
14	0.029169	0.005699	0.504554	0.510483	1	balanced		0.01	rbf	{'C': 1, '	12	0.513514	0.515982	0.490909	0.519362	0.527273	0.510251	0.486239	0.484091	0.504587	0.522727	0.000333	3.65E-05	0.014962	0.013825	
15	0.02858	0.005653	0.489982	0.49091	1	balanced		0.001	rbf	{'C': 1, '	14	0.513514	0.515982	0.481818	0.485194	0.481818	0.485194	0.486239	0.484091	0.486239	0.484091	0.000211	8.13E-05	0.01201	0.012545	
16	0.026977	0.005161	0.639344	0.663474	10	balanced		0.01	rbf	{'C': 10,	10	0.693694	0.650251	0.663636	0.656036	0.627273	0.676538	0.642202	0.672727	0.568807	0.656818	0.000199	8.80E-05	0.041643	0.009203	
17	0.028909	0.005699	0.509911	0.50409	10	balanced		0.001	rbf	{'C': 10,	13	0.486486	0.484018	0.490909	0.519362	0.536364	0.510251	0.486239	0.484091	0.504587	0.522727	0.00032	7.83E-05	0.018957	0.0186	
18	0.026157	0.004857	0.655738	0.667574	50	balanced		0.01	rbf	{'C': 50,	5	0.702703	0.666667	0.654545	0.667426	0.654545	0.66287	0.66055	0.661364	0.605505	0.679545	0.000375	0.000142	0.030857	0.00601	
19	0.027432	0.005326	0.626594	0.654369	50	balanced		0.001	rbf	{'C': 50,	11	0.684685	0.659817	0.609091	0.624146	0.645455	0.667426	0.633028	0.668182	0.559633	0.652273	0.000246	4.38E-05	0.041393	0.016179	

5. 小车行驶见视频

六　模型评价:

1. 图片预处理：用 Canny 处理过的图片能明显看到物体边，且降噪效果相对拉普拉斯等边检测器好
2. 每个图片的特征点个数不同，而 SIFT 是 CV 中常用的特征之一
3. 向量量化：在计算机视觉中，SIFT 向量维度较高，运算速度慢，可以在此对向量进行聚类，而词袋模型最初用于文本分析领域，将每个文档看成袋子，根据袋中词语进行分类。又由于每幅图对应多个向量，且个数不定，所以需要做归一化处理。

七　分工:

王诗俊	廖钰蕾	孟妍廷	刘笑
1. 解决第一次展示中的问题，寻觅并购买新电池 2. 组装小车、固定手机 3. 安装软件实现手机端拍摄图片实时传至电脑端 4. 编写截图所用的程序 5. 完成2600张原始图片数据的采集 6. 完成1700+张可用图片数据的筛选 7. 编写主函数与小车通信功能的代码 8. 修改展示用PPT 9. 课堂展示	1. 解决第一次展示中的问题，寻觅并购买新电池 2. 安装软件实现手机端拍摄图片实时传至电脑端 3. 查找有关算法的文献 4. 选择合适算法 5. 编写训练模型的程序 6. 编写主函数判断小车转弯方向的程序	1. 组装小车、固定手机 2. 完成2600张原始图片数据的采集 3. 完成1700+张可用图片数据的筛选 4. 制作展示用PPT初稿 5. 制作成果报告 6. 检查代码	1. 提出本阶段功能目标 2. 查找有关算法的文献 3. 选择合适的算法 4. 安装软件 5. 完成1700+张图片的标注

八　参考文献:

Canny 参考：http://homepages.inf.ed.ac.uk/rbf/HIPR2/canny.html
SIFT 参考：https://www.cs.ubc.ca/ lowe/papers/ijcv04.pdf
k-means 参考：http://www.onmyphd.com/?p=k-means.clustering&ckattempt=1