# 1.Getting started with Open Source CV

## 1.1, Introduction to OpenCV

What is OpenCV? Its full name is Open source Computer Vision Library, an open source computer vision library. As shown in the picture above, what we see is the OpenCV logo. We can see three small circles made of distinct R, G, and B colors. Composition, that is to say, it is a set of open source API function libraries about computer vision. This also means,

(1) Whether it is scientific research or commercial application, it can be used for development;

(2) The source code of all API functions is public, and you can see the program steps of its internal implementation;

(3) You can modify the source code of OpenCV and compile and generate the specific API functions you need.

Image processing on ROSMASTER uses certain functions of the OpenCV function library, or it can be said that it is inseparable from most image processing design fields. It has been used in intrusion detection and specific target tracking many years ago. , target detection, face detection, face recognition, face tracking and other fields, OpenCV can be said to show its talents, and these are just the tip of the iceberg of its applications. Now that we realize that OpenCV is so universal, in this chapter we will introduce you to some very basic image processing functions that we use in our courses, as well as some universal functions. Here we first have a general understanding of this knowledge. After a while, there are two practical projects in the back to teach you how to get started, including color recognition and tracking, and face recognition and tracking. However, the powerful application functions provided by OpenCV are far more than this. If you are interested in the development of Opencv computer vision library and want to go deeper. If you know more about it, here are several websites for your reference and study:

OpenCV official homepage: https://www.opencv.org

OpenCV Chinese Forum: http://www.opencv.org.cn

OpenCV CSDN Forum: https://bbs.csdn.net/forums/OpenCV

## 1.2. OpenCV image reading and display

### 1.2.1. Reading of images:

img = cv2.imread('yahboom.jpg', 0) The first parameter is the path of the image, and the second parameter is how to read the image.

cv2.IMREAD_UNCHANGED: Keep the original format unchanged, -1;

cv2.IMREAD_GRAYSCALE: Read the image in grayscale mode, which can be represented by 0;

cv2.IMREAD_COLOR:, read a color picture, which can be represented by 1; default value

cv2.IMREAD_UNCHANGED: Read in an image and include its alpha channel, which can be represented by 2.

### 1.2.2. Image display

cv.imshow('frame', frame): Open a window named frame and display frame data (image/video data)

Parameter meaning:

The first parameter represents the name of the window that is created and opened.

The second parameter represents the image to be displayed

### 1.2.3. Code and actual effect display

Run program

**jetson motherboard/Raspberry Pi 4B**

```
cd ~/transbot_ws/src/transbot_visual/opencv
python 1_1.py
```

**Raspberry Pi 5**

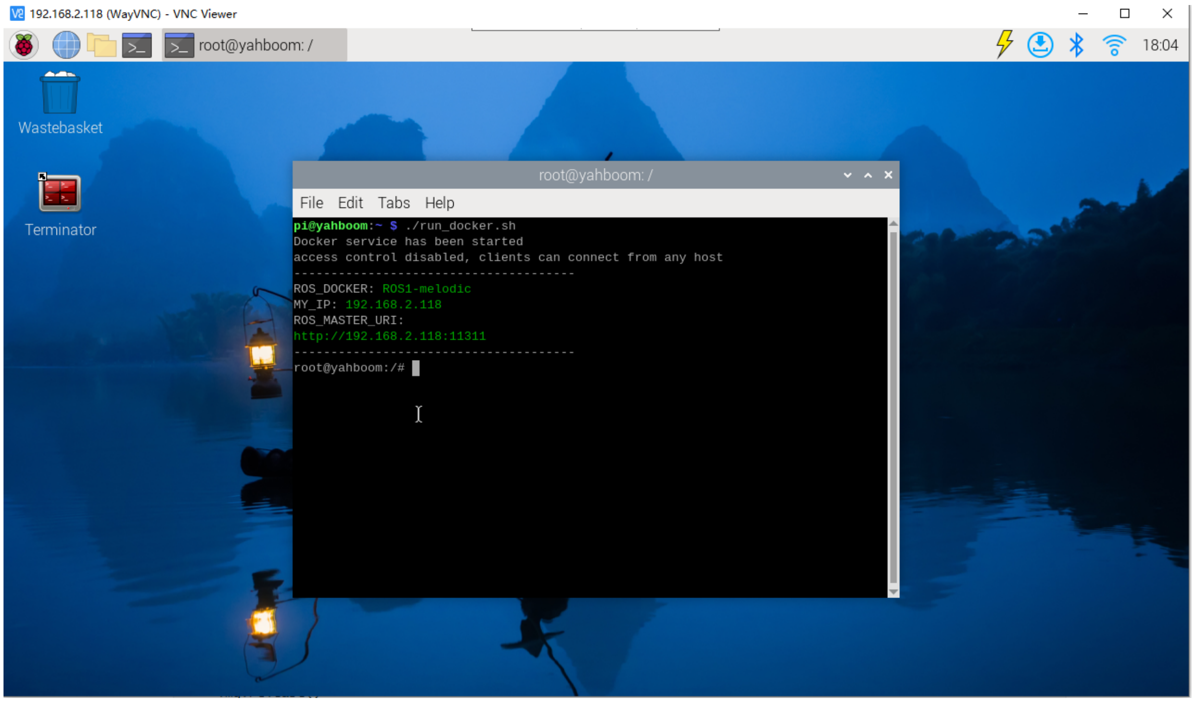
**Before running, please confirm that the large program has been permanently closed**

Enter docker

**Note: If there is a terminal that automatically starts docker, or there is a docker terminal that has been opened, you can directly enter the docker terminal to run the command, and there is no need to manually start docker**

Start docker manually

```
./run_docker.sh
```



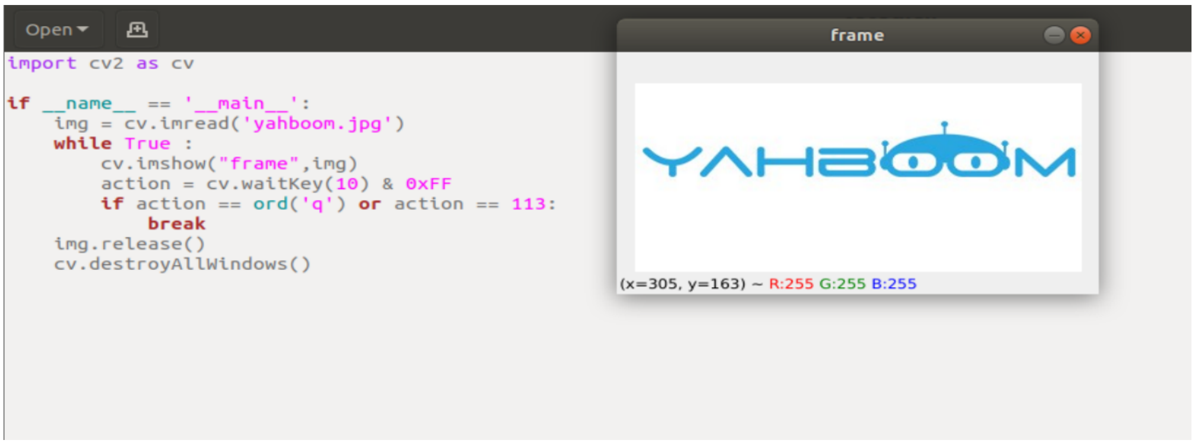
```
pi@yahboom:~ $ ./run_docker.sh
Docker service has been started
access control disabled, clients can connect from any host
--------------------------------------
ROS_DOCKER: ROS1-melodic
MY_IP: 192.168.2.118
ROS_MASTER_URI:
http://192.168.2.118:11311
--------------------------------------
root@yahboom:/#
```

Run program

```
cd ~/transbot_ws/src/transbot_visual/opencv
python 1_1.py
```

```python
import cv2 as cv

if __name__ == '__main__':
img = cv.imread('yahboom.jpg')
while True :
cv.imshow("frame",img)
action = cv.waitKey(10) & 0xFF
if action == ord('q') or action == 113:
break
img.release()
cv.destroyAllWindows()
```



```python
import cv2 as cv

if __name__ == '__main__':
    img = cv.imread('yahboom.jpg')
    while True :
        cv.imshow("frame",img)
        action = cv.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv.destroyAllWindows()
```

```
(x=305, y=163) ~ R:255 G:255 B:255
```

## 1. 3. OpenCV image writing

### 1.3.1. Function method: cv2.imwrite('new_img_name', img)

Parameter meaning:

The first parameter is the saved file name

The second parameter is the saved image

### 1.3.2. Code and actual effect display

**jetson motherboard/Raspberry Pi 4B**

```
cd ~/transbot_ws/src/transbot_visual/opencv
python 1_2.py
```

**Raspberry Pi 5**

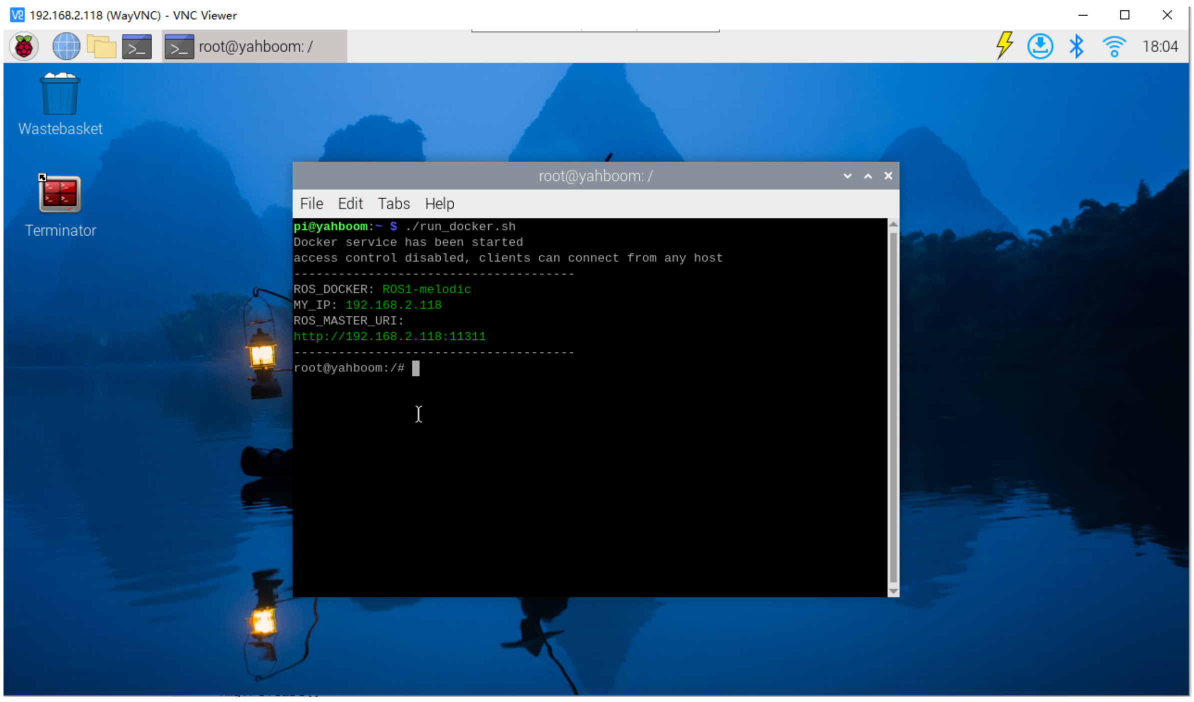
**Before running, please confirm that the large program has been permanently closed**

Enter docker

**Note: If you have a terminal that automatically starts docker, you can directly enter the docker terminal to run the command, and there is no need to manually start docker**

Start docker manually

```
./run_docker.sh
```



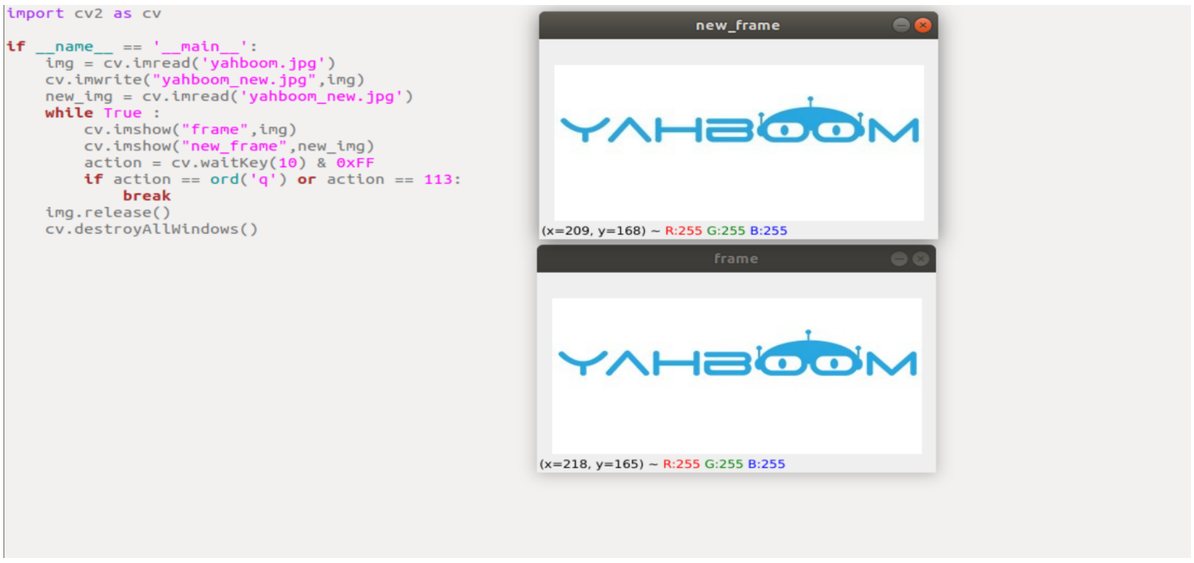
```
pi@yahboom:~ $ ./run_docker.sh
Docker service has been started
access control disabled, clients can connect from any host
-------------------------------------
ROS_DOCKER: ROS1-melodic
MY_IP: 192.168.2.118
ROS_MASTER_URI:
http://192.168.2.118:11311
-------------------------------------
root@yahboom:/#
```

Run program

```
cd ~/transbot_ws/src/transbot_visual/opencv
python 1_2.py
```

```
import cv2 as cv
```

```
if __name__ == '__main__':
img = cv.imread('yahboom.jpg')
cv.imwrite("yahboom_new.jpg",img) #Create a new file yahboom_new.jpg and write
yahboom.jpg into it
new_img = cv.imread('yahboom_new.jpg') #Read the newly written picture
while True :
cv.imshow("frame",img)
cv.imshow("new_frame",new_img)
action = cv.waitKey(10) & 0xFF
if action == ord('q') or action == 113:
break
img.release()
cv.destroyAllWindows()
```



```
import cv2 as cv

if __name__ == '__main__':
    img = cv.imread('yahboom.jpg')
    cv.imwrite("yahboom_new.jpg",img)
    new_img = cv.imread('yahboom_new.jpg')
    while True :
        cv.imshow("frame",img)
        cv.imshow("new_frame",new_img)
        action = cv.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv.destroyAllWindows()
```

```
(x=209, y=168) ~ R:255 G:255 B:255
```

```
(x=218, y=165) ~ R:255 G:255 B:255
```

## 1.4. OpenCV camera reading and displaying video

### 1.4.1. Camera reading

capture=cv.VideoCapture(0)

Parameter meaning:

The parameter in VideoCapture() is 0, which means opening the laptop's built-in camera. If the parameter is the video file path, the video will be opened, such as cap = cv2.VideoCapture("../test.avi")

### 1.4.2. Display camera video

ret,img = frame.read()

Return value meaning:

ret: ret is a bool value to determine whether the correct frame is read back

img: image data of each frame

### 1.4.3. Code and actual effect display

**jetson motherboard/Raspberry Pi 4B**

```
cd ~/transbot_ws/src/transbot_visual/opencv
python 1_3.py
```

**Raspberry Pi 5**

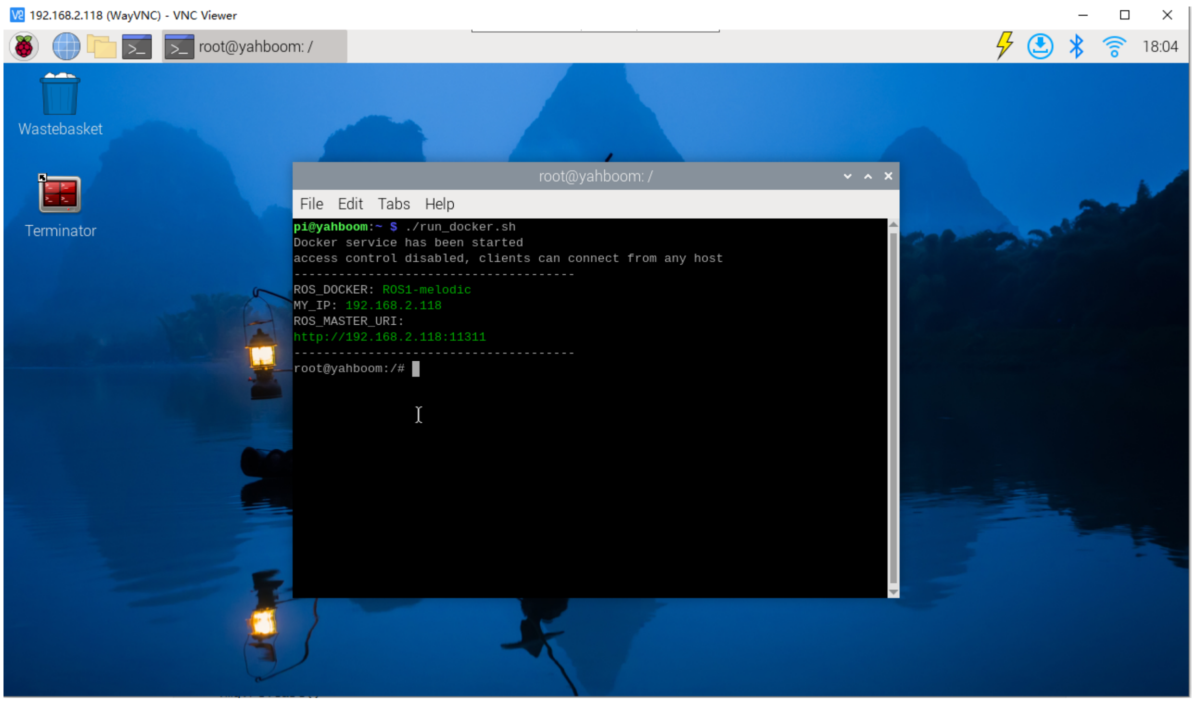
**Before running, please confirm that the large program has been permanently closed**

Enter docker

**Note: If you have a terminal that automatically starts docker, you can directly enter the docker terminal to run the command, and there is no need to manually start docker**

Start docker manually

```
./run_docker.sh
```

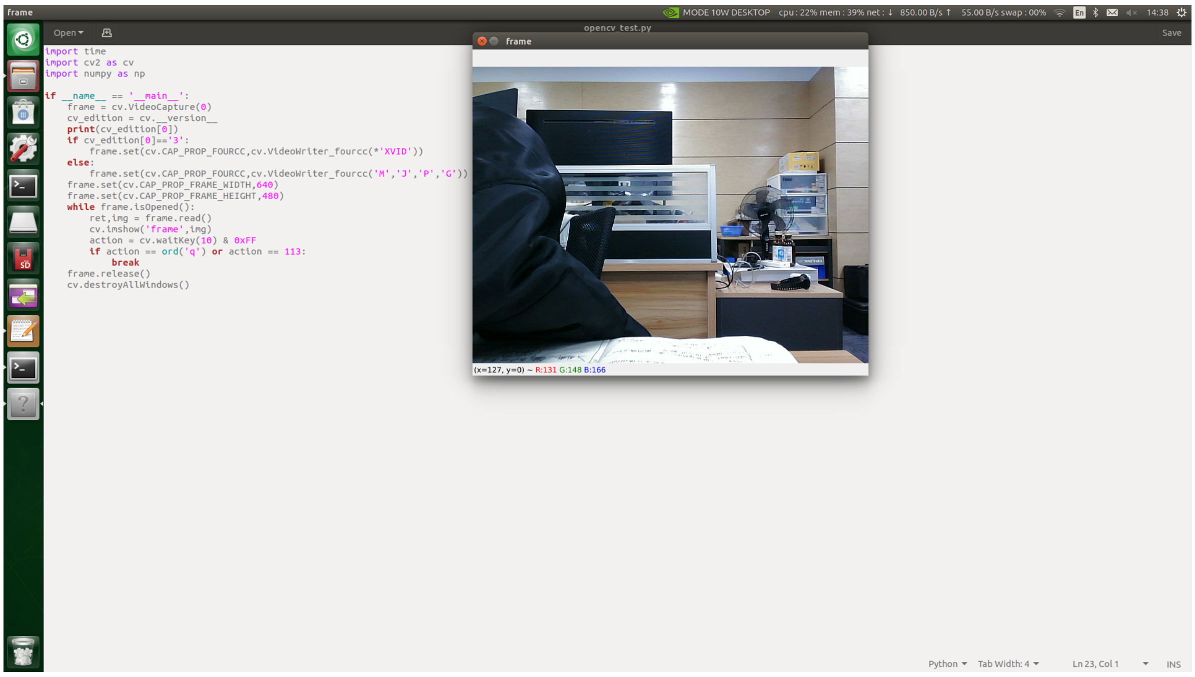


```
pi@yahboom:~ $ ./run_docker.sh
Docker service has been started
access control disabled, clients can connect from any host
--------------------------------------
ROS_DOCKER: ROS1-melodic
MY_IP: 192.168.2.118
ROS_MASTER_URI:
http://192.168.2.118:11311
--------------------------------------
root@yahboom:/#
```

Run program

```
cd ~/transbot_ws/src/transbot_visual/opencv
python 1_3.py
```

```
import cv2 as cv
if __name__ == '__main__':
frame = cv.VideoCapture(0)
     while frame.isOpened():
ret,img = frame.read()
         cv.imshow('frame',img)
         action = cv.waitKey(10) & 0xFF
if action == ord('q') or action == 113:
break
frame.release()
cv.destroyAllWindows()
```



```
import time
import cv2 as cv
import numpy as np

if __name__ == '__main__':
    frame = cv.VideoCapture(0)
    cv_edition = cv.__version__
    print(cv_edition[0])
    if cv_edition[0]=='3':
        frame.set(cv.CAP_PROP_FOURCC,cv.VideoWriter_fourcc(*'XVID'))
    else:
        frame.set(cv.CAP_PROP_FOURCC,cv.VideoWriter_fourcc('M','J','P','G'))
    frame.set(cv.CAP_PROP_FRAME_WIDTH,640)
    frame.set(cv.CAP_PROP_FRAME_HEIGHT,480)
    while frame.isOpened():
        ret,img = frame.read()
        cv.imshow('frame',img)
        action = cv.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    frame.release()
    cv.destroyAllWindows()
```

## 1.5. OpenCV pixel operation

### 1.5.1. Pixel operation, we can change any position to a new pixel color.

First, we need to read the image, then modify the value of bgr and assign an area to black.

### 1.5.2. Code and actual effect display

**jetson motherboard/Raspberry Pi 4B**

```
cd ~/transbot_ws/src/transbot_visual/opencv
python 1_4.py
```

**Raspberry Pi 5**

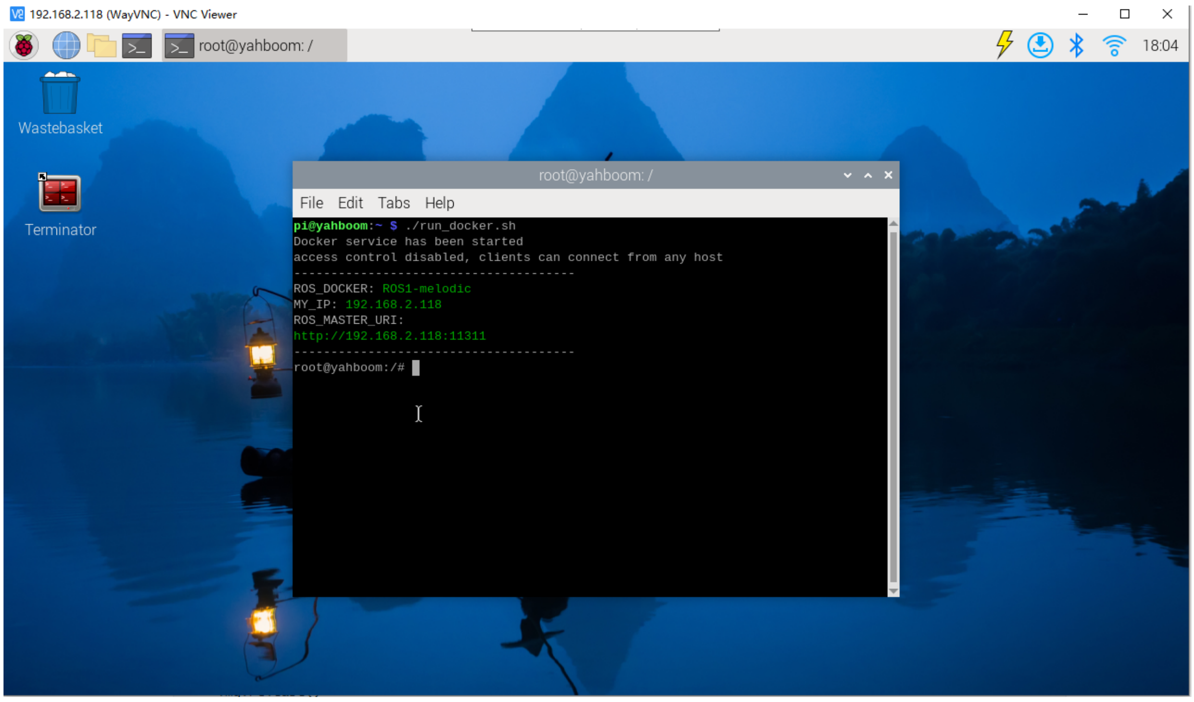
**Before running, please confirm that the large program has been permanently closed**

Enter docker

**Note: If you have a terminal that automatically starts docker, you can directly enter the docker terminal to run the command, and there is no need to manually start docker**

Start docker manually

```
./run_docker.sh
```

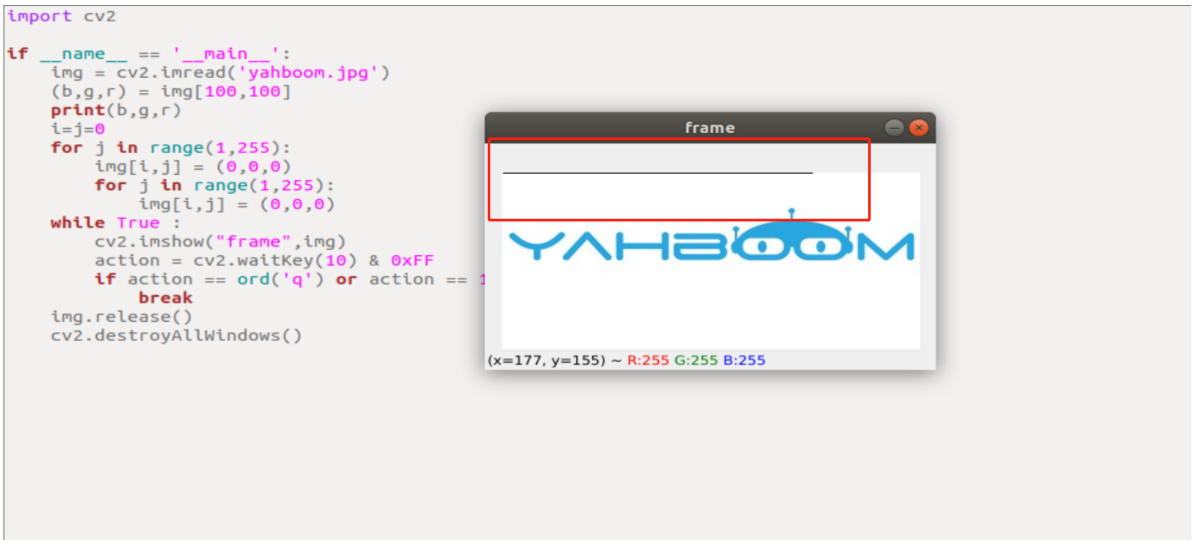


```
pi@yahboom:~ $ ./run_docker.sh
Docker service has been started
access control disabled, clients can connect from any host
--------------------------------------
ROS_DOCKER: ROS1-melodic
MY_IP: 192.168.2.118
ROS_MASTER_URI:
http://192.168.2.118:11311
--------------------------------------
root@yahboom:/#
```

Run program

```
cd ~/transbot_ws/src/transbot_visual/opencv
python 1_4.py
```

```
import cv2
if __name__ == '__main__':
img = cv2.imread('yahboom.jpg')
(b,g,r) = img[100,100]
print(b,g,r)
i=j=0
for j in range(1,255):
img[i,j] = (0,0,0)
for j in range(1,255):
img[i,j] = (0,0,0)
while True :
cv2.imshow("frame",img)
action = cv2.waitKey(10) & 0xFF
if action == ord('q') or action == 113:
break
img.release()
cv2.destroyAllWindows()
```



```
import cv2

if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    (b,g,r) = img[100,100]
    print(b,g,r)
    i=j=0
    for j in range(1,255):
        img[i,j] = (0,0,0)
        for j in range(1,255):
            img[i,j] = (0,0,0)
    while True :
        cv2.imshow("frame",img)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 1
            break
    img.release()
    cv2.destroyAllWindows()
```

The red box part is the modified pigment value.