|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 任务 | 具体要求 | 是否完成 | 总结 |
| 1 | 安装Ubuntu16.04系统  （虚拟机或者双系统） | 学会制作系统驱动盘并安装Ubuntu系统，并熟悉Ubuntu系统常用的命令行操作。 | √ | 我安装的是Ubuntu18.04  旧版本16.04虽说也是也是稳定版，但是在我安装过程，新版安装过程中没有遇到奇奇怪怪的问题 |
| 2 | 安装QT5并配置opencv3.4.3 | 在Ubuntu系统下安装QT5并配置opencv3.4.3 | √ | 我安装的是Qt5.8和OpenCV3.4.7  安装新版OpenCV3没有劣势，只要不是OpenCV4还是可以从网上取得准确适用的学习资源 |
| 3 | 图片读写与显示，调节图像大小 | 1. 配置好QT5与OpenCV后，学习新建QT工程以及如何利用OpenCV进行图片1的读写 | √ | imread（） |
| 1. 用窗口显示所读图片1。 | √ | imshow（） |
| 1. 将图像大小转换为1280\*720 | √ | resize（） |
| 1. 将转换大小后的图片1保存在本地 | √ | imwrite（） |
| 4 | 色彩分割 | 将素材图片1中红色通过色彩分割的方法二值化成黑白图像（目标颜色为白，其余背景为黑） | √ | 通过设置不同的HSV值提取红色，但是红色的定义不同意，提取的部分不一，但提起到的部分大致相同，开连通和闭连通进行优化 |
| 5 | 边缘信息提取 | （1）将色彩分割后的图进行边缘提取，并画出边缘（绿色）,背景为黑色  （提示：色彩分割后用Canny算法） | √ | 在提取到黑白图像之后，用Canny求边缘，再FIndContours/DrawContours进行找边画边 |
| 6 | 视频读写以及摄像头读取 | 1. 读取本地视频并用窗口播放 | √ | Videocapture xxx.open（） |
| 1. 打开手提前置摄像头，并用窗口显示 | √ | Videocapture xxx.open（0） |
| 1. 录制摄像头视频30s并保存 | √ | VideoWriter xxx.write （） |
| 7 | 输出摄像头信息以及调节摄像头 | 1. cout输出的摄像头信息主要包括视频图像大小、帧率（FPS） | √ | FRAME\_WIDTH/HEIGHT/FPS |
| 1. 调节摄像头曝光时间（10、40、80）对比其效果 | √ | PROP\_EXPOSURE |
| 8 | 滑动条应用 | （1）拖动滑动条调节摄像头曝光时间 | √ | createTrackbar（） |
| 9 | 鼠标事件应用 | 1. 用鼠标在图1中框选目标，然后保存框选的图片。 | √ | Rect（） |
| 1. 鼠标拖动过程中要求显示框的线条以及鼠标当前像素点信息（RGB值，坐标） | √ | Img.at<Vec3b>（x,y），（x,y） |
| 1. 拖动完成后单独显示框取的图像 | √ | imshow（） |
| 1. 拖动完成后cout输出框中心像素点坐标 | √ | （（x1+x2）/2，（y1+y2）/2） |
| 10 | 按键事件 | （1）用按键进入不同的处理进程 | √ | switch（） |
| （2）按键数字“1”进入进程1：输出“进程1”按键数字“2”进入进程2：打开摄像头  按键数字“3”进入进程3：imshow图片1 | √ | watikey（）使用的键盘上面的数字键  1按键=49 2按键=50 3按键=51 |
| （3）进入前三个进程其中之一后按“0”退出并且清空所有窗口按0后重新按1、2或3可以重新进入进程1、2或3 | √ | while（1）+switch（key） |
| 11 | 使用SVM和KNN分类 | （1）进行数字识别，要求对鼠标框选的图片进行SVM/KNN分类来判断数字大小 | √ | Ptr<KNearest> model  Ptr<SVM > SVM\_params  训练时，我是用OpenCV Sample->data里面的digits.png进行训练，该图里有5000个20\*20的手写数字，用3000个进行训练，其他用于测试判断准确率 |
| （2）分析处理后Cout输出对应的数字  （图片2作参考 还要找自己去找几张类似的去识别数字）  (SVM 和 KNN都要做一个) | √ | model->predict()  SVM\_params ->predict()  这里两个分类，我暂时没有提取特征，粗暴的把全部像素丢进去,之后在更深入的学习好的提取特征法时会再去修正 |
| 12 | 图像处理 | 以下操作都以图片1为材料  （1）使用高斯混合模型来检测动的物体（以二值图表现）（用笔记本的摄像头，不用图片1） | √ | creatBackgroundSubtractorMOG2()这个算法功能我觉得很有意思 |
| （2）用霍夫直线来检测图片1的灯条 | √ | 霍夫检测直线这个算法深层原理我又去了解，十分感叹想到该算法的人说多聪明，在完成该例子时，我参数调节的不太好，效果不太尽人意。这里我用的是HoughLinesP，进行线段的检测。效果不好 |
| （3）图像增强（低照度增强算法和直方图均衡化各一个） | √ | 这两个算法中，我在课堂上认真学习过直方图均衡化，深层原理我很理解，所以看代码一下子就懂了，至于低照度初步网上学习它的原理 |
| （4）图像的仿射变换（把图像变成一个45度角的平行四边形） | √ | 仿射变换这个算法功能很容易理解，拿三个点，然后求变换矩阵，然后再整图转换，便得到转换后的图，属于透视变换的一子集 |