OpenCL accelerated GPU binary morphology image filters for ITK

Release 0.01

Zoltan Bardosi[1]

October 2, 2015

[1]Medical University of Innsbruck

Abstract

Binary morphological operations are fundamental tools in image processing but the processing time scales with the number of pixels thus making them expensive operations on the CPU for larger 3D datasets that typically appear in medical imaging. Since erosion and dilatation are special neighborhood operators, each pixel in the output depends only on the neighborhood region which makes them fit for massive GPU parallelization. This document introduces a new ITK module that implements generic (OpenCL based) GPU accelerated binary morphology image filters for erosion and dilatation. The filter can be executed within the standard ITKGPU pipeline.

Latest version available at the Insight Journal [http://hdl.handle.net/10380/1338]

Contents

1 Introduction

Binary morphology is a fundamental tool in image processing, object recognition or segmentation. It has various applications in the field of medical imaging where large 3D datasets are processed. In such an environment the current ITK implementations (such as `itkBinaryMorphologyErosionImageFilter`, `itkBinaryMorphologyDilationImageFilter`) can be very time consuming as they only utilize a single

CPU core. The nature of the morphological computations make them fit for GPU based parallel computations. Though GPU based implementations of the morphological operations already exist (e.g. in CUDA-Insight Toolkit), they are not available in the standard ITK installation and are based on the proprietary CUDA architecture that restricts their usage to a specific set of hardware. The base ITK library on the other hand has an OpenCL based based pipeline included for GPU based images represented by the `GPUImage` class. (OpenCL aims to be a royalty-free and cross platform standard for GPU based parallel computing.) The proposed filters fit into this existing GPU pipeline and can be used as drop-in replacements of the CPU implementations.

2 Using the GPU accelerated morphology filters

The **GPUBinaryMorphology** module provides two new classes (`itkGPUBinaryErosionImageFilter` and `itkGPUBinaryDilationImageFilter`). They provide simple but effective implementations of binary morphological erosion and dilation that can be used in the standard ITKGPU filtering pipeline. They are applicable to 1, 2 or 3 dimensional images. This code snippet shows a typical application:

```
typedef itk::GPUImage< unsigned char, 3 > GPUImageType;
typedef itk::GPUBinaryErosionImageFilter<GPUImageType, GPUImageType> ErosionFilterType;
typedef itk::BinaryBallStructuringElement< PType, dim > SRType;

SRType kernel;
ErosionFilterType::Pointer filter = ErosionFilterType::New();
GPUImageType::Pointer image = ImageType::New();

...

filter->SetInput(image);
filter->SetKernel(kernel);
filter->SetErodeValue(255);
filter->SetBackgroundValue(0);
filter->SetBoundaryToForeground(false);
filter->Update();
```

3 Speed impovement

To evaluate the increase in performance the new GPU implementations were tested on a computer tomography volumetric dataset (171 slices, 512x512 pixels in each slice). The GPUImage class was instantiated with short (2 byte) pixel type. A 3D ball structuring element with a radius of 2x2x2 pixels was used.The platform for the test was an Intel i7-2600K CPU running at 3.4Ghz and an NVidia GTX 970 (MSI OC edition) GPU. Each measurement was repeated 10 times to give a reliable average time estimate. Figure 1. shows the results. The speedup in this case is 12-20x on the GPU.

Operation	Platform	Time (seconds +- std)
Erosion	CPU	5.642 ± 0.297
Erosion	GPU	0.248 ± 0.018
Dilation	CPU	2.723 ± 0.037
Dilation	GPU	0.224 ± 0.098

Figure 1: Processing times for a 512x512x171 3D dataset