Chapter 18
Analysis of a Step-Based Watershed Algorithm Using CUDA

Giovani Bernardes Vitor
Universidade Estadual de Campinas, Brazil

André Körbes
Universidade Estadual de Campinas, Brazil

Roberto de Alencar Lotufo
Universidade Estadual de Campinas, Brazil

Janito Vaqueiro Ferreira
Universidade Estadual de Campinas, Brazil

ABSTRACT

This paper proposes and develops a parallel algorithm for the watershed transform, with application on graphics hardware. The existing proposals are discussed and its aspects briefly analysed. The algorithm is proposed as a procedure of four steps, where each step performs a task using different approaches inspired by existing techniques. The algorithm is implemented using the CUDA libraries and its performance is measured on the GPU and compared to a sequential algorithm running on the CPU, achieving an average speed of twice the execution time of the sequential approach. This work improves on previous results of hybrid approaches and parallel algorithms with many steps of synchronisation and iterations between CPU and GPU.

DOI: 10.4018/978-1-4666-1574-8.ch018

1. INTRODUCTION

The identification of objects on images needs in most cases a pre-processing step, with algorithms based on segmentation by discontinuity or the opposite, by similarity. Segmentation itself is not a trivial task, being among the hardest ones in image processing (Gonzalez & Woods, 2002). Some of its inherent problems are illumination variation through image sequences, dynamic change of background where the camera and/or target move, as well as changes on the topology of the target or its partial or full occlusion, resulting on a higher complexity of the scene and therefore a more difficult problem.

The watershed transform is a useful tool that is a part of some segmentation processes, for tasks of region labelling. Since its introduction by Digabel and Lantuéjoul (1978) constant efforts have been made to improve its performance, in sequential architectures such as the first fast algorithm by Vincent and Soille (1991) until the recent ones by Osma-Ruiz et al. (2007) and Cousty et al. (2009), as well as in parallel, such as the framework developed by Moga et al. (1994) until the algorithm of Galilée et al. (2007). These efforts, combined with the recent dissemination of out of the box SIMD (Single Instruction Multiple Data) parallel architectures through GPUs (Graphics Process Unit) and the languages and libraries that ease its development, suggest a new approach on research for accelerating the watershed implementations.

This paper aims for the development of a watershed transform algorithm suited for GPU processing, developed with NVIDIA's CUDA (Compute Unified Device Architecture) platform, in four major steps. A fully parallel SIMD algorithm is proposed, taking advantage of the massive parallelism using specialised algorithms for each necessary step. The proposed algorithm is tested on a set of images and compared to the faster sequential algorithm that the authors built, based on the works by Lin et al. (2006) and Osma-Ruiz et al. (2007), that is very similar in function to the algorithm proposed.

2. WATERSHED TRANSFORM

Taking different approaches, the watershed transform is defined diversely on the literature, each of which producing a different set of solutions (Vincent & Soille, 1991; Meyer, 1994; Lotufo & Falcão, 2000; Bieniek & Moga, 1998; Cousty et al., 2009). The definitions are based on regional or global elements, such as influence zones and shortest-path forests with max or sum of weights of edges, or on local elements, such as the steepest descent paths, where the neighbour's information is used to create a path to the corresponding minimum. These definitions are thoroughly revised on the literature (Audigier & Lotufo, 2007; Roerdink & Meijster, 2000; Cousty et al., 2009). In this work the local condition definition is used, henceforth named LC-WT (Local Condition Watershed Transform).

The LC-WT definition was introduced by Bieniek et al. (1997) with the purpose of achieving speedups on the parallel watershed transform, once it mimics the behaviour of a drop of water on a surface, requiring less steps of global processing, and thus requiring less communication and synchronisation. Next, we discuss the algorithms that implement this definition on their sequential and parallel versions.

Sequential Watershed Algorithms

The recent faster sequential watershed transform algorithms are the result of the evolution of the arrowing technique for watershed of Bieniek and Moga (2000) and the union-find of Meijster and Roerdink (1998). The arrowing is the algorithmic representation of the drop of water, where for every pixel of an image, an arrow is drawn from the current to the next pixel creating a path that ultimately leads to a regional minimum. The arrow indicates the direction that a drop of water would slide, considering the image as a surface. For the LC-WT definition, every pixel that does not belong to regional minima will have one and only one arrow. The union-find technique for