#importing libraries

import collections

import Tkinter as tk

import threading

import serial

from collections import defaultdict

from playsound import playsound

import os

#path to audio files

base\_dir = r'C:/Python27/Lib/site-packages/visual/examples/New folder/60%/New folder'

sen\_val = collections.deque()

max\_words = collections.deque()

q = collections.deque()

arduinoSerialData = serial.Serial('com3',9600) #communcation port

#initializing variables

count = 0

ch = 2

compre =[0,0,0,0,0,0,0,0]

word=" "

previous="NO WORD"

ind = [-1]

arr = ["","","","","","","",""]

ar\_val = [0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0]

to\_output=" "

prev\_out=""

flag=0

def myfunc(): #start function

left.start()

right.start()

def capture(direction, nextSource): #sensor value capture function

global count

while(1):

if(arduinoSerialData.inWaiting()>0):

myData=arduinoSerialData.readline()

md = myData.rstrip()

sen\_val.append(md)

count=count+1

#print list(sen\_val)

return

def processing\_func(direction, nextSource): #processing function

while True:

global count,flag

if(count >= 8):

for i in range(8):

compre[i]=int(sen\_val[i])/8 #divide by 8

if(compre[i]<0 or compre[i]>31):

flag=1

sen\_val.popleft()

sen\_val.popleft()

sen\_val.popleft()

sen\_val.popleft()

sen\_val.popleft()

sen\_val.popleft()

sen\_val.popleft()

sen\_val.popleft()

count-=8

if(flag==0):

out\_fun()

flag=0

return

def out\_fun() : #calculate signal value

c=0

for i in range(0,8):

if compre[i]<10:

buf = "%d0%d.txt" % (i, compre[i])

else:

buf = "%d%d.txt" % (i, compre[i])

fo=open(buf,"r+")

#print buf

strg=fo.readline();

while(strg!=""):

c=0

sig=fo.readline();

for j in range(0,ind[0]+1):

if strg == arr[j]:

c=1

ar\_val[j] = ar\_val[j] + 0.02

break

if c==0 :

ind[0]=ind[0]+1

arr[ind[0]]="%s" % strg

ar\_val[ind[0]]=0.02

strg=fo.readline();

fo.close();

trig()

def trig(): #find maximum word array and output word

global to\_output

max\_occur\_word=""

max\_occur=0

#print arr, ar\_val

pos=ar\_val.index( max(ar\_val) )

new\_out=arr[pos].rstrip()

max\_words.append(new\_out)

print max\_words

if(len(max\_words)>=6):

d=defaultdict(int)

for i in max\_words:

d[i]+=1

result=max(d.iteritems(), key=lambda x: x[1]) #find max occurance ie to\_output

max\_occur\_word=result[0]

max\_occur=result[1]/(6\*1.0)

if(to\_output!=max\_occur\_word and max\_occur>=0.5):

disp(to\_output)

for i in range(0,6):

max\_words.popleft()

max\_words.append(max\_occur\_word)

to\_output=max\_occur\_word

elif(max\_occur<0.5):

disp(to\_output)

for i in range(0,ind[0]+1):

ar\_val[i]=0.0

if(len(max\_words)==6):

max\_words.popleft()

def disp(buff): #print function

global prev\_out

if(prev\_out!=buff):

print (buff)

q.append(buff)

prev\_out=buff

left = threading.Thread(target=capture, args=('Left', sen\_val.popleft))

right = threading.Thread(target=processing\_func, args=('Right', sen\_val.pop))

def UI\_label(label): #UI function

end=".mp3"

global word

global base\_dir

def count():

global previous

if(q):

word=q.popleft()

previous=word

filename=os.path.join(base\_dir, word)

filename1="%s.mp3" % filename

abc=filename1.replace("/","\\")

playsound(r'C:/Python27/Lib/site-packages/visual/examples/New folder/60%/New folder/NAME.mp3')#

else:

word=previous

label.config(text=word)

label.after(2000, count)

count()

#UI design format

root = tk.Tk()

root['bg']='snow3'

root.title("Sign language Interpreter")

root.geometry("900x600")

label2 = tk.Label(root,text=" ",bg='snow3', font = "Times 100 italic ")

label2.pack()

label1 = tk.Label(root,text="GESTURE GLOVE", fg="dark blue",bg='snow3', font = "Times 60 italic ")

label1.pack()

label\_space1 = tk.Label(root,text=" ",bg='snow3', font = "Times 10 italic ")

label\_space1.pack()

label = tk.Label(root, fg="dark green",bg='snow3', font = "Times 50 bold")

label.pack()

label\_space2 = tk.Label(root,text=" ",bg='snow3', font = "Times 10 italic ")

label\_space2.pack()

UI\_label(label)

button = tk.Button(root, text='START', width=20,bg="PeachPuff4", fg="white", font = "Times 12 bold " , command=myfunc)

button.pack()

button1 = tk.Button(root, text='STOP', width=20, bg="salmon3", fg="white", font = "Times 12 bold " , command=root.destroy)

button1.pack()

root.mainloop()

#closing port

arduinoSerialData.close()