DOCUMENTATION OF ROSA-i

Author: Ananth (Version1)

Ramyashree B K(Version1.1)

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 garbage collector.
    - Then, made a window which can fit to the screen with labels as 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.

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

- 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.

```
#1 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="reject1.png")
        circleebtn=Button(window2,image = circlee,border=0,bg="white")
        circleebtn.image = circlee
        circleebtn.place(relx=0.4,rely=0.5,relwidth=0.27,relheight=0.12)
        count()
        today = datetime.now()
        d1 = today.strftime("%d/%m/%Y %H:%M:%S")
        row_list = ['Rejected',d1]
        writer = csv.writer(file)
#currentdatetime = datetime.now()
        writer.writerow(row_list)
        window2.update()
        window2.after(2000,delete_img1())
```

- 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). 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="accept1.png")
    circle2btn=Button(window2,image = circle2,border =0,bg="white")
    circle2btn.image = circle2
    circle2btn.place(relx=0.4,rely=0.5,relwidth=0.30,relheight=0.12)
    count1()
    today1 = datetime.now()
    d2 = today1.strftime("%d/%m/%Y %H:%M:%S")
    row_list = ['Accepted',d2]
    writer = csv.writer(file)
rentdatetime = datetime.now()
    writer.writerow(row_list)
    window2.update()
    window2.after(2000,delete_img2())
```

Else:

Accepted

accepted image is shown as per the validation. 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.