一种快速实用的直线检测算法*

孙 涵，任明武，杨静宇
(南京理工大学 计算机科学与技术系，江苏 南京 210094)

摘 要：总结了目前几个主要的直线检测算法，并分析了各个算法的优缺点，然后提出了一个新的基于链码的快速直线检测算法，新算法仅需两个约束参数，即最小直线段长度和最小直线段近似度。实验表明，新算法检测速度快、实用性强，适合实时处理。

关键词：链码；直线检测；直线近似度；图像理解
中图法分类号：TP391 **文献标识码**：A **文章编号**：1001-3695(2006)02-0256-02

A Fast and Practical Algorithm for Line Detection

SUN Han, REN Ming-wu, YANG Jing-yu
(Dept. of Computer Science & Technology, Nanjing University of Science & Technology, Nanjing Jiangsu 210094, China)

Abstract: Gives a survey of current line detection algorithms and analyzes their merits and defects. Then, the new line detection algorithm based on chain code is proposed, which only needs two parameters: minimum line length and minimum line similarity degree. And experiments show that the new algorithm is very fast, practical, and suitable to real-time application.
Key words: Chain Code; Line Detection; Line Similarity Degree; Image Understanding

直线检测是图像处理中的一个重要内容，为图像理解提供了直接依据。现有的直线检测算法主要有两大类：①通过对图像的处理得到目标的边界点集合，然后利用 Hough 变换提取目标边界上的直线；②在对图像预处理后，直接获取目标的边界线集合，然后在该集合中进行直线段识别。

基于 Hough 变换的直线检测方法[1,2]日趋成熟。该方法的优点是对图像中的噪声不敏感，个别非边界像素点不影响直线检测结果。但有两点不足，即计算量比较大，不利于实时应用和变换过程中丢失了线段的端点和长度信息。

方法②一般是先对目标边界进行链码跟踪，然后在得到的链码串集合中进行直线段提取。这类方法的优点是计算量小，并且能同时得到直线段的位置、长度、方向等信息。不足之处是算法性能受目标边界的跟踪算法制约。但随着边界跟踪算法的改进，该影响已越来越弱。因此，在实时应用中，该类方法越来越受到重视。本文提出的直线检测算法也属于这一类。

1 基于链码的直线检测

早在 20 世纪 70 年代 Freeman 就提出了直线链码必须满足的三条准则[3]：

(1) 链码中至多出现两个方向码，且它们在模 8 意义下相差 1；

(2) 若有两个方向码出现，则其中之一必单个出现；

(3) 单个出现的方向码总是尽可能均匀地出现在链码串中。

文献[4]中又给出了 Freeman - 吴定理和增量式直线链码识别算法，进一步证明了这三条准则是直线链码的充要条件。

收稿日期：2005-01-21；修返日期：2005-03-22
基金项目：国防科技预研基金资助项目

但是，上述的准则仅适用于理想直线。众所周知，图像的获取过程中，噪声的引入是不可避免的。在实际图像中，通过边界跟踪得到的链码几乎不能满足这三条直线判别准则。因此，研究人员又不断改进，使之能够满足实际应用。如文献[5]中采用上下边界和必经区域约束来检验链码是否属于当前直线段，但该方法没有摆脱 Freeman 准则的束缚，抗噪性不强。文献[6]中采用 BL 算法进行直线链码检验，该算法抗噪性有所提高，在一定程度上能够解决一些实际问题，但其判别方式也过于复杂。

在上述分析的基础上，提出了一种新的基于链码的快速直线检测算法。采用以下形式表示目标边界的链码串：

$$I: Len, X, Y, d_1, d_2, \cdots, d_n \quad (1)$$

其中 I 为当前链码串的编号；Len 为当前链码串中链码的总长度；X, Y 为当前链码串起始点的图像坐标；$d_1, d_2, \cdots, d_n$ 为当前链码串上各像素的方向码。

(1) 判断当前链码串中链码的总长度是否满足最小直线段长度约束。如果链码总长度小于预设的最小直线段长度阈值 LT，则认为该段链码中没有满足要求的直线段，直接舍弃。

(2) 对链码串逐段进行直线段相似度检验。从起点开始，依次选择满足最小直线段长度约束的子链码串。同时计算子链码串的实际长度和两个端点之间的理想直线距离，并根据公式(2)计算该段子链码串的直线近似度：

$$S = \frac{\| p_s - p_e \|}{Len(p_s, p_e)} \quad (2)$$

其中 $\| p_s - p_e \|$ 表示端点 p_s 与 p_e 之间的理想直线距离；$Len(p_s, p_e)$ 表示子链码串从端点 p_s 到端点 p_e 经过像素的实际长度；S 则表示直线段近似度。另外，在计算链码串的实际长度时，若方向码为 0,2,4,6，则链码连接的两像素间的实际长度为 1；若方向码为 1,3,5,7，则两像素间的实际长度为 2(图 1)。

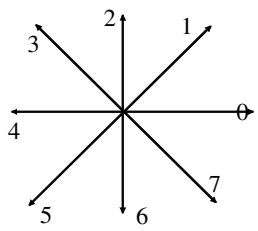

图 1　八链码示意图

显然, S 值越大, 该段链码串越能表示一条直线。如果 S 值小于预设的阈值 ST, 则认为该段链码串不是一条直线。

2　算法描述与分析

2.1　算法描述

根据上述分析, 本文提出的基于链码的直线检测算法具体步骤如下:

输入　目标边界跟踪得到的链码串。

输出　满足预设参数约束的直线段。

(1) 最小直线段长度验证

顺序判断各条链码串。

如果链码串的长度小于预设的最小直线段长度阈值 LT, 则舍弃该条链码串。

(2) 根据直线段近似度准则从链码串中提取直线段

顺序判断各条链码串。

①从起点开始, 依次选择满足最小直线段长度约束的子链码串, 根据式(2) 计算该段的直线段近似度 S。若 $S \geq ST$, 则该段满足直线约束, 转②, 否则舍弃该段, 继续判断下一段子链码串。

②若前一段子链码串也满足直线约束, 则考虑相邻的两条子链码串能否合并成一条更长的直线段。将前一段链码串的起点到本段链码串的终点间的这段链码看作一条新的链码串, 计算其直线段近似度 S'。若 $S' \geq ST$, 则进行合并, 否则将本段子链码串标记成一条新的直线段。

③若已到达链码串终点, 则结束, 否则转①继续判断下一段子链码串。

2.2　算法分析

本文提出的算法与文献[6] 中的 BL 算法相比, 有以下两点不同:

(1) 本文采用的最小直线段长度阈值与 BL 算法中的长度阈值不同。BL 算法中的长度阈值是指链码串中相同方向码持续的行程(判据 1) 和两个方向码持续的行程(判据 2)。在自然图像中, 由于噪声是不可避免的, 相同方向码和两个方向码能够持续的长度是有限的, 因此, BL 算法中的判据 1 和判据 2 能提取的直线段有限。

本文中最小直线段长度阈值是指链码串的实际长度, 与具体的方向码排列方式无关, 能够克服噪声影响。另外, 它有两点优势: ①能够通过该阈值筛选掉一些极短的链码串, 减小后续判决的计算量; ②为后续的直线段提取提供了一个度量单位, 方便直线段相似度的计算。

(2) 为了满足 Freeman 准则约束, 同时又要克服噪声影响, BL 算法分别考虑两个方向码交替出现和多个方向码同时出现的情况, 先进行检测, 然后需要验证, 同时还要考虑直线段的分裂和合并, 判据显得过于复杂。而且, 通过计算像素点到理想直线的偏离程度来检验链码串是否为直线, 计算量较大。

本文提出的算法则采用直线段近似度来进行判别, 不必关心方向码的具体排列分布, 从而避免了 BL 算法中出现的复杂的判据。同时只需考虑相邻两段是否要合并, 不必进行分裂。而且, 以最小直线段长度为单位逐段进行检验, 计算量很小。

另外, 我们算法有一个关键的地方: 在链码串中依次选择满足最小直线长度约束的子链码串进行判别。如何依次选择是一个关键点, 也就是下段子链码串的起点到当前子链码串起点的间距如何确定。如果间距小, 则检验精度高, 但需要进行判别的次数多; 如果间距大, 则检验精度低, 但需要进行判别的次数少。因此, 可以在具体应用中选择合适的间距, 从而在不影响精度要求的前提下提高直线检测速度。

3　实验结果与分析

为验证新算法的性能, 我们进行了 BL 算法和新算法的对比实验(图 2)。图 2(a) 为 Lena 原始图像, 大小为 256 ×256。采用灰度阈值 128 对原始图像进行二值化, 并进行链码跟踪, 共得到 199 条链码串, 如图 2(b) 所示。两个算法的检测结果如图 3(c) 和图 3(d) 所示, 其中, BL 算法中的最小直线段长度 LT 值为 25 个像素, 点到理想直线最大距离 DT 值为 5 个像素, 两直线的最大夹角 AT 值为 5°; 新算法的最小直线段长度 LT 值也为 25 个像素, 最小直线相似度 ST 值为 0.92。图中直线段的端点已用十字线标出。

(a)Lena 原始图像 (256×256)　(b)边界链码跟踪 (199 条链码串)　(c)BL 算法检测结果(28 条直线)　(d)新算法检测结果(35 条直线)

图 2　基于链码的直线检测结果图

从实验结果可知: ①BL 算法有一个缺陷。在 BL 算法的后处理步骤中, 两相邻直线段的合并仅考虑了直线间的夹角关系, 忽视了一种特殊情形, 如图 2(c) 所示, 右上角和左下角出现了两条首尾相接且方向相反的平行直线被合并。它们虽然满足首尾相邻并且夹角很小的条件, 但显然是不能合并的。可见, BL 算法的合并准则不够严密, 应该再加上直线走向需一致的约束。本文采用基于直线相似度约束的直线段合并方法避免了这样的问题, 如果两条直线走向相反, 则其直线相似度会显著下降, 从而阻止合并。②由于提出的算法彻底抛弃了 Freeman 准则的约束, 因此该算法抗噪性强, 直线检出率高, 例如图 2 中新算法的直线检出率比 BL 算法提高了 25%, 从而更有利于后续的图像理解。③直线相似度准则是有效的。在直线相似度大于 90% 时, 直线检测结果是可以接受的。

4　结论

直线检测一直是图像处理领域的热点之一。在总结现有直线检测算法的基础上, 本文提出了一种新的基于链码的快速直线检测算法。该方法从实际应用出发, 摆脱了 Freeman 理想直线准则的约束, 仅采用最小直线段长度和最小直线相似度两个约束参数, 就能快速准确地实现直线检测。实验结果表明, 新算法检测速度快、实用性强, 适合实时处理。(下转第 260 页)

充好, 结束循环。

2.3 权值的设计及分析

为了保留图像的边缘特性和尽量减少模糊, 要使得模板中位置不同的各点对应权重也不同。由于把图像的修补过程设计成了修补区域边界的向内演化过程, 所以我们要综合考虑 p, q 点位置, 修补区域轮廓以及 q 点颜色变化等信息, 来设计各点的权重 $W(p, q)$。

$$W(p, q) = dir(p, q) \cdot dst(p, q) \cdot lev(p, q) \quad (8)$$

设 p 点坐标为(i, j), q 点坐标为$(i - ii, j - jj)$, 则式(8)中的三个分量分别为

$$dir(p, q) = \frac{(p - q)}{|p - q|} \nabla T(p) = \frac{1}{\sqrt{ii^2 + jj^2}} \left(ii\, g \frac{T(p)}{x} \right) + \left(jj\, g \frac{T(p)}{y} \right)$$

$$dst(p, q) = \frac{1}{|p - q|^2} = \frac{1}{ii^2 + jj^2} \quad (9)$$

$$lev(p, q) = \frac{1}{1 + |T(p) + T(q)|^2}$$

这里, $dir(p, q)$ 的作用是控制修补区域边界法线方向上的点对修补点的影响, 也就是说距离 $\nabla T(p)$ 方向近的点对 p 的贡献大; $dst(p, q)$ 是 p, q 两点之间几何距离的倒数, 这很容易理解, 即在几何距离上较远的点对 s 的影响相对小; $lev(p, q)$ 描述了 q 点与修补区域边界距离对 p 点的影响, 也就说明了距离修补区域轮廓线越近, 对 p 点的贡献越大。

我们用图 2 来表示三个分量的效果。在图 2(a) 中, 白色环形为修补区域, 其宽度大于 30 个像素。图 2(c) 只用 dir 分量, 图 2(d) 用 dir 和 dst 分量, 图 2(e) 用 dir 和 lev 分量, 图 2(f) 用 dir, dst 和 lev。图 2 为四种情况的修复效果。可见, 只用 dir 分量时, 造成的模糊很明显; 用 dir 和另外一个分量时, 模糊程度减小; 三个分量全用时效果最好。

3 总结与讨论

下面将设计的新修补算法同 Bertalmio 等的 BSCB 算法进行比较。如图 3 所示, 对同一幅图(图 3(a)) 用 BSCB 和我们的算法修补, 得到的效果图如图 3(b) 和图 3(c) 所示, 可以看出两者的效果相差不大, 但是我们设计的算法速度要快很多。在文献[1] 中, 用 CPU 为 300MHz 的电脑修补该图要用 5 分钟左右, 而本文的算法用 Visual C++ 编程在 CPU 为 1GHz 的电脑上只需 3 秒左右。显然, 该算法可以节省大量时间, 提高效率。所以, 它不像现在的 BSCB, TV, CDD 等算法由于太耗时经常造成电脑无响应、死机等现象, 该算法不太耗资源, 一般能将修补时间控制在 5 秒之内, 所以非常适合作为图像处理软件的插件使用。

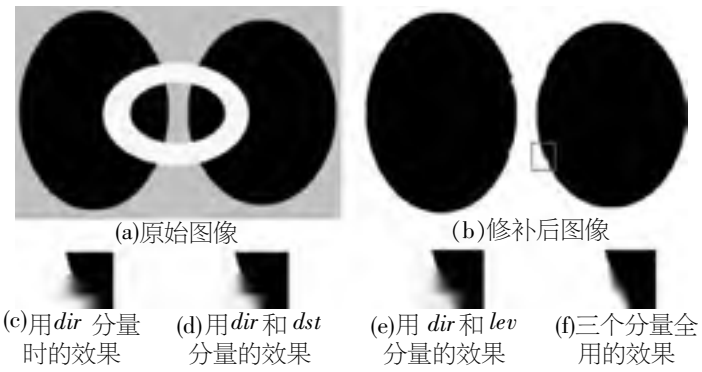

图 2 三分量效果图

图 3 新算法与 BSCB 算法的比较

参考文献:

[1] arcelo Bertalmio, Guillermo Sapiro, Vicent Caselles. Image Inpainting [J]. SIGGRAPH, 2000, (1): 417-424.

[2] T F Chan, J Shen. Mathematical Models for Local Non-texture Inpaintings [J]. SIAM Journal on Applied Mathematics, 2002, 62: 1019-1043.

[3] Criminisi A, Perez P, Toyama K. Objects Removal by Exemplar-based Inpainting[J]. IEEE Proc. CVPR, 2003, 2: 721-728.

[4] T F Chan, J Shen. Non-Texture Inpainting by Curvature-Driven Diffusion (CDD) [J]. Journal of Visual Communication and Image Representation, 2001, 12: 436-449.

[5] Osher, S Sethian J. Fronts Propagating with Curvature Dependent speed: Algorithms Based on the Hamilton-Jacobi Formulation [J]. Journal of Computational Physics, 1988, 79: 12-49.

[6] Sethian J A. Fast Marching Methods [J]. SIAM Review, 1999, 41: 199-235.

[7] Sethian, J. A Fast Marching Level Set Method for Monotonically Advancing Fronts [J]. Proceedings of the National Academy of Sciences, 1996, 93: 1591-1595.

作者简介:

周世生(1963-), 男, 山东潍坊人, 教授, 博士, 主要研究方向为颜色信息处理与印刷复制技术、高功能印刷版材技术的研究开发、现代票证印刷技术的研究开发等; 孙帮勇(1980-), 男, 山东青岛人, 硕士研究生, 研究方向为图像修补及图像处理。

(上接第 257 页)

参考文献:

[1] laus Hansen, Jens Damgaard Andersen. Understanding the Hough Transform: Hough Cell Support and Its Utilisation [J]. Image and Vision Computing, 1997, 15(3): 205-218.

[2] John Immerk r. Some Remarks on the Straight Line Hough Transform [J]. Pattern Recognition Letters, 1998, 19(12): 1133-1135.

[3] H Freeman. Boundary Encoding and Processing[A]. B S Lipkin, A Rosenfeld[C]. Picture Processing and Psychopictorics, Academic, New York, 1970. 241-266.

[4] 顾创, 翁富良, 吴立德. 直线链码的线性提取算法[J]. 模式识别与人工智能, 1990, 3(2): 34-38.

[5] Jianxing Yuan, Ching Y Suen. An Optimal O(n) Algorithm for Identifying Line Segments from a Sequence of Chain Codes [J]. Patter Recognition, 1995, 28(5): 635-646.

[6] 史册, 徐胜荣, 荆仁杰, 等. 实时图像处理中一种快速的直线检测算法[J]. 浙江大学学报(工学版), 1999, 33(5): 482-486.

作者简介:

孙涵(1978-), 男, 博士研究生, 主要研究方向为数字图像处理、模式识别与智能机器人; 任明武(1969-), 男, 副教授, 主要研究方向为数字图像处理、计算机视觉、图像压缩与编码; 杨静宇(1941-), 男, 教授, 博导, 研究方向为模式识别、数字图像处理、计算机视觉和智能机器人。