DOI：10.7652/xjtuxb201808015

一种适于硬件实现的快速连通域标记算法

张国和，徐快，段国栋，赵晨，梁峰
（西安交通大学电子与信息工程学院，710049，西安）

摘要：针对模式识别、计算机视觉和图像处理中常用的特征提取和选择问题，提出了一种适于硬件实现的快速连通域标记算法。首先进行行扫描，判定同一行内连续的前景像素，即游程，并记录游程的起始坐标和结束坐标；然后进行游程标记和等价游程对合并，对上述标记的游程根据连通情况对其赋予临时标记；最后扫描上一行游程，通过检测相应的标志位判断上一行游程是否真正结束，若已结束，将已结束区域信息进行输出，否则继续进行下一行的扫描。使用不同的二值图像进行实验，并与已有算法性能进行比较，仿真结果表明，所提出的快速连通域标记算法在速度和资源需求方面具有明显优势，对图像处理的平均帧率可以达到20帧/s以上，对于分辨率为2 048×1 536像素的图像，需求的片上存储资源约为3.45 Mbit，仅为块决策表算法的21.9%、He算法的7.6%左右。

关键词：图像处理；二值图像；连通区域标记；硬件
中图分类号：TN492 **文献标志码**：A **文章编号**：0253-987X(2018)08-0095-07

A Fast Labeling Algorithm of Connected Components Applicable for Hardware Implementation

ZHANG Guohe, XU Kuai, DUAN Guodong, ZHAO Chen, LIANG Feng
(School of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an 710049, China)

Abstract: A fast labeling algorithm of connected components applicable for hardware implementation is proposed for feature extraction and selection that are commonly used in pattern recognition, computer vision and image processing. First, row scanning is performed to determine consecutive foreground pixels in the same row, i.e., runs, and to record the start and end coordinates of the runs. Then, the run labels and the equivalent run pairs are merged, and temporary values are assigned to these labeled runs according to the connection conditions. Finally, runs of the last row is scanned to determine whether the run of the last row is really over by detecting the corresponding flag bit if it is really finished, and the information of finished region is output, otherwise the next row is scanned. Experiments using different binary images and comparisons with the performance of existing algorithms show that the proposed algorithm has obvious advantages in speed and resource requirements. The average frame rate of image processing speed reaches up to 20 frames/s, and the algorithm requires about 3.45 Mbit on-chip memory resource for an image with a resolution of 2 048×1 536 pixels, which is only 21.9% and 7.6% of the requirements of the block based decision table algorithm and the He algorithm, respectively.

Keywords: image processing; binary image; connected components labeling; hardware

收稿日期：2018-03-12。 作者简介：张国和(1981—)，男，副教授，博士生导师。 基金项目：国家自然科学基金资助项目(61474093)。
网络出版时间：2018-06-08 网络出版地址：http://kns.cnki.net/kcms/detail/61.1069.T.20180608.1809.010.html

在计算机视觉、遥感、医学、生物特征学、文档分析、机器人等领域，图像中连通区域信息的提取和标记具有重要的作用[1]。在基于硬件实现的图像、视频处理系统中，与可流水实现的图像滤波、边缘检测和阈值分割等算法相比，连通区域标记算法往往需要占用更多的硬件资源和处理时间[2]，成为高速图像处理的瓶颈。因此，如何提高连通区域标记算法的执行效率是整个图像处理系统中的关键问题[3]。

通过对近些年提出的典型的二值图像连通域标记算法进行详细描述，总结其优缺点，对相应算法复杂度和资源需求进行了分析，以供设计出适合于硬件实现的二值图像连通域标记算法。当前已有的区域连通标记算法根据其实现方式可以分为 2 类：软件可实现算法和硬件可实现算法。

Rosenfeld 等提出的两遍扫描算法被视为经典的区域连通标记算法[4]，但是由于存储等价标记所需的内存空间和合并等价标记所需的时间都很大，此算法仅仅适用于软件实现。Chang 等提出的轮廓追踪(CT)算法[5]只需扫描图像一次，然而轮廓像素点可能被扫描多次。由于算法中对内存的访问非常没有规律，此算法也仅仅适用于软件实现。何立风等人设计的基于游程的两遍扫描连通域标记算法[6-8]，引入游程的概念，对像素批量进行标记和等价标记合并。经过测试，该算法在同类算法中的效率最高，且随着图像像素和复杂度的增大具有较稳定的线性特性，本文的连通域标记算法借鉴了此算法对于等价标记表的实现方式。在目前已知的区域连通算法中，Grana 等提出的块决策表(BBDT)算法[9]具有最好的性能，该算法增大了相邻像素的扫描窗口，缺点是只能支持 8 连通规则。

由于对大存储空间的要求，上述区域连通算法往往无法通过硬件逻辑加速，由此又出现了一些适用于硬件实现的连通域标记算法。Kofi 等提出一种基于游程长度的区域连通算法[10]，可以通过片上 RAM 实现，但对于大于 1 024×1 024 像素的图像，过高的存储空间要求依然成为瓶颈。Johnston 等提出的一次扫描算法[11]在 FPGA 实现过程中，由于需要对堆栈进行复杂的写入读出操作，使得电路的最高工作频率很低，并不利于图像的实时高速处理。

国内许多高校也对区域连通标记算法进行了一定的研究[12-14]，如沈阳工业大学的冯海文等人提出的高效一遍扫描连通域标记算法[12]、四川大学的王凯等人设计的基于图像分块的连通域标记算法[13]、华东理工大学的刘关松提出的新型二值图像连通域标记算法[14]等都是在一些经典算法的基础之上做了一定的改进，对算法效率的改善不大，且大多基于软件实现，本文主要研究连通域标记算法的硬件电路实现，因此这类算法仅作参考。

从国内外研究现状来看，比较主流的连通域标记算法仍然是基于对图像的光栅扫描，这些算法受限于过高的资源需求，只适用于软件实现，因此速度成为其主要的衡量指标。

为了减少存储等价标记所需的空间和处理等价标记所需的时间，本文以游程为单位批量地对行内连续像素进行处理。在逐行对图像进行光栅扫描的过程中，将已经结束的连通区域即时输出，不仅减少了存储这些连通域信息所需的空间，而且只需对图像完整地扫描一遍，就可以完成对连通域信息的提取或者连通域像素的标记，极大地提高了运行速度，使其能高速、高效完成连通域的标记，用于实现图像的快速扫描与追踪。

1 二值图像连通域标记算法的设计

常见的连通区域分 8 邻域和 4 邻域连通 2 种，由于 8 邻域连通更为全面，通用性好[15]，因此本文以 8 邻域连通为例进行描述。本文所提出的快速连通域标记算法需要经过行扫描、游程标记和等价游程对合并、已结束游程检测和已结束区域信息输出几个主要阶段，完成整幅图片的连通域标记和特征提取。

本文所提出的连通域标记算法采用光栅扫描顺序[16]，逐行逐列扫描图像的每个像素，将每一行内连续的前景像素归为一个游程，以游程为基本单位进行预标记和等价标记合并，降低了所需记录标记的数量和处理等价标记所需的时间。为了减小存储已扫描完游程所需的存储空间，在当前行扫描结束后对上一行游程进行检测，输出已结束区域包含的所有游程。该算法只需对图像完整地扫描一遍，就可以完成所有连通域的标记，避免了对整幅图像的临时标记值进行缓存。本文算法的流程图如图 1 所示，主要分为以下几个步骤。

(1)行扫描。从左至右逐个对当前行的像素进行扫描，判定行内的所有游程，并将游程的起始坐标和结束坐标存入 A_o 或 A_e 数组(奇数行存入 A_o 数组，偶数行存入 A_e 数组)。当目标像素和前一个像素为“01”时，意味着一个游程的开始，当目标像素和前一个像素为“10”时，意味着一个游程的结束，需要注意的是在行开始和行结束像素的边界情况。

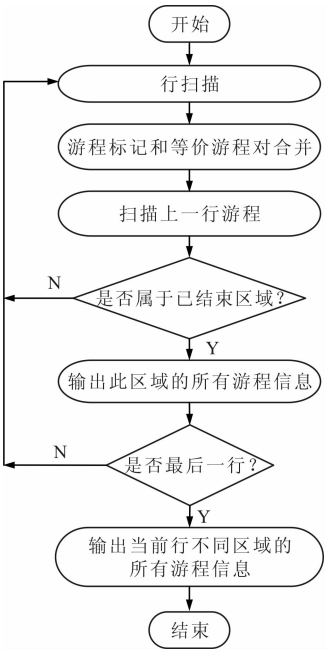


图 1　本文提出算法流程图

(2)游程标记和等价游程对合并。对于步骤(1)中在当前行产生的每个游程，需要对其赋予临时标记。对于当前行的某个游程 R_1，需要扫描上一行的所有游程信息，若当前行为奇数行，则扫描 A_e 数组中上一行的游程信息，若当前行为偶数行，则扫描 A_o 数组中上一行的游程信息。2 行间的 2 个游程由于位置不同存在的连通情况如图 2 所示。

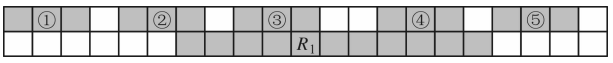


图 2　行间游程可能存在的连通关系示意图

若上一行的游程与当前行的游程的位置关系如图 2 中的游程②、③、④的情况所示，则认定两者之间存在连通关系。算法中的实现方式为：设当前行游程的起始坐标为 X_0，结束坐标为 X_1，上一行游程的起始坐标为 Y_0，结束坐标为 Y_1，若坐标满足下列 2 个公式，则认定这 2 个游程存在连通关系，可描述为

$$X_0 \leqslant Y_1;\ X_1 \geqslant Y_0\text{(四连通)} \tag{1}$$

$$X_0 \leqslant Y_1+1;\ X_1 \geqslant Y_0-1\text{(八连通)} \tag{2}$$

若当前行游程 R_1 仅与上一行的一个游程 R_2 连通，则将 R_2 的临时标记 L_2 赋予 R_1 即可，同时需要将该标记值存入 A_o 或 A_e 数组中相应的游程信息中，以供后续的等价标记合并时查看，还需将游程信息(起始坐标、结束坐标、所在行数、临时标记)记录到 A_n 数组(记录所有游程的信息)中，以供后续输出已结束游程的信息时查看。如图 2 所示，若当前行游程 R_1 同时与上一行的多个游程②、③、④连通，则将这些游程中最左侧的(即最早被扫描到的)游程②的临时标记 L_2 赋予 R_1 即可，并将相应信息记录到 A_o、A_e 和 A_n 数组中。同时，游程③、④的标记值 L_3、L_4 与 L_2 被认定为等价标记对，需要在后续的等价标记对合并操作中进行合并。

若当前行游程 R_1 与上一行的游程的位置关系如游程①或⑤的情况所示，说明未与上一行的任何游程连通，则需要为 R_1 新建一个临时标记 L_1，新的临时标记 L_1 的代表标记也为 L_1，并将相应信息记录到 A_o、A_e 和 A_n 数组中。

对于上述记为等价标记对的 2 个临时标记，需要进行合并。合并的主要操作是将 2 个临时标记所在的代表标记集合进行合并。首先取出这 2 个临时标记值的代表标记 S_1 和 S_2，若 $S_1=S_2$，则不需要合并；否则，将较大的代表标记集合中的所有临时标记归入较小的代表标记集合中，实现了 2 个游程所代表的区域的合并。

(3)扫描上一行游程。在当前行游程标记和等价游程对合并结束后，需要对上一行的游程信息进行扫描，主要检测 F_l(代表该游程是否与下一行的游程发生连通关系)标志位。若 $F_l=1$，意味着该游程与当前行的游程发生连通关系，跳过；若 $F_l=0$，意味着该游程未与当前行的游程发生连通关系，但这只能说明该游程所在的区域在局部结束，是不是真正结束还要根据该游程的代表标记与当前行所有游程代表标记的关系来判定。如图 3 所示的图像形状，当扫描到第 5 行时，第 4 行的游程②未与当前行的任何游程连通，但其所在区域在游程①、③的位置与当前行游程④、⑤连通，因此并未结束。

考虑到这种特殊情况，本文采取的策略是对于上一行未与当前行的连通的游程，取出其代表标记值 S_2 与当前行所有游程的代表标记值进行比较(代表标记表征了游程所在的区域)，若没有与之相等的，则说明该游程未与当前行的任何游程存在等价关系，换言之就是该游程未与当前行的任何游程处在同一区域，说明此区域在上一行的这个游程位置已经结束，将其代表标记值 S_2 作为结束标记存入 A_f 数组(记录上一行已结束游程的代表标记)，不重复记录；若有与之相等的，则该游程所在的区域在当前行还在延伸，不作任何操作。

(4)输出已结束区域信息。若步骤(3)中检测出上一行存在已经结束的游程，则从 A_f 数组中取出其代表标记 L_f，遍历 A_n 数组中游程的临时标记，将代

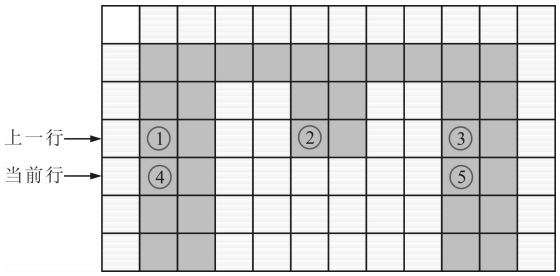


图 3 判定游程结束的特殊案例

表标记和 L_f 相等的所有游程的信息输出，输出后要对 A_n 数组中空出的空间做出填充以节省存储空间。

2 存储机制和资源需求优化

2.1 游程信息表存储机制

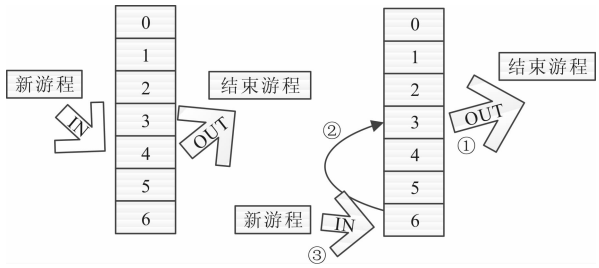
根据上文的描述，本文算法需要的资源需求主要是来自 A_o、A_e、A_n、A_f、B_r（记录临时标记集合中所有标记的代表标记）、B_n（记录临时标记集合中所有标记的排列顺序）、B_t（记录临时标记集合的结束点）等数组，其中 A_n 数组用来记录扫描过程中所有游程的信息，所需的位宽和深度最大。为了降低 A_n 存储块的大小，本文对第 1 章中算法步骤(2)中 A_n 数组的存储机制进行一定的优化，主要的思想就是将已结束游程的信息输出后，将新建立的游程信息存入空闲的存储空间里，避免建立新的存储空间。如图 4 所示，位于 A_n 数组中地址 4 的游程被检测为结束游程，将其游程信息输出，这样地址 4 指向的空间就闲置了出来，当扫描下一行像素检测到新的游程时，把新的游程信息存入地址 4，避免建立新的地址空间，这样可以降低存储所有游程信息需要的存储块深度。

通过对多幅分辨率为 2 048×1 536 像素的图片进行测试，经过优化后，A_n 数组的深度需求大大减少，最大降低的幅度高达 90.3%，最少的也有 32.6%，其中资源节约率为后者相对于前者降低的幅度，计算公式为

$$P_r = \frac{H_{before} - H_{A_n}}{H_{before}} \tag{3}$$

式中：P_r 为资源节约率；H_{before} 为优化前应有深度；H_{A_n} 为 A_n 数组实际占用深度。

经过优化后的 A_n 数组的存储机制虽然极大地降低了资源需求，但是考虑到每次有新的游程需要存储时，都需要对 A_n 数组中已占用过的所有的空间进行遍历，寻找闲置的空间，这个过程会随着游程数量的增多耗费更多的时间。因此也需要对 1 中算法步骤(4)A_n 数组中游程信息的输出机制进行优化，主要的思想就是当某个地址的游程信息输出后，将数组末尾的游程信息存至该位置，地址指针减 1，这样有新的游程需要存储时，就可以省去遍历 A_n 数组的时间，直接将数组信息存入地址指针指向的空间，即原来末尾游程的存储空间，如图 4 所示。



(a)优化前 (b)优化后

图 4 A_n 数组输出机制优化前后示意图

2.2 硬件实现资源需求分析及优化

在硬件实现时，首先要考虑算法需求的片上资源。由上述分析过程可知，该算法共需要 7 个数据阵列：A_o、A_e、A_n、A_f、B_n、B_t 和 B_r，对于分辨率 $M\times N$ 图像，每组存储块所需的位宽和深度分析如下。

对于分辨率 $M\times N$ 图像，区域个数的极限值为 $M\times N/2$，考虑到发生这种情况的时候，图像检测区域已经没有意义，所以即使求出区域信息也是没有必要的。故可以约定一个可以接受的区域上限，来减少资源的占用。如果输入图像对单独一个点的区域做了过滤，那么区域上限为 $M\times N/4$。

当图像像素分辨率为 2 048×1 536 像素时，本文连通域标记算法在硬件实现时理论上所需的存储资源大约为 88.87 Mbit，这大大超出了可以接受的范围。但是，在实际测试中发现一些 RAM 的使用率远远小于其理论所需设计的深度，而且一些变量的最大值也远远小于其理论预估的极值，因此需要对上述 RAM 深度和宽度的分配进行一定的优化。

首先需要优化的变量为临时标记和代表标记的最大值。根据上文的分析，认定临时标记和代表标记的最大值为区域数量的上限，对于分辨率 $M\times N$ 的图像，最大的连通域数量为 $M\times N/4$。考虑到本文算法最大支持的图像分辨率为 2 048×1 536 像素，在 MATLAB 中对于大量分辨率为 2 048×1 536像素、且构成复杂的图像进行测试，发现其游程的最大临时标记和代表标记值远远小于理论最大值 $M\times N/4$。出于节约硬件资源的考虑，这里取 $2^{14}=16\ 384$ 即可以满足实际需求，也就是临时标记和代表标记的记录需要占用位宽 14 bit，深度为 16

kbit。因此 B_n、B_t 和 B_r 数组的 RAM 需求为16 000×14 bit。A_o 和 A_e 数组的位宽需求为 40(40=2×12+14+2)bit，A_f 数组的位宽需求为 14 bit，A_n 数组的位宽需求为 50(50=2×12+12+14)bit。

表 1　本文算法各数组对应的 RAM 需求

数组	RAM 深度/bit	RAM 宽度/bit	RAM 总需求/kbit
A_o，A_e	512	40	20.48
B_n，B_t，B_r	16 000	14	224
A_f	256	14	3.584
A_n	64 000	50	3 200

接下来对各数组所需求 RAM 的深度进行优化。如之前 2.1 所述，通过对 A_n 数组存入读出机制进行一定的调整，实现了对其存储深度的最大利用，根据表 1 统计的信息，将 A_n 数组对应 RAM 的深度设为 64 kbit 即可满足实际需求。为了节省硬件资源，将 A_o 和 A_e 数组的 RAM 深度设为 $M/4=512$ bit，将 A_f 数组的 RAM 深度设为 $M/8=256$ bit，即可满足实际需求。因此，对于分辨率为 2 048×1 536的图像，RAM 总需求量为 20.48+224+3.584+3 200=3 448 kbit≈3.45 Mbit。

表 2　5 种算法硬件实现需求的片上资源对比

算法	BBDT	He	Kofi	Prasad	本文
存储资源/Mbit	15.75	45.02	69.75	252.0	3.45
倍率	4.56	13.04	20.21	73.04	1.00

如表 2 所示，对比可见本文算法硬件实现时需要的片上资源为 BBDT 算法的 21.9%，为 He 算法[8]的 7.6%。该配置所提供的硬件资源空间足以处理分辨率 2 048×1 536 像素及其以下的绝大多数二值图像。

3　算法的硬件架构

本文所提出标记算法的硬件系统结构框图如图 5 所示。其中对外连接的数据接口主要分为 3 组，第一组接口连接图像源输入模块，接受来自图像源的数据和数据有效信号，并反馈行结束信号给图像输入模块，以开启下一行的图像数据输入；第二组接口连接参数配置模块，主要完成对图像分辨率以及二值图像连通规则的配置，图像分辨率的配置一定要与图像的像素个数匹配，避免出现数据不足或溢出的情况；第 3 组接口通过输出使能信号的控制，输出已结束的区域游程息。

在二值图像连通域标记硬件结构内部，主要分为 4 个操作模块、2 个控制模块以及 4 组片上 RAM，这 4 个操作模块都受状态机控制模块的控制，与片上 RAM 交互标记过程中产生中间信息，其中 A_n 数组对应 RAM 的存储机制较为复杂，要受到 A_n 存储控制模块的控制。这 4 个操作模块组成了一个完整的行操作流程，完成一幅分辨率 $M\times N$ 的二值图像的连通域标记需要 N 次行操作流程。

扫描模块根据输入的图像数据，判定当前行的所有游程，将游程的起始坐标和结束坐标存入 A_o、A_e 数组中；标记及合并模块读取当前行和上一行的游程坐标，判定当前行每一个游程与上一行游程的连通关系，将游程的临时标记和标志位送回 A_o、A_e 数组中，将新建的代表标记和合并等价标记对的信息传递给 B_n、B_t、B_r 存储器，最终还要将游程的信息存入 A_n 存储器中；结束区域检测模块根据 A_o/A_e 存储器中存的上一行游程的标记位，将上一行中未

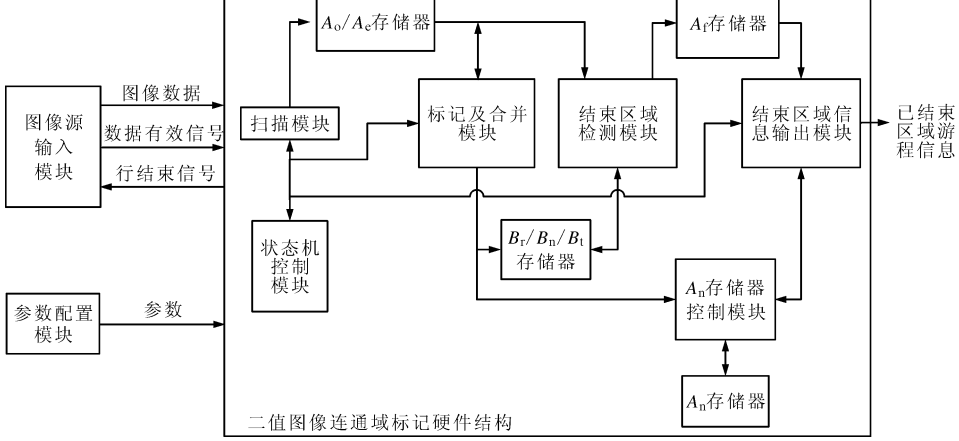


图 5　本文算法的硬件系统结构图

与当前行连通的游程取出作为待检测游程，将这些游程的代表标记与当前行所有游程的代表标记进行比较(需要与 B_r 存储器交互信息)，锁定已结束的游程，再将这些游程的代表标记存入 A_f 存储器中；结束区域信息输出模块根据 A_f 中存储的代表标记，将 A_n 存储器中所有与之相等的游程全部输出。在算法的硬件实现过程中，为了提高运行速度，使用多个电路同时工作以充分利用硬件系统的并行处理能力。

4 实验结果及分析

为了对比本文算法与其他算法的性能，选取了 2 幅分辨率为 512×512 像素的图像，分为记为图 A 和图 B 在图像源一定的情况下，对比不同算法处理的运行时间。由于其他算法都是在高速 PC 工作站上基于软件运行，因此这里引入频率换算因数来有效地对比工作速度。各算法的类型和仿真平台如表 3 所示。并且将本文提出的算法在 Altera 公司的 StrtixIV 系列 EP4SGX530H H35C2 板子上进行了验证，证实了其硬件可行性。

表 3 不同算法的类型和仿真平台

算法	算法类型	仿真平台	频率换算因数
Contour Tracing[5]	单次扫描	1	30
Hybrid[17]	多次扫描	1	30
He[8]	2 遍扫描，基于游程	1	30
BBDT[9]	2 遍扫描	2	24
本文	一遍扫描，基于游程	3	1

仿真结果如表 4 所示，其中用到 3 个不同的仿真平台，平台 1 为 C language on a PC-based workstation (Intel Pentium D 3.0 GHz + 3.0 GHz CPUs，2GB-Memory，Mandriva Linux OS)；平台 2 为 C++language on a PC-based workstation(Inter Core i5 2.4 GHz+2.4 GHz CPUs，2GB-Memory)；平台 3 为 Verilog language on Modelsim 10.2c @ 100 MHz。从表 4 可以看出，在频率换算以后，本文的算法要快于其他 4 种算法，其中比 He 算法要快至少 3 倍以上，比 BBDT 算法要快至少 9 倍以上。考虑到仿真频率较低，在最大工作频率限度内提高运行频率能够极大提高处理速度，而且 SMIC 0.18 um 仿真库并不是当前最先进的工艺，因此若将工艺尺寸减小，本文算法的工作性能还有很大的提升潜力。

表 4 对比测试仿真结果

算法	运行时间/ms		频率换算后运行时间/ms	
	图 A	图 B	图 A	图 B
Contour Tracing[5]	2.30	1.90	69.00	57.00
Hybrid[17]	14.50	14.00	435.00	420.00
He[18]	1.30	1.30	39.00	39.00
BBDT[9]	4.14	3.66	99.36	87.84
本文	9.73	3.54	9.73	3.54

为了进一步验证本算法的正确性和与已有算法相比较的优势，在 SIMPLIcity 数据库[18]中选取了 500 张分辨率都为 512×512 像素的图片，将其分为 9 个不同疏密等级(即前景像素占总像素的比例)，图 6 为各算法在图像密度不同时的柱型统计图。

由图 6 分析可得，本文算法在图像疏密等级较低，图像构成复杂度稍低时具有明显优势，游程数量巨大，图像复杂度太高时标记速度会有所减慢，分析其原因主要是本文算法在扫描过程中实时输出已结束游程的信息，这样可以大大减少需要存储的游程信息条目，但是代价是，在游程信息表中遍历查找已结束游程需要耗费绝大部分运行时间，特别是游程数量巨大的情况。这主要是资源需求和运算速度之间的折中问题。

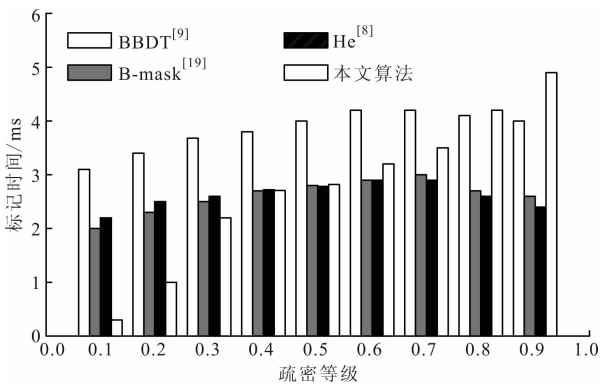


图 6 各算法在图像疏密不同时标记时间对比图

5 结 论

本文提出了一种快速连通区域标记算法的硬件实现方法。算法的主要特点是在扫描过程中实时输出检测到的已结束区域信息，只需完整地扫描一遍图像，就可以完成对所有区域的标记，不仅避免了对整幅图像标记值的缓存，还减少了需要记录的游程信息。通过对游程信息表存取机制的优化，大大降低了算法的存储资源需求，但是由于算法在扫描过程中实时输出检测的已结束游程信息，这样可以减

少存储资源，而在游程信息表中遍历查找已结束游程需要耗费绝大部分运行时间，特别是游程数量巨大、图像复杂的情况，这主要是资源需求和运算速度之间的折衷问题。因此，本文所提出的算法是一种对于小图像特别高效的算法。

参考文献：

[1] HUGO H，FREDRIK K，VIKTOR O. Implementation of a labeling algorithm based on contour tracing with feature extraction [C]// Proceedings of IEEE International Symposium on Circuits and Systems. Piscataway，NJ，USA：IEEE，2007：1101-1104.

[2] AYMAN A，RAMI Q，STAN I，et al. One scan connected component labeling technique [C]// Proceedings of 2007 IEEE International Conference on Signal Processing and Communications. Piscataway，NJ，USA：IEEE，2007：1283-1286.

[3] 王凯，施隆照. 基于 FPGA 的快速连通区域标记算法的设计与实现 [J]. 计算机工程与应用，2016，52(18)：192-198.
WANG Kai，SHI Longzhao. Design and implementation of fast connected region marking algorithm based on FPGA [J]. Computer Engineering and Applications，2016，52(18)：192-198.

[4] ROSENFELD A. Sequential operations in digital picture processing [J]. Journal of the ACM，1966，13(4)：471-494.

[5] CHANG Fu，CHEN Chunjin，LU Chijin. A linear-time component-labeling algorithm using contour tracing technique [J]. Computer Vision and Image Understanding，2004，93(2)：741-745.

[6] HE Lifeng，CHAO Yuyan，SUZUKI K. A run-based two-scan labeling algorithm [J]. IEEE Transactions on Image Processing，2008，17(5)：749-756.

[7] HE Lifeng，CHAO Yuyan，SUZUKI K，et al. Linear-time two-scan labeling algorithm [C]// Proceedings of the IEEE International Conference on Image Processing. Piscataway，NJ，USA：IEEE，2007：241-244.

[8] HE Lifeng，CHAO Yuyan，SUZUKI K，et al. Fast connected-component labeling [J]. Pattern Recognition，2009，42(9)：1977-1987.

[9] COSTANTINO G，DANIELE B，RITA C. Optimized block-based connected components labeling with decision trees [J]. IEEE Transactions on Image Processing，2010，19(6)：1596-1609.

[10] KOFI A，ANDREW H，PATRICK D，et al. A run-length based connected component algorithm for FPGA implementation [C]// Proceedings of the International Conference on Field-Programmable Technology. Piscataway，NJ，USA：IEEE，2008：177-184.

[11] JOHNSTON CT，BAILEY DG. FPGA implementation of a single pass connected components algorithm [C]// Proceedings of the IEEE International Symposium on Electronic Design，Test and Applications. Piscataway，NJ，USA：IEEE，2008：228-231.

[12] 冯海文，牛连强，刘晓明. 高效的一遍扫描式连通区域标记算法 [J]. 计算机工程与应用，2014，50(23)：31-35.
FENG Haiwen，NIU Lianqiang，LIU Xiaoming. Efficient one-pass scanning connected area labeling algorithm [J]. Computer Engineering and Design，2014，50(23)：31-35.

[13] 王凯，黄山，赵瑜，等. 面向图像目标提取的改进连通域标记算法 [J]. 计算机工程与设计，2014，35(7)：2438-2441.
WANG Kai，HUANG Shan，ZHAO Yu，et al. Improved connected domain marking algorithm for image object extraction [J]. Computer Engineering and Design，2014，35(7)：2438-2441.

[14] 刘关松，吕嘉雯. 一种新的二值图像标记的快速算法 [J]. 计算机工程与应用，2002，38(4)：57-59.
LIU Guansong，LÜ Jiawen. A new fast algorithm for binary image marking [J]. Computer Engineering and Applications，2002，38(4)：57-59.

[15] DIEGO J C，SANTIAGO R，CAVALCANTI G，et al. Efficient 2×2 block-based connected components labeling algorithms [C]// Proceedings of the IEEE International Conference on Image Processing. Piscataway，NJ，USA：IEEE，2015：4818-4822.

[16] CHEN Baisheng. A new algorithm for binary connected components labeling [J]. Computer Engineering and Applications，2006，42(25)：46-47.

[17] SUZUKI K，HORIBA I，SUGIE N. Linear-time connected-component labeling based on sequential local operations [J]. Computer Vision and Image Understanding，2003，89(1)：1-2.

[18] WANG J Z，LI J，WIEDERHOLD G. Semantics-sensitive integrated matching for picture libraries [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence，2001，23(9)：947-963.

[19] SUTHEEBANJARD P，PREMCHAISWADI W. Efficient scan mask techniques for connected components labeling algorithm [J]. Eurasip Journal on Image and Video Processing，2011(1)：1-20.

（编辑　刘杨）