基于改进的概率 Hough 变换的直线检测优化算法

刁燕*，吴晨柯，罗华，吴必蛟
四川大学制造科学与工程学院，四川 成都 610065

摘要 针对概率 Hough 变换耗费大量内存以及直线端点搜索容易受到网状聚集点干扰的缺陷，提出一种基于概率的局部 Hough 变换优化算法。将边界分为有序和无序两类，前者通过随机抽取采样点并结合其相邻点进行直线搜索，后者采用在随机抽取点周围建立感兴趣区域并进行局部 Hough 变换，检测到直线后进行全局搜索并实时修正直线斜率，对因网状聚集点产生的错误直线采用间隔计数和限制总间隔长度的方法进行排除。使用 500 张图片进行实验验证，算法耗时均低于概率 Hough 变换耗时的 1/3，且对网状聚集边界点的直线错检具有较高抵抗性，检测结果比概率 Hough 变换直线检测更加准确，内存消耗减少超过 90%以上。

关键词 光学检测；直线检测；概率 Hough 变换；机器视觉
中图分类号 TP391.9 **文献标识码** A **doi**：10.3788/AOS201838.0815016

Line Detection Optimization Algorithm Based on Improved Probabilistic Hough Transform

Diao Yan*, Wu Chenke, Luo Hua, Wu Bijiao
School of Manufacturing Science and Engineering, Sichuan University, Chengdu, Sichuan 610065, *China*

Abstract Aiming at the drawbacks that for probabilistic Hough transform it consumes a lot of memory and the line endpoints searching is vulnerable to interference from reticular aggregation points, a probability-based local Hough transform optimization algorithm is proposed. The edge is classified into two categories: sortable and non-sortable. For the former, sampling points are randomly picked and combined with their adjacent points for straight line searching. For the latter, the local probabilistic Hough transformation is carried out in the region of interest which is established around the random edge point, the endpoints are searched after the line is detected and the slope is fixed in real time. The error lines resulting from mesh aggregation points are excluded by the interval counting and the total interval length limit method. Experiments were carried out by 500 images. The proposed algorithm consumes less than 1/3 of the time of the probabilistic Hough transform, and it is highly resistant to line mis-detection of meshed aggregation edge points. Line detection is more accurate and memory consumption is reduced by more than 90%, compared with the probabilistic Hough transform.

Key words optical inspection; line detection; probabilistic Hough transform; machine vision
OCIS codes 120.4630; 100.3008; 150.1135; 150.0155

1 引 言

物体特征提取中直线提取在机器视觉领域非常关键，目前对直线的检测方法大多基于 Hough 提出的 Hough 变换[1]算法，1972 年 Duda 等[2]改进了此方法，将极坐标的参数方程引入 Hough 变换，解决了在斜率无穷大情况下的参数转换，Hough 变换实现了从图像空间到参数空间的映射关系，噪声图像稳定性和稳健性良好，且易于实现，因而被应用于定位与识别[3]，农业作物方位检测[4]，零件特征检测[5]等多个领域，张国英等[6]利用 Hough 变换提高检测高分辨率遥感影像中线性目标的精度。但标准 Hough 变换算法在实现过程中存在计算复杂度和空间复杂度高、运算量大、直线结果精确度不高等问题，国内外研究人员提出了诸多基于标准 Hough 变换的改进算法。Xu 等[7]提出了随机 Hough 变换(RHT)，克服了标准 Hough 变换中的消耗内存大、精度低、速度慢等缺点。刘通等[8]利用概率 Hough 变换解决了空间碎片漫反射激光测距回波数据信噪比低，难以快速高效地提取有效数据点的问题，鄢然

收稿日期：2018-03-20；**修回日期**：2018-04-07；**录用日期**：2018-05-08
* E-mail：diaoyan2@163.com

等[9]利用概率 Hough 变换建立了一种编布花边的实时识别算法，巩学美等[10]改进概率 Hough 变换进行了遥感图像的道路识别，Chutatape 等[11]则提出一种将 Hough 空间转换为一维空间的改进算法，大大减少了计算的复杂度和需要的内存。张振杰等[12]提出基于一维的 Hough 变换算法，为边缘分组、直线编组和直线精确处理 3 个过程加速并优化了 Hough 变换的结果。段汝娇等[13]提出基于像素点聚类的方法来加速 Hough 变换算法。

现有的算法相对于概率 Hough 变换，在计算量、内存使用减小以及计算加速上，已经有良好改善，但在直线端点查找中还有不足之处，并且霍夫空间的内存减小还有优化空间，在此基础上，本文针对这两种情况提出一种基于概率和感兴趣区域的改进概率 Hough 变换算法。

2　概率 Hough 变换

概率 Hough 变换（Progressive Probabilistic Hough Transform，PPHT）由 Matas 等[14]提出，是在 Hough 变换基础上，对边界点进行随机采样，在直线到原点距离 ρ 和直线角度 $\theta(0\sim\pi)$ 形成的累加器空间进行投票，达到阈值即认为有直线，该方法不像 Hough 变换对全部边界点进行检索，仅部分抽样，大大减少了计算时间和计算量。

2.1　概率 Hough 变换的缺陷

1）消耗内存大，假设通过一个像素点的直线簇的采样角度间隔为 $\Delta\theta$，ρ 的分辨率为 $\Delta\rho$，对于一张 $M\times N$（pixel×pixel）的图像，保证图像中每个像素点能映射到 Hough 空间所需要的储存单元数量 n_c 可表示为

$$n_c=\frac{(1+N+\sqrt{M^2+N^2})\cdot\pi}{\Delta\rho\Delta\theta}。\tag{1}$$

2）计算量大，设一张图像中有边界点数量 m，直线采样的渐进角度 $\Delta\theta$ 取 $\pi/180$，那么，ρ 最多需要计算的次数为 $180m$，当 m 较大而 $\Delta\theta$ 较小时，计算量将变得非常巨大。

2.2　Hough 变换算法实现中的缺陷

很多文章都讨论了 Hough 变换算法的缺陷，王竞雪等[15]对 Hough 变换在算法实现中的缺陷做了一定的分析，也提出了一定的规避方法。但仍有一些问题应归于用计算机语言实现 Hough 变换过程中代码造成的缺陷，主要集中于：

1）邻近的边界点或噪点会使 Hough 空间的投票产生虚假峰值，造成直线错误检测；

2）直线过连接，有间断的多段直线被视为一条直线或将错误点纳入直线，这是由 Hough 空间的投票方式所导致的，仅能从峰值获知该位置和该角度有直线，无法获知有多少段直线；

3）当有散点聚集或网状边界出现时，直线的允许间断 G_{ap} 设定不为 0，直线端点搜索将受到严重干扰，会将间断线识别为直线，如图 1 所示，$G_{ap}=6$，树叶处产生大量的网状边界，无法先验设定某些参数去掉这种干扰状况下，很容易被识别为有间断的伪直线，如图 1(c)所示。

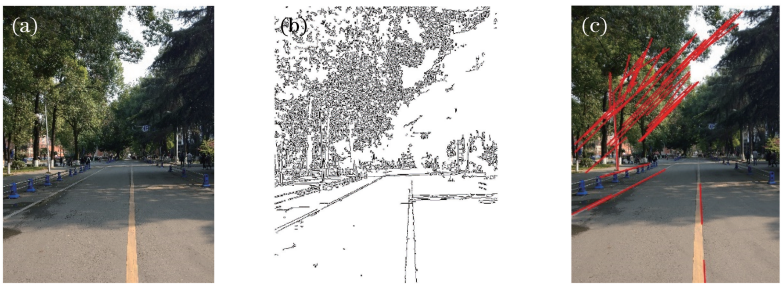

图 1　树叶产生网状边界影响直线结果。(a)原图；(b) Canny 边界；(c)直线结果

Fig. 1　Leaves produce a mesh edge that affects the line results. (a) Original image; (b) Canny edge; (c) line detected result

3　算法描述

3.1　参数确定

2.2 节中 3 个问题都和 Hough 变换的参数及代码实现方法有关。虚假峰值和参数设定、图像噪点过滤有关。直线的过连接问题一是由于 $\Delta\rho$ 和 $\Delta\theta$ 的设定精度不当，使靠近的点也会在 Hough 空间的同一个位置投票而被错误地识别是一条直线；二是在端点搜索过程中，没能删除偏移像素点的干扰。

$\Delta\rho$ 和 $\Delta\theta$ 直接影响直线检测的精度和检测速度，滕今朝等[16]对 $\Delta\theta$ 的设定精度做了测试，提出分式查表法以提高运算速度，但该文章中对 $\Delta\theta$ 的设定并不妥当，对一张分辨率为 800 pixel×600 pixel 的图像，设定的 $\Delta\theta$ 并非越小越好，该图像分辨率下

能产生的两直线（对角线处）夹角最小为 arctan（600/800）－arctan（599/800）＝0.0008 rad，即0.046°，如果 $\Delta\theta$ 小于这个值，$\Delta\theta$ 的可取值精度将大于直线的斜率精度，这会增加 Hough 变换的计算量和 Hough 空间需要的内存。而 $\Delta\rho$ 则关乎检测到直线距离原点的距离精度，设定得越小越可以避免邻近同向的直线被识别为一条直线，像素平面上，平行直线最小距离也是 1，故设置为 1 或略小于 1 即可。直线检测需要设定直线最小长度为 $L_{\min}$，直线方向上容许间距为 G_{ap}，如果 G_{ap} 设定过大，且在求取边界时没过滤细小边界聚合，则会使端点查找极为不准，将聚集散点识别为直线。对于上述问题，本文提出解决思路：

1）有序的边界（单像素）坐标点，采用间隔采样，提取若干个相邻采样点进行直线判定，如果检测到直线，则延伸直线检查端点。

2）无序的边界点，随机抽取一个边界点，建立感兴趣区域 R，在区域内进行较低精度的概率 Hough 检测，当检测到直线时，延伸搜索端点，并实时修正直线斜率。加入直线线宽 d，解决直线延伸过程中略有偏移的像素点漏检问题，并且对 G_{ap} 不为 0 的情况，增加设定每条直线最大总间断数 $G_{\max}$（一条直线允许总的间隔出现次数），以及总间隔容量 Q（一条直线允许的总间隔像素数），出现连续间断时，如果有不足 5 pixel 的小线段，依然认为是间断区间，凡是单次间隔超过 G_{ap}、间隔次数达到 $G_{\max}$、Q 消耗为 0 都认为直线端点检索结束。

3.2　有序边界点情况下的算法实现

此类一般是物体外轮廓或机器人的轨迹曲线，设排序的边界点是点集 D，假定最短直线的投票阈值 $T=3$（threshold），即如果有 4 个连续采样点形成直线，则认为此处出现直线，操作示意如图 2 所示，具体实现步骤如下：

1）从 D 起始位置开始，按照每间隔 $g=10$ 采样 1 个点，放置到点集 S，减少点的选择空间（图 2 中圆点即采样点）；

2）从 S 中随机抽取一点 a_i，获取 a_i 在 S 中位置序号，提取序号前 3 和后 3 个点组成相邻点点集 $G=\{a_{i-3},a_{i-2},a_{i-1},a_i,a_{i+1},a_{i+2},a_{i+3}\}$，$a_i$ 前后如果不足 3 个点，则在其反方向多取点，只要点数大于 1 就进行步骤 3），仅有 1 个点，重新执行步骤 1）；

3）对 G 中的点，相邻位置点两两求斜率，则会有 S_{1-6} 斜率数据，建立累加器 A_C（范围为[0，π]，分辨精度为 $\Delta\theta$），将 S_{1-6} 在 A_C 对应位置进行投票，并记录进行了运算的点，如果 G 中某个点 a_j（$a_j\in G$）在 A_C 中得分大于阈值 T，则认为过该点有直线，提取该点，跳至步骤 5），否则跳至步骤 4）；

4）A_C 投票低于阈值 T，从 S 中去除在步骤 3）中抽取的 7 个点，更新 S，继续步骤 2）；

5）A_C 投票达到阈值 T，提取产生得分达到阈值 T 的点 a_j 和 A_C 达到阈值的角度 θ_j，从 a_j 的序号开始向前向后搜索 S 中的采样点，考察这些点是否在以 a_j 点为中心以 θ_j 为方向的直线上，并将这些点归入这条直线，当点偏离直线，说明后面的点不在该直线，执行步骤 6）；

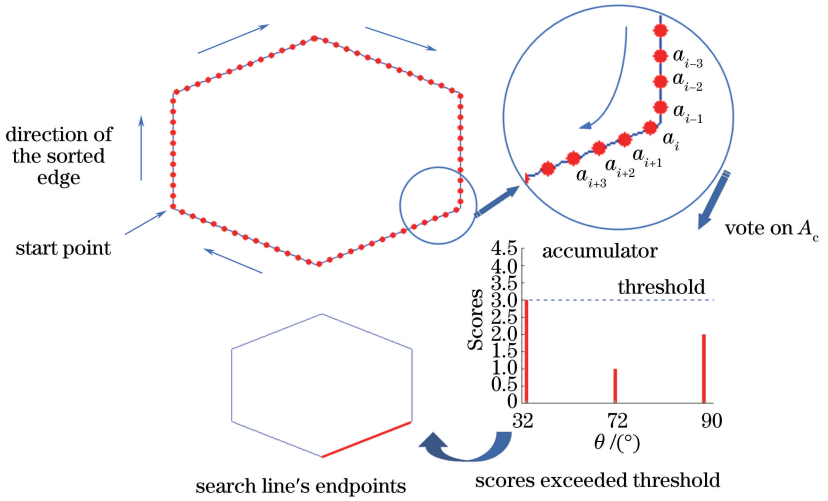

图 2　有序边界点情况下的直线检测步骤

Fig. 2　Steps of line detection in sorted edge points

6) 检测从步骤5)中获得的直线端点序号,从端点位置开始,沿直线方向前后搜索 D 中的点进行端点精确匹配,将直线上的点更新为直线端点,当点开始偏离直线时,直线端点搜索即停止,执行步骤7);

7) 检查直线长度,如果该直线达到预设直线长度 $L_{\min}$,提取出直线。继续步骤2),直到 S 中的点全部耗尽,或达到需要的直线数量。

以上步骤,无需建立需要巨大内存的Hough空间,而是建立一个角度投票的累加器,通过对边界点采样,大大减少了检索空间,并且直线端点的检索基于序号,直线的漏检和错检及过连接问题将不会出现。

3.3 无序边界点情况下的算法实现

如果1个点在1条直线上,那么该点周围投票出现直线的概率很大,设无序边界点集 D,从中随机抽取1点 d_i($d_i \in D$),以 d_i 为中心建立矩形感兴趣区域 R,尺寸为 $l \times l$(pixel×pixel),且设定 $l = L_{\min}/\sqrt{2}$,即检测的最短直线长度 $L_{\min}$ 刚好为矩形 R 对角线长度,如果有直线产生,一定通过 R 的边界,由于 R 的范围较小,在此处进行局部Hough变换时,需要的累加器空间大大减少,并且 R 的四边和图像边框分别平行,在 R 中检测到直线的角度相对于整个图像不变,可直接延伸搜索端点,操作示意如图3所示,该方法实现步骤如下:

1) 从 D 中随机抽取一点 d_i 建立感兴趣区域 R,抽取 R 范围内的边界点;

2) 检查 R 内的边界点数量是否超过累加器的跳出阈值,如果没有,此处属于干扰区域,无法产生直线,从原图中移除这片区域,继续步骤1),否则进行步骤3);

3) 建立 R 范围的Hough空间累加器 A_H(图3中Hough accumulator),设定 $\Delta\theta = \pi/360$,$\Delta\rho = 1$,该累加器可以重复使用,节约内存;

4) 对 R 中的点随机抽取进行概率Hough变换并在 A_H 中进行投票,记录已经计算过的点,当有投票超过阈值时,提取该点 p,并跳到步骤5),否则继续步骤1),直到耗尽,如果没有,则认为此处无直线,为干扰区域,移除 R 内的点,清空 A_H,执行步骤1);

5) 从步骤4)中获取的 p 开始,按照直线方向,搜索直线端点,建立连续计数器 W,判断直线的斜率 θ_j 是否属于[45°,135°],若属于该范围,则沿 x 方向每步+1 pixel进行端点搜索,否则沿 y 方向每步+1 pixel搜索,以线宽 $d = 1$,移除被检测到的点及4邻域的边界点,每检测到边界点则 W 增加1,当 W 达到20重新计算该条直线的斜率并将 W 清零,计算新纳入直线的端点的边界点和起始点 p 的斜率,更新该直线斜率继续搜索,当遇到间断时执行步骤6);

6) 若发生间断,G_{ap} 统计增加1,总间断次数 $G_{\max}$ 减小1,如果 W 小于5,总容量 $Q = Q - W$(即表示该次间断之前的连续长度不足以认为是可靠直线长度,为间断直线),间断区间完成检索,从 Q 中减去该间断区间总长度,如果 G_{ap} 达到设定阈值或 $G_{\max}$ 减小到0或 Q 消耗到0,都认为直线产生过多间隔,该处网状散点干扰概率大,端点搜索应该停,执行步骤7),否则继续执行步骤5),搜索端点;

7) 更新直线端点,检查直线长度,如果大于设定 $L_{\min}$,提取出直线,执行步骤8);

8) 从原图删除 R 范围中的点,清空 A_H,继续执行步骤1),直到原图点耗尽或直线数量达到需要值。

算法总流程图如图4所示,橙色代表无序模式的流程,蓝色代表有序模式的流程。

3.4 计算量和存储空间

从内存的使用情况看,当概率Hough变换 θ 的精度为 $\Delta\theta$,ρ 的精度为 $\Delta\rho$,图像尺寸为 $M \times N$,由(1)式,累加器需要 $(1+N+\sqrt{M^2+N^2})\cdot\pi/(\Delta\rho\Delta\theta)$ 个存储单元。在有序边界情况下,无需建立巨大的Hough空间,只需要建立一个一维动态大小累加器(最大 $\pi/\Delta\theta$ 个储存单元),而无序模式下,累加器需要的储存单元数量 n_t 可表示为

$$n_t = \frac{(1+l+\sqrt{2}l)\cdot 360}{\Delta\rho}, \tag{2}$$

仅需一个可以重复使用的储存单元数为 n_t 的累加器,大大减少了对内存资源的消耗,两种状态下内存占用比 r_a 设为

$$r_a = \frac{n_t}{n_c} = \frac{\dfrac{360\cdot(1+l+\sqrt{2}l)}{\Delta\rho}}{\dfrac{\pi(1+N+\sqrt{M^2+N^2})}{\Delta\rho\Delta\theta}} = \frac{360\cdot\Delta\theta(1+\sqrt{2}l+l)}{\pi(1+N+\sqrt{M^2+N^2})}。 \tag{3}$$

若某张图像尺寸为800 pixel×600 pixel,设检测出的最短直线长度 $L_{\min} = 100$ pixel,则取 $M = 800$,$N = 600$,$\Delta\theta = \pi/1800$,$\Delta\rho = 1$,$l = 70.71$,可得到 $r_a = 0.0214$,内存占用减少97%。

计算速度上,本算法和概率Hough变换一样,都是采取随机抽取方式,但在局部范围使用了较低

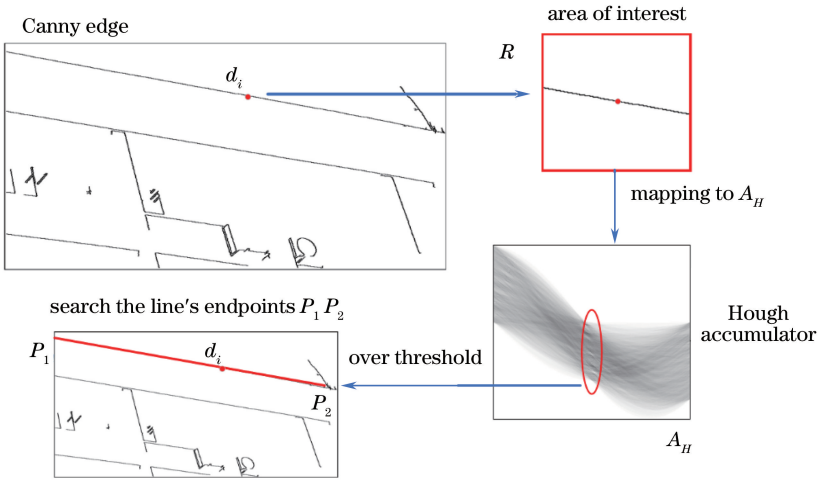

图 3　无序边界点情况下的直线检测步骤

Fig. 3　Steps of line detection in unsorted edge points

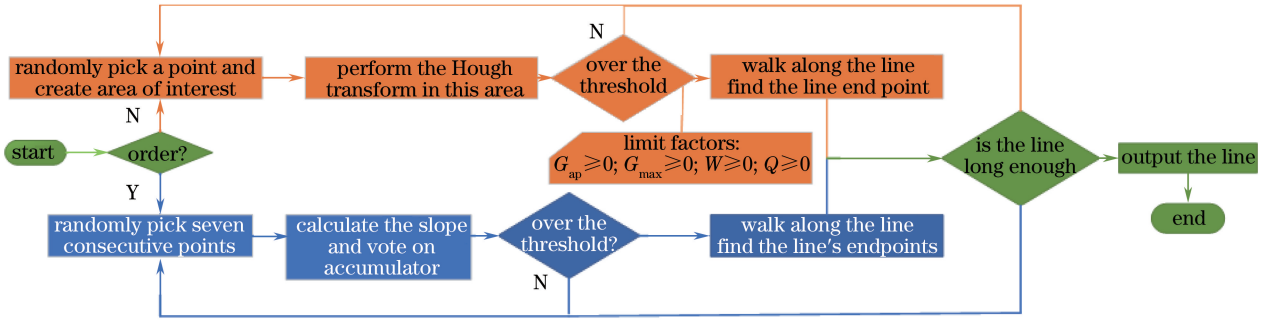

图 4　算法运行流程图

Fig. 4　Algorithm operation flow chart

精度较小范围的霍夫空间投票方法，并在进行局部霍夫变换前检测此处是否可能出现直线，不可能有直线的不进行霍夫空间投票，大大减少了运行时间。但由于基于概率，无法用表达式给出准确的加速倍数，故进行实验测试，采用时间倍数关系数 T_{ratio} 来对比时间耗费，计算方式为

$$T_{ratio}=\frac{T_{PPHT}}{T_{ours}},\tag{4}$$

式中：T_{PPHT} 表示概率 Hough 变换检测耗费时间，T_{ours} 表示本文算法耗费时间，单位 ms。$T_{ratio}>1$ 表示本文算法耗费时间更短，反之则是概率 Hough 变换耗费时间更短。

4　实验与分析

实验分为速度测试和效果验证。处理器：Intel i7-6700HQ；程序环境：Qt Creator 5.10；编译器：Desktop Qt 5.10.0 MSVC2017 64 bit；对比程序实现：Opencv 3.4 的 HoughlinesP 函数。

4.1　检索速度实验

概率霍夫变化和本文算法都是基于概率，由于不同图像的参数较多，无法用数学方式准确给出时间关系，故进行大量的图像实验验证。实验采用尺寸分别为 3264 pixel × 2448 pixel、2560 pixel × 1920 pixel、2048 pixel × 1536 pixel 一共 500 张拍摄外景图像（随机顺序）进行两种算法耗时对比。并且设置相同的参数：最短直线长度 $L_{min}=100$ pixel，单次间隔允许 $G_{ap}=6$ pixel，霍夫空间量化参数 $\Delta\rho=1$ pixel，$\Delta\theta=\pi/1800$，间断总次数允许 $G_{max}=6$，间断总长度允许 $Q=15$。

实验中 500 组实验的时间对比和两种算法时间关系如图 5(a)所示，红色为概率 Hough 变换的 500 次实验耗费时间状况，蓝色为本文算法（标示为 ours）的时间耗费，可见本文算法在同样参数同一图像下耗费时间明显较少，图 5(b)为每组实验所用时间的倍数关系，最高 $T_{ratio}=10.987$，最低 $T_{ratio}=3.59587$，500 次实验平均时间倍数 $T_{ratio}=6.21$（图 5(b)中虚线），

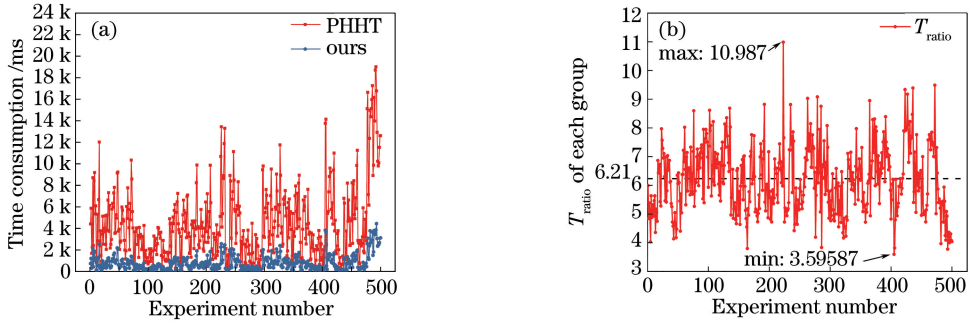

图 5　两种算法效率对比图。(a)每组实验耗费时间对比;(b)每组实验耗费时间倍率

Fig. 5　Efficiency comparison of two kinds of algorithms. (a) Time consumption comparison of each group of experiments; (b) time ratio of each group of experiments

500 次实验中全部满足 $T_{ratio}>3$,大量图像测试结果表明,概率 Hough 变换耗时是本文算法的 3 倍以上。

4.2　有序边界实验

有序实验采用一张模拟图像,该图像可以提取有序边界,分别用概率Hough变换和本文算法检测,时间统计加入所有的前处理时间,单次间隔 $G_{ap}=5$,总间隔容量 $Q=10$,间断总次数 $G_{max}=10$。结果图像如图 6 所示,可见概率 Hough 变换和本文算法的效果基本一致,但从表 1 最后一栏时间中可看出本文算法时间消耗更少,$T_{ratio}=23.7$。

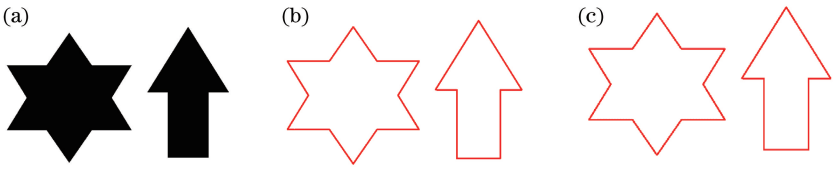

图 6　可以提取有序边界的直线检测。(a)模拟图像;(b) PPHT 结果;(c)本文算法结果

Fig. 6　Straight line detection on a sorted edge image. (a) Analog image; (b) PPHT results; (c) our algorithm results

表 1　有序边界点下概率 Hough 变换和本文算法结果对比

Table 1　Performance comparison of PPHT and our algorithm in sorted edge points

Algorithm	Image size / pixel×pixel	L_{min}/pixel	G_{ap}/pixel	$\Delta\rho$ /pixel	$\Delta\theta$ /rad	Lines	Time /ms
PPHT	4500×2728	80	5	1	π /1800	20	7362
Ours						19	311

4.3　无序边界实验

通过三组实验,进行无序边界点检测效果对比。无序边界实验 1 用于检测算法在有树叶形成的网状边界线检测中的抗干扰性,图 7 是一张 540 pixel×960 pixel 的带有树叶干扰的图像,设定线宽为 1,设定单次间隔 $G_{ap}=5$,总间隔容量 $Q=10$,间断总次数 $G_{max}=10$,由图 7(c)和 7(d)的对比看出,本文算法对于树叶形成的边界纹理有很强的抵抗性,且图 7(b)中的主要直线也被提取出来,相对于概率 Hough 变换,效果更佳,具有更强的稳定性。

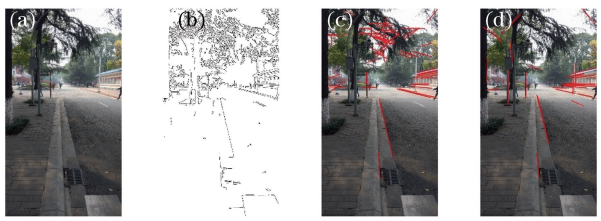

图 7　算法对边界聚合的抵抗效果。(a)原图像;(b) Canny 边界;(c) PPHT 结果;(d)本文算法结果

Fig. 7　Resistance of two algorithms to the boundary polymerization. (a) Original image; (b) Canny edge; (c) PPHT results; (d) our algorithm results

无序边界实验 2 使用一张拍摄建筑图，如图 8(a)所示，分辨率为 960 pixel×540 pixel。可以看到本文算法相较于概率 Hough 变换，直线基本都检测到，还检测到更多窗框的结果。在窗框内产生的边界线聚集使得概率 Hough 变换处产生大量的重合直线，如图 8(c)所示，而本文算法的结果并没受到严重干扰[图 8(d)]。

图 8　建筑物图像检测结果比较。(a)原图；(b) Canny 边界；(c) PPHT 结果；(d)本文算法结果

Fig. 8　Comparison of a building image detection results. (a) Original image; (b) Canny edge; (c) PPHT results; (d) our algorithm results

无序边界实验 3 采用一张有大量树叶和墙砖形成的聚集边线的图像，如图 9(a)所示，本次实验设定单次间隔 $G_{ap}=10$，总间隔容量 $Q=40$，间断总次数 $G_{max}=10$，图像一共有 1070444 个边界点，在 $\Delta\theta=\pi/18000$ 的高精度下，计算量和内存占用巨大，概率 Hough 变换耗时 320 s，整个程序占用了超过 1 G的内存。由(1)式可计算出仅 Hough 空间累加器 A_H 就需要至少 141822000 个储存单元，由 (2) 式可得本文算法仅用了 123480 个储存单元，内存占用比例 $r_a=0.00087$，可见在图像巨大且 $\Delta\theta$ 设定极小的状况下，内存节约更加明显。从结果中看出概率 Hough 变换的结果受到了严重的干扰(图 9(b))，将树叶部分也识别为直线，而建筑外观的直线却有所断开和缺失，从图 9(c)中看出本文算法受到的干扰小得多，并且建筑上的直线检测结果也很良好，时间耗费仅为概率 Hough 变换的 2%，$T_{ratio}=41.8$。

图 9　大图片的算法效果对比。(a) Canny 边界；(b) PPHT 结果；(c)本文算法结果

Fig. 9　Comparison of algorithm effect on a large picture. (a) Canny edge; (b) PPHT results; (c) our algorithm results

无序边界点的三组实验的参数设定和检测直线数量及耗时等比较如表 2 所示，可见直线检测的数量相对准确，且可以计算出三组的时间倍率 T_{ratio} 均在 3 以上。

表 2　无序边界下概率 Hough 变换和本文算法比较

Table 2　Performance comparison of PPHT and our algorithm in unsorted edge points mode

NO.	Algorithm	Image size / pixel×pixel	L_{min}/pixel	G_{ap}/pixel	$\Delta\rho$ /pixel	$\Delta\theta$ /rad	Detected lines	Time /ms
1	PPHT	540×960	120	5	1	π /1800	69	810
	Ours						23	209
2	PPHT	960×540	120	5	1	π /1800	159	715
	Ours						63	227
3	PPHT	4608×2592	200	10	1	π /18000	430	320671
	Ours						208	7674

5 结 论

针对概率 Hough 变换的运行内存大、计算量大的算法问题及在直线端点搜索中容易受到聚集网状边界点干扰的问题，提出了一种减小内存占用、计算速度更快、且对网状边界有一定抵抗性的优化算法。该算法将边界类型分为有序边界和无序边界。

对有序边界使用依据序号的相邻采样点搜索直线的加速算法，且不建立需要大量内存的 Hough 空间累加器，用一维的累加器完成直线判定，采取依据序号顺序向前向后的搜索方式进行直线端点的定位。

对无序边界则随机取出一个边界点并建立感兴趣区域，在区域内进行小范围的概率 Hough 变换，找出直线后进行全局端点检查，这样无需建立整个图的 Hough 空间，仅需一个可重复利用的局部 Hough 空间累加器进行投票，大大减少了计算量和需要的内存；在直线端点搜索中加入了间断区间累计来屏蔽网状边界的干扰，并利用线宽来减少像素点错位带来的错误检测，使得到的直线更加准确。

本文算法在 500 组图像对比测试中全部优于概率 Hough 变换，且时间比率最高 $T_{\text{ratio}}=10.987$，最低 $T_{\text{ratio}}=3.59587$，实验平均时间倍率 $T_{\text{ratio}}=6.21$，相比概率 Hough 变换提速 3 倍以上。进行 4 次效果试验(一次有序，三次无序)。有序测试中，本文算法耗时是概率 Hough 变换的 1/4，且结果准确；无序测试中，第 1、2 个实验中为尺寸 960 pixel×540 pixel的图像，本文算法速度是概率 Hough 变换的 1/3，且直线效果较概率 Hough 更好。第 3 个实验为 4608 pixel×2592 pixel 的大尺寸图像，采用较高精度的参数及大量网状的边界点干扰，概率 Hough 变换耗时 320 s，本文算法仅用 7.6 s，概率 Hough 变换的结果受到网状边界的干扰严重，而本文算法将主要的直线段都抓取出来，并没有受到网状边界的干扰，由于使用了可重复利用的小范围 Hough 累加器，在巨大图像上的内存节约效果尤为明显，内存使用减少 99.91%。

该算法虽然加快了运算、大幅度减小了内存、加入了线宽检查，但是仍不能彻底解决直线重叠的问题，因此还有待进一步提高。

参 考 文 献

[1] Shapiro S D. Feature space transforms for curve detection[J]. Pattern Recognition, 1978, 10(3): 129-143.

[2] Duda R O, Hart P E. Use of the Hough transformation to detect lines and curves in pictures [J]. Communications of the ACM, 1972, 15(1): 11-15.

[3] Zhao L J, Liu E H, Zhang W M, *et al*. Feature extraction of target based on global information[J]. Acta Optica Sinica, 2014, 34(4): 0415001.
赵连军, 刘恩海, 张文明, 等. 利用全局信息提取靶标特征的方法[J]. 光学学报, 2014, 34(4): 0415001.

[4] Jiang G Q, Ke X, Du S F, *et al*. Crop row detection based on machine vision[J]. Acta Optica Sinica, 2009, 29(4): 1015-1020.
姜国权, 柯杏, 杜尚丰, 等. 基于机器视觉的农田作物行检测[J]. 光学学报, 2009, 29(4): 1015-1020.

[5] Yan H R, Yang M S. Line extraction based on improved hough transform[J]. Infrared Technology, 2015, 37(11): 970-975.
闫怀仁, 杨慕升. 基于改进的 Hough 变换的直线提取算法[J]. 红外技术, 2015, 37(11): 970-975.

[6] Zhang G Y, Cheng Y Y, Zhu H. Detection of linear target based on improved hough transform [J]. Computer Engineering and Design, 2014, 35(2): 536-540.
张国英, 程益钰, 朱红. 基于改进 Hough 变换的线性目标检测[J]. 计算机工程与设计, 2014, 35(2): 536-540.

[7] Xu L, Oja E, Kultanen P. A new curve detection method: Randomized Hough transform (RHT)[J]. Pattern Recognition Letters, 1990, 11(5): 331-338.

[8] Liu T, Chen H, Shen M, *et al*. Effective echo extraction for space debris laser ranging using randomized Hough transform[J]. Chinese Journal of Lasers, 2016, 43(4): 0408002.
刘通, 陈浩, 沈鸣, 等. 随机 Hough 变换提取空间碎片激光测距有效回波[J]. 中国激光, 2016, 43(4): 0408002.

[9] Yan R, Zhang L C, Zhang Y S, *et al*. Tricot lace real-time recognition method based on feature recognition[J]. Laser & Optoelectronics Progress, 2015, 52(11): 111002.
鄢然, 张李超, 张宜生, 等. 基于特征识别的经编布花边实时识别算法[J]. 激光与光电子学进展, 2015, 52(11): 111002.

[10] Gong X M, Gao K, Wang Y, *et al*. A novel linear target detection method based on improved probability Hough transform in remote sensing imagery[J]. Imaging Science & Photochemistry, 2017, 35(2): 162-167.
巩学美, 高昆, 王研, 等. 一种基于概率 Hough 变换

的遥感图像中线目标检测新方法[J]. 影像科学与光化学, 2017, 35(2): 162-167.

[11] Chutatape O, Guo L. A modified Hough transform for line detection and its performance[J]. Pattern Recognition, 1999, 32(2): 181-192.

[12] Zhang Z J, Hao X Y, Liu S L, *et al*. Line detection based on Hough one-dimensional transform[J]. Acta Optica Sinica, 2016, 36(4): 0412005.
张振杰, 郝向阳, 刘松林, 等. 基于Hough一维变换的直线检测算法[J]. 光学学报, 2016, 36(4): 0412005.

[13] Duan R J, Zhao W, Huang S L, *et al*. Fast line detection algorithm based on improved Hough transformation[J]. Chinese Journal of Scientific Instrument, 2010, 31(12): 2774-2780.
段汝娇, 赵伟, 黄松岭, 等. 一种基于改进Hough变换的直线快速检测算法[J]. 仪器仪表学报, 2010, 31(12): 2774-2780.

[14] Matas J, Galambos C, Kittler J. Robust detection of lines using the progressive probabilistic Hough transform[J]. Computer Vision and Image Understanding, 2000, 78(1): 119-137.

[15] Wang J X, Zhu Q, Wang W X, *et al*. Straight line extraction algorithm by Hough transform combining edge grouping[J]. Journal of Remote Sensing, 2014, 18(2): 378-389.
王竞雪, 朱庆, 王伟玺, 等. 结合边缘编组的Hough变换直线提取[J]. 遥感学报, 2014, 18(2): 378-389.

[16] Teng J Z, Qiu J. Fast and precise detection of straight line with Hough transform[J]. Journal of Image and Graphics, 2008, 13(2): 234-237.
滕今朝, 邱杰. 利用Hough变换实现直线的快速精确检测[J]. 中国图象图形学报, 2008, 13(2): 234-237.