

375-380

(10)

利用梯度方向信息的随机 Hough 变换*

陈燕新 戚飞虎

(上海交通大学计算机科学与工程系, 上海, 200030)

TP391.41

摘要 针对随机 Hough 变换(RHT)在处理复杂图像时随机采样造成的大量无效累积提出一种改进的用于直线检测的 RHT,较好地解决了无效累积问题,并使改进后的算法具有计算速度更快、占用内存更少以及检测性能更好等优点。

关键词 随机 Hough 变换(RHT)、直线检测、梯度方向信息。

RHT

引言

Hough 变换^[1]是检测直线的有效方法,在计算机视觉中常用它来提取边缘直线特征。但传统的 Hough 变换存在几种较大缺陷:(1) 计算量大,每个边缘点映射成参数空间的一条曲线,是一到多的映射;(2) 占用内存大;(3) 提取的参数受参数空间的量化间隔制约。为克服上述缺陷,Xu 等^[2~3]提出了随机 Hough 变换(RHT),采用多到一的映射,避免了传统 Hough 变换一到多映射的庞大计算量;采用动态链表结构进行参数累积,降低了内存需求,使得 RHT 具有参数空间无限大、参数精度任意高等优点。尽管 RHT 只对多到一映射所得到的参数分配单元进行累积,但在处理复杂图像时,由于随机采样仍引入大量无效单元,造成了大量的无效累积。本文提出一种利用梯度方向信息的随机 Hough 变换(RHT_GD)用于直线检测,它不仅利用梯度方向信息来决定是否对采样到的两点进行参数计算和累积,而且还将梯度方向信息用于直线确认,从而较好地解决了无效累积问题,并使改进后的算法具有计算速度更快、占用内存更少以及检测性能更好等优点。

边缘梯度方向包含了丰富的边缘形状信息,一些学者已将其用于 Hough 变换,如 Gupta 等^[4]提出了一种新的 Hough 变换,它利用边缘梯度方向和直线段的连贯性进行物体轮廓的多边形近似。考虑到计算获得的梯度方向精度不高,他们预先指定一个 θ_{ω} ,对于梯度方向为 θ 的边缘点,不仅仅对 θ 参数单元进行累积,而是对 $(\theta \pm \theta_{\omega})$ 范围内的参数单元都进行累积,这就导致对于近似成直线段的曲线上的边缘点,它们的梯度方向最大可以相差 $2\theta_{\omega}$,利用这一性质实现了物体轮廓的多边形近似。由此可以推知:对于有些学者直接将边缘梯度方向作为待检测直线的一个参数进行直线检测,其结果将不会理想。本文不直接将边缘梯度方向作为待检测直线的一个参数,而是利用它来解决无效累积问题和进行直线确认。另外,在利用基于模板或方向导数的边缘检测算子(如 Sobel 算子、Robert 算子、Haralik 算子^[5]、Canny 算子^[6]等)进行边缘检测时,在获得边缘强度的同时可以获得边缘梯度方向,因此获

* 国防预研基金(编号 96J2.4.2)资助项目
稿件收到日期 1998-01-05,修改稿收到日期 1998-02-26

得边缘梯度方向所需的计算量很小. 本文提出的方法对计算获得的梯度方向精度要求不高, 因此不会对边缘检测提出额外要求.

本文首先简要介绍了随机 Hough 变换(RHT), 然后给出了一种利用梯度方向信息的随机 Hough 变换(RHT_GD), 最后是实验结果和分析.

1 Hough 变换原理及算法

1.1 随机 Hough 变换(RHT)

一条直线可以用参数方程表示为

$$\rho = x \cos \theta + y \sin \theta, \quad (1)$$

其中 (x, y) 为图像空间坐标, (ρ, θ) 为参数空间坐标, θ 是 X 轴到直线的法线的逆时针转角, ρ 是坐标原点到直线的代数距离. 对于任意给定的两个边缘点, 由上述参数方程可以求得由这两点所确定直线的参数.

RHT 的基本思想是随机选取(这是随机采样)两个边缘点唯一地确定参数空间的一个点, 这是多到一的映射, 避免了传统 Hough 变换一到多映射的庞大计算量. 在实现累积时, 采用动态链表结构, 只对多到一映射所得到的参数分配单元进行累积, 从而降低了内存需求, 并使得 RHT 具有参数空间无限大、参数精度任意高等优点.

算法描述如下: 设 D 为图像空间的边缘点集, P 为参数空间的参数单元集, 它是一个动态链表结构. 对于参数空间两点 p 和 p' , 用 $\|p - p'\| \leq \delta$ 表示 $|\rho - \rho'| \leq \delta \rho$, $|\theta - \theta'| \leq \delta \theta$. 设参数单元的计数值为 score, 当参数单元 p_c 的 score 大于等于指定的阈值 N_c (它是一个很小的数, 例如 2、3) 时, 该参数 p_c (本文用相同符号表示参数单元和它对应的参数) 对应的直线成为候选直线, 判断图像空间中落到该候选直线上的点数 M_{p_c} , 若其大于直线所允许的最小长度 M_{\min} , 则确认该候选直线为真实直线, 记下该直线并继续进行下一直线的检测. 当检测到的直线数达到规定数目时, 结束检测过程; 若事先不知道直线数, 则可规定检测一条直线过程中所允许随机采样的最大点数 K_{\max} , 当采样点大于 K_{\max} , 而参数单元点集 P 中仍没有参数单元的 score 大于等于 N_c , 则认为 Hough 变换已不能检测到更多的直线而结束.

算法的具体步骤如下: (1) 构造边缘点集 D , 初始化参数单元集 $P = \text{NULL}$, $k = 0$; (2) 从 D 中随机选取两点 d_1, d_2 ; (3) 计算两点所确定的直线参数 p ; (4) 在 P 中找一个 p_c , 满足 $p = p_c$ (或 $\|p - p_c\| \leq \delta$, δ 是容许误差), 若找到了则转步骤(6), 否则转步骤(5); (5) 将 p 插入 P , 令其 score 为 1, 转步骤(7); (6) 将 p_c 的 score 加 1, 若小于指定的阈值 N_c , 则转步骤(7), 否则转步骤(8); (7) $k = k + 2$, 若 $k > K_{\max}$, 则结束, 否则转步骤(2); (8) p_c 为候选直线的参数, 若该参数对应直线上的点数 $M_{p_c} > M_{\min}$, 则转步骤(9), 否则为虚假直线, 从 P 中去除 p_c , 转步骤(2); (9) 检测到参数为 p_c 的真实直线, 判断已检测到的直线数是否已达到规定的数目, 若是, 则结束, 否则将落在参数 p_c 对应直线上的点从 D 中去掉, 重置 $P = \text{NULL}$, $k = 0$, 转步骤(2).

1.2 利用梯度方向信息的随机 Hough 变换(RHT_GD)

尽管 RHT 只对多到一映射所得到的参数分配单元进行累积, 但在处理复杂图像时, 由于随机采样仍引入大量无效单元, 造成了大量的无效累积. 图 1(a) 是一个黑三角形, (b) 是

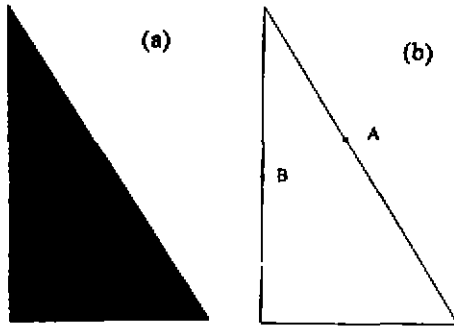


图 1 黑三角形及其边缘

Fig. 1 The black triangle image and its edge image

它的边缘图, A、B 两点显然不在同一直线上, 但由于随机采样, RHT 可能会选取这两点, 导致需要计算经过这两点的直线参数 p , 并搜索参数单元集 P (它是一个动态链表), 如果在 p 中没有 P , 还需在 P 中插入新的参数单元, 这就使得链表规模变大, 这一方面意味着占用了更多的内存, 另一方面使得后继映射过程中搜索链表的工作量变大. 如果我们利用梯度方向信息, 靠它就完全可以确定这两点不在同一直线上, 因而可以省去计算参数 p 和搜索参数单元集 P 的工作, 另外也不会引入新的参数单元, 这既减少了占用的内存,

又使链表保持在一个较小的规模, 使得后继映射过程中搜索链表的工作量较小. 这就是本文提出 RHT_GD 的出发点.

在 RHT 和 RHT_GD 中, 存储单元根据映射得到的参数进行分配, 其参数可以在整个参数空间取值, 对于参数方程(1)表示的直线, 其参数空间为 $\theta \in [0, \pi)$, $\rho \in [-R, R]$, 其中 R 为图像平面的半径.

设边缘在边缘点 n 的梯度方向为 α_n , 法线方向为 β_n , α_n 在 $[0, 2\pi)$ 之间取值, β_n 在 $[0, \pi)$ 之间取值, 它们之间的关系是

$$\beta_n = \begin{cases} \alpha_n, & \alpha_n < \pi \\ \alpha_n - \pi, & \alpha_n \geq \pi \end{cases} \quad (2)$$

在理想情况下, 直线边缘上各点的梯度方向 α_n 都是一样的, 各点的法线方向 β_n 和直线的法线方向 θ (参见式(1))也是一样的. 考虑到计算获得的梯度方向精度不高, 我们预先指定一个 θ_{tol} , 对于随机采样获得的两点 d_1 和 d_2 , 首先判断边缘在这两点处的梯度方向 α_1 和 α_2 是否满足 $|\alpha_1 - \alpha_2| \leq \theta_{tol}$, 若不满足, 则认为这两点不在同一直线上, 完成采样点计数后立即重新采样, 而不进行后继处理; 若满足, 则认为这两点可能在同一直线上, 计算过这两点直线的法线方向 θ , 结合边缘在这两点处的法线方向 β_1 和 β_2 判断 $|\theta - \beta_1| \leq \theta_{tol}$ 和 $|\theta - \beta_2| \leq \theta_{tol}$ 是否成立, 若不成立, 则认为这两点不在同一直线上, 完成采样点计数后立即重新采样, 而不进行后继处理; 若成立, 则认为这两点可能在同一直线上, 进行累积等后继处理, 这样就较好地解决了无效累积问题.

尽管我们允许计算获得的梯度方向可以有一定的误差 (即预先指定的 θ_{tol}), 在实际图像中的一条边缘直线上, 仍可能会存在一些点, 这些点由于噪声等原因使得计算获得的梯度方向误差超过允许值, 而不能对该直线对应的参数进行累积, 但由 RHT_GD 的随机采样、累积以及直线确认 (后面将论述) 机理可知: 只需该边缘直线上大多数点能对该直线对应的参数进行累积即可, 这意味着 RHT_GD 仍具有较强的抗噪性能, 同时对计算获得的梯度方向精度要求并不高, 不会对边缘检测提出额外的要求.

对于一条有较多边缘点的假想直线 (不是真实直线), 用 RHT 检测时, 对该假想直线参

数的累积会达到指定阈值 N_t , 而需对其进行确认. 而用 RHT_GD 检测时, 由于利用了梯度方向信息, 使得能对该假想直线参数进行累积的点数远小于假想直线上的边缘点数, 从而参数累积达到指定阈值 N_t 的可能性大大减小, 这就带来了如下好处: (1) 大大减小了产生虚假直线的可能性, 从而提高了算法的检测性能; (2) 减少了确认直线的次数, 使得算法计算量更小, 计算速度更快.

在直线确认时, RHT 仅仅利用落到候选直线上的点数, 这给检测较短的直线段带来了困难, 因为在检测直线时为了不将具有较多边缘点的假想直线(不是真实直线)确认为真实直线, 直线所允许的最小长度 M_{\min} 取值不能过小, 而 M_{\min} 取值较大, 会导致将较短的直线段判为虚假直线. 而 RHT_GD 一方面在累积时利用了梯度方向信息, 使得当对具有较多边缘点的假想直线参数进行累积时, 能达到指定阈值 N_t 的可能性大大减小; 另一方面, 在确认直线时再次利用梯度方向信息: 设候选直线的法线方向为 θ , 落在候选直线上点的法线方向为 β_i , 则只有当 $|\theta - \beta_i| \leq \theta_{\max}$ 成立时, 才将落到候选直线点的计数加 1, 这样就大大提高了算法的检测性能.

算法的具体步骤如下: (1) 构造边缘点集 D , 初始化参数单元集 $P = \text{NULL}$, $k = 0$; (2) 从 D 中随机选取两点 d_1, d_2 ; (3) 判断边缘在这两点处的梯度方向 α_1, α_2 是否满足 $|\alpha_1 - \alpha_2| \leq \theta_{\max}$, 若满足, 则转步骤(4), 否则转步骤(9); (4) 计算过这两点直线的法线方向 θ , 结合边缘在这两点处的法线方向 β_1 和 β_2 判断 $|\theta - \beta_1| \leq \theta_{\max}$ 和 $|\theta - \beta_2| \leq \theta_{\max}$ 是否成立, 若成立, 则转步骤(5), 否则转步骤(9); (5) 计算过这两点直线的代数距离 ρ 和步骤(4)计算得到的 θ 构成过这两点直线的参数 p ; (6) 在 P 中找一个 p_c , 满足 $p = p_c$ (或者 $\|p - p_c\| \leq \delta$, δ 是容许误差), 若找到了则转步骤(8), 否则转步骤(7); (7) 将 p 插入 P , 令其 score 为 1 转步骤(9); (8) 将 p_c 的 score 加 1, 若小于指定阈值 N_t , 转步骤(9), 否则转步骤(10); (9) $k = k + 2$, 若 $k > K_{\max}$, 则结束, 否则转步骤(2); (10) p_c 为候选直线的参数, 判断该候选直线是否为真实直线, 若是, 则转步骤(11), 否则为虚假直线, 从 P 中去除 p_c , 转步骤(2); (11) 检测到参数为 p_c 的真实直线, 判断已检测到的直线数是否已达到规定的数目, 若是, 则结束, 否则将落在参数 p_c 对应直线上的点从 D 去掉, 重置 $P = \text{NULL}$, $k = 0$, 转步骤(2).

2 实验结果与分析

本文用检测一条直线所需的平均时间 T 、检测一条直线所占用的平均参数单元个数 CN_{av} 以及检测一条直线所占用的最大参数单元个数 CN_{\max} 来表征算法的性能, 它们定义如下: 设算法检测到 LN 条直线, 所花时间为 T_i , 检测第 i 条直线所占用的最大参数单元数为 CN_i , 则

$$\left. \begin{aligned} T &= \frac{T_i}{LN}, \\ CN_{av} &= \frac{\sum_{i=1}^{LN} CN_i}{LN}, \\ CN_{\max} &= \max\{CN_1, CN_2, \dots, CN_{LN}\}. \end{aligned} \right\} \quad (3)$$

其中 T 表征算法的计算速度, CN_{av} 和 CN_{\max} 表征算法占用内存的大小.

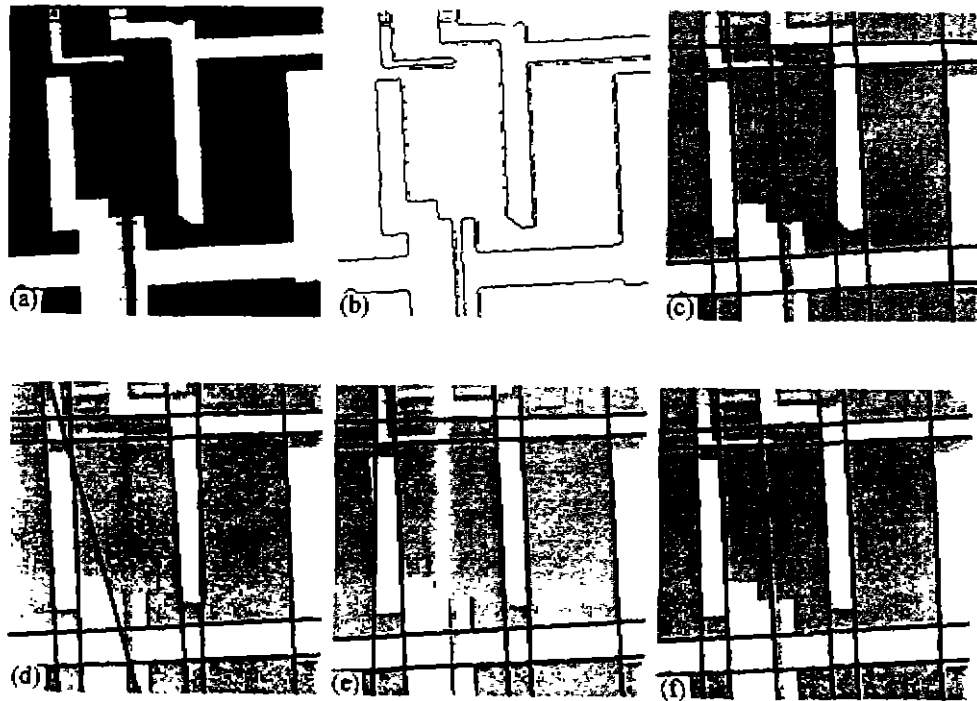


图 2 测试图像及检测结果

(a) 测试图像, (b) 边缘图像, (c) RHT 检测结果 1, (d) RHT 检测结果 2,

(e) RHT 检测结果 3, (f) RHT_GD 的检测结果

Fig. 2 The test images and its detection results

(a) the test image, (b) the edge image, (c) the first result of RHT, (d) the second result of RHT,

(e) the third result of RHT, (f) the result of RHT_GD

实验用 120MHz586 微机进行, 编程语言是 Visual C++5.0. 测试图像如图 2(a) 所示, 其大小为 256×256 . 用计算量较小的 Sobel 算子提取边缘并细化后, 结果如图 2(b) 所示, 图像边缘点数为 2510. 实验中参数取值尽可能使算法检测到 10 条长直线 (直线的长度以它上面的点数来衡量), 这是通过规定所需检测的直线数和 M_{\min} , 并将 K_{\min} 取得较大实现的. 算法主要参数取值如下: 所需检测的直线数取 10, N 取 3, θ_{\max} 取 15, K_{\min} 取 10 倍的边缘点数, 在参数单元累积时 $\delta\rho$ 和 $\delta\theta$ 分别取 0.5 和 $\frac{\pi}{360}$, 在确认真实直线时, $\delta\rho$ 和 $\delta\theta$ 分别取 2.5 和 0. 用 RHT 检测直线, 当 M_{\min} 在合理范围内取值时, 出现如图 2(c) 所示的结果次数最多, 该结果是我们期望的结果. 但是当 M_{\min} 取值相对较小时, RHT 有时会检测到虚假直线, 如图 2(d) 所示 ($M_{\min}=130$), 在 M_{\min} 取 130 时, 在 50 次运行结果中, 只有 36 次 RHT 得到期望的结果. 如果提高 M_{\min} 的取值, 又会导致 RHT 检测不到 10 条直线, 如图 2(e) 所示 ($M_{\min}=140$), 在 M_{\min} 取 140 时, 在 50 次运行结果中, 有 41 次 RHT 得到期望的结果, 另外 9 次不仅没有检测到 10 条直线, 而且检测时间大大增加. 我们尝试了很多的 M_{\min} 取值, 对于每一取值, 在 50 次运行结果中, 得到期望结果的次数均小于 44. 用 RHT_GD 检测直线, 当 M_{\min} 取值 130

时,在50次运行结果中,有48次RHT_GD得到期望的结果,如图2(f)所示,这说明RHT_GD的检测性能要优于RHT.由于RHT_GD也采用随机采样,因此每次运行结果并不完全相同,本文在RHT和RHT_GD的 M_{\min} 都取130的情况下,将50次运行结果的 T 、 CN_{\min} 以及 CN_{\max} 的值进行平均,作为 T 、 CN_{\min} 以及 CN_{\max} 的值,如表1.由表1可见RHT_GD比RHT计算速度快得多,占用内存小得多.

由上述实验可以看出:本文提出的利用梯度方向信息的随机Hough变换(RHT_GD)与RHT相比,具有计算速度更快、占用内存更少以及检测性能更好等优点,完全可用于实时处理系统.

表1 算法的性能指标

Table 1 The performance indexes of algorithms

考察的性能指标	T	CN_{\min}	CN_{\max}
RHT性能指标	0.0536	663	1547
RHT_GD的性能指标	0.0137	77	230
RHT和RHT_GD的性能指标之比	3.9	8.6	6.7

REFERENCES

- 1 Illingworth J, et al. *Comput. Vision Graphics Image Process*, 1988, 44(1): 87~116
- 2 XU L, et al. *Pattern Recognition Letters*, 1990, 11(5): 331~338
- 3 XU L, et al. *Comput. Vision Graphics Image Process: Image Understanding*, 1993, 57(2): 131~154
- 4 Gupta A K, et al. *Pattern Recognition*, 1993, 26(7): 1069~1086
- 5 Haralick R. *IEEE Trans. PAMI*, 1984, 6(1): 58~68
- 6 Canny J. *IEEE Trans. PAMI*, 1986, 8(6): 679~698

A RANDOMIZED HOUGH TRANSFORM USING GRADIENT DIRECTION INFORMATION*

CHEN Yan-Xin QI Fei-Hu

(Department of Computer Science and Engineering, Shanghai Jiaotong University, Shanghai 200030, China)

Abstract To solve the problem of a large amount of useless accumulations yielded by random sampling by means of the randomized Hough transform (RHT) in handling complex images, an improved RHT used to detect lines was proposed. It uses gradient direction information, not only can well solve the problem of useless accumulations, but also has higher speed, smaller storage and better detection performance.

Key words randomized Hough transform (RHT), line detection, gradient direction information.

* The project supported by the National Defence Preliminary Research Foundation
Received 1998-01-05, revised 1998-02-26