

# Python编程与人工智能实践

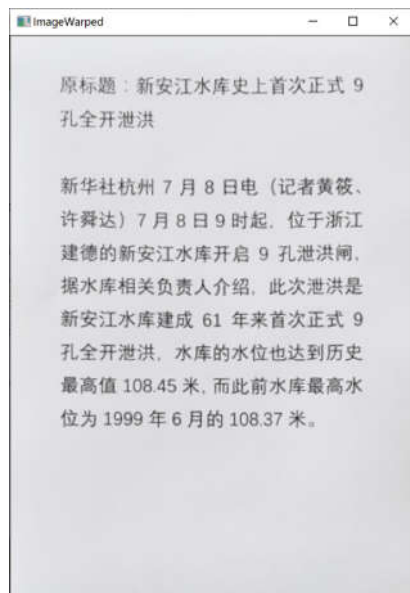
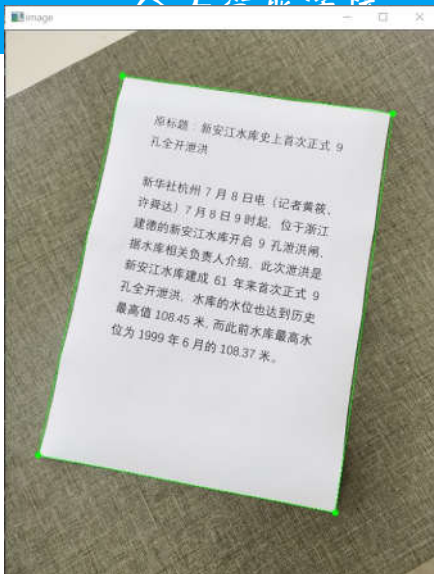
## 应用篇：文字扫描仪 (tesseract)

于泓

鲁东大学

信息与电气工程学院

2021.4.12



PS D:\工作相关\我设计的课程\python与人工智能课程设  
原 标 题：新安江水库史上首次正式9  
孔全开泄洪  
新华社杭州7月8日电（记者黄筱、  
许舜达）7月8日9时起，位于浙江  
建德的新安江水库开启9孔泄洪闸，  
据水库相关负责人介绍，此次泄洪是  
新安江水库建成61年来首次正式9  
孔全开泄洪，水库的水位也达到历史  
最高值108.45米，而此前水库最高水  
位为1999年6月的108.37米。

# tesseract

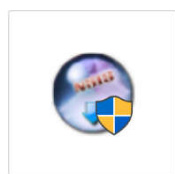
- Tesseract: 开源的OCR识别引擎, 初期Tesseract引擎由HP实验室研发, 后来贡献给了开源软件业, 后由Google进行改进、修改bug、优化, 重新发布。
- OCR(Optical Character Recognition): 光学字符识别, 是指电子设备(例如扫描仪或数码相机)检查纸上打印的字符, 通过检测暗、亮的模式确定其形状, 然后用字符识别方法将形状翻译成计算机文字的过程。

## tesseract 安装

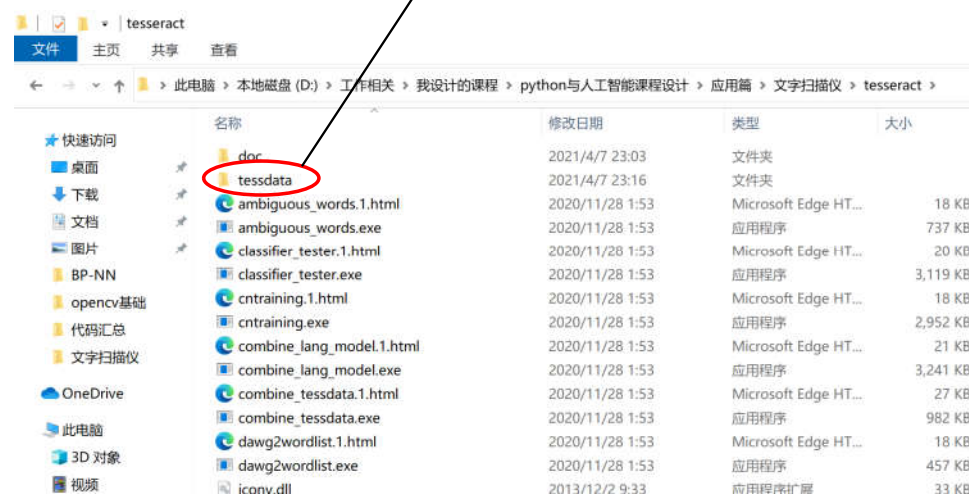
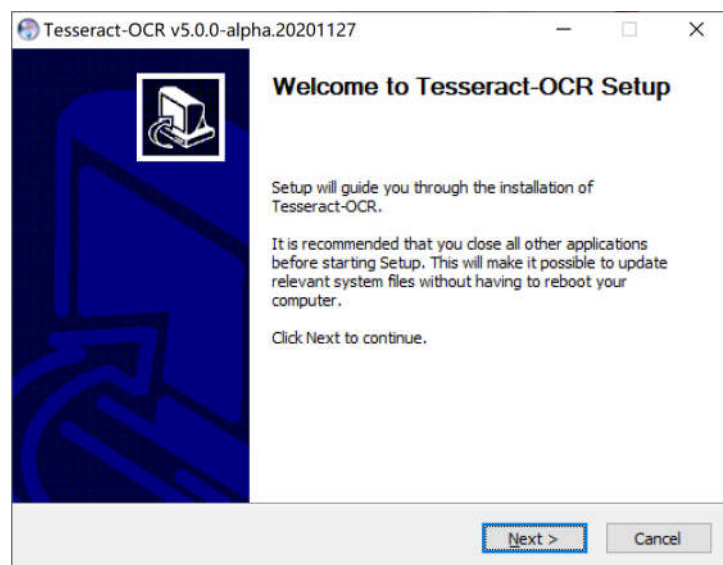
Win版下载地址: <https://digi.bib.uni-mannheim.de/tesseract/tesseract-ocr-w64-setup-v5.0.0-alpha.20201127.exe>

Win版下载页面: <https://github.com/UB-Mannheim/tesseract/wiki>

下载模型文件放入tessdata文件夹





tesseract-ocr-w  
64-setup-v5.0.0  
-alpha.2020112  
7.exe



中文模型下载页面

[https://github.com/tesseract-ocr/tessdata\\_best](https://github.com/tesseract-ocr/tessdata_best)

下载简体中文识别模型 放入tessdata 文件夹

 chi_sim.traineddata	Initial import (on behalf of Ray)
 chi_sim_vert.traineddata	Fix extra intra-word spacing for Chinese (GitHub issue #991)

安装tesseract 支持Python包

pip install pytesseract

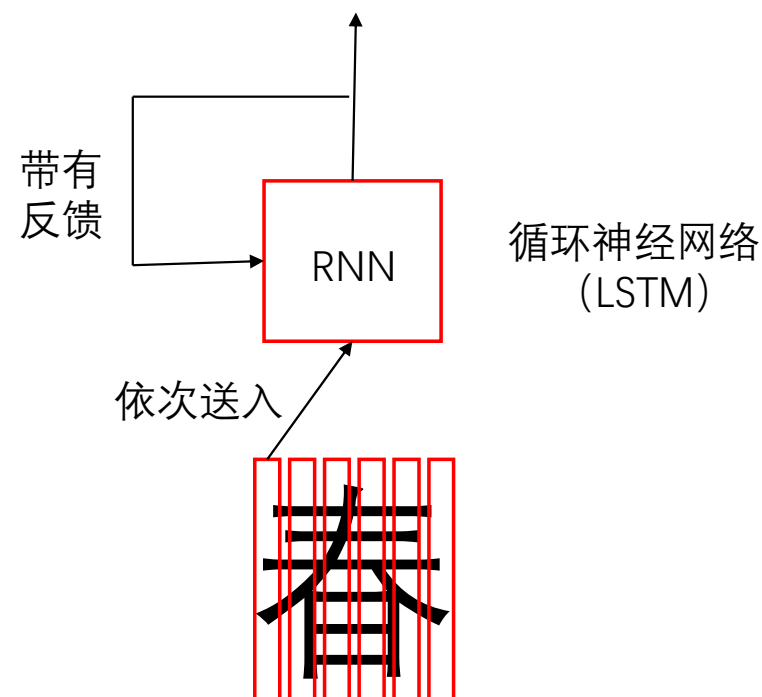
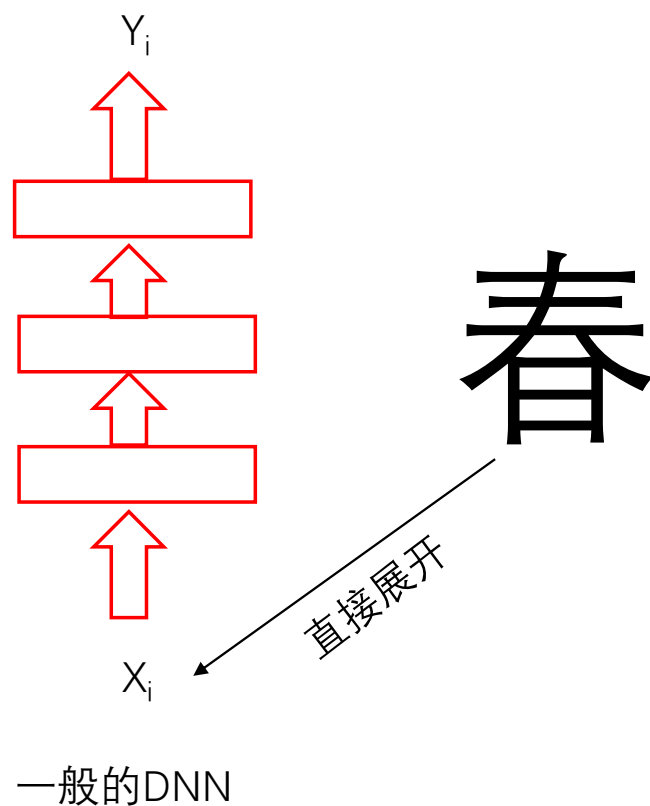
进入 tesseract 安装路径

运行 .\tesseract.exe -list-langs

检测安装结果

```
PS D:\工作相关\我设计的课程\python与人工智能课程设计\应用篇\文字扫描仪\tesseract> .\tesseract.exe --list-langs
List of available languages (4):
chi_sim
chi_sim_vert
eng
osd
```

## tesseract 模型 OCR 基本原理



代码实现:

## 1 利用仿射变换实现图像矫正

- (1) 收集源图像的四个顶点
- (2) 构建源图像顶点与目标图像顶点的关系
- (3) 计算仿射矩阵
- (4) 实现仿射变换 (图像矫正)

## 2 利用tesseract实现文字识别

```
# 扫描文件四个顶点的收集程序
# 双击进行收集
def points_collect(event,x,y,flags,param):
    dic_points = param
    if event == cv2.EVENT_LBUTTONDOWNCLK:
        if len(dic_points['ps'])>=4:
            dic_points['ps']=[]
            dic_points['ps'].append((x,y))
        else:
            dic_points['ps'].append((x,y))

    if event == cv2.EVENT_MOUSEMOVE:
        dic_points['p_move']=(x,y)
```

### 定义鼠标回调函数

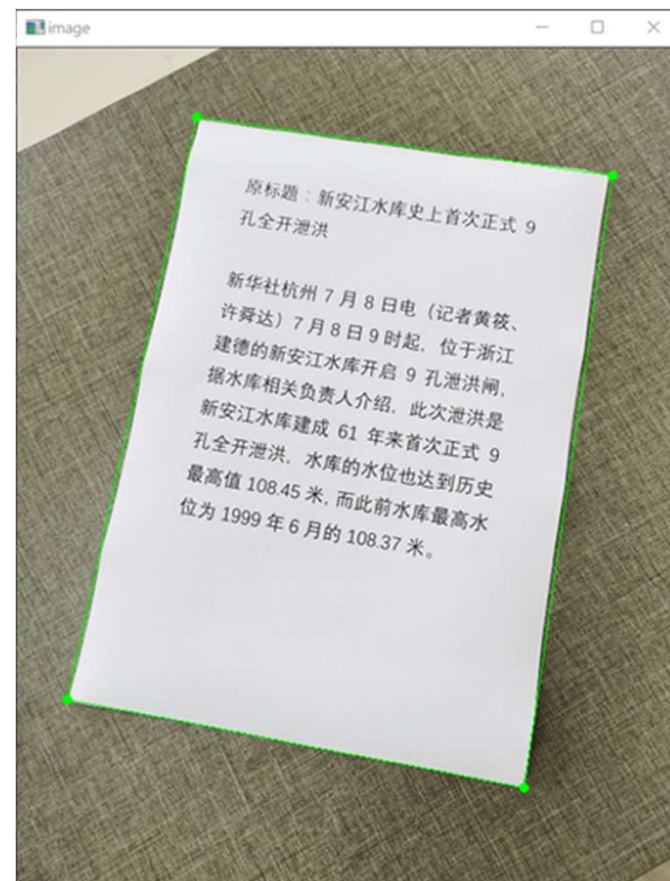
```
if __name__ == "__main__":

    # 记录坐标的字典
    dic_points = {}
    dic_points["ps"]=[]
    dic_points["p_move"]=()

    # 设置回调函数
    cv2.namedWindow('image')
    cv2.setMouseCallback('image', points_collect,param=dic_points)
```

## 选点过程绘图

```
def drawlines(img,dic_points):  
    color = (0,255,0)  
  
    # 已记录顶点复制  
    points = dic_points['ps'][:]  
    #追加移动动点  
    points.append(dic_points['p_move'])  
    if len(points)>0 and len(points)<5:  
        for i in range(len(points)-1):  
            cv2.circle(img,points[i],4,color,cv2.FILLED)  
            cv2.line(img,points[i],points[i+1],color,1)  
    elif len(points)>=5:  
        for i in range(3):  
            cv2.circle(img,points[i],4,color,cv2.FILLED)  
            cv2.line(img,points[i],points[i+1],color,1)  
  
    cv2.circle(img,points[3],4,color,cv2.FILLED)  
    cv2.line(img,points[3],points[0],color,1)
```



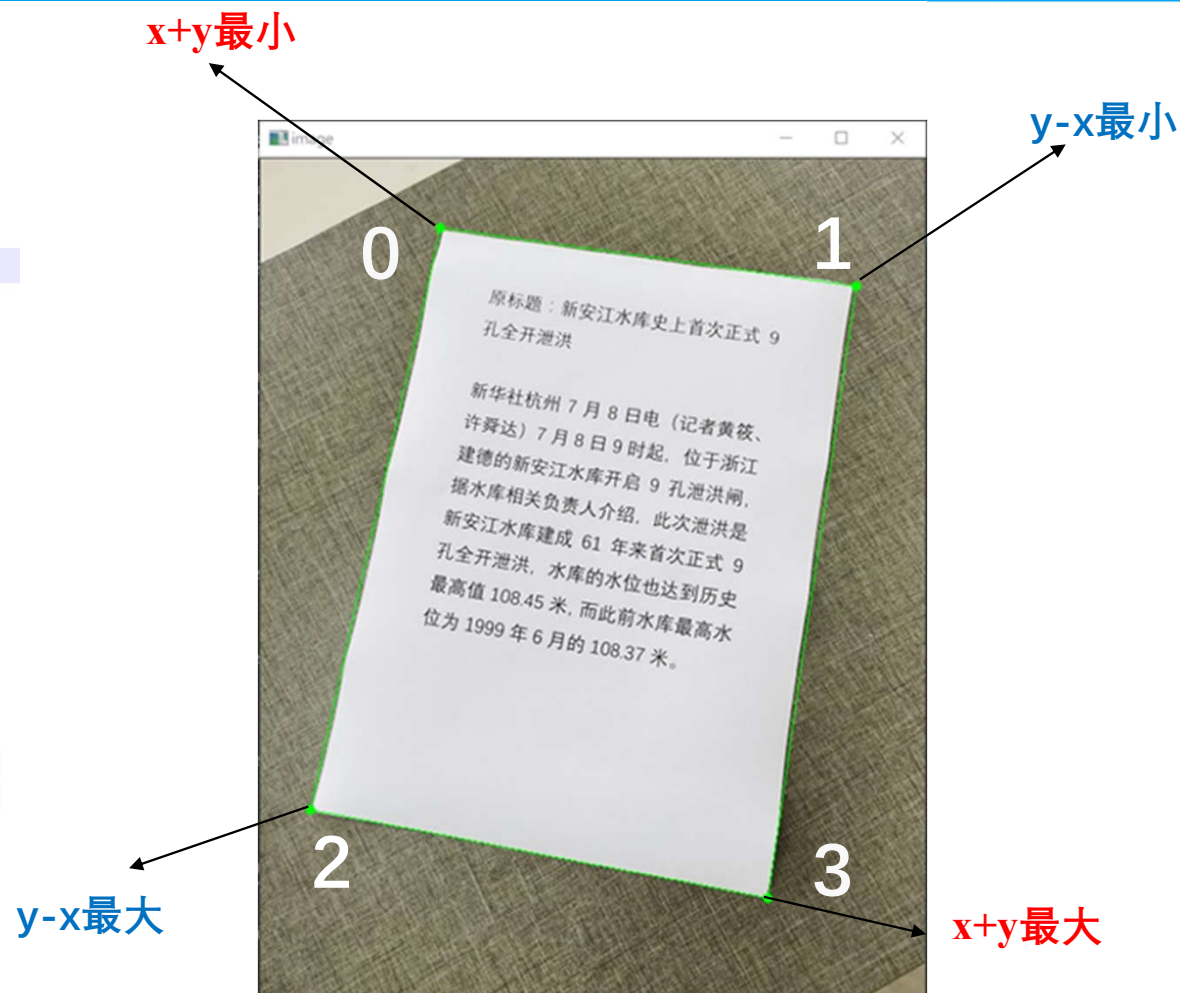


将收集到的点进行重新排序

```
# 将收集的四个顶点按照[左上, 右上, 左下, 右下]
# 顺序进行重新排列
def reorder(points):
    points = np.array(points)
    ordered_points = np.zeros([4,2])

    # 将横纵坐标相加,
    # 最小为左上角, 最大为右下角
    add = np.sum(points,axis=1)
    ordered_points[0] = points[np.argmin(add)]
    ordered_points[3] = points[np.argmax(add)]

    # 将横纵坐标相减 diff 为后减前 即 y-x
    # 最小为右上角, 最大为左下角
    diff = np.diff(points,axis=1)
    ordered_points[1] = points[np.argmin(diff)]
    ordered_points[2] = points[np.argmax(diff)]
    return ordered_points
```





## 进行仿射变化实现图像矫正

```

# 实现图像的仿射变换
# ordered_points : 需要变换的4个顶点
# size_wrapped: 变换后 图像的大小 (w,h)
def getWarp(img,ordered_points,size_wrapped):
    w,h = size_wrapped

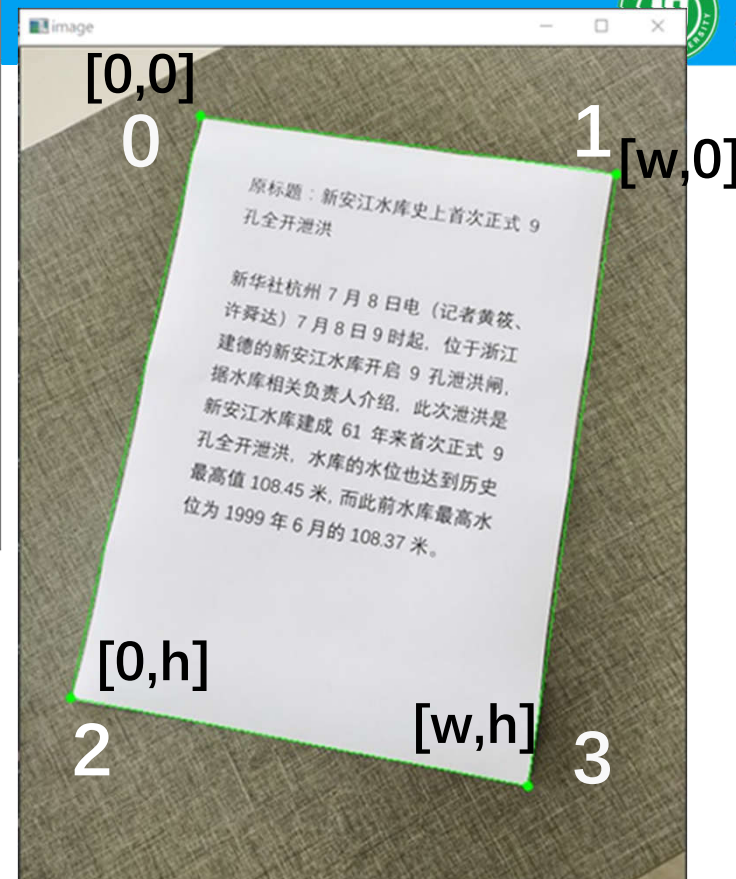
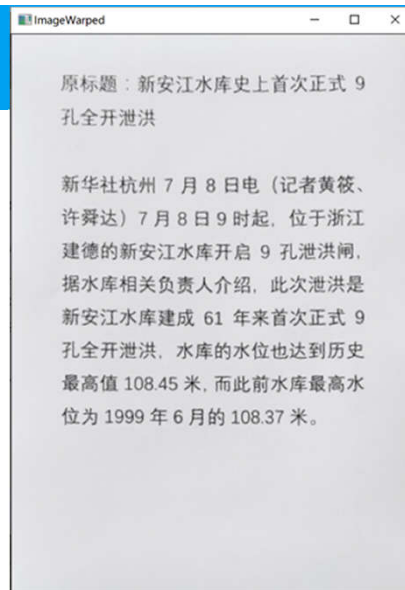
    # 源图像坐标点
    ps1 = np.float32(ordered_points)

    # 目标图像坐标点
    ps2 = np.float32([[0,0],[w,0],[0,h],[w,h]])

    # 计算仿射矩阵
    matrix = cv2.getPerspectiveTransform(ps1, ps2)

    # 进行仿射变换
    imgOutput = cv2.warpPerspective(img, matrix, (w, h))

    # 对边界进行简单裁剪
    imgCropped = imgOutput[20:imgOutput.shape[0]-20,20:imgOutput.shape[1]-20]
    imgCropped = cv2.resize(imgCropped,(w,h))
    return imgCropped
  
```



```
# 需要扫描的文件
file_scan = "paper.jpg"

# 扫描文件的大小
size_warped = (420,600)

# 定义tesseract的位置
pytesseract.pytesseract.tesseract_cmd = 'D:\\工作相关\\我设计的课和

while True:
    img = cv2.imread(file_scan)

    drawlines(img,dic_points)

    key=cv2.waitKey(10) & 0xFF
    cv2.imshow('image',img)

    if key == ord('q'):
        break

    if key == ord('w'):
        key = 0
        if len(dic_points['ps'])==4:

            # 图像仿射变换
            ordered_points = reorder(dic_points['ps'])
            img_warped = getWarp(img,ordered_points,size_warped)
            cv2.imshow("ImageWarped",img_warped)

            # 颜色转换
            img_warped_RGB = cv2.cvtColor(img_warped, cv2.COLOR_BGR2RGB)

            # 文字识别
            txt = pytesseract.image_to_string(img_warped_RGB, lang='chi_sim')
            print(txt)
```