# GPU上基于旋转不变细化算法的并行加速策略

## Abstract

## 1. 引言

随着GPU的发展，GPGPU多线程并行处理能力的增强和CUDA等通用并行编程模型的出现，使其能够解决很多领域内的大型数据密集和计算密集的问题，如图像处理、科学计算等领域。

在图像处理中，图像细化算法为将形状宽度变为1像素宽度的过程。细化算法能够在保持形状骨架的基础上减少冗余信息，因此细化是很多复杂图像算法的一个重要的预处理步骤，如字符识别、模式识别等。Lam 等[8]于1992年调研了上百种图像细化算法，将图像细化算法分为两类：并行细化算法和串行细化算法。并行细化算法通过每次迭代删除边缘冗余点，串行细化算法则一次删除所有冗余点。并行细化算法通常比串行细化算法快，但串行细化算法往往能产生更好的结果。近年来，图像大小逐渐增大，如医学图像和卫星遥感图像，图像细化的处理时间也相应增加。大图像对细化算法的实时性来说是一个极大的挑战。举例说明，细化一个2048\*2048像素大小的灰度图像，在i7 CPU上，Ahmed和Ward（AW）细化算法[2]需要17s左右。

相关研究已经注意到细化算法的实时性问题。Zhang和Sune（ZS）等[7]在1984年提出了一种快速并行细化算法，通过迭代两子步骤删除图像中的冗余点。金汉均等[10]在2014年将ZS算法简单移植到GPU上。将算法简单移植到GPU上仅利用了GPU的硬件优势，往往不能最大化利用GPU的硬件优势。Hu BingFeng等[9]针对D12-PK-C 3D细化算法，根据GPU硬件特性对算法进行改进，取得152倍的加速比。

AW细化算法是第一个旋转不变的并行迭代细化算法。如果原始图像旋转了一定角度，那么其细化结果也将旋转相同的角度。该算法很好的解决了断线问题，对噪声点具有一定的抗干扰能力。图1列出了简单移植并行细化算法到GPU上的加速结果，本文实验测试了四个并行细化算法。可见，ZS算法、GH算法和PS算法能够较好的利用GPU硬件优势，获得平均93.7倍加速比。而AW算法的加速比仅为21倍。主要原因是AW算法的旋转不变特性导致其算法流程比其他细化算法更加复杂，核心步骤中包含有大量条件判断语句。条件分支会降低GPU内核心利用率，对性能产生很大影响。

本文基于AW细化算法，提出了一种新的并行策略，将模板转化为查找表(Templates-to-Lookup Table, **TTL**)，该策略能够减少分支，提高资源利用率，得到更好的加速比。后面章节详细讨论了：

1. 基于AW细化算法提出了一种新的并行策略（TTL），该策略能够很好地减少分支；

2. 从细节上讨论了加速比与图像大小、线程块大小、查找表存储位置的关系，同时还讨论了TTL策略的在其他细化算法上的加速效果。

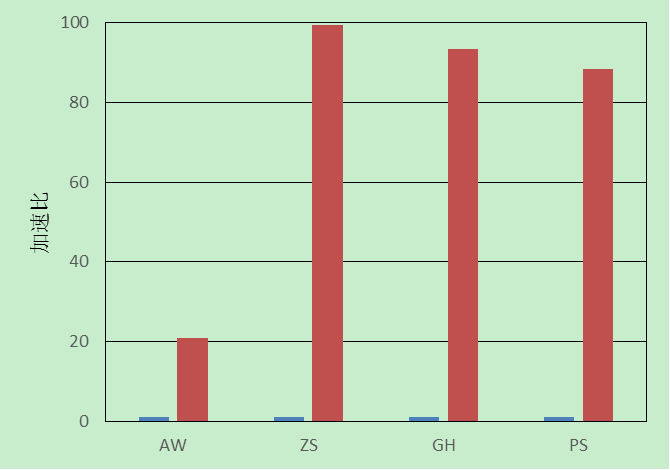


图 1 并行细化算法简单移植到GPU上的加速比。 (AW) Ahmed和Ward算法. (ZS) Zhang和Suen算法. (GH) Guo和Hall算法. (PS) Petrosino和Salvi算法.