一种高效检测图像中是否有三角形的算法

黎海兵 易卫东
(中国科学院研究生院信息学院,北京 100049)

摘 要 在停车场的停车位检测系统中,要识别停车位是否空闲,就必须对检测系统获取的图像中的任何可能停靠在停车场中的汽车进行识别,但由于汽车种类、形状、大小等千变万化,因此通过识别图像中是否有汽车来判断停车位是否空闲是不现实的。由于可以先在每个停车位上画上一个实心黑三角形,然后通过识别图像中是否包含有三角形来判断停车位是否空闲,如果图像中没有实心黑三角形,则表明该停车位已经被汽车所占据;否则表明停车位空闲,因此,对空闲停车位的检测就转换为检测图像中是否有三角形,这要比识别所有的汽车容易得多。而传统的 Hough 变换则不能有效地检测图像中是否包含有三角形,为了准确检测三角形,提出了一种有效的检测图像中是否有三角形的算法。该算法首先利用 Sobel 算子检测出图像的边缘信息;然后抽取一条连通的边缘,并对当前抽取出来的连通边缘所围成的区域进行填充;接着利用三角形面积与它的 3 条边的关系来判断当前被填充的区域是否是三角形。当分析完该条边缘后,再继续抽取图像中的下一条边缘进行分析,如此反复,直到图像中的所有边缘被抽取完,则停止循环;最后输出结果,如果图像中有三角形,则输出三角形的个数;如果图像中没有三角形,则输出 0。实践表明,该算法具有运算量小、运算速度快、所需内存少的优点。

关键词 边缘抽取 连通边缘 膨胀 区域填充 三角形检测
中图法分类号:TP391.41 **文献标识码**:A **文章编号**:1006-8961(2008)03-0456-05

An Effective Algorithm to Detect Triangles in Image

LI Hai-bing, YI Wei-dong
(*Information Engineering*, *CAS*, *Beijing* 100049)

Abstract In the parking space detecting system, it must recognize all kinds of cars which may park on the parking plot to decide is a parking space is free or not. But vaviable shapes of cars make it difficult for computer to make decision. After drawing a black solid triangle on a parking space and detecting whether there is a solid triangle in image, it shows that whether the parking space is free. If there is no triangle in image, it shows that the parking space is taken by a car. The problem of detecting free parking space is transformed into detecting whether there is a triangle in image. It is easier to detect whether there is a triangle in image than to detect a car. Traditional Hough transform cannot effective detect whether there are triangles in image. The algorithm presented in this paper proposes an effective algorithm to detect whether there are triangles in image. It first detects the edges of image by sobel algorithm, then extracts connected edges and fills the area enclosed by the edge extracted just now. Finnaly a judgement can be made that the area filled just now is a triangle or not by the relationship of its area and its edges. When the edge extracted just now has been analyzed, it continues to extract next edges in the image until all edges in the image are extracted. If there are triangles in the image, it outputs the number of triangles in the image otherwise, it outputs zere. Exper imental results, it show that the algorithm presented in the paper has little mathematical calculation and it is fast and with small memory cost .

Keywords edges extracting, connected edge, dilate, A filling, triangles detection

收稿日期:2006-07-25;**改回日期**:2006-09-25

第一作者简介:黎海兵(1981 ~),男。2004 年毕业于华中科技大学,现在中国科学院研究生院信息学院硕士研究生。主要研究方向为图像处理、计算机视觉。E-mail:gucashbli04@ gmail.com

1　引　言

在计算机视觉、模式识别、自动化、人工智能等领域当中,由于许多应用都必须首先检测出图像中的直线或者曲线,因而图像中的直线或曲线的检测一直是图像处理领域的研究重点。Hough 变换是当今直线或曲线检测领域应用最广泛的方法,Hough 变换虽然能较好检测出直线和一些能参数化的比较规则曲线,例如圆、椭圆等[1~5],但是对于那些很不规则并且不能简单参数化的曲线,例如不规则的三角形,Hough 变换就没办法检测出来。

本文则提出了一种基于抽取连通边缘、区域填充的算法,它能有效地检测图像中是否含有三角形。

2　抽取连通边缘

假设图 1 是一幅经过边缘检测,并且二值化后的图像,其中边缘像素的像素值为 255,背景像素的像素值为 0,图中三角形的 3 个顶点分别是 A,B,C。s 是 ΔABC 的 AB 边上的任意一点,点 s 是起始点。令 $\boldsymbol{I}$ 表示抽取 ΔABC 前的图像,令 $\boldsymbol{D}$ 为一个适当的结构元素($\boldsymbol{D}$ 是一个 3×3 大小的全 1 的矩阵),则抽取连通边缘时,首先反复循环迭代下式[1]:

$$\boldsymbol{X}_k = (\boldsymbol{X}_{k-1} \oplus \boldsymbol{D}) \cap \boldsymbol{I} \qquad (1)$$

其中,$X_0 = s$,s 是一个已知起始点,$\boldsymbol{X}_k$ 表示对该公式迭代 k 次所抽取出来的与点 s 连通的连通分量,$\boldsymbol{X}_{k-1} \oplus \boldsymbol{D}$ 表示对连通分量 $\boldsymbol{X}_{k-1}$ 进行形态学上的膨胀操作;再与原始图像相交把与点 s 连通的所有像素点(在本例中为 ΔABC 的 3 条边)都抽取出来,当 $\boldsymbol{X}_k = \boldsymbol{X}_{k-1}$ 时,循环结束,连通分量不再增长,算法收敛,此时令 $\boldsymbol{Y} = \boldsymbol{X}_k$,则抽取的 ΔABC 的 3 条边缘图像 $\boldsymbol{Y}$ 如图 2 所示;然后更新变量 $\boldsymbol{I} = \boldsymbol{I} - \boldsymbol{Y}$,将三角形的边缘从原边缘图像中擦除掉,此时的图像 $\boldsymbol{I}$ 如图 3 所示。

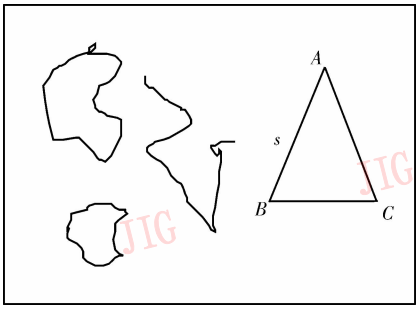


图 1　抽取 ΔABC 前的图像 $\boldsymbol{I}$

Fig. 1　Image $\boldsymbol{I}$ before ΔABC is extracted

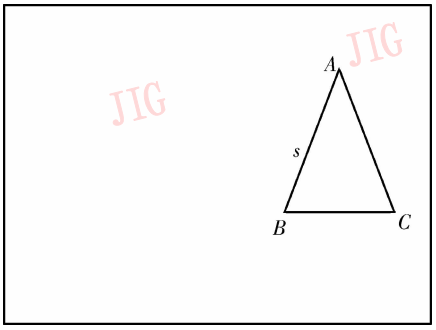


图 2　抽取的 ΔABC 的 3 条边缘图像 $\boldsymbol{Y}$

Fig. 2　Image $\boldsymbol{Y}$ containing ΔABC

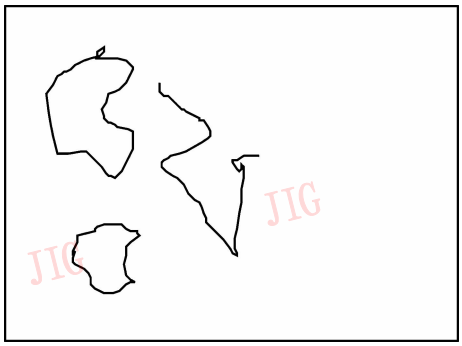


图 3　抽取 ΔABC 后的图像 $\boldsymbol{I}$

Fig. 3　Image $\boldsymbol{I}$ after ΔABC is extracted

3　对抽取出来的连通边缘进行填充

对抽取出来的连通边缘图像进行填充。其具体步骤就是,从顶端按行依次从上往下、从左到右扫描图像。

若当前扫描的像素的像素值为 0 时,则先按该像素所在的列分别向上、向下扫描,接着按该像素所在的行分别向左向右扫描,如果这 4 个扫描方向都碰到像素值为非 0 的像素,则该当前被扫描的像素在由连通边缘所围成的区域内,且当前被扫描的像素的像素值被置为 127;如果 4 个方向至少有一个方向上没有扫描到非 0 点,则当前被扫描的像素在由连通边缘所围成的区域外,且当前被扫描的像素的像素值不变,依然为 0;

若当前扫描到的像素的像素值为非 0,则不采取任何动作。例如,对图 2 中的图像 $\boldsymbol{Y}$ 进行填充后的图像如图 4 所示。

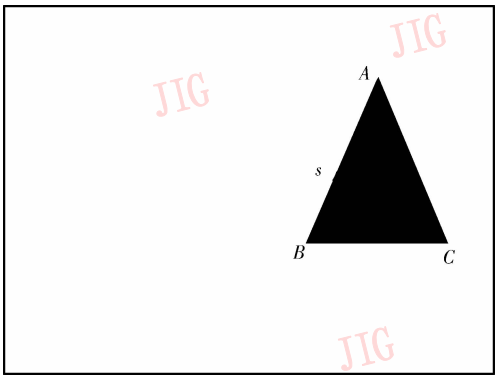


图 4 对图 2 中的图像 $\boldsymbol{Y}$ 进行填充后的新图像 $\boldsymbol{Y}$

Fig. 4 Image $\boldsymbol{Y}$ after being filled

4 判断填充后的区域是否是三角形

具体步骤是首先初始化变量 $S_{\text{region}}=0$, $N_{\text{edge}}=0$, $sum_x=0$, $sum_y=0$, $x_{\text{center}}=0$, $y_{\text{center}}=0$, $x_{\text{upper}}=0$, $y_{\text{upper}}=0$, $x_{\text{down}}=0$, $y_{\text{down}}=0$(其中 S_{region} 代表区域面积, N_{edge} 代表边缘点的个数, sum_x、sum_y 分别代表边缘点的 x、y 坐标的累加和, x_{center}、y_{center} 分别代表边缘所围成区域重心的 x、y 坐标值, x_{upper}、y_{upper} 分别代表边缘最左上像素的 x、y 坐标值, x_{down}、y_{down} 分别代表边缘最右下的像素的 x、y 坐标值)。

对填充后的图像按行从上到下、从左到右的次序进行扫描。

若扫描的当前像素值为 255,则表明当前被扫描的像素点为边缘点,然后采取如下动作:更新 N_{edge} 和面积 S_{region} 的值, $N_{\text{edge}}=N_{\text{edge}}+1$, $S_{\text{region}}=S_{\text{region}}+1$;更新 x_{down} 和 y_{down} 的值为当前像素的 x 和 y 坐标;如果该像素为扫描过程中碰到的第 1 个像素值为 255 的像素,则将该像素的 x、y 坐标分别记为 x_{upper}、y_{upper},如果该像素不是第 1 个像素值为 255 的像素,则不更改 x_{upper}、y_{upper} 的值。

若扫描的当前像素值为 127,则更新面积 S_{region} 的值, $S_{\text{region}}=S_{\text{region}}+1$。若扫描的当前像素值为 0,则不采取任何动作。

当整幅图像被扫描后,再计算区域重心的坐标值,即 $x_{\text{center}}=sum_x/N_{\text{edge}}$, $y_{\text{center}}=sum_y/N_{\text{edge}}$,此时 x_{center} 和 y_{center} 分别代表了边缘所围成区域的重心的 x 和 y 坐标;接着计算另一个顶点的 x 和 y 坐标值, $x_{\text{T}}=3\times x_{\text{center}}-x_{\text{upper}}-x_{\text{down}}$, $y_{\text{T}}=3\times y_{\text{center}}-y_{\text{upper}}-y_{\text{down}}$,此时 x_{T} 和 y_{T} 为保存第 3 个顶点的 x 和 y 坐标值;接着由这 3 点的坐标值分别计算这 3 点之间的距离,分别记为 a, b, c。

对于图 4 来说,当上述步骤执行完后,坐标 $(x_{\text{upper}},y_{\text{upper}})$、$(x_{\text{dowm}},y_{\text{dowm}})$ 和 $(x_{\text{T}},y_{\text{T}})$ 分别保存了点 A、C 和 B 的坐标值,而 a, b, c 则分别是三角形 3 条边的长度。

如果 a, b, c 中任何一个值小于某个阈值(在本实验中为 20,即 20 个 pixels),则判断该区域不是三角形。如果 a, b, c 均大于阈值,则利用以下三角形面积 S_{triangle} 和各边的关系:

$$S_{\text{triangle}}=\sqrt{(a+b+c)(a+b-c)(b+c-a)(a+c-b)}/4 \tag{2}$$

利用上述公式,即可计算 S_{triangle} 的值,并做除法运算 $T=\dfrac{S_{\text{region}}}{S_{\text{triangle}}}$。如果 T 的值在 0.90 和 1.10 之间,则判断该区域为三角形;否则判断该区域不是三角形。

5 算法的详细步骤

(1) 将输入的 RGB 图像转换成灰度图像,即利用公式 $f=0.3\times R+0.59\times G+0.11\times B$ 进行计算。如果输入的图像本身是灰度图像,则不需要执行该步骤。

(2) 利用 Sobel 算子对图像进行处理,以检测出图像的边缘[1]。

(3) 自适应设置阈值,该阈值为图像中各个像素点的梯度的平均值的 4 倍,利用该阈值先将图像转化为二值图像 $\boldsymbol{Y}$,然后细化图像。

(4) 修补二值图像 $\boldsymbol{Y}$ 中的断裂点。构造一个新图像 $\boldsymbol{I}$,图像 $\boldsymbol{I}$ 的宽、高分别是 $\boldsymbol{Y}$ 的宽、高的一半,将图像 $\boldsymbol{I}$ 的所有像素点的像素值初始化为 0;对于图像 $\boldsymbol{I}$ 的每个像素点,应考察 $\boldsymbol{Y}$ 中与之对应的 2×2 的小块,如果该 2×2 小块中的 4 个像素中至少有一个像素的像素值为非 0,则与该小块对应的图像 $\boldsymbol{I}$ 的当前像素点的像素值被置为非 0。该步骤的目的就是修补 $\boldsymbol{Y}$ 图像的边缘的断裂点,使得检测出来的边缘都是连续的,该步骤最多能修补不大于 4 个 pixels 的断裂。以下所有分析都是针对图像 $\boldsymbol{I}$。

(5) 初始化变量 $N_{\text{triangle}}=0$, N_{triangle} 为保存图像中三角形的个数;初始化新图像 $\boldsymbol{T}$, $\boldsymbol{T}$ 的大小和图像 $\boldsymbol{I}$ 一样,同时将图像 $\boldsymbol{T}$ 所有像素的像素值置为 0;按

行从上到下、从左到右的顺序扫描图像 $\boldsymbol{I}$。

(6) 扫描图像　如果扫描的当前像素的像素值为非 0,则记录该点为 X_0,并重置 $\boldsymbol{T}$,使 $\boldsymbol{T}$ 的每个像素点的像素值为 0,然后转步骤(7);如果当前扫描的像素值为 0,则继续扫描下一个像素。如果图像已经被扫描完,转步骤(11)。

(7) 将 X_0 所在的连通边缘抽取出来　依据第 2 小节介绍的原理,循环如下公式:

$$\boldsymbol{X}_k = (\boldsymbol{X}_{k-1} \oplus \boldsymbol{D}) \cap \boldsymbol{I}$$

$$\boldsymbol{T} = \boldsymbol{X}_k$$

直到当 $\boldsymbol{X}_k = \boldsymbol{X}_{k-1}$ 时,则结束循环,算法收敛。($\boldsymbol{D}$ 为一个适当大小的结构元素,是一个 3 × 3 的全 1 的矩阵)

(8) 更新原始图像 $\boldsymbol{I} = \boldsymbol{I} - \boldsymbol{T}$。将抽取出来的连通边缘从原图像中擦除掉。

(9) 对图像 $\boldsymbol{T}$ 进行填充　依据本文第 3 小节所介绍的填充算法对图像 $\boldsymbol{T}$ 进行填充。

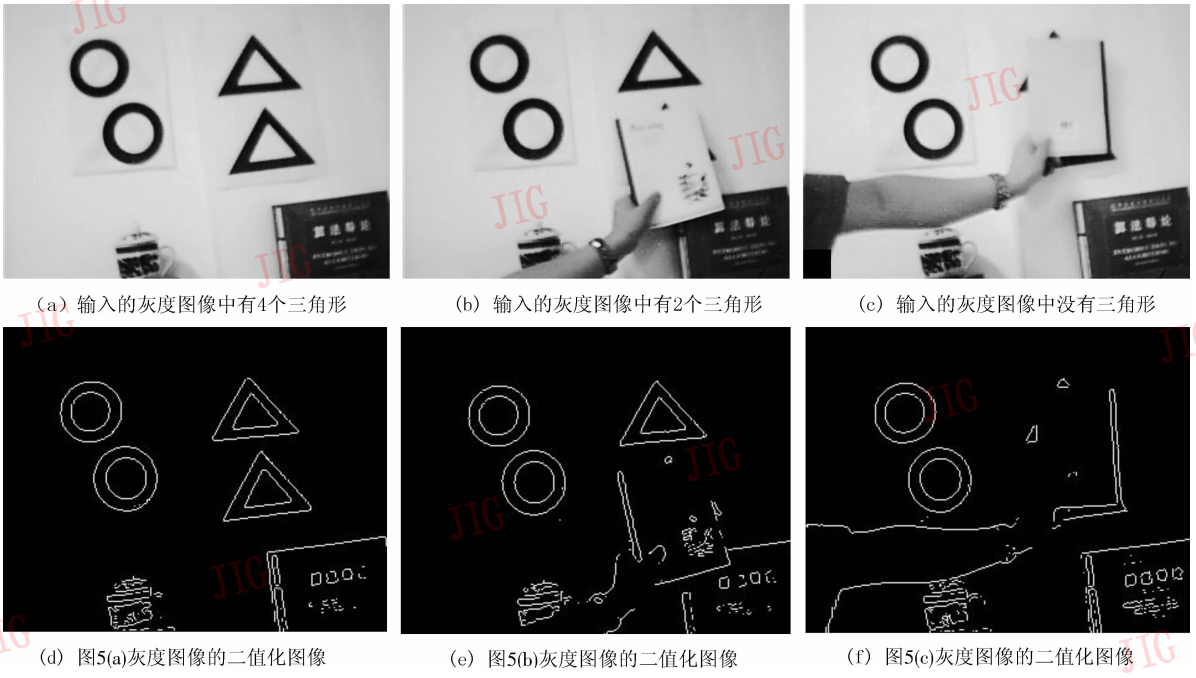
(10) 分析图像 $\boldsymbol{T}$ 中的非 0 区域是否是三角形。依据本文第 4 小节介绍的算法判断图像 $\boldsymbol{T}$ 中的区域是否是三角形。如果是三角形,则更新 N_{triangle} 的值。

转步骤(6)。

(11) 输出 N_{triangle} 的值,算法结束。

6　实验结果

为了有效验证算法的正确性,分别选取了 3 幅灰度图像(图 5(a)和图 5(b)和图 5(c))作为实验数据对本文算法进行测试。而图 5(d)、图 5(e)、图 5(f)分别是与图 5(a)、图 5(b)、图 5(c)对应的二值化图像。在 Visual C + +6.0 环境下编程实现本算法,当输入 5(a)、图 5(b)、图 5(c)3 幅图像时,程序输出结果分别是 4,2,0,这就验证了本算法的正确性。



(a) 输入的灰度图像中有4个三角形　(b) 输入的灰度图像中有2个三角形　(c) 输入的灰度图像中没有三角形

(d) 图5(a)灰度图像的二值化图像　(e) 图5(b)灰度图像的二值化图像　(f) 图5(c)灰度图像的二值化图像

图 5　具有不同三角形的灰度图像及其对应的二值化图像

Fig. 5　Gray images with different triangles and their binary images

需要说明的是,如今曲线检测领域当中关于三角形检测的论文极少,至今笔者还没有查到有关三角形检测的相关论文。

而在曲线检测领域当中,应用最广泛的莫过于 Hough 变换了,但由于标准 Hough 变换只适用于能参数化的曲线检测(比如 Hough 变换广泛用于对圆的检测),而不适用于检测三角形这种不能简单参数化的曲线,至今笔者未查到相关用 Hough 变换检测三角形的资料。因此为了对本算法与应用广泛的用于检测曲线的 Hough 变换算法进行比较,笔者修改了本算法的步骤(10),使之能检测圆的个数,用来与标准 Hough 圆检测算法做对比。

在 matlab 仿真条件下,本文以图 5(a)作为输入

图像来对比标准 Hough 变换圆检测算法和本文修改后的圆检测算法的检测效果,对比结果见表 1(在本次实验中,所有输入图像大小为 320 × 240,用标准 Hough 变换检测圆时,圆心坐标参数化的粒度为 1,半径参数化的粒度为 2)。

表 1 输入图像为图 5(a)时两种算法的对比表

Tab. 1 The comparing of the 2 algorithms when input image

	所耗内存(Kbyte)	运算时间(s)	输出圆个数
标准 Hough 变换	约 900	90.120	4
修改后本文算法	约 300	4.015	4

7 结论与分析

由表 1 可看出,与标准的 Hough 变换算法相比较,本算法在运算速度提高了 20 多倍,所需内存也只是标准 Hough 变换的 1/3。

本算法的另外一个优点就是只要适当修改算法的步骤(10),就能有效地检测图像中是否含有圆,长方形等其他一些面积与边缘有确定的和比较简单的函数关系的物体。

本算法的精度与第 4 小节中设置的阈值大小有关,即阈值越大,精度越低,阈值太小,就有误判的危险,阈值设置在 20 个 pixels 左右比较合适。因为由第 4 小节可看出,抽取出来的区域面积和区域的边的长度都是由离散的像素个数来表示,由于这种用离散量来表示诸如面积、边的长度等连续量,本身就存在量化误差,抽取出来的区域越大,这种量化误差与实际值相比就越小,反之就越大,因此如果抽取出来的区域过小,则即使该区域本身不是三角形,可能也会由于量化误差的原因而被误判为三角形;反之,有可能该区域本身是三角形,但是由于区域太小,也会因量化误差的原因而被误判为不是三角形,因此,为了消除掉量化误差可能带来的误判,要设定一个阈值,一般设置阈值在 20 个 pixels 左右就可以过滤掉量化误差的影响了,当然小于阈值的区域,即使是三角形也就被过滤掉了。

本算法的缺点就是必须要保证所要检测的对像的边缘与背景的灰度差别比较明显,这样检测出来的边缘就比较清晰连续,并且具有较少的断裂点,然后就可以利用本文算法步骤(4)来修补断裂点,使所要检测的对像边缘连续;同时必须保证不能有其他的干扰物体的边缘与所要检测的对像的边缘相连接,这样所要检测的对像的边缘就不会和其他干扰物体的边缘相连。

参考文献(References)

1 Gonzalez Rafael C. Digital Image Processing (Second edition), Ruan Qiu-qi *et al* Translation[M]. Beijing: Publishing House of Electronics Industry, 2003:475 ~ 479. [R. C. 冈萨雷斯. 数字图像处理(第二版),阮秋琦等译[M],北京:电子工业出版社,2003:475 ~ 479, 435,107.]

2 Sun Yi-nan, Liu Wei-jun, Wang Chao-yue, *et al*. A method for circle detection using modified Hough transform[J]. Computer Technology and Application, 2003,**39**(20):35 ~ 37. [孙易南,刘伟军,王超越等. 一种用于圆检测的改进 Hough 变换方法[J]. 计算机工程与应用,2003,**39**(20):35 ~ 37.]

3 Chen Yan-xin, Qi Fei-hu. A new ellipse detection method using randomized Hough transform [J]. Journal. Infrared Millim Waves, 2000,**19**(1):43 ~ 47. [陈燕新,戚飞虎. 一种新的基于随机 Hough 变换的椭圆检测方法[J]. 红外与毫米波学报,2000,**19**(1):43 ~ 47.]

4 Yu Li-na, Hu Zheng-ping, Lian Qiu-sheng. Hybird circle and ellipse fast detection using improved randomized Hough transform [J]. Journal of Electronic Measurement and Instrument, 2004,**18**(2): 92 ~ 97. [于莉娜,胡正平,练秋生. 基于改进随机 Hough 变换的混合圆/椭圆快速检测方法[J]. 电子测量与仪器学报,2004,**18**(2):92 ~ 97.]

5 Shu Zhi-lin, Qi Fei-hu. A novel algorithm for fast circle detection using randomized Hough transform [J]. Computer Engineering, 2003,**29**(6):87 ~ 89. [束志林,戚飞虎. 一种新的随机 Hough 快速圆检测算法[J]. 计算机工程,2003,**29**(6):87 ~ 89.]