6、Face tracking

6.1 Introduction to gameplay

This course mainly uses the robot's camera. Based on face detection, it adds the function of PID algorithm to control the camera pan/tilt to track faces.

6.2Core content analysis

Initialize the initial angle of the camera pan/tilt. The default settings are S1=90 degrees and S2=30 degrees. The default angles can be modified according to actual needs.

```
from MutoLib import Muto
g_bot = Muto()

g_last_x = 90
g_last_y = 30

def reset_servo():
    global g_last_x, g_last_y
    g_last_x = 90
    g_last_y = 30
    g_bot.Gimbal_1_2(g_last_x, g_last_y)
reset_servo()
```

In order to make the camera pan/tilt track faces more efficiently and quickly, an incremental PID algorithm is used here to control the rotation angle of the robot servo pan/tilt.

```
from simplePID import PID
PID_X_Kp = 0.03
PID_X_Ki = 0
PID_X_Kd = 0.02
PID_Y_Kp = 0.03
PID_Y_Ki = 0
PID_Y_Kd = 0.02
PID_X = PID(0, PID_X_Kp, PID_X_Ki, PID_X_Kd)
PID_Y = PID(0, PID_Y_Kp, PID_Y_Ki, PID_Y_Kd)
```

Analyze the image, extract the face position information, and send it to the robot to control the movement of the servo and gimbal.

```
def task_processing():
    global g_stop_program, g_start_function
    t_start = time.time()
    m_fps = 0
    while g_camera.isOpened():
        if g_stop_program:
            break
        ret, frame = g_camera.read()

        PID_slider_value()
        if g_start_function:
            state, (face_x,face_y,face_w,face_h) = face_detect(frame)
            if state != None:
                cv2.rectangle(frame, (face_x, face_y), (face_x+face_w,
face_y+face_h), (0,255,255), 1)
                robot_control(face_x+face_w/2-320, face_y+face_h/2-240)

        m_fps = m_fps + 1
        fps = m_fps / (time.time() - t_start)
        if (time.time() - t_start) >= 2:
            m_fps = fps
            t_start = time.time() - 1
        cv2.putText(frame, "FPS " + str(int(fps)), (10,20),
cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0,255,255), 1)

        # 图像传输给显示组件 The image is transmitted to the display component
        image_widget.value = bgr8_to_jpeg(frame)
        time.sleep(.01)
```

According to the detected coordinate (x, y) value of the human face, the robot's camera pan/tilt is controlled to rotate. The ideal effect is to keep the human face in the middle area of the image.

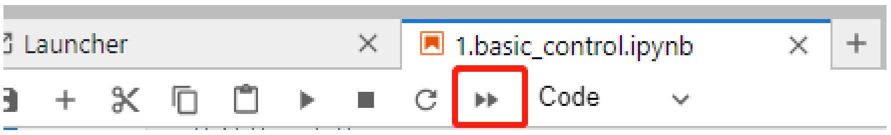
```
def robot_control(x, y):
    global g_last_x, g_last_y
    # print("raw:", x, y)
    if -20 < x < 20:
        x = 0
    if -20 < y < 20:
        y = 0
    value_x = int(PID_X.incremental(x))
    value_y = int(PID_Y.incremental(y))
    value_x = limit_value(value_x + g_last_x, 0, 180)
    value_y = limit_value(value_y + g_last_y, 0, 115)
    g_last_x = value_x
    g_last_y = value_y
    # print("angle:", value_x, value_y)
    g_bot.Gimbal_1_2(value_x, value_y)
```

6.3 operation

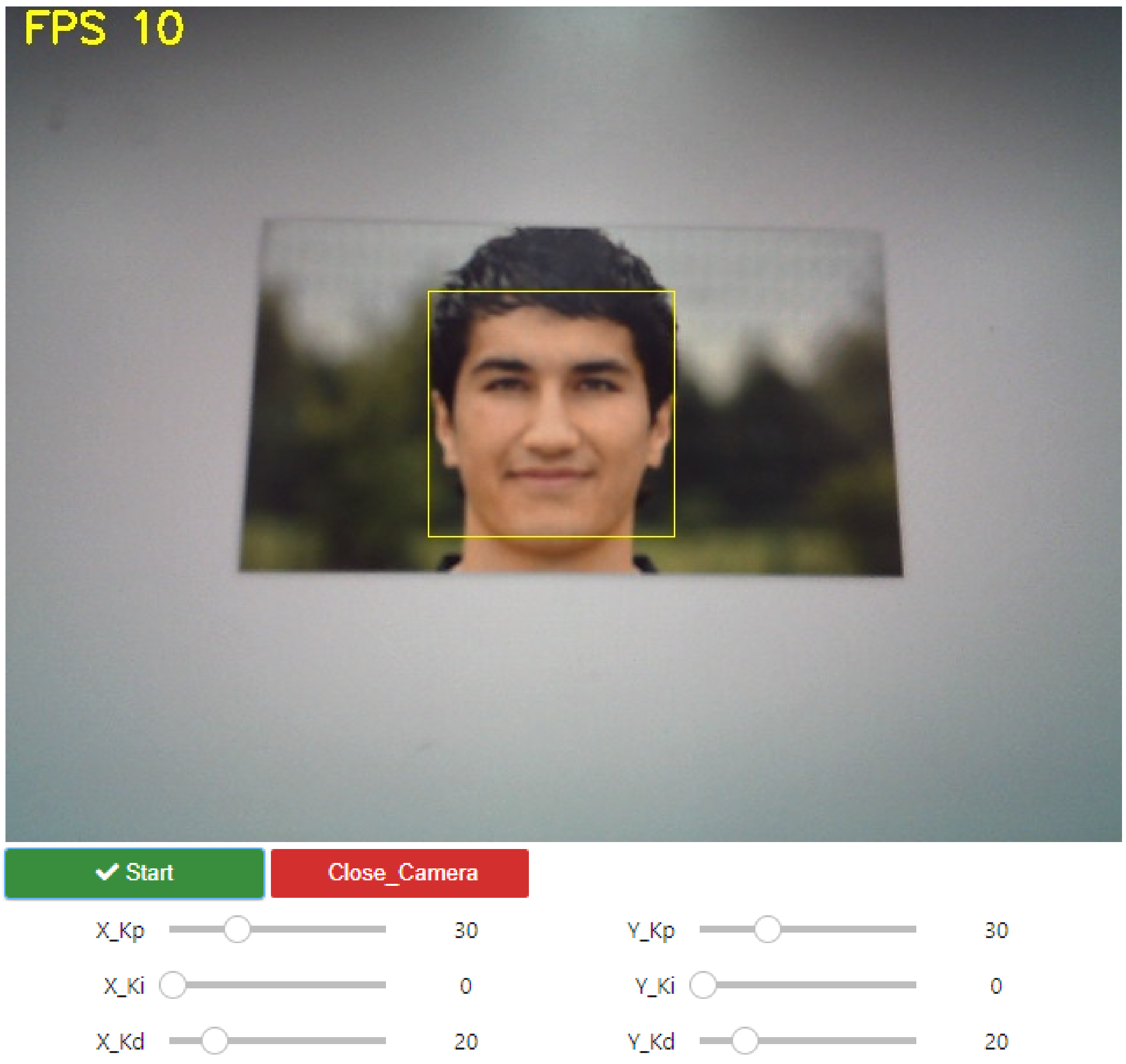
Open the jupyterLab client and find the code path:

```
muto/Samples/AI_Samples/02_color_tracking/color_tracking.ipynb
```

Click to run all cells, and then scroll to the bottom to see the generated controls.



When the display control is started, it only displays the camera image. You need to click the Start button first to turn on the face detection and camera pan/tilt movement functions. At this time, if a face enters the camera detection area, the face will be automatically framed, and the camera pan/tilt will Rotate to a suitable angle and keep the face in the middle of the image.



If you find that the camera PTZ tracks faces too fast or too slowly, you can adjust the PID parameters appropriately to optimize the effect of the camera PTZ tracking faces. Among them, PID_X adjusts the angle of the S1 servo, and PID_Y adjusts the angle of the S2 servo.

Finally click the Close_Camera button to close the camera.