

2012年学期总结

连高欣

导师：朱远平教授

天津师范大学计算机与信息工程学院

July 2, 2012

主要任务&问题描述

- 任务：
根据<<自然场景文本定位>>实现算法框架, 完成初步的程序
- 目的：
得到图像中文本所在的位置或者刚好包围文本的矩形区域

主要任务&问题描述

- 任务：
根据<<自然场景文本定位>>实现算法框架, 完成初步的程序
- 目的：
得到图像中文本所在的位置或者刚好包围文本的矩形区域

Figure: 目标图像



Figure: 结果图像



- ① 金字塔分解
- ② 边缘提取
- ③ 形态学运算
- ④ 先验知识限制
- ⑤ 结果合成
- ⑥ 评价

- 原因：
图像内部的字符大小可能不一致，跨度大，算法对字符大小敏感
- 算法：
分割图像，至 $1/1$, $1/2$, $1/4$, $1/9$ 四副子图，每幅子图采取相同的算法，最后做结果合成。

```
//创建个子图3
Mat div2img, div4img, div9img;
//每个子图对应的输出结果
Mat out, out2, out4, out9;
resize(img, div2img, Size(), 0.5, 0.5);
resize(img, div4img, Size(), 0.25, 0.25);
resize(img, div9img, Size(), 0.11, 0.11);
```

目的：取图像的基本边缘框架

垂直Sobel算子 $\begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}$

边缘密度-算法实现

```
/*分别声明行数*/  
const uchar* previous = img.ptr<const uchar>(i-1);  
const uchar* current = img.ptr<const uchar>(i);  
const uchar* next = img.ptr<const uchar>(i+1);  
uchar* output = dst.ptr<uchar>(i);  
... ..  
/* 算子运算sobel*/  
    y = int(previous[j-1]+2*previous[j]+previous[j  
    +1]-next[j-1]-2*next[j]-next[j+1]);  
/*将运算得到的值做范围限制并赋值给对应的行数*/  
    output[j] = saturate_cast<uchar>(abs(y));
```

$$\sum_{i=-w/2}^{i=w/2} E(x+i, y) = EI(x, y)$$

$EI(x, y)$ 表示边缘密度

$E(x+i, y)$ 表示边缘

w 表示水平窗口大小，暂定为9

边缘密度-边缘宽度

```
for (i=1; i<img.rows;i++)
{
    uchar* output = dst.ptr<uchar>(i);

    for (j=1; j<img.cols;j++)
    {
        uchar sum=0;
        for (int k=-4;k<5;k++)
            sum += img.at<uchar>(i+k, j);
        if (sum > 245) output[j] = 255;
    }
}
```

边缘提取 二值化边缘密度图像

采用双阈值的Ostu全局二值化方法，二值化边缘密度图像

- ① 水平闭运算, 连接字符笔画称为矩形

- ① 水平闭运算, 连接字符笔画称为矩形
- ② 水平开运算, 去除孤立的背景

```
/*闭运算*/  
dilate (img, dst, Mat ());  
erode (dst, dst, Mat ());  
/*开运算*/  
erode (dst, dst, Mat ());  
dilate (img, dst, Mat ());
```

$$W_{\text{dilation}} = \min(\text{height}, \text{width}/8)$$

$$W_{\text{erosion}} = \text{width}/4$$

每个连通域做 W_{dilation} 宽度的膨胀运算, W_{erosion} 宽度腐蚀运算

形态学限制2

```
findContours (imgclone,
              contours, //用来存放轮廓信息
              CV_RETR_LIST, //获取所有的轮廓信息
              CV_CHAIN_APPROX_NONE); //获取为点

for (int i = 0; i < contours.size(); i++)
{
    r[i]= boundingRect (Mat (contours [i]));
    int dilationWidth= min (r [i].height, r [i].width) ;
    int erosionWidth = r [i].width / 2 ;
}

/*画出轮廓信息*/
drawContours (dst, contours,
              -1, //draw all contours
              Scalar (255), // in black
              2);
```

目的：用常见的字符规则对结果进行限制，精细化结果

对形态学运算后的连通域进行划分，
符合规则的归为备选文本区域，等待更多 规则的筛选

$$\frac{\text{width}}{\text{height}} > r \wedge \frac{\text{whitedots}}{\text{blackdots}} > t$$

$$r = 1.0 \quad t = 2.3$$

先决条件

```
for (int i=ccxmin[m]; i<=ccxmax[m];i++) {
    for (int j=ccymin[m]; j <= ccymax[m];j++) {
        if ((int)label_image.at<uchar>(j, i) == 2
            whitedots[m] +=1;
        }
        else
            blackdots[m] +=1;
    }
}
a = float(ccxwid[m])/float(ccyhei[m])+0.5;
b = float(whitedots[m])/float(blackdots[m])+1.9
}
```

大于阈值T的像素个数符合一定比例

$$\frac{\text{Sum}(p > T)}{\text{Sum}(p)} \in (a, b)$$

$$a = 0.2 \quad b = 0.8$$

进一步对获取的备选文本区域进行分解

投影分析

目的：文本区域向x轴的投影曲线有明显的波峰和波谷，波峰对应字符的笔画采用投影曲线的波峰数量Num和Variance做为判断标准

$$\text{Variance} = \frac{1}{\text{Width} * \text{Height}^2} \sum_{i=\text{Left}}^{i<\text{Right}} (\text{Sum}(i) - \text{mean})^2$$

$$\text{mean} = \frac{1}{\text{Width}} \sum_{i=\text{Left}}^{i<\text{Right}} \text{Sum}(i)$$

Sum(i) 曲线在x轴的i点对应值 Width Height Left Right
为对应连通域文本框的宽度，高度，左侧，右侧横坐标

投影分析的Tricks

```
for (int i=0; i<=num; i++)
{
    if(h[i] > h[i-1]) {b[i] = 1; f = 1;}
    else if (h[i] == h[i-1]) { b[i]= f;}
    else { b[i] = -1; f = -1;}
}
if(b[i]+b[i+1]==0)
{
    if(b[i]>b[i+1])
        //极大值，默认为波峰
}
```


目的：合并子图结果 合并公式：

$$\frac{A(R_1 \cap R_2)}{\min(A(R_1), A(R_2))} > p \vee \frac{A(R_1 \cap R_2)}{\min(A(R_1), A(R_2))} > q \vee \frac{\max(A(R_1 \cap R_2))}{\min(A(R_1), A(R_2))} > s$$

其中 R_1 R_2 为相交的文本区域，而 p , q , s 则为预先设定的常量值

综合测评公式:

$$f = \frac{1}{1/p' + (1 - a)/r'}$$

$$a=0.5$$

P' 表示准确率, P' 越高误检率越低, 即检测到的非文本区域越少;

r' 表示召回率,

r' 越高遗漏率越低, 即没有被检测到的文本区域越少

涉及到读取xml文件, 思路 and 结果合并类似

- ① 学习OpenCV (中文版)
- ② Learning Opencv
- ③ OpenCV 2 Computer Vision Application Programming Cookbook
- ④ 数字图像处理 (Java算法描述)

- ① 注意力分散，精力太散，1个拳头想打10个鸡蛋

- ① 注意力分散，精力太散，1个拳头想打10个鸡蛋
- ② 时间分配不够

- ① 注意力分散，精力太散，1个拳头想打10个鸡蛋
- ② 时间分配不够
- ③ 理论知识学习还要加强

- ① 注意力分散，精力太散，1个拳头想打10个鸡蛋
- ② 时间分配不够
- ③ 理论知识学习还要加强
- ④ 写程序，查文档时需要更细致

假设

rows \rightarrow y cols \rightarrow x 访问坐标系 (x, y) 坐标

img.at<uchar>(y, x)	使用img
Point(x, y)	使用Point

2套坐标系，吐槽无力