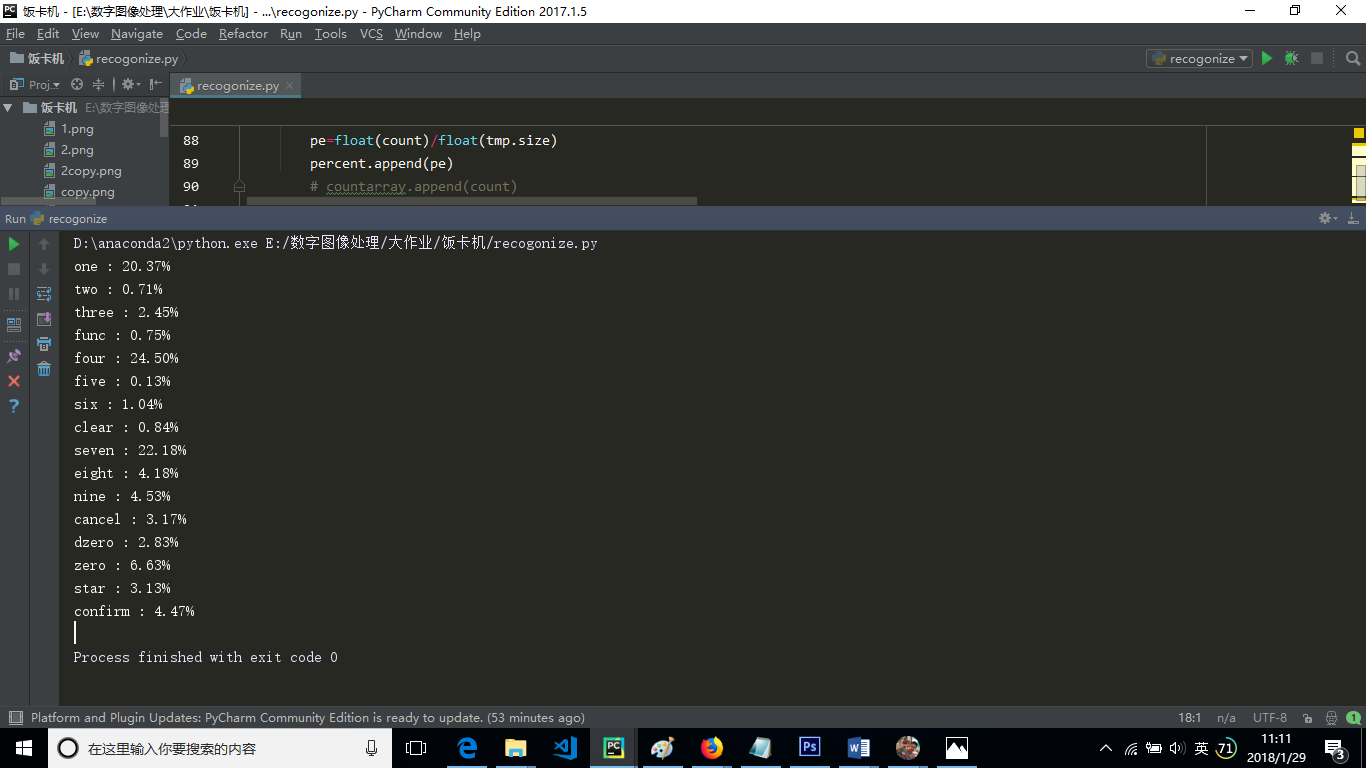
# 键盘磨损评估实验报告

## 1.项目简介

本项目利用opencv,python对校园智能终端键盘进行磨损程度评估。

## 2.安装与使用说明

在支持python，opencv的IDE中打开饭卡机文件夹下recogonize.py文件运行，会显示出磨损掉的部分占未磨损的部分的比例，在5%左右或大于5%的为磨损的，概率越大说明磨损程度越重。



由图片可以看出，1，4，7，0，确认分别有不同程度的磨损。

与实际结果（见下图）高度吻合。（0虽然磨损程度较大，但是没有通过概率显示出来的原因是未磨损键盘的概率是通过手工描点得到的，很难找到没磨损的键盘，所以带来一定的偏差）



## 3.设计文档

## （1）设计思路：

1.图像预处理：对原图进行中值滤波后进行自适应阈值，然后对阈值后的图片再做两次中值滤波，这样做的目的是消除图像中噪声，使得到的效果更好。

2.统计图像中黑色像素点的个数。预处理之后，图像中键盘数字部分变为黑色，为了切割出来每一个键盘的区域，我采用先横着统计每一排黑像素点个数，再竖着统计每一列，得到两个数组。

def jSums(i,j,src):#统计每一行黑像素点数  
 trans=0  
 count=0  
 while trans<j:  
 if src[i,trans]==0:  
 count=count+1  
 trans=trans+1  
 return count  
  
def iSums(i,j,src):#统计每一列黑像素点个数  
 traa=0  
 countt=0  
 while traa<i:  
 if src[traa,j]==0:  
 countt=countt+1  
 traa=traa+1  
 return countt

3.然后对这两个数组进行处理，找到键盘的边缘，我采用的方式是，

对每一排：

if counti[lengthi-1]<5 and counti[lengthi]>5:  
 row.append(lengthi)  
if counti[lengthi-1]>5 and counti[lengthi]<5:  
 row.append(lengthi)

则说明是一个边界

对每一列：

if countj[lengthj-1]<10 and countj[lengthj]>10:  
 col.append(lengthj)  
if countj[lengthj-1]>10 and countj[lengthj]<10:  
 col.append(lengthj-1)

则说明是一个边界

4.由于可能出现边界不明显，而且有些键盘的边儿有两个，这样会导致得到的row,col数组并不是严格的边界，（也就是说没有8条边），利用一个函数对这两个数组进行处理。

def dealarray(a,len):#处理得到的数组  
 tmp=a[0]  
 m=[]  
 m.append(tmp)  
 for i in range(len-2):  
 if a[i+1]-tmp>100:  
 tmp=a[i+2]  
 m.append(a[i+1])  
 m.append(tmp)  
 m.append(a[-1])  
 return m

处理的目的是消除那些间隔很小的数据，得到严格边界。

5.row,col中的元素两两相组合形成十六个区域，对每一个区域中的黑色像素点数再进行统计。（这里代码很长，不在报告中赘述，详见recogonize.py）

6.将最后得到的每个键盘区域的黑色像素点数占整个区域的比重和未磨损的像素点数占整个区域的比重进行比较。（这样做，而不直接用黑色像素点个数，是为了避免由于图片大小不一致导致模板数据不能使用）

## （2）模板的构成：

由于找不到没有磨损的图，我手工将磨损的一些点涂成黑色，由这个图得出的一个数组作为模板去和其他的图比较。



由此图得到的二值化后的图片，可以看出基本跟没磨损的图得出来的相同。



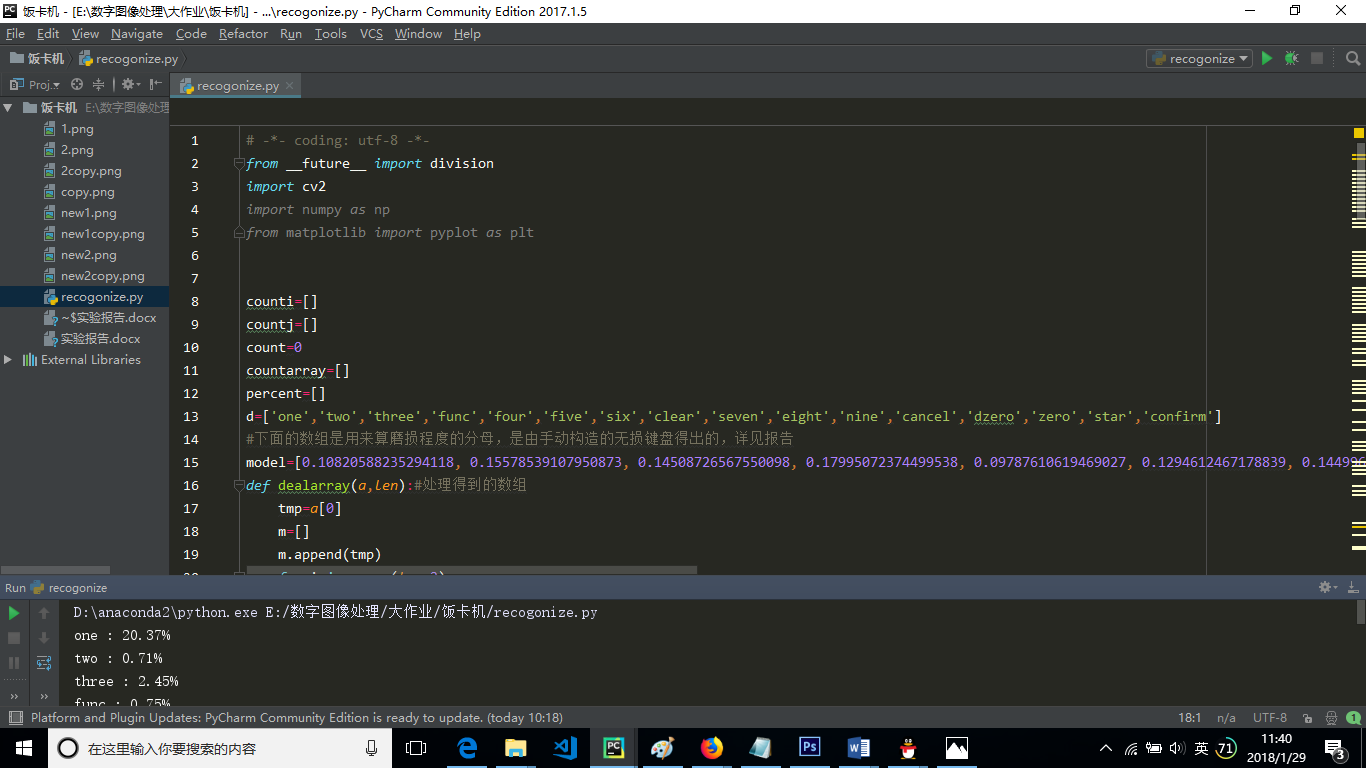
由这个图得到的数据

[0.10820588235294118, 0.15578539107950873, 0.14508726567550098, 0.17995072374499538, 0.09787610619469027, 0.1294612467178839, 0.14499659632402995, 0.15685956539869342, 0.0902122641509434, 0.11494401824590504, 0.14057640472734811, 0.16605749283809149, 0.1308128078817734, 0.134655984409679, 0.1173334055107454, 0.15286410646583964]

这个数据是每个键盘区域内，像素黑点数占整个区域像素点数的比例。

以后这个数组就作为算磨损程度的分母

这个数据就是源代码中的model数组



## 4.特色与创新

本实验让我体会颇多，让我对机器学习在数字图像处理这方面的应用产生了极大的兴趣，虽然中间过程很多数据都是先输出，然后再用某个函数转化成希望的形式，（详见注释掉的代码）。我认为最大的创新就是将所学的知识应用到生活中，并且在图像预处理的过程中，我发现仅仅用一次滤波可能结果并不尽如人意，需要多做几次滤波。而且在切割中，对数组的处理方法，我认为也是本实验的一大特色，相当于去掉数组中的一些“噪声”，即不希望出现的数据，来获得精确的结果。

创新的地方就是没有使用边缘检测，直接人工写函数去找边缘，虽然速度很慢，但是更加精确可控，从而对每个区域的统计更加精确。