AMD Accelerated Parallel Processing TECHNOLOGY

SAMPLE

Recursive Gaussian

1 Overview

1.1 Location

$<*APPSDKSamplesInstallPath*>\samples\opencl\cl\

1.2 How to Run

See the *Getting Started* guide for how to build samples. You first must compile the sample.

Use the command line to change to the directory where the executable is located. The pre-compiled sample executable is at `$<APPSDKSamplesInstallPath>\samples\opencl\bin\x86\` for 32-bit builds, and at `$<APPSDKSamplesInstallPath>\samples\opencl\bin\x86_64\` for 64-bit builds. Ensure that the OpenCL 2.0 environment is installed.

Type the following command(s).

1. `RecursiveGaussian_ProgramScope`
   This applies recursive Gaussian filter on input image.
2. `RecursiveGaussian_ProgramScope -h`
   This prints the help message.

1.3 Command Line Options

Table 1 lists, and briefly describes, the command line options.

Table 1 Command Line Options

Short Form	Long Form	Description
-h	--help	Shows all command options and their respective meaning.
	--device	Devices on which the program is to be run. Acceptable values are cpu or gpu.
-q	--quiet	Quiet mode. Suppresses all text output.
-e	--verify	Verify results against reference implementation.
-t	--timing	Print timing.
	--dump	Dump binary image for all devices.
	--load	Load binary image and execute on device.
	--flags	Specify compiler flags to build the kernel.
-p	--platformId	Select platformId to be used (0 to N-1, where N is the number of available platforms).
-d	--deviceId	Select deviceId to be used (0 to N-1, where N is the number of available devices).
-v	--version	AMD APP SDK version string.
-i	--iterations	Number of iterations for kernel execution.

2 Introduction

This sample demonstrates the use of Program Scope Variables, a new feature of OpenCL 2.0. Program scope variables can be defined either in the constant address space or in the global address space and their scope is the program lifetime. These variables could be shared by more than one compute unit. Program scope variables are demonstrated in this sample by using the Recursive Gaussian filter implementation.

The Gaussian filter is a filter whose impulse response is a Gaussian function. It has the minimum possible group delay. Gaussian filter modifies the input signal by convolution with a Gaussian function. The advantages of using the Gaussian filter are:

- Combined effect of differentiation and low pass filtering yields a filter with a smooth impulse response.
- Used to compute running averages.

The Gaussian function is non-zero for $x \in [-\infty, \infty]$ and would theoretically require an infinite window length. However, since it decays rapidly, it is often reasonable to truncate the filter window and implement the filter directly for narrow windows, in effect by using a simple rectangular window function.

The Gaussian filters can be implemented using scale space windowing technique. In this sample, Gaussian filter implementation is done using Recursive filters which is a type of scale space windowing technique.

3 Implementation

The Recursive Gaussian filter is implemented using two kernels, one a RecursiveGaussian Kernel, and another, Transpose Kernel.

In the RecursiveGaussian Kernel, both Forward and reverse filter are used to ensure the response is symmetrical.

The Forward filter is as follows,:

```
Yf(m,n) = a0 * X(m,n) + a1 * X(m,n-1) - b1*Yf(m,n-1) – b2 * Yf(m,n-2),
```

The Reverse filter is as follows:

```
Yr(m,n) = a2 * X(m,n) + a3 * X(m,n+1) - b1*Yr(m,n+1) – b2 * Yr(m,n+2),
```

where a0, a1, a2, a3 , b0 andb1 are Gaussian filter parameters. X is the input image pixels and Yf, Yr is the output of the forward and reverse recursive filter respectively, and the m and n axis are parallel to width and height respectively.

The Final Output will be :

```
Y(m,n)  =    Yf(m,n)      +   Yr(m,n).
```

After the execution of the Recursive Gaussian Kernel, data is transposed using transpose kernel. Further, Recursive Gaussian kernel is applied on the transposed image to filter in the other direction. Finally again transpose kernel is applied to bring back the filtered image to original position. In all these data transitions, the intermediate results of the Recursive Gaussian kernel are held in a program scope array.

4 References

1. http://en.wikipedia.org/wiki/Gaussian_filter
2. http://en.wikipedia.org/wiki/Scale_space_implementation
3. http://www.cwp.mines.edu/Meetings/Project06/cwp546.pdf
4. http://homepage.tudelft.nl/e3q6n/publications/1998/ICPR98LVTYPV/ICPR98LVTYPV.pdf

Contact

Advanced Micro Devices, Inc.
One AMD Place
P.O. Box 3453
Sunnyvale, CA, 94088-3453
Phone: +1.408.749.4000

For AMD Accelerated Parallel Processing:
URL: developer.amd.com/appsdk
Developing: developer.amd.com/
Support: developer.amd.com/appsdksupport
Forum: developer.amd.com/openclforum