

PROGRAMA PYTHON

```
import serial#Llibreria per comunicacions per port sèrie
import time #Llibreria per poder fer delays
import cv2 # Llibreria OpenCv, per processar imatges
import glob #llibreria per accedir a carpetes i arxius
import os #llibreria per obtenir dades dels arxius de les carpetes
import numpy as np #Llibreria per operar amb matrius i operacions matemàtiques
import sys #Llibreria que ens permet fer el shutdown del programa

ts=0 #variable que guarda la data de la creació de la foto anterior
comptador=0 #Variable per comptar fotogrames negres
rang_negre_baix = np.array([0,0,0], dtype=np.uint8)#Establim límit mínim màscara
rang_negre_alt = np.array([10,10,10], dtype=np.uint8)#Límit màxim màscara
captura=0 #Variable per indicar que encara s'estan fent fotos
foto=1#Variable per saber la foto en la que estem
x_tot=0 #Valor acumulat de la coordenada x del rectangle delimitador
y_tot=0 #Valor acumulat de la coordenada y del rectangle delimitador
w_tot=0 #Valor acumulat de l'ampada w del rectangle delimitador
h_tot=0 #Valor acumulat de l'alçada h del rectangle delimitador

#Demanam informació a l'usuari per pantalla

print("Introdueix el port COM de l'arduino:")
Port=input()
print("Introdueix la ruta de la carpeta d'origen:")
Carpeta_origen = input()
print("Introdueix la ruta de la nova carpeta:")
Carpeta_nova=input()

arduino=serial.Serial(Port,9600)#Establim el port sèrie de l'arduino

while captura==0: #Mentre estem en el mode captura
    list_of_files = glob.glob(r'+Carpeta_origen+'*.jpg')#Llegim arxius jpg
    if list_of_files!=[]:#Si la carpeta no està buida
        latest_file = max(list_of_files, key=os.path.getctime)#Agafem l'últim arxiu
        imatge=cv2.imread(latest_file)#Llegim la imatge
        mask = cv2.inRange(imatge, rang_negre_baix, rang_negre_alt)#Creem màscara
        moments = cv2.moments(mask) #Generem el moment dels píxels dintre del rang
        area = moments['m00'] #Calculem l'àrea dels objectes que detecta la càmera
        if area >= 5300000000.0:#Si detecte un fotograma completament negre
            comptador=comptador+1 #Incrmentem comptador
            if comptador==3:#Si detecta tres fotogrames negres seguits
                arduino.write(b'1')#Enviem un "1" a l'arduino per parar la captura
                captura=1#Indiquem que s'ha acabat la captura
                comptador=0#Reiniciem comptador
            else:#Si és un fotograma normal reinicia el comptador
                comptador=0

for img in glob.glob(r'+Carpeta_origen+'*.jpg'):#Llegim fotos de carpeta d'origen
    fts=os.path.getmtime(img) #Llegim la data de creació de la foto
    if fts > ts: #Si la data de creació de la foto actual és superior a l'anterior.
        ts=fts #Guardem la data actual amb la variable antiga
        imatge=cv2.imread(img)#Llegim la imatge
        num_rows, num_cols = imatge.shape[:2]#Agafem el nombre de files i columnes
        #Calculem la matriu de rotació considerant el punt mig i per girar 180°
        rotation_matrix = cv2.getRotationMatrix2D((num_cols/2, num_rows/2), 180, 1)
        #Girem la imatge
        imatge_girada =cv2.warpAffine(imatge, rotation_matrix, (num_cols,num_rows))
        fotograma=imatge_girada.copy()#Fem una còpia de la imatge girada
        #La convertim de RGB a escala de grisos per detectar el fons negre
        imgray = cv2.cvtColor(imatge_girada,cv2.COLOR_BGR2GRAY)
        #Definim el llindar del color
        ret,thresh = cv2.threshold(imgray, 10, 255,cv2.THRESH_BINARY)
        #Trobem els contorns
```

```

imggray, contours, hierarchy = cv2.findContours(thresh, cv2.RETR_LIST,
cv2.CHAIN_APPROX_SIMPLE)
#Trobem el contorn amb l'àrea més gran
contours = sorted(contours, key = cv2.contourArea, reverse = True)[:1]
cnt = contours[0] #Guardem els punts del contorn en una matriu
#Trobem les coordenades del rectangle delimitador
x,y,w,h = cv2.boundingRect(cnt)
if foto>1: #Excepte el fotograma
    #Si el retall difereix molt de la mitjana
    if (x>(x_mitjana+80) or x<(x_mitjana-80)):
        x=x_mitjana #Agafem les coodenades del retall anterior
    if (y>(y_mitjana+80) or y<(y_mitjana-80)):
        y=y_mitjana
    if (w>(w_mitjana+80) or w<(w_mitjana-80)):
        w=w_mitjana
    if (h>(h_mitjana+80) or h<(h_mitjana-80)):
        h=h_mitjana
x_tot=x_tot + x #Valor acumulat de x
x_mitjana=int(x_tot/foto) #Mitjana de x
y_tot=y_tot + y #Valor acumulat de y
y_mitjana=int(y_tot/foto) #Mitjana de y
w_tot=int(w_tot + w) #Valor acumulat de w
w_mitjana=int(w_tot/foto) #Mitjana de w
h_tot=h_tot + h #Valor acumulat de h
h_mitjana=int(h_tot/foto) #Mitjana de h
#Reatallem pel rectangle delimitador
crop = fotograma[y_mitjana:y_mitjana+h_mitjana,x_mitjana:x_mitjana
+w_mitjana]
IMG=str(foto) #Convertim la variable foto a string
#Guardem cada imatge a la carpeta triada
cv2.imwrite(r''+Carpeta_nova+'\IMG'+IMG+'.jpg',crop)
foto=foto+1 #Incrementem el comptador de fotogrames

```

```

time.sleep(5) #Delay de 5 segons
sys.exit #Tanquem el programa

```