

# 智能系统与控制

树莓派网络控制:

人脸识别登陆



人脸检测登陆



于泓

鲁东大学

信息与电气工程学院

2022.6.18

任务：设计一个基于人脸检测的登陆页面，当出现允许登陆的人脸时，可以进入系统，实现小灯控制，环境检测等功能。



检测到允许  
的人脸  
后进行页面  
跳转



## 人脸识别部分

- (1) 在 main.py 中构造一个全局变量  
存储人脸识别所需的  
**识别器** **存储的人脸特征** 等

```
print("加载人脸检测器")
det_face = dlib.get_frontal_face_detector()

# 加载标志点检测器
det_landmark = dlib.shape_predictor("shape_predictor_68_face_landmarks GTX.dat") # 68点5
sp = dlib.shape_predictor("shape_predictor_5_face_landmarks.dat") # 5点

# 加载人脸特征提取器
facerec = dlib.face_recognition_model_v1("dlib_face_recognition_resnet_model_v1.dat")

# 加载训练好的人脸模型
model = np.load('trainer.npz')
face_vectors = model['face_vectors']
face_ids = model['ids']

opt_face = dict()
opt_face['detector'] = det_face # 人脸检测器
opt_face['recongnizer'] = facerec # 人脸特征提取器
opt_face['sp']=sp # 关键点提取器
opt_face['face_ids'] = face_ids # 已存储的人脸id
opt_face['face_vectors'] = face_vectors # 已存储的人脸矢量
opt_face['face_allowed'] = ['yuhong'] # 允许通过认证的人脸ID
opt_face['flag'] = False # 认证成功标志
```

人脸检测标志位

## (2) 构造生成器 实现视频显示和 人脸检测

视频

```
def gen_face_recognition(camera, opts)
```

参数

```
    detector = opts['detector']
    sp = opts['sp']
    facerec = opts['recongnizer']
    face_vectors = opts['face_vectors']
    face_ids = opts['face_ids']
    face_allowed = opts['face_allowed']

    while True:

        img = camera.get_frame()
        if img is None:
            time.sleep(0.1)
            continue

        # BGR 转 gray
        img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

        # BGR 转 RGB
        img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

        # 进行人脸检测
        dets = detector(img_gray, 1)
```

```
# 遍历检测的人脸
for k, d in enumerate(dets):
    # 画框
    cv2.rectangle(img, (d.left(), d.top()), (d.right(), d.bottom()), (255, 0, 0), 3)

    # 标志点检测
    shape = sp(img_rgb, d)

    # 获取人脸特征
    face_vector = facerec.compute_face_descriptor(img_rgb, shape)

    # 进行识别返回ID与距离
    face_id, dis = face_recognize(np.array(face_vector), face_vectors, face_ids)

    if (dis < 0.45):
        str_out = "%s %.2f"%(face_id, dis)

        if face_id in face_allowed:
            opts['flag'] = True

    else:
        str_out = "unknown %.2f"%(dis)

    # 检测结果文字输出
    cv2.putText(img, str_out, (d.left()+5, d.top()), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 2)

ret, jpeg = cv2.imencode('.jpg', img)
# 对图像进行编码输出
yield(b'--frame\r\n'+b'Content-Type: image/jpeg\r\n\r\n' + jpeg.tobytes() + b'\r\n\r\n')
```

## main.py 构建路由

```
@app.route('/', methods=['GET', 'POST'])
def face():
    if request.method == 'GET':
        opt_face['flag'] = False
        return render_template('video_face.html', videourl=url_for('video_face_rec'))

@app.route('/video_face_rec')
def video_face_rec():
    return Response(gen_face_recognition(m_camera, opt_face), mimetype='multipart/x-mixed-replace; boundary=frame')

@app.route('/checkFace', methods=['GET', 'POST'])
def checkFace():
    print(opt_face['flag'])
    if opt_face['flag'] == True:
        print("跳转")

    return jsonify({'flag': 'True'})
    else:
        return jsonify({'flag': 'Flase'})
```

主页面

主页面中视频显示部分

变量检测  
查看是否检测到人脸

```
<!DOCTYPE html>
<html lang="en">
<head>
  <!-- <meta http-equiv="refresh" content="1" charset="UTF-8"> -->
  <title>人脸检测登陆</title>
</head>

<script>
  function getFlag() {
    var xmlHttp = new XMLHttpRequest();
    var baseUrl = "/checkFace";
    xmlHttp.open( "GET", baseUrl, false );
    xmlHttp.send( null );
    return xmlHttp.responseText;
  }

  function changeUrl() {
    var Ino_Flag = getFlag();
    var json = JSON.parse(Ino_Flag);
    if (json.flag == "True")
    {
      window.location.assign("/light")
    }
  }

  function addTimer() {
    setInterval(changeUrl, 5000);
  }
</script>
```

```
<body onload="addTimer();"

  <h1>人脸检测登陆</h1>
  
  <br>

</body>
</html>
```