4. Image beautification

4.1. OpenCV repair pictures

1. Image inpainting is a type of algorithm in computer vision whose goal is to fill in areas within an image or video. The area is identified using a binary mask, and filling is usually done based on the area boundary information that needs to be filled. The most common application of image restoration is to restore old scanned photos. It is also used to remove small unwanted objects from images.
2. In OpenCV, dst = cv2.inpaint(src, inpaintMask, inpaintRadius, flags) is provided to repair the image.

Parameter meaning:

src: source image, which is the image that needs to be repaired

inpaintMask: Binary mask indicating which pixels to inpaint.

dst: result image

inpaintRadius: represents the radius of repair

flags: Repair algorithm, mainly INPAINT_NS (Navier-Stokes based method) or INPAINT_TELEA (Fast marching based method)

Navier-Stokes based repairs should be slower and tend to produce blurrier results than fast marching methods. In practice, we did not find this to be the case. INPAINT_NS produced better results in our tests and was slightly faster than INPAINT_TELEA.

3. Code and actual effect display

(1) First, we first add damage to the intact picture, which can be understood as modifying the pixel value of a specific part of it.

Run program

jetson motherboard/Raspberry Pi 4B

```
cd ~/transbot_ws/src/transbot_visual/opencv
python4_1.py
```

Raspberry Pi 5

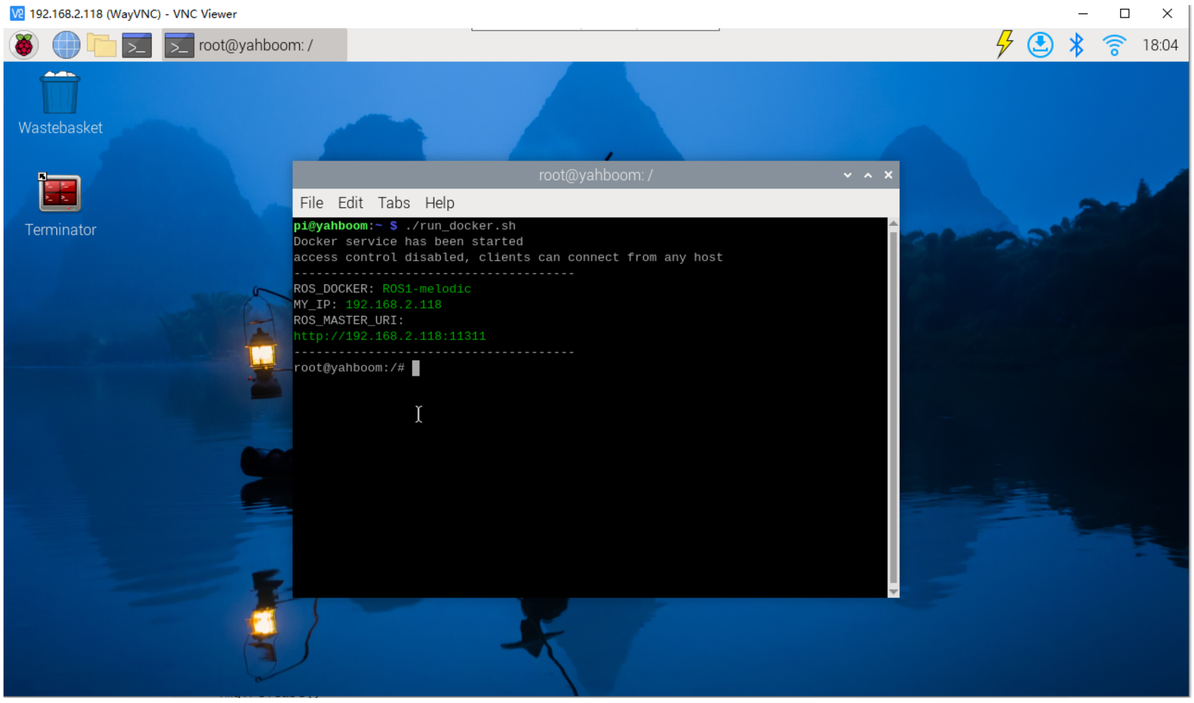
Before running, please confirm that the large program has been permanently closed

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
```



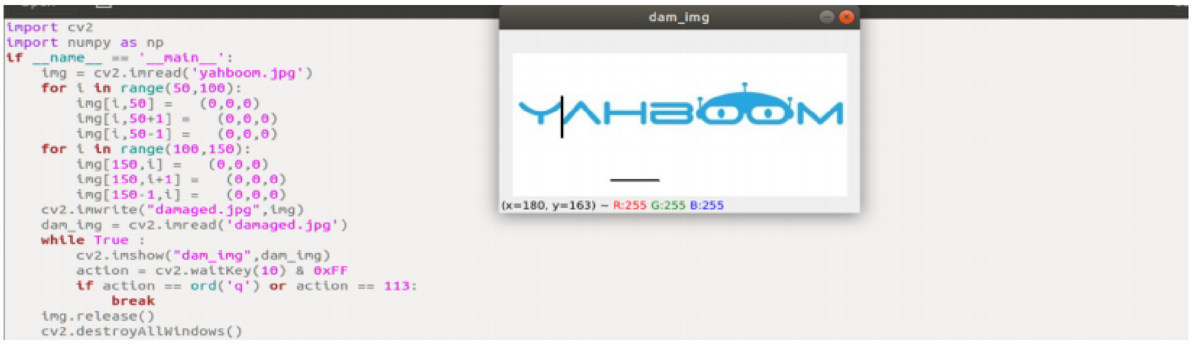
```
pi@yahboom:~ $ ./run_docker.sh
Docker service has been started
access control disabled, clients can connect from any host
-------------------------------------
ROS_DOCKER: ROS1-melodic
MY_IP: 192.168.2.118
ROS_MASTER_URI:
http://192.168.2.118:11311
-------------------------------------
root@yahboom:/#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

Run program

```
cd ~/transbot_ws/src/transbot_visual/opencv
python 4_1_1.py
```

```
import cv2
import numpy as np
if __name__ == '__main__':
img = cv2.imread('yahboom.jpg')
for i in range(50,100):
img[i,50] = (0,0,0)
img[i,50+1] = (0,0,0)
img[i,50-1] = (0,0,0)
for i in range(100,150):
img[150,i] = (0,0,0)
img[150,i+1] = (0,0,0)
img[150-1,i] = (0,0,0)
cv2.imwrite("damaged.jpg",img)
dam_img = cv2.imread('damaged.jpg')
while True :
cv2.imshow("dam_img",dam_img)
action = cv2.waitKey(10) & 0xFF
if action == ord('q') or action == 113:
break
img.release()
cv2.destroyAllWindows()
```

After running, a picture will be generated, which is regarded as a damaged picture of the original picture.



```
import cv2
import numpy as np
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    for i in range(50,100):
        img[i,50] =   (0,0,0)
        img[i,50+1] =   (0,0,0)
        img[i,50-1] =   (0,0,0)
    for i in range(100,150):
        img[150,i] =   (0,0,0)
        img[150,i+1] =   (0,0,0)
        img[150-1,i] =   (0,0,0)
    cv2.imwrite("damaged.jpg",img)
    dam_img = cv2.imread('damaged.jpg')
    while True :
        cv2.imshow("dam_img",dam_img)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv2.destroyAllWindows()
```

(2) To repair the photo just created, first read it, then create the mask, and finally use the function to repair it

Run program

jetson motherboard/Raspberry Pi 4B

```
cd ~/transbot_ws/src/transbot_visual/opencv
python 4_1_2.py
```

Raspberry Pi 5

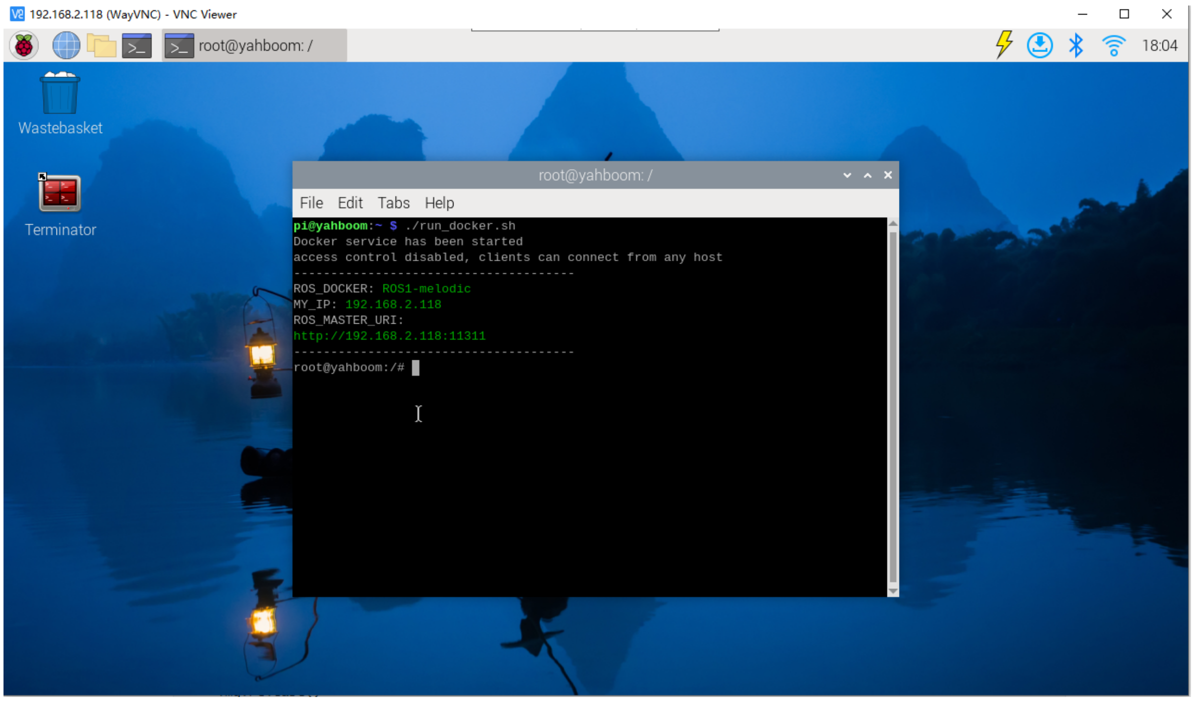
Before running, please confirm that the large program has been permanently closed

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
```

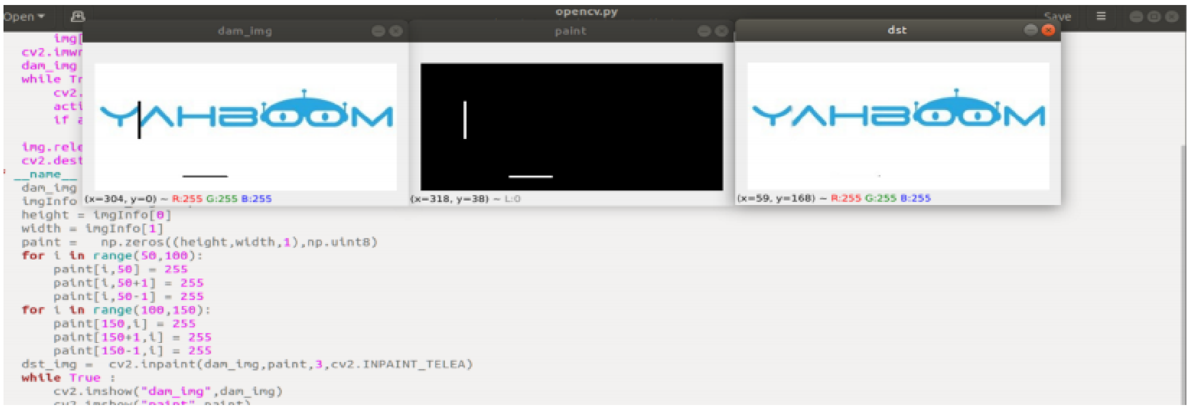


```
pi@yahboom:~ $ ./run_docker.sh
Docker service has been started
access control disabled, clients can connect from any host
----------------------------------------
ROS_DOCKER: ROS1-melodic
MY_IP: 192.168.2.118
ROS_MASTER_URI:
http://192.168.2.118:11311
----------------------------------------
root@yahboom:/#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

Run program

```
cd ~/transbot_ws/src/transbot_visual/opencv
python 4_1_2.py
```

```
import cv2
import numpy as np
if __name__ == '__main__':
dam_img = cv2.imread('damaged.jpg')
imgInfo = dam_img.shape
height = imgInfo[0]
width = imgInfo[1]
paint = np.zeros((height,width,1),np.uint8)
for i in range(50,100):
paint[i,50] = 255
paint[i,50+1] = 255
paint[i,50-1] = 255
for i in range(100,150):
paint[150,i] = 255
paint[150+1,i] = 255
paint[150-1,i] = 255
dst_img = cv2.inpaint(dam_img,paint,3,cv2.INPAINT_TELEA)
while True :
cv2.imshow("dam_img",dam_img)
cv2.imshow("paint",paint)
cv2.imshow("dst",dst_img)
action = cv2.waitKey(10) & 0xFF
if action == ord('q') or action == 113:
break
img.release()
cv2.destroyAllWindows()
```


Open
opencv.py
Save
dam_img
paint
dst
(x=304, y=0) ~ R:255 G:255 B:255
(x=318, y=38) ~ L:0
(x=59, y=168) ~ R:255 G:255 B:255

```
height = imgInfo[0]
width = imgInfo[1]
paint =   np.zeros((height,width,1),np.uint8)
for i in range(50,100):
    paint[i,50] = 255
    paint[i,50+1] = 255
    paint[i,50-1] = 255
for i in range(100,150):
    paint[150,i] = 255
    paint[150+1,i] = 255
    paint[150-1,i] = 255
dst_img =  cv2.inpaint(dam_img,paint,3,cv2.INPAINT_TELEA)
while True :
    cv2.imshow("dam_img",dam_img)
```

As shown in the figure, the left is before repair, the middle is the mask image, and the right is the original image after repair.

4.2. OpenCV image brightness enhancement

1. Implementation process: synchronously amplify the three-channel value of each pixel while keeping the channel value between 0-255. In fact, it is to traverse each pixel, add and subtract values to them, and then judge the three channels. Whether rgb is in the range of 0-255, if it is greater or less than 255 or 0.
2. Code and actual effect display

Run program

jetson motherboard/Raspberry Pi 4B

```
cd ~/transbot_ws/src/transbot_visual/opencv
python4_2.py
```

Raspberry Pi 5

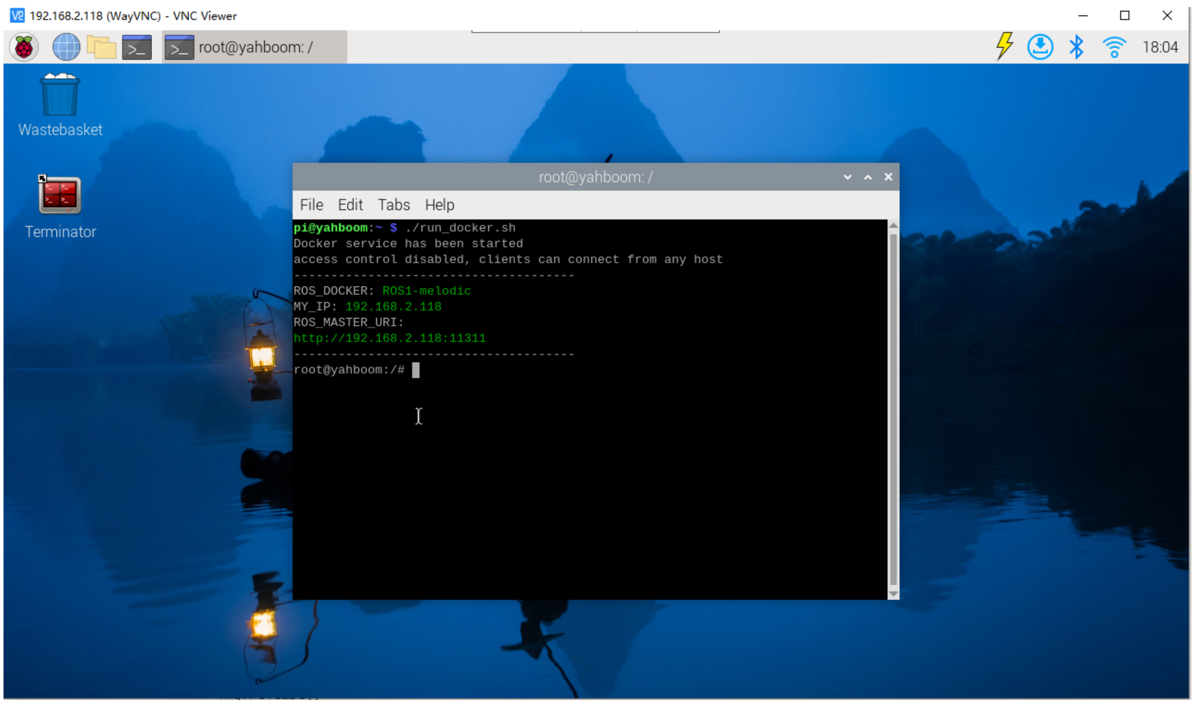
Before running, please confirm that the large program has been permanently closed

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
```



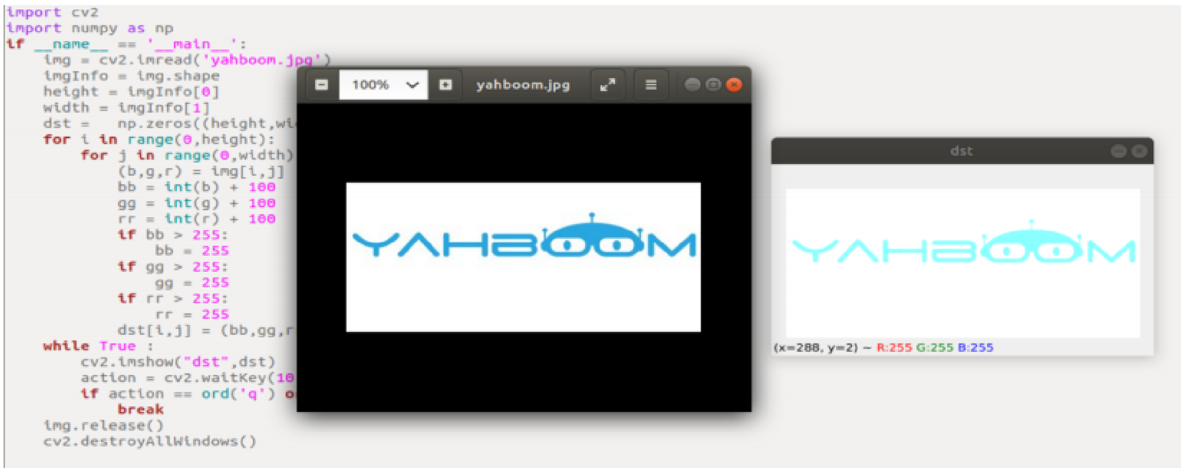
```
pi@yahboom:~ $ ./run_docker.sh
Docker service has been started
access control disabled, clients can connect from any host
--------------------------------------
ROS_DOCKER: ROS1-melodic
MY_IP: 192.168.2.118
ROS_MASTER_URI:
http://192.168.2.118:11311
--------------------------------------
root@yahboom:/#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

Run program

```
cd ~/transbot_ws/src/transbot_visual/opencv
python4_2.py
```

```
import cv2
import numpy as np
if __name__ == '__main__':
img = cv2.imread('yahboom.jpg')
imgInfo = img.shape
height = imgInfo[0]
width = imgInfo[1]
dst = np.zeros((height,width,3),np.uint8)
for i in range(0,height):
for j in range(0,width):
(b,g,r) = img[i,j]
bb = int(b) + 100
gg = int(g) + 100
rr = int(r) + 100
```

```
ifbb>255:
b = 255
if gg > 255:
gg = 255
ifrr > 255:
rr = 255
dst[i,j] = (bb,gg,rr)
while True :
cv2.imshow("dst",dst)
action = cv2.waitKey(10) & 0xFF
if action == ord('q') or action == 113:
break
img.release()
cv2.destroyAllWindows()
```



```
import cv2
import numpy as np
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    imgInfo = img.shape
    height = imgInfo[0]
    width = imgInfo[1]
    dst =   np.zeros((height,wi
    for i in range(0,height):
        for j in range(0,width)
            (b,g,r) = img[i,j]
            bb = int(b) + 100
            gg = int(g) + 100
            rr = int(r) + 100
            if bb > 255:
                bb = 255
            if gg > 255:
                gg = 255
            if rr > 255:
                rr = 255
            dst[i,j] = (bb,gg,r
    while True :
        cv2.imshow("dst",dst)
        action = cv2.waitKey(10
        if action == ord('q') o
            break
    img.release()
    cv2.destroyAllWindows()
```

The picture on the left is the original picture, and the picture on the back is the photo after increasing the brightness.

4.3. OpenCV image skin whitening

1. OpenCV implements the function of skin resurfacing and whitening images. The principle of implementation is basically the same as that of "1.20 OpenCV Image Brightness Enhancement", except that here we do not need to process the r value. We only need to follow this formula, p = p(x)*1.4+ y, where p(x) represents the b channel or g channel, and y represents the value that needs to be increased or decreased. Similarly, after adding the value, we need to make a judgment on the value.
2. Code and actual effect display

Run program

jetson motherboard/Raspberry Pi 4B

```
cd ~/transbot_ws/src/transbot_visual/opencv
python 4_3.py
```

Raspberry Pi 5

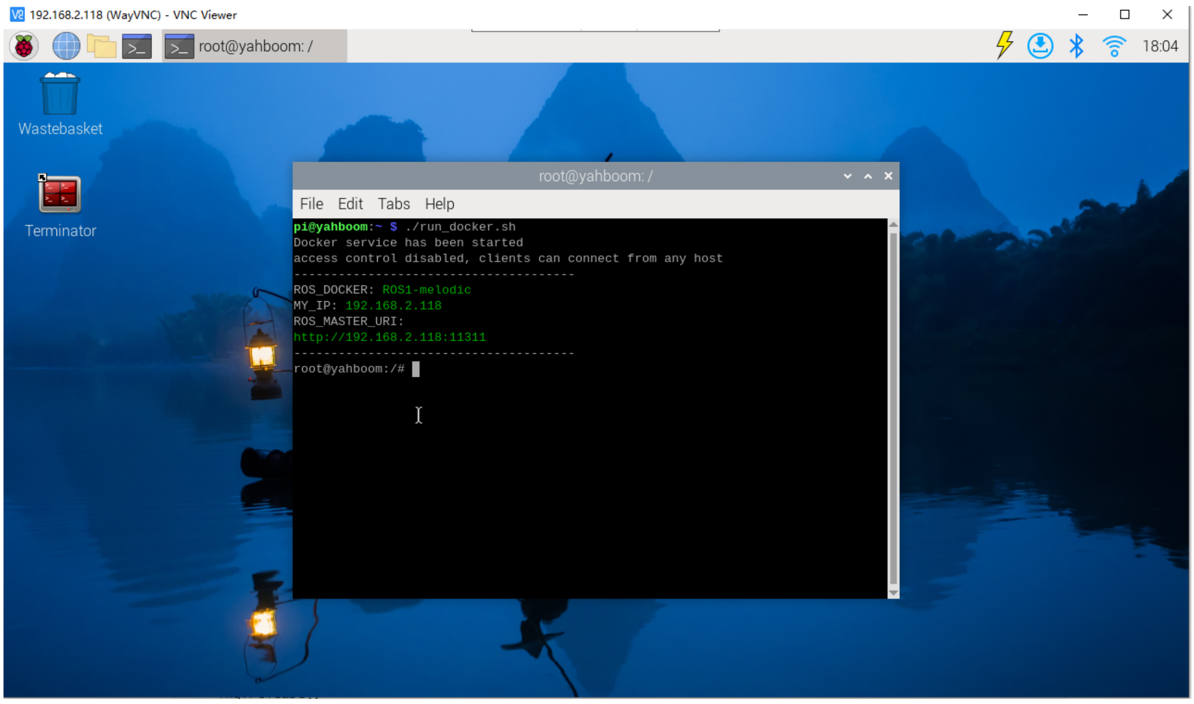
Before running, please confirm that the large program has been permanently closed

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
```

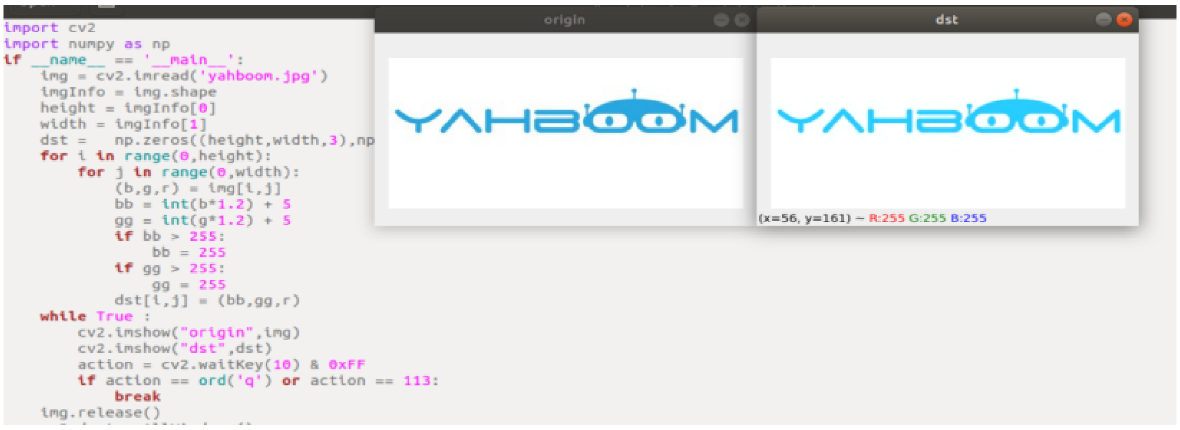


```
pi@yahboom:~ $ ./run_docker.sh
Docker service has been started
access control disabled, clients can connect from any host
--------------------------------------
ROS_DOCKER: ROS1-melodic
MY_IP: 192.168.2.118
ROS_MASTER_URI:
http://192.168.2.118:11311
--------------------------------------
root@yahboom:/#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

Run program

```
cd ~/transbot_ws/src/transbot_visual/opencv
python 4_3.py
```

```python
import cv2
import numpy as np
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    imgInfo = img.shape
    height = imgInfo[0]
    width = imgInfo[1]
    dst =   np.zeros((height,width,3),np.uint8)
    for i in range(0,height):
        for j in range(0,width):
            (b,g,r) = img[i,j]
            bb = int(b*1.4) + 5
            gg = int(g*1.4) + 5
            if bb > 255:
                bb = 255
            if gg > 255:
                gg = 255
            dst[i,j] = (bb,gg,r)
    while True :
        cv2.imshow("origin",img)
        cv2.imshow("dst",dst)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv2.destroyAllWindows()
```



```
import cv2
import numpy as np
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    imgInfo = img.shape
    height = imgInfo[0]
    width = imgInfo[1]
    dst =   np.zeros((height,width,3),np
    for i in range(0,height):
        for j in range(0,width):
            (b,g,r) = img[i,j]
            bb = int(b*1.2) + 5
            gg = int(g*1.2) + 5
            if bb > 255:
                bb = 255
            if gg > 255:
                gg = 255
            dst[i,j] = (bb,gg,r)
    while True :
        cv2.imshow("origin",img)
        cv2.imshow("dst",dst)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
```