DOCUMENTATION OF ROSA-i

Author: Ananth (Version1)

Ramyashree B K(Version1.1, 1.4)

Impana S(Version 1.2,1.3)

1. HOW TO SETUP RASPBERRY PI:

HERE ARE THE STEPS TO SETUP THE RASPBERRY PI USING COMMAND PROMPT:

- sudo nano/etc/dphys-swapfile
- Comment or use # in the place of

  CONF_SWAPSIZE=100

  CONF_SWAPSIZE=2048
- 1. sudo apt-get install build-essential cmake pkg-config

  2. sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev

  3. sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev
  4. sudo apt-get install libxvidcore-dev libx264-dev
  5. sudo apt-get install libgtk2.0-dev libgtk-3-dev
  6. sudo apt-get install libatlas-base-dev gfortran
- 1. sudo apt-get install python3-dev
  2. sudo apt-get install python3-pip
- 1. wget -O opencv.zip https://github.com/opencv/opencv/archive/4.1.0.zip
  2. wget -O opencv_contrib.zip https://github.com/opencv/opencv_contrib/archive/4.1.0.zip
  3. unzip opencv.zip
  4. unzip opencv_contrib.zip
- sudo pip3 install numpy
- 1. cd ~/opencv-4.1.0
  2. mkdir build
  3. cd build
  4. cmake -D CMAKE_BUILD_TYPE=RELEASE \
  -D CMAKE_INSTALL_PREFIX=/usr/local \
  -D INSTALL_PYTHON_EXAMPLES=ON \
  -D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib-4.1.0/modules \
  -D BUILD_EXAMPLES=ON ..
- make -j4(This step will take time…)
- sudo make install && sudo ldconfig
- sudo reboot

2. INSTALLATION OF ROSA I

- OPEN BROWSER ON RASPBERRY PI

- LOGIN TO GITHUB (USERNAME: hydaxiso@gmail.com, PASSWORD: Hydax@iso2018)

- Go to hydax hydraulics ROSA I folder
- Copy url
- Open terminal and type code git clone ctrl+v
- New folder will be created named ROSA-i
- Open Rosa-I folder
- Create new folder (Name: train_img)
- Inside this folder create 4 new folders (folder A, folder B, folder C & folder D)

3. Creation of sub-folders in Rosa-i:

create 4 folders, it should be case sensitive with name: folderA, folderB, folderC, folderD.

(This is Optional)

- installation process of raspberry pi OS in a new sd card:

.

- 1.https://www.raspberrypi.org/documentation/installation/installing-images/
- download this image
- run
- select os: raspberry with recommended software
- select storage: memory card through card reader or pen drive
- write

.

- The OS will be installed and ready for the boot up and start

.

- -->ANY ISSUES WITH THE RASPBERRY PI, FOLLOW THIS INSTRUCTIONS:
- https://learn.pimoroni.com/tutorial/pi-lcd/getting-started-with-raspberry-pi-7- touchscreen-lcd

.

- -->if PIL, imageTk, Image gives error,
- go to the terminal
- check python version
- it should be above 3.0 version
- if yes , follow the two commands
- -- - - *- - - - --
- python3 -m pip install --upgrade pip
- python3 -m pip install --upgrade Pillow

- *----------

4. BOOT PROCESS:

Things to do to run the program on bootup :

- Modify the. bashrc file
  To do that, open terminal type:

  sudo nano /home/pi/.bashrc

  Then a screen appears, which is bashrc file

- We need to edit that file; we can follow the following steps.
- Just type,

  echo running at boot

  sudo python3.7 /home/pi/rosa---i/updation_for_rosa-i.py

  These were the commands ,that to be used .

  here /home/pi is default. rosa---i/updation_for_rosa-i.py is my folder .

- U can direct it to ur python path ,by just clicking the path shown in destination file.
- Just after running two command lines , press CTRL+x Y enter.
- sudo reboot

Before all,

- check the python version, if not upgraded. kindly upgrade to latest version or at least python 3 and above.
- install

  pip install pip

  sudo apt-get install python3-pil.imagetk (if you are using python3 and above).

- Hopefully, this will clear all the import errors.

Important steps to follow when dealing with os:

- On the top of the program, let's say line one type: -
  - #! /usr/bin/env
  - python import os

os.chdir("home/pi/rosa-i/") [inside brackets, we need to provide path of the file.]

5. THE ROSA-i CODE EXPLAINATION:

This is a program which uses python with tkinter framework for GUI.

1. Imported all dependencies required for ROSA-i.

2. First three lines is for bootup process, it is directly linked to Linux storage
3. Setting up the tkinter UI window to the required size.
4. Initialized all the photoIMAGE, for UI
5. Then we defined function, according to the four main parts: TEACH, OPEN, TOOLS and HELP
6. Let's move to the TEACH option.
    - Here we are declaring global variable for some photos, which is done because sometimes the photos goes to
    - Then, made a window which can fit to the screen with labels as garbage collector.per the requirement
    - Coming to the:: pathA:: and remaining paths, we have initialized the path as iteration path because in future we will be using glob.glob
    - window. Destroy is used to destroy the window
    - Then each button has specific function which is to be executed according to the need. below will be discussing about the button functionalities.
    - When teach is pressed, the user can have only four snaps as per this rosa-i (ver 1.0) software.
    - There are 4 folders, where we can save the snaps in the required folders. Choosing it takes to the particular folder creates file as 1,2,3,4. After this, it starts estimating the nonblack pixels going into the folder
    - Def est1():
        - here, i created two empty list, applying for loop over path folder using glob.glob
    - The reason i=2, because we have comparing first image and other

three imagesin that folder in for loop.

- By using subtract (photo1, photo2), we are able to extract the non-colliding pixels out.

```
if photo1.shape==photo_all.shape:
    diff = cv2.subtract(photo1,photo_all)
    non_black_pix = np.sum(diff!=0)
```

- Diff!=0, extracts all the non-white pixels and assigned to non_black_pix.
- We need to have 4 comparisons, 1-2,1-3,1-4,2-3
- Use the same way of reading for comparison and append the values in the empty list NonBP=[]
- Then we need to find standard deviation, use the formulas with help of code.
- Subtracting the mean from each element and taking its square.All these operations are performed in below:
- Variance by raising it to the power of 0.5

```
Mean = sum(NonBP)/len(NonBP)
#Var = sum(pow(i_v - Mean,2) for i_v in NonBP)/len(NonBP)
# Std_dev = math.sqrt(Var)
Var=sum((i_v - Mean)**2 for i_v in NonBP)/len(NonBP)
Std_dev=Var**0.5
print("Mean is ",Mean)
#calc.append(Mean)
print("Std_dev",Std_dev)
#calc.append(Std_dev)
Uthreshold = Mean+Std_dev
Lthreshold = Mean-Std_dev
print("Uthreshold",Uthreshold)
calc.append(Uthreshold)
print("Lthreshold",Lthreshold)
#calc.append(Lthreshold)
```

- Append the uthreshold to the calc[], so that we can parse and put it in the csv file.

format. Path = (initialize a new if do not exist)
with open (Path,"w", newline="") as file:
(any name) writer = csv.writer

writer.writerow(row_list)

- To create csv file, follow the Follow the same pattern to create similar functionalities with help of function.
- Checking the file if it contains any file then disable. The procedure is this. line from 641 to 673.

- OpenCV camera initialization. And then changing its frame to black and white format and linking the path to the file.
- End of teach option.

7. OPEN window:
    - Here, we have test window and a folder option.
    - We have to select the folder and then press test button to test the testing object with the master object.
    - Here, we have created the choose the folder function, where we have created the empty list
    - We are comparing the first image of any of the folders present in the directory (which is defined as path or set according to the path) to the test image.
    - Test image is captured when the test button is pressed.
    - The value of uthreshold is stored in csv file .so to read that csv file and used in as if nonblack pixels >uthreshold or not, then resulting in Accepted and Rejected accordingly.

    - So, reading the csv file happens from these lines, were before these lines initialization and path is set to respective folders.

```
#i need to load csv file here
filee = path_for_csvfile
print(filee,"this path is for csv file")
with open(filee,"r") as letscsv:
    csv_reader = csv.reader(letscsv)
    for line in csv_reader:
        print(line,":")
        print(len(line))
        compute_value=line.pop(0)
        print("compute_value",compute_value)
        Uthreshold = int(float(compute_value))
        print(type(Uthreshold))
        break
```

Comparing of images is done as we did in teach screen,

```
if imggrayed.shape == template.shape:
    difference = cv2.subtract(imggrayed,template)

    x= np.sum(difference!=0)
    #print(difference.shape)
    print("non_black_pixels",x)
    non_black_pixels = x.item(0)
    print(type(non_black_pixels))
```

- According to the nonblack pixels value from teach window -any folders file –first image (which is done by reading the path of that path and assigning to any random variable to the test image path (difference value of them)

```
ff = (fla+"/Result.csv")
#ff=("/home/pi/ROSA---i/train_images/folderA/Results.csv")
with open(ff,"a",newline="") as file:
    if non_black_pixels > Uthreshold:
        print("non black pixel is greater than uthreshold")
        circlee= PhotoImage(file="rejectl.png")
        difference = abs(non_black_pixels - Uthreshold)
        quotient = difference/Uthreshold
        percentage = quotient * 100
        #print("percentage:",  percentage)
        formatted_num = format(percentage, ".2f")
        l4.configure(text="" + str(formatted_num)+ "%")
        circleebtn=Button(window2,image = circlee,border=0,bg="w
        circleebtn.image = circlee
        circleebtn.place(relx=0.4,rely=0.5,relwidth=0.27,relheig
        count()
        hello()
        today = datetime.now()
        dl = today.strftime("%d/%m/%Y %H:%M:%S")
        row_list = ['Rejected',formatted_num,dl]
        writer = csv.writer(file)
```

- Create a Reslut.csv to store result of test image.
- if non_black_pixels > Uthreshold:

  then its rejected so i have given the image which is rejected image on the live video (which is done by OpenCV). Find the percentage of NonblackPixels.

Call **count()** function (refer below image)to count the rejected images and store the result with current date and time in csv file.

```
def count():
    global buttonclick
    global l3
    buttonclick += 1
    l3.configure(text= " 0" + str(buttonclick))
```

```
else:
    print("this statement is for lthreshold")
    circle2=PhotoImage(file="acceptl.png")
    difference = abs(non_black_pixels - Uthreshold)
    quotient = difference/Uthreshold
    percentage = quotient * 100
    #print("percentage:",  percentage)
    formatted_num = format(percentage, ".2f")
    l4.configure(text="" + str(formatted_num)+ "%")
    circle2btn=Button(window2,image = circle2,border =0,bg="
    circle2btn.image = circle2
    circle2btn.place(relx=0.4,rely=0.5,relwidth=0.30,relheig
    count1()
    accep()
    today1 = datetime.now()
    d2 = today1.strftime("%d/%m/%Y %H:%M:%S")
    row_list = ['Accepted',formatted_num,d2]

    writer = csv.writer(file)
entdatetime = datetime.now()
    writer.writerow(row_list)
    window2.update()
    window2.after(2000,delete_img2())
```

Else:

Accepted

accepted image is shown as per the validation. Find the percentage of nonblackpixels of an Test Image.

Call **count1()** function (refer below image) to count the rejected images and store the result with current date and time in csv file.

```
def count1():
    global buttonclick1
    global l1
    buttonclick1 += 1
    l1.configure(text= " 0" + str(buttonclick1))
```

- Again, the validation tag will only remain in the window up to required time and the window refreshes. So, line 801-804 does that.
- The test image is automatically deleted after every first image, and the line 828-831.
- We have set the live camera and pic resolution to 320x240
- OpenCV live camera is opened using the same syntax and stored in the file using the imwrite inbuilt function from OpenCV.

8. TOOLS WINDOW.

- Setting window, labelling, photoimage and function creation goes as above code.

ROSA-I V1.2

Author's Name:IMPANA S

Pyttsx3 is a text-to-speech conversion library,it works offline and is compatible with both Python2 and Python3

- Steps to install pyttsx3 are as follows:
- pip install pyttsx3
- Along with pyttsx3,install libespeak as
- sudo apt-get install libespeak

Mainly,import pyttsx3 is defined,the code for application of voice is given below:

```
def too():
    engine=pyttsx3.init()
    engine.say("Welcome to rosaa-i")
    engine.runAndWait()
too()
```

A function is defined,where pyttsx3.init() will get a reference to a pyttsx3.engine.say is where a text message is written to speak .Here a text is written as "Welcome to rosa-I" is defined where the output is in the form of speech.

Again,here the function which is defined is called.

- In case1 of Rejected or Accepted the code is as follows:

```
    def hello():
        engine=pyttsx3.init()
        engine.say("Rejected")
        engine.runAndWait()

    hello()
```

A function called as hello() is defined ,pyttsx3.init() is used to get a reference to a pyttsx3 where the text called as "Rejected" when this code is executed a speech called rejected is heard to you.

- In case2 Accepted the code is as follows:

```
def welcome():
    engine=pyttsx3.init()
    engine.say("Accepted")
    engine.runAndWait()
welcome()
```

Here,a function is define where,pyttsx3.init() is used to get a reference to pyttsx3 where the text called as "Accepted" when this code is executed a speech called accepted is heard to you.

- The audio is even given to teach which is as follows:

```
def teach():
    engine=pyttsx3.init()
    engine.say("teach")
    engine.runAndWait()
teach()
```

Here,a dialog box is displayed where upon clicking on the teach button a voice is given to it simultaneously as a teach button is clicked.

- The audio is given to the open which is as follows:

```
def whenopen():
    engine=pyttsx3.init()
    engine.say("open")
    engine.runAndWait()
whenopen()
```

Here,a dialog box is displayed where upon clicking on the open button a voice is given to it simultaneously as an open button is clicked.

- The audio is given to the help which is as follows:

```
def whenhelp():
    engine=pyttsx3.init()
    engine.say("help")
    engine.runAndWait()
whenhelp()
```

Here,a dialog box is displayed where upon clicking on the help button a voice is given to it simultaneously as an help button is clicked.

- The audio is given to the tools which is as follows:

```
import pyttsx3

engine=pyttsx3.init()
engine.say("tools")

engine.runAndWait()
```

Here,a dialog box is displayed where upon clicking on the tools button a voice is given to it simultaneously as an tools button is clicked.

- For the selection of folder where it asks the user to choose the folder,an audio is given to it which is as follows:

```
def choosethefolder():
    engine=pyttsx3.init()
    engine.say("choose folder")
    engine.runAndWait()
choosethefolder()
```

- An audio is given to instruct the user to choose a folder.

ROSA-I CODE EXPLANATION(v1.3)

Author's Name:IMPANA S

```
# Simple demo of reading each analog input from the ADS1x15 and printing it to t
from subprocess import call
import time
import RPi.GPIO as GPIO
import os
import Adafruit_ADS1x15

pi_gpio_out_09 =21
pi_gpio_out_22 =15

GPIO.setmode(GPIO.BOARD)

GPIO.setup(pi_gpio_out_09, GPIO.OUT)
GPIO.setup(pi_gpio_out_22, GPIO.OUT)

# Create an ADS1115 ADC (16-bit) instance
adc = Adafruit_ADS1x15.ADS1115()

# Or create an ADS1015 ADC (12-bit) instance.
```

- We,need to import packages such as Adafruit_ADS1x15
- To install the packges of Adafruit_ADS1x15 in terminal window

  I.e:pip3 install Adafruit_ADS1x15

  The above packages will be successfully installed

- Certain procedures need to be followed for the proper execution of battery monitoring code go to the terminal:
- 1) sudo apt-get install cron
- 2)crontab –e
- It will ask you to choose 1-3[1]:enter 1
- A window opens you need to mention your file path where the file is present
- Ex:@reboot/home/pi/ROSA---I/raspberry_pi_GPIO_with_RTC_ADC_UPS/battery_monitoring.sh
- Then save it and exit

```
GAIN_BATTERY =2/3
RATIO_BATTERY =0.1875
print('Reading ADS1x15 values, press Ctrl-C to quit...')
print('| {0:>6} | {1:>6} | {2:>6} | {3:>6} |'.format(*range(4)))
print('-' * 37)
```

- See table 3 in the ADS1015/ADS1115 datasheet for more info on gain
- It is used to print nice channel column headers

```
# Main loop.
def task1():
    while True:
        GPIO.output(pi_gpio_out_09, True)
        values = [0]*4
        value_Current =[0]*4
        value_Voltage =[0]*4
        value_Voltage_Battery =0
        value_battery_channel =0
        values[1] = adc.read_adc(1,1)
        value_battery_channel = adc.read_adc(0,GAIN_BATTERY)
        time.sleep(0.9)
        value_Current[1] =((values[1]*0.125)/100)+0.0289999999999999
        value_Voltage[1]=(values[1]*0.125/1000)
        value_Voltage_Battery=(value_battery_channel*RATIO_BATTERY/1000)+((value_battery_channel*RATIO_BATTERY/1000)*0.24)-.03
        #Print the ADC values.
        print(round(value_Voltage_Battery,2))
        if round(value_Voltage_Battery,2)<=3.9:
            GPIO.output(pi_gpio_out_22, True)
            call("sudo shutdown -h now", shell=True)

        else:

           GPIO.output(pi_gpio_out_22, False)
           time.sleep(0.5)

    t1=threading.Thread(target=task1,name='t1')
    t2=threading.Thread(target=task2,name='t2')

    t1.start()
    t2.start()
)
```

- GPIO.output(pi_gpio_out_22,False)
- Read all the ADC channel values in a list
- For I in range (4)
- It reads the specified ADC channel using thr previously set gain value
- Note you can also pass in an optional data_rate parameter that controls
- The ADC conversion time(in samples/second).Each chip has a different set of allowed data rate values,datasheet table 9 config register
- DR bit values
- Values[I] =adc.read_adc(I,gain=GAIN,data_rate=128)
- Each value will be a 12 or 16 bit signed integer value depending on the ADC(ADS1015=12-bit,ADS1115=16-bit)
- time.sleep(0.5) indicates a pause for half a second
- print(round(value_Current[1],2))
- Multithreading concept is used to perform multiple functions at the same time.

To Reduce Flickering

```
while True:
    _ , frames = cam.read()
    img1 = cv2.cvtColor(frames, cv2.COLOR_BGR2GRAY)
    edges=cv2.Canny(img1,200,180)
    edgesvid = ImageTk.PhotoImage(Image.fromarray(edges))
    #edgesvid1 = ImageTk.PhotoImage(Image.fromarray(edges))
    l1["image"] = edgesvid

    window.update()

    if cv2.waitKey(1)& 0xFF==ord('q'):
        break
cam.release()
cv2.destroyAllWindows()
```

- Here,is the code where the values are altered at cv2.Canny(img1,200,180)
- In case of teach,to avoid the entire screen from flickering at a rapid rate.

For Open:

```
while True:
    _,frame =cam.read()

    hellovid= cv2.cvtColor(frame,cv2.COLOR_BGR2RGB)
    lower_red = np.array([30,150,50])
    upper_red = np.array([255,255,180])

    mask = cv2.inRange(hellovid,lower_red,upper_red)
    res = cv2.bitwise_and(frame,frame,mask=mask)
    edges1 =cv2.Canny(hellovid,200,180)
    edgesvid2 = ImageTk.PhotoImage(Image.fromarray(hellovid))
    edgesvid1 = ImageTk.PhotoImage(Image.fromarray(edges1))
    bt1["image"]=edgesvid1
    key = cv2.waitKey(1) & 0xFF
    if key==32:
        break
```

- Here,is the code where the values are altered at cv2.Canny(img1,200,180)
- In case of Open,to avoid the entire screen from flickering at a rapid rate