```python
import threading
import socket
import time
import cv2
import numpy as np



host = '0.0.0.0'
port = 8889
locaddr = (host, port)


# Create a UDmport tello

sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

tello_address = ('192.168.10.1', 8889)

sock.bind(locaddr)

def recv():
    while True:
        try:
            data, server = sock.recvfrom(1518)
            print(data.decode(encoding="utf-8"))
        except Exception:
            print('\nExit . . . RECV\n')
            break
#recvThread create
recvThread = threading.Thread(target=recv)
recvThread.start()



sock.sendto(b'command', tello_address)
print('command ok')
time.sleep(0.5)
sock.sendto(b'streamon', tello_address)
print('stream on')
time.sleep(1)
#sock.sendto(b'takeoff', tello_address)
time.sleep(2)

# #cmd = 'rc {} {} {} {}' rc{roll}{pitch}{throttle}{yaw}
#rc_command = 'rc 0, 0, 20, 0'
#sock.sendto(rc_command.encode('utf-8'), tello_address)
time.sleep(2)


LOCAL_IP = '192.168.10.1'
LOCAL_PORT ='11111'
addr = ("udp://%s:%s?overrun_nonfatal=1&fifo_size=50000000" % ('192.168.10.1',
'11111'))

cap = cv2.VideoCapture(addr)
#cap.set(cv2.CAP_PROP_BUFFERSIZE, 2)

print('start cap')
```

```
59
60
61
62
63 pid = [0.4, 0.4, 0]
64 pError = 0
65 w,h = 480, 360
66
67
68 def findFace(img):
69             global face_center_y
70
71             face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades +
   'haarcascade_frontalface_default.xml')
72
73
74             gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
75             faces = face_cascade.detectMultiScale(gray, 1.3, 5)
76
77
78             FaceList_center = []
79             FaceList_area = []
80             face_center_y = 0
81
82
83             for (x, y, w, h) in faces:
84
85                cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2)
86                face_center_x = int(x + (w / 2))          #顔の中心x軸
87                face_center_y = int(y + (h) / 2)
88
89                face_area =int(w * h)       #面積
90                cv2.circle(img,(face_center_x, face_center_y), 10, (0,0,225),
   cv2.FILLED)  #中心point
91                cv2.putText(img, f'facecenter:({face_center_x}, {face_center_y})',
   org=(50, 50), fontFace=cv2.FONT_HERSHEY_SIMPLEX, fontScale=0.5, color=(204, 0, 255),
   thickness=1, lineType=cv2.LINE_4)
92                cv2.putText(img, f'facearea: '+ str(face_area), org=(30, 30),
   fontFace=cv2.FONT_HERSHEY_SIMPLEX, fontScale=0.5, color=(204, 0, 255), thickness=1,
   lineType=cv2.LINE_4)
93
94                FaceList_center.append([face_center_x, face_center_y])
95                FaceList_area.append(face_area)
96
97                print(FaceList_center, FaceList_area)
98
99
100
101            if len(FaceList_area) != 0:
102                  i = FaceList_area.index(max(FaceList_area))
103                  return img, [FaceList_center[i], FaceList_area[i]]
104            else:
105                  return img, [[0, 0], 0]
106
107
108 a = b = c = d =0         # #cmd = 'rc {a} {b} {c} {d}' rc{roll}{pitch}{throttle}
   {yaw}
109 def TrackFace( info, w, pid, pError):   # x軸に対するPI制御
110           global b, d
111 #          pError = error
```

```
#          pid_output = d
          area = info[1]
          x, y = info[0]
          b = 0   #bを初期化
          #yaw(左右運動)算出用
          error = x - w/2
          d = pid[0] * error + pid[1] * (error - pError)    #PID制御のうちPI制御
          d = int(np.clip(d, -100, 100))    #RCコマンドのリミッター
          print('pErroe: ' + str(pError))
          print('error' + str(error))
          #pitch(前後運動)算出用

          if area > 6200 and area < 6800:
             b = 0
          elif area > 6800:
             b = -20
          elif area < 6200 and area != 0:
             b = 20
          if x == 0:
             d = 0
             error = 0

          print(b, d)
          return error


def trottle(face_center_y):
          global c

          c = 0
          print(face_center_y)
          if face_center_y > 180 and face_center_y < 40:
                 c = 0
          elif 280 > face_center_y > 180 + 10 :   #frame h:360の1/2=180 顔の下降（座標が大きくなる）
                 c = -20
          elif 50 < face_center_y < 110  :
                 c = 20
#          c = int(np.clip(c, -100, 100))

          print('throttle: ' + str(c))
          return



while True:
       for i in range(5):
          ret, frame = cap.read()
       if frame is None or frame.size == 0:
          continue
       cv2.circle(frame,(int(480), int(360)), 5, (0,225,0), cv2.FILLED) #中心point

       img = cv2.resize(frame, (w, h))
       img, info = findFace(img)
       pError = TrackFace(info, w, pid, pError)
       face_center_y = trottle(face_center_y)
       rc_command = "rc {0} {1} {2} {3}".format(a, b, c, d)     # #cmd = 'rc {a} {b} {c} {d}' rc{roll}{pitch}{throttle}{yaw}
    #   sock.sendto(rc_command.encode('utf-8'),tello_address)
       print("Sending RC command with values:", a, b, c, d)
```

```
169       cv2.putText(img, f' roll: '+ str(a) + f'   pitch:' + str(b) + f' throttle: ' + str(c) + f'   yaw: ' + str(d), org=(100, 100), fontFace=cv2.FONT_HERSHEY_SIMPLEX, fontScale=0.5, color=(204, 0, 255), thickness=1, lineType=cv2.LINE_4)
170       cv2.imshow('frame', img)
171
172       if cv2.waitKey(1) & 0xFF == 27  :
173 #
174    #      sock.sendto(b'land', tello_address)
175          cap.release()
176          cv2.destroyAllWindows()
177          sock.close()
178          break
```