Extended tutorial : KNN background segmenter for moving target recognition

！ Note: When running the program of this course, there must be a desktop for displaying pictures. It is recommended that you use VNC to log in to the system so that the pictures can be displayed.

We choose KNN as the background selector for this use.

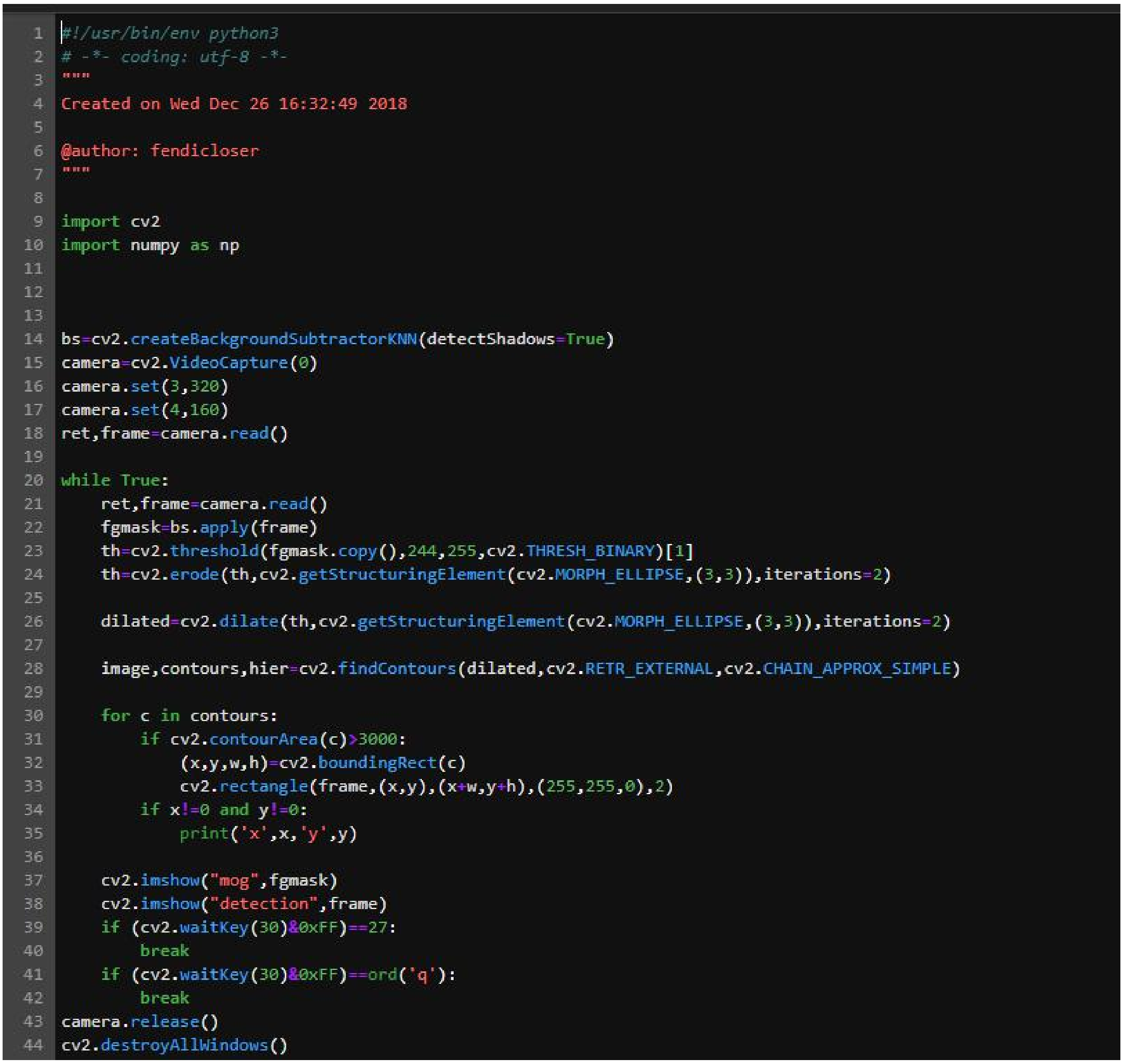
```
bs = cv2.createBackgroundSubtractorKNN(detectShadows=True)
```

In the lighting environment, the object will produce a shadow, but in fact the shadow is not part of the moving target. detectShadows=True means to calculate the shadow. By calculating the shadow, the shadow area in the image can be excluded to improve the accuracy.
At the core of this method is the .apply() , which returns a foreground mask to distinguish which objects are the background and which objects are the targets. Once the target is obtained, the outline of the target is detected.

The source code of the program is located at: **/home/pi/yahboom/motion_detect/KNN.py**

!!!Note: Please try the actual code below on the PC.

```
  fgmask = bs.apply(frame)
  th = cv2.threshold(fgmask.copy(), 244, 255, cv2.THRESH_BINARY)[1]
  th = cv2.erode(th, cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (3, 3)), iterations=2)
  dilated = cv2.dilate(th, cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (8, 3)), iterations=2)
image, contours, hier = cv2.findContours(dilated, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
```



```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Wed Dec 26 16:32:49 2018

@author: fendicloser
"""

import cv2
import numpy as np



bs=cv2.createBackgroundSubtractorKNN(detectShadows=True)
camera=cv2.VideoCapture(0)
camera.set(3,320)
camera.set(4,160)
ret,frame=camera.read()

while True:
    ret,frame=camera.read()
    fgmask=bs.apply(frame)
    th=cv2.threshold(fgmask.copy(),244,255,cv2.THRESH_BINARY)[1]
    th=cv2.erode(th,cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(3,3)),iterations=2)

    dilated=cv2.dilate(th,cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(3,3)),iterations=2)

    image,contours,hier=cv2.findContours(dilated,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)

    for c in contours:
        if cv2.contourArea(c)>3000:
            (x,y,w,h)=cv2.boundingRect(c)
            cv2.rectangle(frame,(x,y),(x+w,y+h),(255,255,0),2)
        if x!=0 and y!=0:
            print('x',x,'y',y)

    cv2.imshow("mog",fgmask)
    cv2.imshow("detection",frame)
    if (cv2.waitKey(30)&0xFF)==27:
        break
    if (cv2.waitKey(30)&0xFF)==ord('q'):
        break
camera.release()
cv2.destroyAllWindows()
```

Please enter the following command to run the program:

python3 KNN.py

The running effect is as follows, the figure1-1 is the real scene, and the figure1-2 is the analysis of the background.

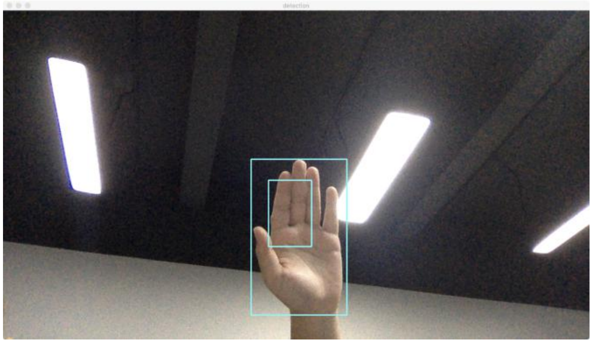


figure1-1

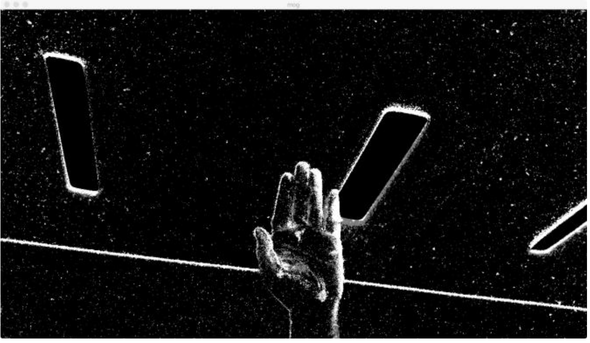


figure1-2