**CODING:**

**MULTITASK:**

import threading

import numpy as np

import cv2

from yolo\_opencv import\*

def task1():

i = 0

cap = cv2.VideoCapture(0)

while(True):

# Capture frame-by-frame

ret, frame = cap.read()

# Our operations on the frame come here

gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)

# Display the resulting frame

cv2.imshow('frame',frame)

i=i+1

# print(i)

if(i == 5):

cv2.imwrite("input.jpg", gray)

i=0

if cv2.waitKey(1) & 0xFF == ord('q'):

break

# When everything done, release the capture

cap.release()

cv2.destroyAllWindows()

#############################################################################################

def task2():

while(True):

try:

image\_read\_recoginition()

except:

print("camera opening")

###############################################################################

if \_\_name\_\_ == "\_\_main\_\_":

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*#

try:

t1 = threading.Thread(target=task1)

except:

t1 = threading.Thread(target=task1)

time.sleep(2)

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*#

try:

t2 = threading.Thread(target=task2)

except:

t2 = threading.Thread(target=task2)

time.sleep(2)

t1.start()

t2.start()

t1.join()

t2.join()

#both threads completely executed

print("Done!")

**YOLO OPENCV:**

import cv2

import numpy as np

from pydub import AudioSegment

from pydub.playback import play

from playsound import playsound

import serial

import time

global classes

global configure

global weights

global images

global label

label = " "

configure = "yolov3.cfg"

weights = "yolov3.weights"

images = "input.jpg"

def get\_output\_layers(net):

layer\_names = net.getLayerNames()

output\_layers = [layer\_names[i[0] - 1] for i in net.getUnconnectedOutLayers()]

return output\_layers

def draw\_prediction(img, class\_id, confidence, x, y, x\_plus\_w, y\_plus\_h):

global COLORS

global label

label = str(classes[class\_id])

color = COLORS[class\_id]

cv2.rectangle(img, (x,y), (x\_plus\_w,y\_plus\_h), color, 2)

cv2.putText(img, label, (x-10,y-10), cv2.FONT\_HERSHEY\_SIMPLEX, 0.5, color, 2)

#print(label)

def image\_read\_recoginition():

global classes

global COLORS

global label

print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

image = cv2.imread(images )

Width = image.shape[1]

Height = image.shape[0]

scale = 0.00392

classes = None

with open("yolov3.txt", 'r') as f:

classes = [line.strip() for line in f.readlines()]

COLORS = np.random.uniform(0, 255, size=(len(classes), 3))

net = cv2.dnn.readNet(weights, configure)

blob = cv2.dnn.blobFromImage(image, scale, (416,416), (0,0,0), True, crop=False)

net.setInput(blob)

outs = net.forward(get\_output\_layers(net))

class\_ids = []

confidences = []

boxes = []

conf\_threshold = 0.5

nms\_threshold = 0.4

for out in outs:

for detection in out:

scores = detection[5:]

class\_id = np.argmax(scores)

confidence = scores[class\_id]

if confidence > 0.5:

center\_x = int(detection[0] \* Width)

center\_y = int(detection[1] \* Height)

w = int(detection[2] \* Width)

h = int(detection[3] \* Height)

x = center\_x - w / 2

y = center\_y - h / 2

class\_ids.append(class\_id)

confidences.append(float(confidence))

boxes.append([x, y, w, h])

indices = cv2.dnn.NMSBoxes(boxes, confidences, conf\_threshold, nms\_threshold)

for i in indices:

i = i[0]

box = boxes[i]

x = box[0]

y = box[1]

w = box[2]

h = box[3]

draw\_prediction(image, class\_ids[i], confidences[i], round(x), round(y), round(x+w), round(y+h))

#cv2.imshow("object detection", image)

#cv2.waitKey()

cv2.imwrite("object-detection.jpg", image)

if(label == "dog"):

print(label)

print("dog sound on")

playsound('1.wav')

elif(label == "horse"):

print(label)

print("horse sound on")

playsound('2.wav')

elif(label == "sheep"):

print(label)

print("sheep sound on")

playsound('3.wav')

elif(label == "cow"):

print(label)

print("cow sound on")

playsound('4.wav')

elif(label == "elephant"):

print(label)

print("elephant sound on")

playsound('5.wav')

elif(label == "bear"):

print("bear sound on")

print(label)

playsound('6.wav')

elif(label == "zebra"):

print("zebra sound on")

print(label)

playsound('5.wav')

elif(label == "giraffe"):

print("giraffe sound on")

print(label)

playsound('2.wav')

elif(label == "giraffe"):

print("giraffe sound on")

print(label)

playsound('6.wav')