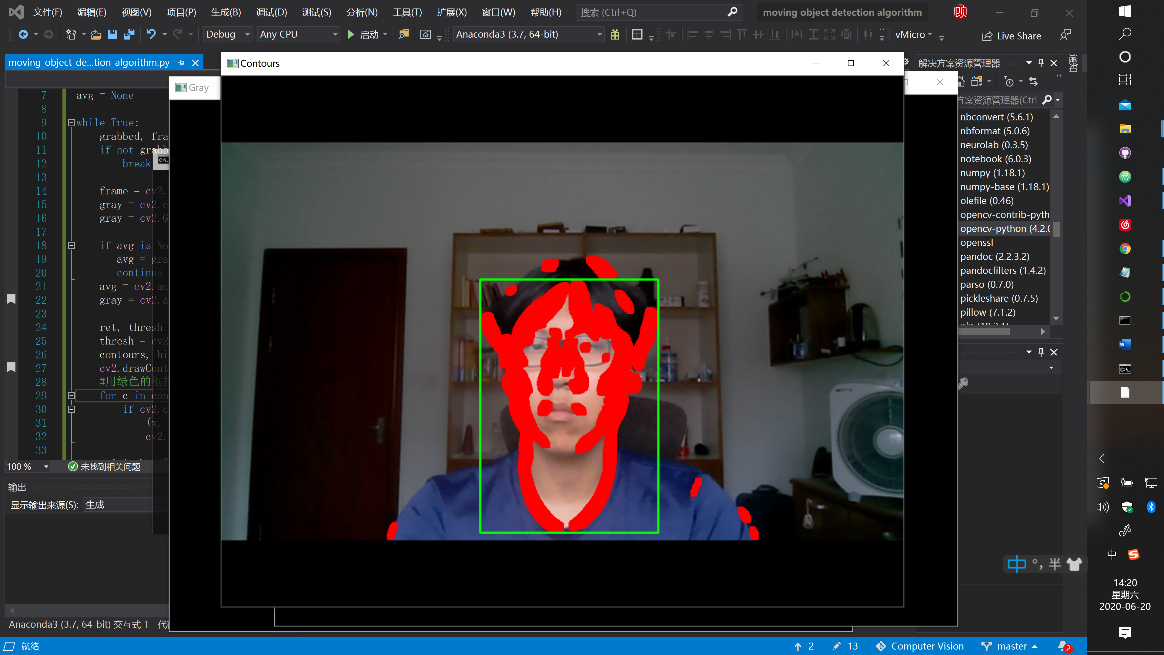
Instructions for Moving Object Detection Algorithm

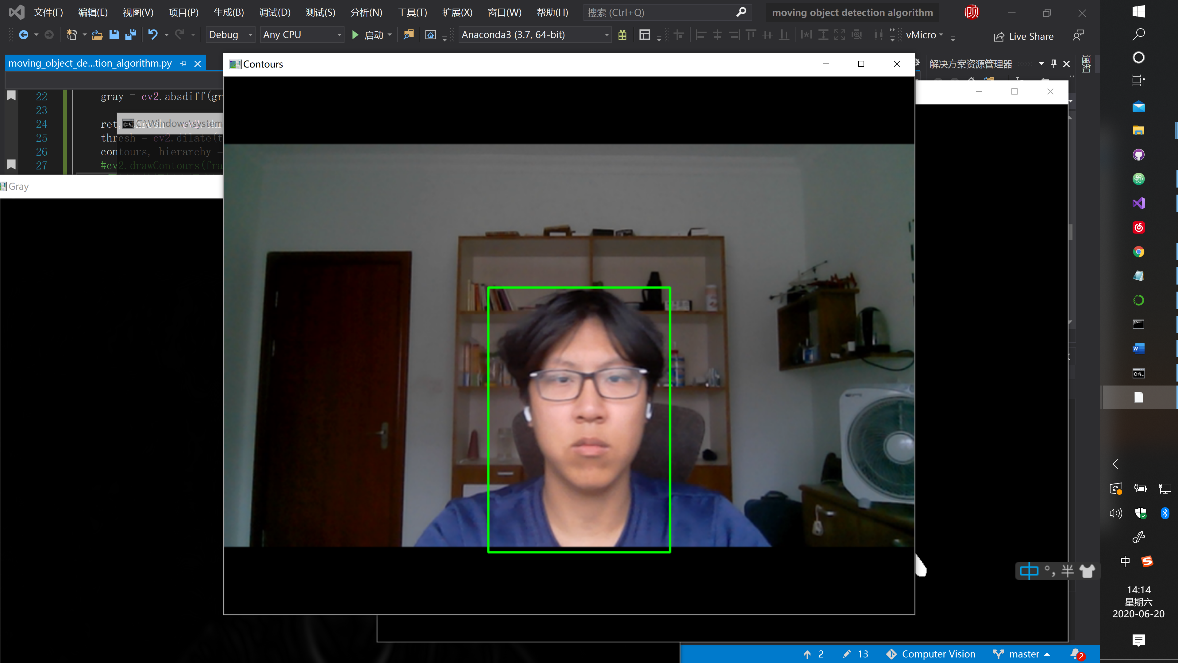
**Zian Gu**

**06/19/2020**

In project 3, I used the OpenCV based on python to realize moving object detection algorithm. The algorithm I chose to realize it is called “frame difference”. It’s one of the most commonly used algorithm to detect moving object. The procedures and instructions are as follows:

1. Define object camera, and use cv2.VideoCapure(0) to open the internal camera of computer.
2. In the while loop, use class method read() to get a frame of the video.
3. Resize the frame and convert it to gray scale. Add Gaussian blur to decrease white noise of the frame.
4. Use accumulateWeighted() function to update the background and make it becomes a running average of a frame sequence. Then use absdiff() function to calculate the absolute difference of two input images. The output is in a gray scale. Here is the core of “frame difference”.
5. Use threshold() function to convert the gray scale to binary scale. The threshold I set here is 5. If it’s too small, it will produce lots of noises. If it’s too large, it will be blind to any difference.
6. After we get the difference in binary scale, we use morphological dilation to make the difference more explicit. Then we find all contours of the frame and calculate the area of them. If the area of the contours is larger than 2000, I use a green rectangle to highlight the moving part of the video.

**Result：**

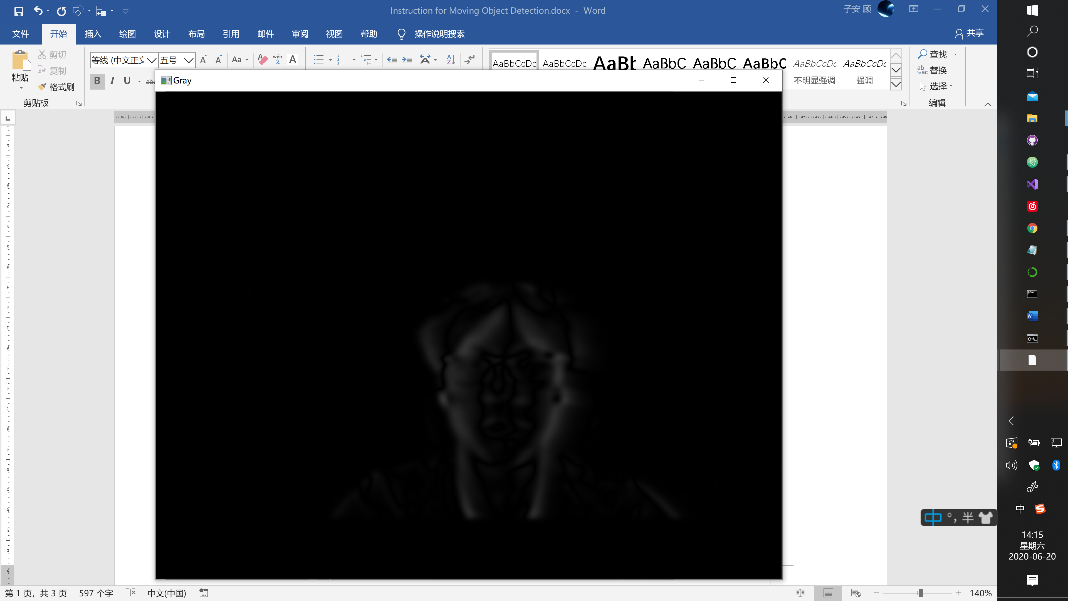
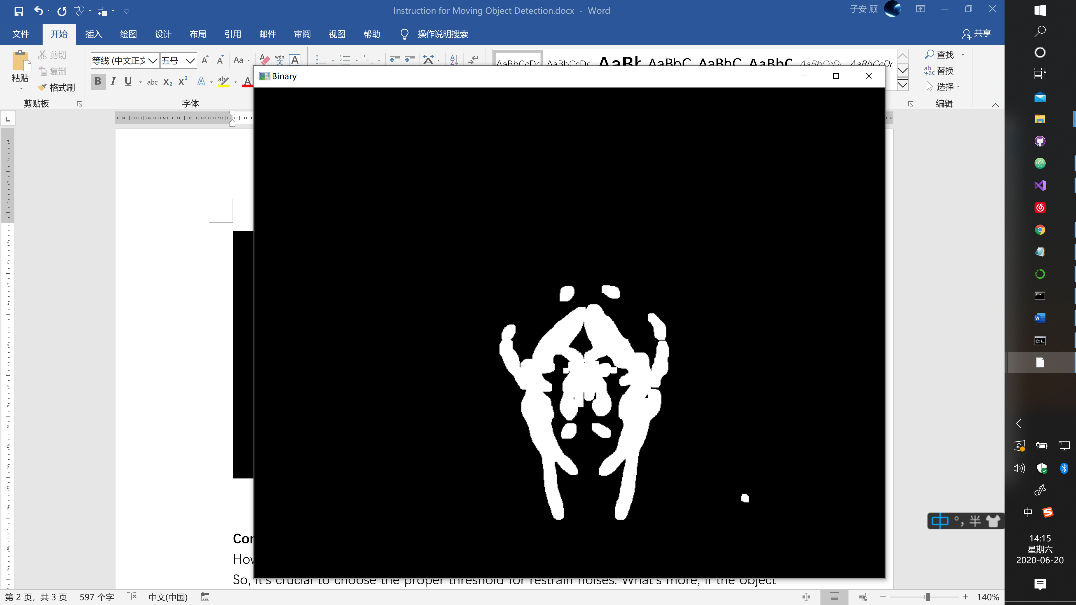


**Figure 2: the detection of moving object (the difference of frames is highlighted by red shadow)**

**Figure 1:the detection of moving object (highlighted by green rectangles)**

**Figure 4: the difference of frames in binary scale which has been dilated**

**Figure 3: the difference of frames in gray scale**



**Conclusion:**

However, the disadvantages of “frame difference” is that it’s sensitive to environment noises. So, it’s crucial to choose the proper threshold for restrain noises. What’s more, if the object moving too fast, the algorithm can’t get the complete moving object. Finally, if the color of the moving object is close to the background, the effect may not be so explicit. To solve these, Gaussian mixture model (GMM) may be a good choice.

**Source code:**

#Moving Object Detection

#by Zian Gu

#06/18/2020

import cv2

#初始化

camera = cv2.VideoCapture(0)#0表示打开笔记本内置摄像头

avg = None

while True:

grabbed, frame = camera.read()

if not grabbed:

break

frame = cv2.resize(frame,(900,700))

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

gray = cv2.GaussianBlur(gray, (23, 23), 0)#0是标准差，若该参数是0则根据高斯矩阵的尺寸自己计算

if avg is None:

avg = gray.copy().astype("float")#初始化平均帧

continue

avg = cv2.accumulateWeighted(gray, avg, 0.5)#更新背景，叠加于avg

gray = cv2.absdiff(gray, cv2.convertScaleAbs(avg))#帧差法计算两帧的差的绝对值

ret, thresh = cv2.threshold(gray,5,255,cv2.THRESH\_BINARY) #对叠加后的灰度图进行二值化处理

thresh = cv2.dilate(thresh, None, iterations=3)#形态学膨胀处理

contours, hierarchy = cv2.findContours(thresh, cv2.RETR\_EXTERNAL,cv2.CHAIN\_APPROX\_SIMPLE)#寻找轮廓（第二个返回参数是层级，实际无用处）

#cv2.drawContours(frame,contours,-1,(0,0,255),-1)#用红色把变化区域轮廓标记出来（填充）

#用绿色的框把大幅度运动的区域框出来

for c in contours:

if cv2.contourArea(c) < 2000:#降低干扰

continue

(x, y, w, h) = cv2.boundingRect(c)#外接最小矩形

cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)

cv2.imshow("Contours", frame)

cv2.imshow("Gray", gray)

cv2.imshow("Binary", thresh)

if cv2.waitKey(20) & 0xFF == 27:

break

camera.release()

cv2.destroyAllWindows()