

# 商空间理论的 Hough 变换直线提取

郑艺泉 谢维波

(华侨大学信息科学与工程学院, 福建 泉州 362021)

**摘要** Hough 变换以其对图像噪声的鲁棒性,一直是参数化图形检测提取的经典工具,尤其是在直线提取方面.但传统的 Hough 变换存在的一些缺点,限制了 Hough 变换的应用场合,粒度计算为这些缺点的改进提供思路.文中在全面分析传统 Hough 变换及其存在缺点的基础上,结合商空间的粒度计算理论模型,提出一种基于商空间理论的 Hough 变换直线提取算法.实验证明,该算法在算法复杂度、实用性和准确度等各方面都有一定的改善.

**关键词** Hough 变换, 粒度计算, 商空间理论, 直线提取

**中图分类号** TP 391.4; TP 301.2

**文献标识码** A

直线提取是数字图像处理、计算机视觉,以及模式识别中非常典型的任务,Hough 变换对图像噪声的鲁棒性,一直是参数化图形检测的主要工具,尤其在直线检测方面,已成为一个经典的检测工具.但传统的 Hough 变换存在以下 3 个主要的缺点<sup>[1]</sup>. (1) 计算量大,占用内存大. (2) 检测性能受参数量化间隔的制约. (3) 只能指出图像中某条直线的存在,不能给出直线段的完整描述(端点坐标和长度信息等),这限制了 Hough 变换在需要更高精确度的场合的应用. 粒度计算是一种智能计算思想,最早由 Zadeh<sup>[2,3]</sup> 提出,他主要讨论了粒度的表示和用“词计算”来解决粒度问题. 张铃等<sup>[4,5]</sup> 在 Zadeh 的理论基础上,给出了关于粒度计算求解的讨论,并提出了空间理论的粒度计算数学模型. 粒度计算为 Hough 变换中存在的缺点的解决提供了思路. 本文在粒度计算思想应用到 Hough 变换的基础上,提出了一种基于商空间理论的 Hough 变换直线提取算法. 实验证明,本算法在计算法复杂度、实用性和准确度等各方面都有一定的改善.

## 1 Hough 变换原理

Hough 变换的原理其实就是一个投票过程(图 1). 二值图中的一个点  $(x, y)$  根据公式  $\rho = x \cos \theta + y \sin \theta$  遍历  $\theta$  求出  $\rho$ , 并在累积矩阵  $(\rho, \theta)$  对应点加 1. 一个点投票后是一条正弦曲线, 根据二值图中所有点的 Hough 变换结果得到累积矩阵  $(\rho, \theta)$ , 如图 2 所示. 从图 2 可以看出,  $\rho$  在原图空间表示从原点到直线的距离,  $\theta$  在原图空间表示直线的法线与  $X$  轴的夹角. 一组特定的  $(\rho, \theta)$  在原图空间中表示的就是区域

1 中所包含的像素点, 如图 2 所示. 由此可以得出, 只要  $(\Delta \rho, \Delta \theta)$  取值适当, 一条具有一定宽度的直线就能投到一个单一的  $(\rho, \theta)$  中这也是 Hough 变换中最理想的情况.

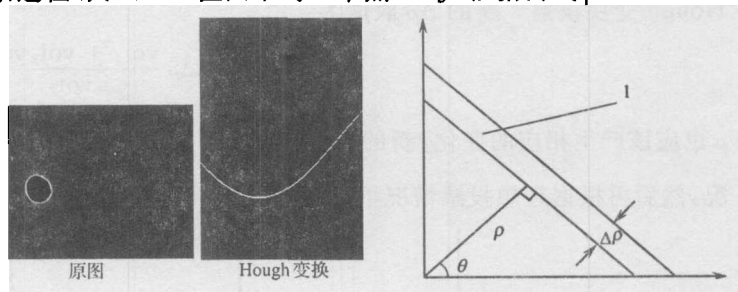


图 1 图像空间投票情况

图 2 特定的  $(\rho, \theta)$  的几何意义

**收稿日期** 2005-08-17

**作者简介** 郑艺泉(1981-),男,硕士研究生,主要从事图像图形处理与模式识别的研究;通信作者:谢维波(1966-),男,副教授,E-mail: xwblxf@hqu.edu.cn

**基金项目** 国务院侨务办公室科研基金资助项目(03QZR06)

## 2 Hough 投票分析与直线特征提取

### 2.1 Hough 投票分析

根据 Hough 变换的原理,一个点会投到参数空间中的  $\theta_{\text{range}}/\Delta\theta$  个点参数空间上,  $\theta_{\text{range}}$  为 Hough 变换中角度搜索的范围,一般为 0 到  $\pi$ ,  $\Delta\theta$  为 Hough 变换时的粒度. 这些点形成一条正弦曲线,处在相同直线上的点或者满足处在同一条直线上的条件的所有点,投票后的正弦曲线会交于参数空间的一个点. 那么,其累加值在参数空间上就会产生一个局部极大值. 当我们根据投票结果进行直线检测时,这个极大值就是我们判断是否为直线的一个很重要的标准.

根据这个原理,接下来讨论一下这个极大值的各种情况.

(1) 理想的情况下,处在同一条直线上的点都投到参数空间中的同一个点. 但是,在这种情况下还可以存在两种不同的情况. (1) 该点正好是直线的准确描述. 即特定的  $(\rho, \theta)$  和特定的  $(\Delta\rho, \Delta\theta)$  在图像空间所表示的区域是该直线的最小上界,即最佳逼近. (2) 该峰值并非是直线的准确描述. 即特定的  $(\rho, \theta)$  和特定的  $(\Delta\rho, \Delta\theta)$  在图像空间所表示的区域是该直线的上界,但不是最小上界. 还有像第 1 种情况下的最小上界,能更精确地描述直线. 在这种情况下,可以减小  $\Delta\rho$ ,然后再进行局部 Hough 变换. 在本文的算法中,对  $\Delta\rho$  采用二分法进行递归搜索,直到最后得到直线的最小上界.

(2) 非理想情况下,在这种情况下直线的投票存在跨越. 这也是整个 Hough 变换中比较不好处理的情况. 在存在跨越的投票中,又可以分为下列几种情况.

(a) 单向跨越. 同一条直线投到参数空间中两个不同的点(图 3a,高度表示投票的多少,下同),产生这种情况可能存在有 2 种原因. (1)  $\rho$  的定位不精确. 这种情况下可以平移  $\Delta\rho$  然后进行局部 Hough 变换. (2)  $\Delta\rho$  太小引起. 这种情况下可以经过合并变成理想情况下的投票结果,合并后的直线的可描述为  $[\rho_1 + (1 + \frac{\min(\text{vot}_1 + \text{vot}_2)}{\max(\text{vot}_1, \text{vot}_2)})\Delta\rho, \theta]$  (其中  $\text{vot}_1, \text{vot}_2$  为第 1 个值和第 2 个值的投票数,下同). 这样合并后再根据理想情况下的不同情况进行处理,即可得到直线的精确描述. 这种情况合并后,有可能变成下面的双向跨越的情况,再根据双向跨越的情况再进行处理. 也可以直接增大  $\Delta\rho$ ,然后再进行局部 Hough 变换,增大的幅值可以为原来  $\Delta\rho$  的两倍. 这样能保证局部 Hough 变换后会变成理想情况下的投票分布. 经过处理后均可得到理想的投票情况,然后再根据理想投票情况下处理方式,得到最终的直线描述.

(b) 双向跨越. 同一条直线投到参数空间中 3 个不同的点(图 3b). 在这种情况下,和单向跨越一样也可能存在有 2 种原因. (1)  $\rho$  的定位不精确. 这种情况下可以平移  $\rho$ ,然后进行局部 Hough 变换. (2)  $\Delta\rho$  太小引起. 它无法使直线上的点都落入到同一个区间中需要适当的增大  $\Delta\rho$ ,然后再进行 Hough 变换投票. 新的  $\Delta\rho$  取值为

$$\Delta\rho = \Delta\rho \frac{\text{vot}_1 + \text{vot}_2 \text{vot}_3}{\text{vot}_2},$$

$\rho$  也应该产生相应的变化,新的  $\rho$  取值为  $(\rho_2 - \Delta\rho \frac{(\text{vot}_1 - \text{vot}_3)}{\text{vot}_2})$ . 经过处理后,均可得到理想的投票情况,然后再根据理想投票情况下处理方式,得到最终的直线描述.

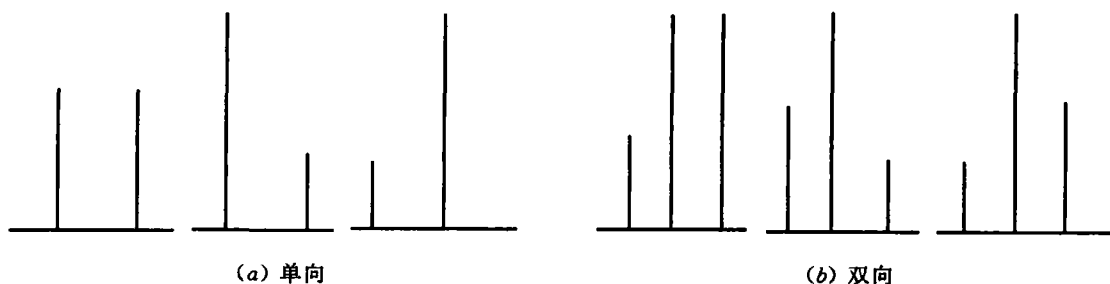


图 3 Hough 变换中的跨越情况

该分析中充分利用论域(Hough 参数空间)的粒度层次,在不同的粒度层次之间进行跳变,不断的进行粒度的合成与分解,最终收敛于优化解. 这也是本文算法的精髓所在.

## 2.2 直线特征提取

根据 Hough 投票分析可以得出,最后直线的峰值都会变成理想情况下,具有直线最小上界的峰值,根据这些理想情况下的峰值,就能得出直线的精确描述.

## 3 算法

### 3.1 具体步骤

(1) 选取一个比较大的  $\Delta\rho$  和固定的  $\Delta\theta$ ,并根据这两个值进行全局 Hough 变换.

(2) 根据 Hough 变换后的累积矩阵,按特征得出一些直线. 这时候所得到的直线只是粗粒度描述的直线,并不是直线的精确描述.

(3) 从选取出的每一条直线的描述,根据 Hough 投票分析得出新的  $\Delta\rho, \Delta\theta$  保持不变. 然后,进行局部的 Hough 变换,变换后根据步骤(2)又可以得出一些直线.

(4) 如果得出的直线比没有局部 Hough 变换前的直线更精确,进行下一条直线的局部 Hough 变换;否则,返回到步骤(3),直到所有的直线都得到精确的描述.

### 3.2 算法分析

(1) 时间复杂度. 根据前面的分析可以得出,算法的计算量主要集中在粗粒度的全局 Hough 变换中. 因为后面的局部 Hough 变换中,  $\rho, \theta$  的搜索范围非常小,其计算量可略去不计. 在本算法粗粒度的搜索过程中,选取比较大的  $\Delta\rho$  和固定的  $\Delta\theta$ (具体  $\Delta\rho$  和  $\Delta\theta$  的确定方法在后面的讨论中将会详细说明). 相比于传统的 Hough 变换中,  $\Delta\theta$  均取为 1(弧度化成角度,下同),本文的算法中考虑到在直线的宽度及离散化后存在的有限精度等原因,  $\Delta\theta$  可以取大于 1 的值. 假设原始的二值图中有  $N$  个像素点,那么初粒度的全局 Hough 变换中所需要的计算量为  $N \cdot \frac{\theta_{range}}{\Delta\theta}$  ( $\theta_{range}$  为 Hough 变换中  $\theta$  变化的范围,下同). 由此,可以得到计算量的大小与  $\Delta\theta$  的大小成反比关系. 本算法中  $\Delta\theta$  取大于 1 时,效率就能提高将近  $(\Delta\theta - 1)100\%$ .

(2) 空间复杂度. 空间复杂度和时间复杂度类似,也只是主要集中在粗的全局 Hough 变换中. 传统的 Hough 变换中,存储 Hough 变换结果的矩阵大小为  $(\rho_{range} \cdot \theta_{range})$  (其中  $\rho_{range}$  为 Hough 变换中  $\rho$  变化的范围,下同). 在本算法中,存储 Hough 变换结果的矩阵大小仅为  $\frac{\rho_{range} \cdot \theta_{range}}{\Delta\rho \cdot \Delta\theta}$ ,而且粗粒度下的  $\rho$  比较大,使这个矩阵的大小迅速变小.

(3) 实验结果. 在本算法中,采用不同的粒度层次进行搜索,在粗粒度和细粒间交替搜索,最后可以得到直线的精确描述. 而且,在最后一遍搜索时的  $\Delta\rho$  正好就是直线的宽度,如表 1 所示. 表中的标志位

表 1 算法在 Matlab 6.5.1 上仿真的结果

$\rho$	$\theta$	投票值	$\Delta\rho$	$\Delta\theta$	标志位
35.000 0	45.000 0	49.000 0	1.000 0	3.000 0	1.000 0
54.000 0	45.000 0	225.000 0	3.000 0	3.000 0	1.000 0
73.000 0	45.000 0	608.000 0	5.000 0	3.000 0	1.000 0
106.000 0	45.000 0	229.000 0	2.000 0	3.000 0	1.000 0

表示最后一次 Hough 变换的峰值的情况,1 为单个峰值,2 为单向跨越的峰值,3 为双向跨越的峰值. 图 4 中有 4 条粗细不同的直线,检测的结果得到不同的  $\Delta\rho$  正好就是直线的宽度. 这也是本算法改进的一个地方. 因为这样不仅可以检测有没有直线,还得到直线的密度信息这是传统的 Hough 变换所无法做到的.

### 3.3 讨论

(1) 本文的算法在直线的特征提取是,根据单一的峰值特征来提取直线. 所以,原图中如果有不在任何直线上的点,都会对检测结果造成一定的影响. 因此,可对原图进行中值滤波去除这些点,或者可以设定一定的阈值,先去除粗粒度下检测出来的投票值比较小的直线,然后再进行细粒度的搜索. 这样可以去除一些虚假直线,降低算法的复杂度.

(2) 本文的算法中,在 *Hough* 变换过程中采用较大的  $\Delta\rho$  和固定的  $\Delta\theta$ . 在实验过程中发现,  $\Delta\theta$  取偶数,离散化后  $\theta$  的值在一次遍历中只能取奇数或偶数;而如果取奇数,在一次遍历中不仅可以取到奇数,也可以取到偶数,这有利于提高精度. 所以,  $\Delta\theta$  值必须取奇数. 在  $\Delta\rho$  的初始化问题中,取大一点的  $\Delta\rho$  有可能会把离得比较近的直线投到参数空间的一个上去,取小一点的  $\Delta\rho$  又会加大算法的空间复杂度,而且还会使粗一点的直线投到参数空间中的多个点上去,导致即使在粗粒度下也不能峰值集中.  $\Delta\rho$  初始化的原则应该是,在保证不同直线上的点投到不同的参数空间中的点的基础上,尽量增大  $\Delta\rho$ .

(3) 在本文的算法中,只考虑到  $\Delta\rho$  粒度层次对检测结果的影响,而暂时忽略了  $\Delta\theta$  粒度层次对检测结果的影响. 这样做可以降低算法的复杂度,但同时也限制了检测出来的直线在角度精度方面的提高. 上面所提到的固定  $\Delta\theta$  的取值问题,也是由于没有考虑到  $\Delta\theta$  的粒度层次引起的,这将是下一步研究工作的重点.

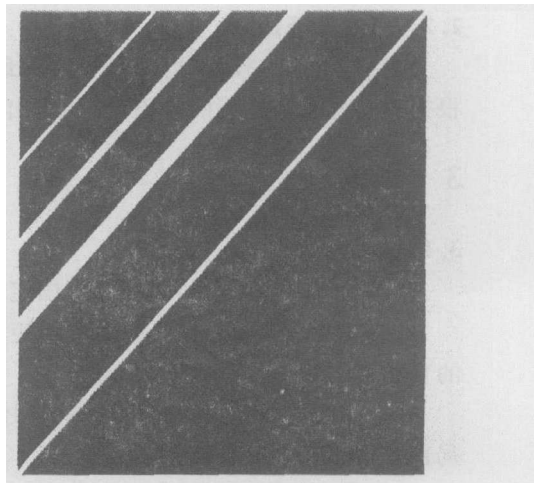


图 4 实例原图

#### 4 结束语

面对比较复杂的,难以在一次求解就得到最优解的问题时,人们通常采用逐步尝试的方法达到有限合理的目标,也就是取得所谓足够满意的解. 人类就是采用这种概略地、由粗到细、不断求精的多粒度分析法,避免了计算复杂度高的困难,使得原来看似难以解决的问题迎刃而解. 笔者将其应用到 *Hough* 变换直线提取中,取得了较好的时间复杂度、空间复杂度和算法精确度.

#### 参 考 文 献

- 1 Heikki Kälviäinen, Petri Hirvonen, Lei Xu, et al. Probabilistic and non-probabilistic hough transforms[J]. Overview and Comparisons. 1995, 13(4):239~252
- 2 Zadeh L A. Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic[J]. Fuzzy Sets and Systems, 1997, 19(1):117~127
- 3 Zadeh L A. Some reflection on soft computing, granular computing and their roles in the conception design and utilization of information/intelligent systems[J]. Soft Computing, 1998, 2(1):23~25
- 4 张 铃, 张 钺. 模糊商空间理论(模糊粒度计算方法)[J]. 软件学报, 2003, 14(4):770~776
- 5 张 钺, 张 铃. 问题求解理论与应用[M]. 北京:清华大学出版社, 1990. 1~126

### Hough Transformation Based on Quotient Space Theory for Line Extraction

Zheng Yiquan Xie Weibo

(College of Information Science and Engineering, Huaqiao University, 362021, Quanzhou, China)

**Abstract** Hough Transformation (HT) is famous for its robustness to image noise, and becomes a classical tool for parameterized shape extraction, especially for line extraction. However, some defects in conventional HT limit its application. But, granular computing provides the ways for the improvement of these defects. With complete analysis of conventional HT and its defects, the paper combined HT and quotient space theory, which is a granular computing theory model, presented a line extraction computing method for HT based on quotient space theory. Experiments show that this algorithm made an improvement in algorithm complexity, usefulness and accuracy in practice.

**Keywords** Hough transformation, granular computing, quotient space theory, line extraction