**Keyboard initialize code:**

[**https://github.com/noddycheung/cv2/blob/03e5f72156c6ec88f823be5a69f80dacd20bc814/KeyPressModule**](https://github.com/noddycheung/cv2/blob/03e5f72156c6ec88f823be5a69f80dacd20bc814/KeyPressModule)

import pygame  
  
def init():  
 pygame.init()  
 win = pygame.display.set\_mode((400,400))  
  
def getKey(keyName):  
 ans = False  
 for eve in pygame.event.get():pass  
 keyInput = pygame.key.get\_pressed()  
 myKey = getattr(pygame,'K\_{}'.format(keyName))  
 if keyInput[myKey]:  
 ans = True  
 pygame.display.update()  
 return ans  
  
def main():  
 if getKey("LEFT"):  
 print("Left key pressed")  
 if getKey("RIGHT"):  
 print("Right key pressed")  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 init()  
 while True:  
 main()

**Core Code:**

[**https://github.com/noddycheung/cv2/blob/03e5f72156c6ec88f823be5a69f80dacd20bc814/dronemap%2Bcam%2Bnoise**](https://github.com/noddycheung/cv2/blob/03e5f72156c6ec88f823be5a69f80dacd20bc814/dronemap%2Bcam%2Bnoise)

import cv2  
from djitellopy import tello  
import cvzone  
import time  
import KeyPressModule as kp  
import winsound  
import numpy as np  
import math  
  
freq=1000  
dur=50  
  
  
##PARAMETERS###  
fspeed= 117/10 #forward speed in cm/s (15cm/s)  
aspeed= 360/10 #angular speed in degree/sec  
interval = 0.25  
  
dinterval = fspeed\*interval  
ainterval = aspeed\*interval  
#####  
kp.init()  
global img  
me = tello.Tello()  
me.connect()  
print(me.get\_battery())  
me.streamoff()  
me.streamon()  
  
  
thres = 0.55  
nmsThres = 0.2  
# cap = cv2.VideoCapture(0)  
# cap.set(3, 640)  
# cap.set(4, 480)  
  
classFile = 'For YOLO/coco.names'  
classNames = []  
with open(classFile, 'rt') as f:  
 classNames = f.read().rstrip('\n').split('\n')  
  
class\_of\_interest = classNames.index('person') + 1  
  
configPath = 'For YOLO/ssd\_mobilenet\_v3\_large\_coco\_2020\_01\_14.pbtxt'  
weightsPath = "For YOLO/frozen\_inference\_graph.pb"  
  
net = cv2.dnn\_DetectionModel(weightsPath, configPath)  
net.setInputSize(320, 320)  
net.setInputScale(1.0 / 127.5)  
net.setInputMean((127.5, 127.5, 127.5))  
net.setInputSwapRB(True)  
  
points = [(0,0),(0,0)]  
  
x,y = 250,250  
a = 0  
yaw = 0  
  
  
def getKeyboardInput():  
 lr,fb,ud,yv = 0,0,0,0  
  
 speed = 15  
 aspeed = 50  
 d=0  
 global x, y, yaw, a  
  
 if kp.getKey("LEFT"):  
 lr=-speed  
 d= dinterval  
 a=-180  
 elif kp.getKey("RIGHT"):  
 lr = speed  
 d = -dinterval  
 a = 180  
  
 if kp.getKey("UP"):  
 fb = speed  
 d = dinterval  
 a = 270  
  
 elif kp.getKey("DOWN"):  
 fb = -speed  
 d = -dinterval  
 a = -90  
  
 if kp.getKey("w"):  
 ud = speed  
 elif kp.getKey("s"):  
 ud = -speed  
  
 if kp.getKey("a"):  
 yv = -aspeed  
 yaw -= ainterval  
 elif kp.getKey("d"):  
 yv = aspeed  
 yaw += ainterval  
  
 if kp.getKey("q"): me.land()  
 if kp.getKey("e"): me.takeoff()  
  
 if kp.getKey('z'):  
 cv2.imwrite(f'Resources/Images/{time.time()}.jpg',img)  
 time.sleep(0.3)  
  
 time.sleep(interval)  
 a += yaw  
 x += int(d\*math.cos(math.radians(a)))  
 y += int(d \* math.sin(math.radians(a)))  
  
 return [lr,fb,ud,yv,x,y]  
  
  
def drawPoints(mapimg, points):  
 for point in points:  
 cv2.circle(mapimg,(point[0],point[1]), 5 ,(0,0,225),cv2.FILLED)  
  
 cv2.circle(mapimg, points[-1],8,(0,255,0),cv2.FILLED)  
 cv2.putText(mapimg, f'({(points[-1][0] - 250) / 100},{(points[-1][1] - 250) / 100})m',  
 (points[-1][0] + 10, points[-1][1] + 30),cv2.FONT\_HERSHEY\_PLAIN,1,  
 (255,0,255),1)  
  
while True:  
 # success, img = cap.read()  
 img = me.get\_frame\_read().frame  
 classIds, confs, bbox = net.detect(img, confThreshold=thres, nmsThreshold=nmsThres)  
 try:  
 for classId, conf, box in zip(classIds.flatten(), confs.flatten(), bbox):  
 if classId == class\_of\_interest:  
 cvzone.cornerRect(img, box)  
 cv2.putText(img, f'{classNames[classId - 1].upper()} {round(conf \* 100, 2)}',  
 (box[0] + 10, box[1] + 30), cv2.FONT\_HERSHEY\_COMPLEX\_SMALL,  
 1, (0, 255, 0), 2)  
 winsound.Beep(freq, dur)  
 except:  
 pass  
  
 vals = getKeyboardInput()  
 me.send\_rc\_control(vals[0], vals[1], vals[2], vals[3])  
  
 cv2.imshow("Image", img)  
 mapimg = np.zeros((500,500,3),np.uint8)  
 if (points[-1][0] != vals[4] or points [-1][1] != vals[5]):  
 points.append((vals[4],vals[5]))  
 drawPoints(mapimg,points)  
 cv2.imshow("Mapping",mapimg)  
 cv2.waitKey(1)