Face recognition tracking

Note: The ROS_DOMAIN_ID of the Raspberry Pi and the microROS control board need to be consistent. You can check [MicroROS Control Board Parameter Configuration] to set the microROS control board ROS_DOMAIN_ID. Check the tutorial [Connect MicroROS Agent] to determine whether the IDs are consistent.

1、Program Description

After running the program, when a person's face is displayed on the screen and a box appears surrounding the person's face, the PTZ camera will move along with the movement of the person's face.

2、Steps

Program code reference path:

```
/root/yahboomcar_ws/src/yahboomcar_astra/yahboomcar_astra/face_fllow.py
```

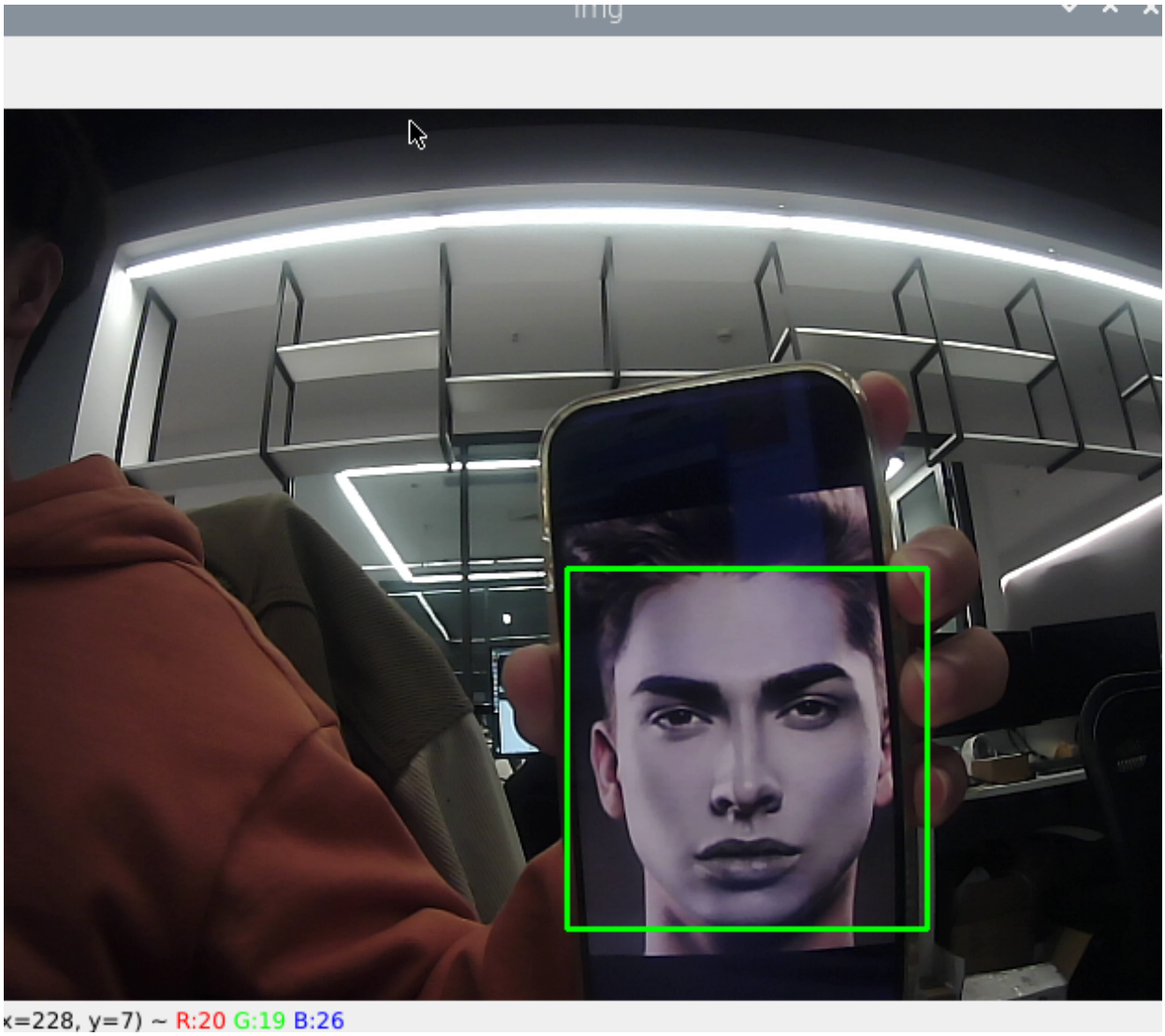
2.1、Start command

After entering the docker container, according to the actual car model, enter the terminal.

```
ros2 run yahboomcar_astra face_fllow
```

After the program is started, the following camera screen will appear.

When the face is recognized, the frame will be selected, and the 2D pan/tilt will follow the movement of the face, and the terminal will print the movement angle.

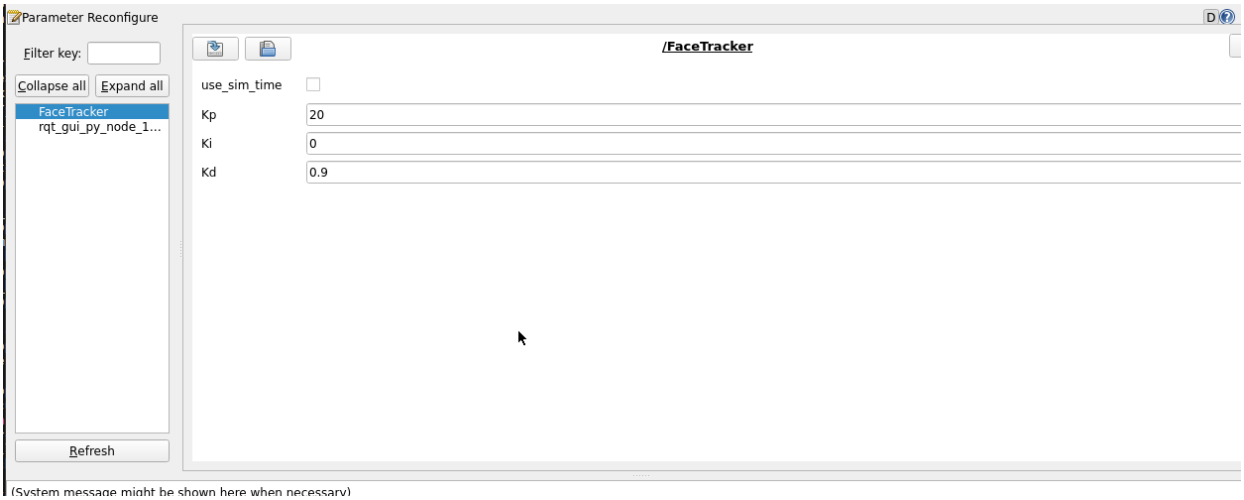



```
servo1 -32.78316886829865
servo1 -30.07844475985071
servo1 -26.97481789417099
servo1 -23.515624985811442
servo1 -21.600336974269787
servo1 -19.257044349261555
servo1 -16.442306542066714
servo1 -13.520620538682824
servo1 -10.684912599783722
servo1 -7.7940241190449076
servo1 -5.077315830114282
servo1 -2.502278319175432
servo1 -0.7169823161321474
servo1 1.269552969667902
servo1 1.9823099893802447
servo1 2.376723120090557
servo1 2.7854900205973907
servo1 3.4503393032376604
servo1 2.5619827777143556
servo1 2.1484540028389736
servo1 1.8792897101748922
servo1 1.6134116784231067
servo1 1.509800274913642
servo1 1.438316729859575
```

2.2、Dynamic parameter adjustment

Enter the following commands in the Docker terminal.

```
ros2 run rqt_reconfigure rqt_reconfigure
```



After modifying the parameters, click on a blank space in the GUI to write the parameter values. As can be seen from the above figure,

- faceTracker mainly adjusts the three parameters of PID to make the gimbal more sensitive

3、core code

3.1、face_fllow .py

The principle of function implementation is similar to that of object tracking. It calculates the rotation angle of the servo based on the center coordinates of the target, and then publishes it to the chassis. Part of the code is as follows.

```
# Calculate center point
 cv.putText(frame, text, (20, 30), cv.FONT_HERSHEY_SIMPLEX, 0.9, (0, 0, 255), 1)
        face_patterns =
cv2.CascadeClassifier('/root/yahboomcar_ws/src/yahboomcar_astra/yahboomcar_astra/haa
rcascade_frontalface_default.xml')
        faces = face_patterns.detectMultiScale(frame , scaleFactor=1.1,
minNeighbors=5, minSize=(100, 100))
        if len(faces)>0:
            for (x, y, w, h) in faces:
                m=x
                n=y
                cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)
            self.execute(m,n)
#Calculate the steering gear rotation based on the center point
 [x_Pid, y_Pid] = self.PID_controller.update([point_x - 320, point_y - 240])
        if self.img_flip == True:
            self.PWMServo_X += x_Pid
```

```
            self.PWMServo_Y += y_Pid
        else:
            self.PWMServo_X  -= x_Pid
            self.PWMServo_Y  += y_Pid

        if self.PWMServo_X  >= 90:
            self.PWMServo_X  = 90
        elif self.PWMServo_X  <= -90:
            self.PWMServo_X  = -90
        if self.PWMServo_Y >= 20:
            self.PWMServo_Y = 20
        elif self.PWMServo_Y <= -90:
            self.PWMServo_Y = -90
```