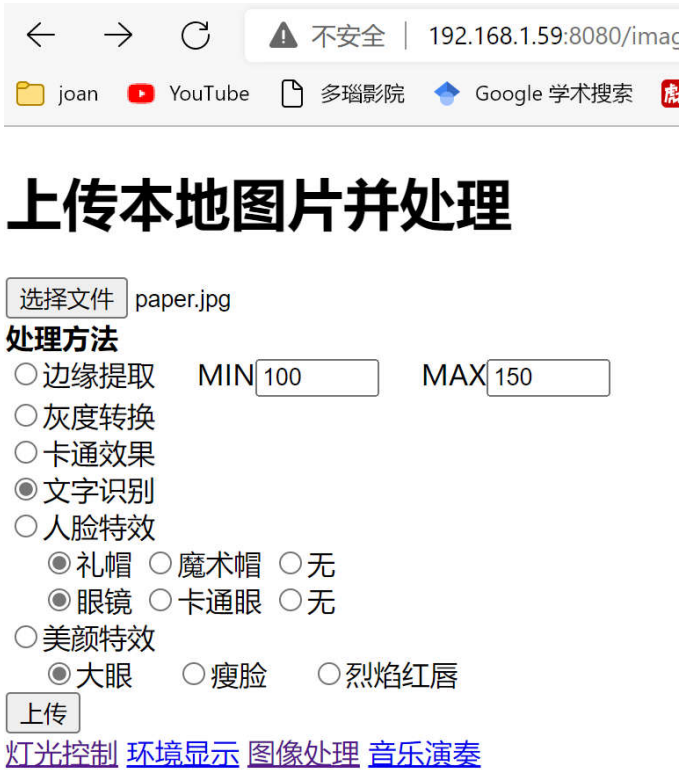
智能系统与控制



树莓派网络控制：搭建图像处理平台

于泓
鲁东大学
信息与电气工程学院
2022.5.1

上传本地图片并处理

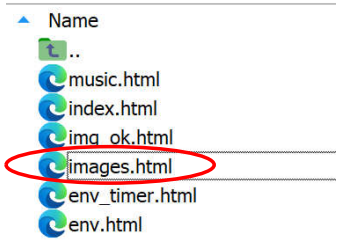
选择文件 paper.jpg

处理方法

- ○ 边缘提取 MIN 100 MAX 150
- ○ 灰度转换
- ○ 卡通效果
- ◉ 文字识别
- ○ 人脸特效
  - ◉ 礼帽 ○ 魔术帽 ○ 无
  - ◉ 眼镜 ○ 卡通眼 ○ 无
- ○ 美颜特效
  - ◉ 大眼 ○ 瘦脸 ○ 烈焰红唇

上传

灯光控制 环境显示 图像处理 音乐演奏




```
<h1>上传本地图片并处理</h1>

<form action="" method='POST' enctype="multipart/form-data" >
    <input type="file" name="file"/>
    <br>

    <b>处理方法</b>
    <br>
    <input type="radio" name="method_img_process" value="edge" checked>边缘提取
    <span>    </span>
    MIN<input type="text" name="th_edge_MIN" value=100 size=5>
    <span>    </span>
    MAX<input type="text" name="th_edge_MAX" value=150 size=5>
    <br>
    <input type="radio" name="method_img_process" value="gray">灰度转换
    <br>
    <input type="radio" name="method_img_process" value="effect_cartoon">卡通效果
    <br>
    <input type="radio" name="method_img_process" value="ocr">文字识别
    <br>
    <input type="radio" name="method_img_process" value="face_det">人脸特效
```

上传本地图片并处理

选择文件 paper.jpg

处理方法

○边缘提取 MIN 100 MAX 150
○灰度转换
○卡通效果
◉文字识别
○人脸特效
◉礼帽 ○魔术帽 ○无
◉眼镜 ○卡通眼 ○无
○美颜特效
◉大眼 ○瘦脸 ○烈焰红唇

上传

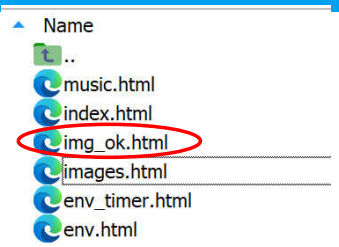
灯光控制 环境显示 图像处理 音乐演奏

```
<input type="radio" name="method_img_process" value="face_det">人脸特效
<br>
<span>    </span><input type="radio" name="face_hat" value="hat1" checked>礼帽
                          <input type="radio" name="face_hat" value="hat2">魔术帽
                          <input type="radio" name="face_hat" value="None">无
<br>
<span>    </span><input type="radio" name="face_eye" value="eye1" checked>眼镜
                          <input type="radio" name="face_eye" value="eye2">卡通眼
                          <input type="radio" name="face_eye" value="None">无

<br>
<input type="radio" name="method_img_process" value="face_beauty">美颜特效

<br>
 <span>    </span>   <input type="radio" name="Face_Beauti" value="bigeye" checked>大眼
 <span>    </span>   <input type="radio" name="Face_Beauti" value="thinface">瘦脸
 <span>    </span>   <input type="radio" name="Face_Beauti" value="colorLip">烈焰红唇

 <br>
  <input type="submit" value="上传" class="button-new" />
  <span>{{message}}</span>
```

图片处理

处理算法：卡通效果

卡通效果展示
灯光控制 环境显示 图像处理 音乐演奏




```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>图片演示</title>
</head>
<body>
    <h1>图片处理</h1>
    <h1>处理算法：{{method}}</h1>

    <table><tr>
        <td><img src="{{url_src}}"  alt="你的图片被外星人劫持了～～"/></td>
        <td> <img src="{{url_dst}}"  alt="你的图片被外星人劫持了～～"/></td
    </tr></table>
    <br>
    <span>{{info_result}}</span>
    <br>
```

```
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

# 允许的图像格式
ALLOWED_EXTENSIONS = set(['png', 'jpg', 'JPG', 'PNG', 'bmp'])
def allowed_file(filename):
    return '.' in filename and filename.rsplit('.', 1)[1] in ALLOWED_EXTENSIONS

"
```

提前加载
人脸检测器

```
@app.route('/images',methods=['GET', 'POST'])
def image():
    # get 方式访问，返回页面
    if request.method == 'GET':
        return render_template('images.html')

    # pos 方式访问 首先进行文件的验证
    if request.method == 'POST':
        if  request.files:
            f = request.files['file']
            print(f)
            if not (f and allowed_file(f.filename)):
                print("file error")
                return render_template('images.html',message="上传文件格式错误")
        else:
            return render_template('images.html',message="文件为空")
```

进行文件格式验证

```python
# 对输入图像文件 随机取名进行保存
basepath = os.path.dirname(__file__)

result_image_pid = str(int(time.time())) + str(random.randint(0, 10000)).zfill(5) + '.' + \
                   f.filename.split('.')[-1]
upload_path = os.path.join(basepath, 'static/images',result_image_pid)
f.save(upload_path)

# 加载需要处理的图像
img = cv2.imread(upload_path)

# 获取图像处理方式
process_method = request.form.get("method_img_process")
print('process_method',process_method)
```

```
<b>处理方法</b>
<br>
<input type="radio" name="method_img_process" value="edge" checked>边缘提取
<span>    </span>
MIN<input type="text" name="th_edge_MIN" value=100 size=5>
<span>    </span>
MAX<input type="text" name="th_edge_MAX" value=150 size=5>
<br>
<input type="radio" name="method_img_process" value="gray">灰度转换
<br>
<input type="radio" name="method_img_process" value="effect_cartoon">卡通效果
<br>
<input type="radio" name="method_img_process" value="ocr">文字识别
<br>
<input type="radio" name="method_img_process" value="face_det">人脸特效
```

```python
# 根据不同的处理方式进行处理
if process_method == "gray":
    message = " 灰度转换"
    img_gray= img2gray(img)
    str_info = "转换成功"
    # 创建结果文件
    dealed_image_pid = str(int(time.time())) + str(random.randint(0, 10000)).zfill(5) + '.' + \
                       f.filename.split('.')[-1]
    dealed_path = os.path.join(basepath, 'static/images',dealed_image_pid)
    cv2.imwrite(dealed_path,img_gray)


    url_src = url_for('static', filename= './images/'+result_image_pid)
    url_dst = url_for('static', filename= './images/'+dealed_image_pid)
    return render_template('img_ok.html',
                           method=message,
                           url_src=url_src,
                           url_dst=url_dst,
                           info_result =str_info)
```

```html
<body>
    <h1>图片处理</h1>
    <h1>处理算法：{{method}}</h1>

    <table><tr>
        <td><img src="{{url_src}}"  alt="你的图片被外星人劫持了～～"/></td>
        <td> <img src="{{url_dst}}"  alt="你的图片被外星人劫持了～～"/></td>
    </tr></table>
    <br>
    <span>{{info_result}}</span>
    <br>
```

```
elif process_method == "edge":
    message = "边缘检测"
    th_min = int(request.form.get("th_edge_MIN"))
    th_max = int(request.form.get("th_edge_MAX"))
    img_edge = img2edge(img,th_min,th_max)
    str_info = "边缘提取成功"

    # 创建结果文件
    dealed_image_pid = str(int(time.time())) + str(random.randint(0, 10000)).zfill(5) + '.' + \
                       f.filename.split('.')[-1]
    dealed_path = os.path.join(basepath, 'static/images',dealed_image_pid)
    cv2.imwrite(dealed_path,img_edge)


    url_src = url_for('static', filename= './images/'+result_image_pid)
    url_dst = url_for('static', filename= './images/'+dealed_image_pid)
    return render_template('img_ok.html',
                           method=message,
                           url_src=url_src,
                           url_dst=url_dst,
                           info_result =str_info)
```

```
elif process_method == "effect_cartoon":
    message = "卡通效果"
    print("卡通效果")
    img_cartoon = img2cartoon(img)

    str_info = "卡通效果展示"

    # 创建结果文件
    dealed_image_pid = str(int(time.time())) + str(random.randint(0, 10000)).zfill(5) + '.' + \
                       f.filename.split('.')[-1]
    dealed_path = os.path.join(basepath, 'static/images',dealed_image_pid)
    cv2.imwrite(dealed_path,img_cartoon)


    url_src = url_for('static', filename= './images/'+result_image_pid)
    url_dst = url_for('static', filename= './images/'+dealed_image_pid)
    return render_template('img_ok.html',
                           method=message,
                           url_src=url_src,
                           url_dst=url_dst,
                           info_result =str_info)
```

```
elif process_method == "ocr":
    message = "文字识别"
    str_ocr = img2text(img)
    str_info = "识别结果为:  " + str_ocr
    url_src = url_for('static', filename= './images/'+result_image_pid)
    url_dst = url_for('static', filename= './images/blank.bmp')
    return render_template('img_ok.html',
                           method=message,
                           url_src=url_src,
                           url_dst=url_dst,
                           info_result =str_info)
```

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>图片演示</title>
</head>
<body>
    <h1>图片处理</h1>
    <h1>处理算法：{{method}}</h1>

    <table><tr>
        <td><img src="{{url_src}}"  alt="你的图片被外星人劫持了～～"/></td>
        <td> <img src="{{url_dst}}"  alt="你的图片被外星人劫持了～～"/></td>
    </tr></table>
    <br>
    <span>{{info_result}}</span>
    <br>
```

```python
elif process_method == "face_det":
    message = "人脸特效  "

    dict_effect = dict()

      dic_effect = {
                "hat": ["Normal","hat1.bmp"],
                "eye": ["glasses","glasses.bmp"],
                # "eye": ["double","left-eye.bmp","left-eye.bmp"]
            }

    style_hat = request.form.get("face_hat")
    if style_hat == "hat1":
        message = message+ "  礼帽"
        dict_effect.update({'hat':[]})
        dict_effect['hat'].append("Normal")
        dict_effect['hat'].append(os.path.join(basepath,'static','img_processing','hat1.bmp'))
    elif style_hat == "hat2":
        message = message+ "  魔法帽"
        dict_effect.update({'hat':[]})
        dict_effect['hat'].append("Normal")
        dict_effect['hat'].append(os.path.join(basepath,'static','img_processing','hat2.bmp'))
```

```html
<input type="radio" name="method_img_process" value="face_det">人脸特效
<br>
<span>    </span><input type="radio" name="face_hat" value="hat1" checked>礼帽
                        <input type="radio" name="face_hat" value="hat2">魔术帽
                        <input type="radio" name="face_hat" value="None">无
<br>
<span>    </span><input type="radio" name="face_eye" value="eye1" checked>眼镜
                        <input type="radio" name="face_eye" value="eye2">卡通眼
                        <input type="radio" name="face_eye" value="None">无
```

```
style_eye = request.form.get("face_eye")

if style_eye == "eye1":
    message = message + "   眼镜"
    dict_effect.update({'eye':[]})
    dict_effect['eye'].append("glasses")
    dict_effect['eye'].append(os.path.join(basepath,'static','img_processing','glasses.bmp'))

elif style_eye == "eye2":
    message = message + "   卡通眼"
    dict_effect.update({'eye':[]})
    dict_effect['eye'].append("double")
    dict_effect['eye'].append(os.path.join(basepath,'static','img_processing','left-eye.bmp'))
    dict_effect['eye'].append(os.path.join(basepath,'static','img_processing','right-eye.bmp'))

flag,img_dealed = face_effect(img,dict_effect,det_face,det_landmark)
```

```
flag,img_dealed = face_effect(img,dict_effect,det_face,det_landmark)
if flag:
    str_info = "特效添加成功"

     # 创建结果文件
    dealed_image_pid = str(int(time.time())) + str(random.randint(0, 10000)).zfill(5) + '.' + \
                       f.filename.split('.')[-1]
    dealed_path = os.path.join(basepath, 'static/images',dealed_image_pid)
    cv2.imwrite(dealed_path,img_dealed)

    url_src = url_for('static', filename= './images/'+result_image_pid)
    url_dst = url_for('static', filename= './images/'+dealed_image_pid)
    return render_template('img_ok.html',
                           method=message,
                           url_src=url_src,
                           url_dst=url_dst,
                           info_result =str_info)
else:
    str_info = "没有找到人脸"
    url_src = url_for('static', filename= './images/'+result_image_pid)
    url_dst = url_for('static', filename= './images/blank.jpg')
    return render_template('img_ok.html',
                           method=message,
                           url_src=url_src,
                           url_dst=url_dst,
                           info_result =str_info)
```

```
elif process_method == "face_beauty":
        message = " 美颜特效   "
        style_beauty = request.form.get("Face_Beauti")
        print("style_beauty",style_beauty)
        if style_beauty == "bigeye":
            message = message + " 大眼 "
            flag,img_out= big_eyes(img,det_face,det_landmark)
        elif style_beauty == "thinface":
            message = message + "  瘦脸"
            flag,img_out = thin_face(img,det_face,det_landmark)
        elif style_beauty == "colorLip":
            print("-----------lip in -----------------")
            message = message + "烈焰红唇"
            flag,img_out = color_lip(img)
            print("-------flag---------------",flag)
```

```
<br>
<input type="radio" name="method_img_process" value="face_beauty">美颜特效

<br>
 <span>    </span>   <input type="radio" name="Face_Beauti" value="bigeye" checked>
 <span>    </span>   <input type="radio" name="Face_Beauti" value="thinface">瘦脸
 <span>    </span>   <input type="radio" name="Face_Beauti" value="colorLip">烈焰红
```

```
if flag:
    str_info = "美颜添加成功"
     # 创建结果文件
    dealed_image_pid = str(int(time.time())) + str(random.randint(0, 10000)).zfill(5) + '.' + \
                   f.filename.split('.')[-1]
    dealed_path = os.path.join(basepath, 'static/images',dealed_image_pid)
    cv2.imwrite(dealed_path,img_out)

    url_src = url_for('static', filename= './images/'+result_image_pid)
    url_dst = url_for('static', filename= './images/'+dealed_image_pid)
    return render_template('img_ok.html',
                    method=message,
                    url_src=url_src,
                    url_dst=url_dst,
                    info_result =str_info)
else:
    str_info = "没有找到人脸"
    url_src = url_for('static', filename= './images/'+result_image_pid)
    url_dst = url_for('static', filename= './images/blank.bmp')
    return render_template('img_ok.html',
                    method=message,
                    url_src=url_src,
                    url_dst=url_dst,
                    info_result =str_info)
```