CUVILib — CUDA for Vision and Imaging Library

Tauseef Rehman, Salman Ul Haq, Usman Aziz, Jawad Masood
{tauseef, salman, usman, jawad}@cuvilib.com

GPU TECHNOLOGY CONFERENCE

INTRODUCTION

CUDA for Vision and Imaging (CUVI) is a software library that provides a set of GPU accelerated computer vision and image processing functions. CUVI can both be utilized as an add-on library to the NVIDIA's NPP (NVIDIA Performance Primitives) as it compliments the functionality present in NPP as well as it can be used as a standalone library ready to be plugged into end-user C/C++ applications.

CUVI comes with an intuitive interface making it easy to integrate into existing applications. Most function calls mimic the behaviour of equivalent functions available in either IPP or OpenCV wherever possible. Programmers familiar with either would find it extremely easy to incorporate these accelerated functions into their exsitng code. CUVI supports all CUDA runtime versions including 1.0 to 2.0 (Fermi).

CUVI Beta is availabe for FREE download at www.cuvilib.com/downloads. The distribution package includes detailed documentation, function binaries , library files, and several samples. CUVILib is supported on both Windows and Linux operating systems (32 and 64 bit variants). Multi-GPU support is also going to become available in future releases.

VISION AND IMAGING LIBRARIES

Background

Computer Vision and Imaging algorithms deal with data in the form of Images and Video that translate into large 2D and 3D data structures and therefore, generally have inherent data parallelism. Therefore, these can take advantage of the SIMD/Many-core architecture of a GPU.

CUVI Lib and Existing Libraries

The two most popular computer vision and image processing libraries are Intel's IPPI (IPP Imaging) and OpenCV. IPPI is optimized for the Intel's processors and is a commercial library while OpenCV is the open source counterpart with over 40,000 active users. With CUVI-Lib we want to bridge the gap between NVIDIA's NPP and these libraries. Additionally, several state of the art algorithms coming out of current research are not yet part of these libraries and as indicated by CUVI-Lib's functional support listed below we want the state of the art available and optimized for accelerated execution on GPUs.

GRAPHICS FUNCTIONALITY

OpenGL is used to render PBO (Pixel Buffer Objects) and VBO (Vertex Buffer Objects) from the GPU's DRAM. CUDA data pointers can be mapped to OpenGL objects to visualize intermediete or final results. This saves the overhead of porting data back to host for visualizing. OpenGL can be set to work on different GPU then the one in which CUDA data is written.

1) 2D images (OpenGL's Pixel Buffer Object)
2) 3D surface plots (OpenGL's Vertex Buffer Objects)
3) Quiver Plots (For rendering Optical Flow's velocity vectors)

CODE SAMPLE

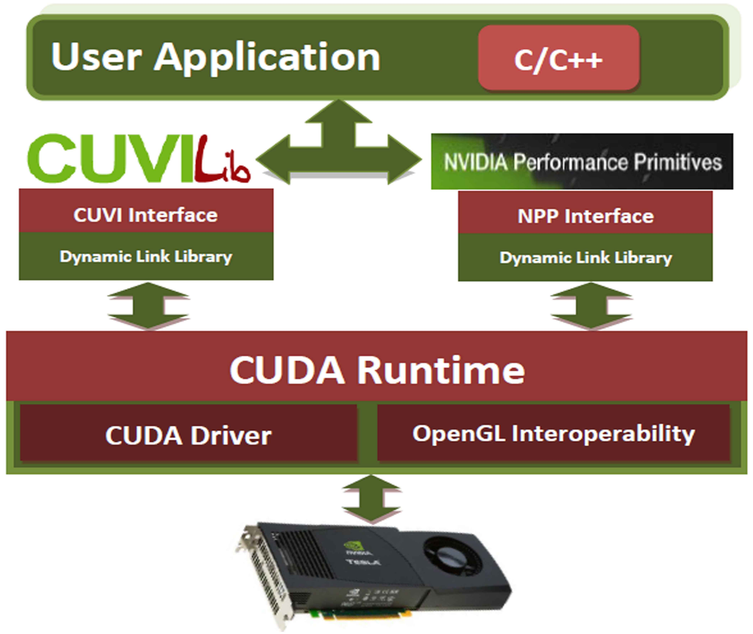
Compute Optical Flow (LK) velocity vectors of 2 frames:

```
/*Read 2 frames using NPP's FreeImage utility functions*/
FIBITMAP* frame0 = FreeImage_Load(FIF_PGM, "absolute_image1_location");
FIBITMAP* frame1 = FreeImage_Load(FIF_PGM, "absolute_image2_location");

/*Fetch image dimensions, pitch and data pointer*/
int h_width  = FreeImage_GetWidth(frame0);
int h_height= FreeImage_GetHeight(frame0);
int h_pitch    = FreeImage_GetPitch(frame0);
unsigned char *imgA  = FreeImage_GetBits(frame0);
/*Repeat for frame1*/

/*Allocate memory for velocity vectors*/
float *velocityX = (float*)malloc(image_size);
float *velocityY = (float*)malloc(image_size);

/*Specify ROI*/
NppiSize ROI = {h_width,h_height};

/*Call Optical Flow (LK) function*/
cuvi::lib::localOpticalFlowLK_8u_C1(imgA, imgB, pitch, ROI, winSize, velocityX, velocityY, velocityPitch);
```

Note: *As you can note from the sample code, you do not have to allocate memory on the GPU since CUVI functions does that and also port the image data to GPU's DRAM. This simplifies the programming effort.*

All the temporary GPU memory used by the CUVI functions is freed before the function returns. This has to be done since there is no automatic garbage collection in GPUs.

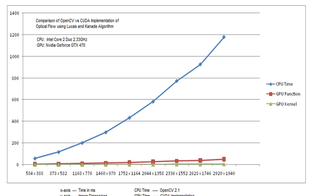
CUVI ARCHITECTURE

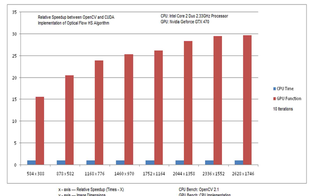


BENCHMARKS

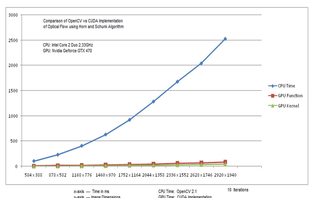
CPU: Intel Core 2 Duo 2.33 GHz
GPU: NVIDIA GeForce GTX 470 (Fermi)

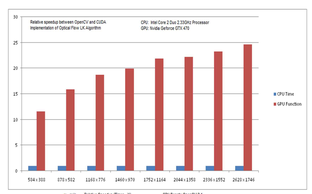
Optical Flow (Lucas & Kanade)



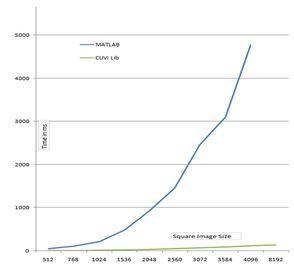


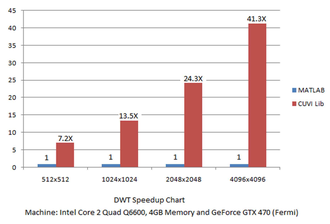
Optical Flow (Horn & Schunk)





DWT (Haar)





DWT Speedup Chart
Machine: Intel Core 2 Quad Q6600, 4GB Memory and GeForce GTX 470 (Fermi)

http://www.cuvilib.com

FUNCTIONALITY

CUVI-Lib presently supports the following:

1) Optical Flow (Lucas & Kanade)
2) Optical Flow (Horn & Schunk)
3) Standard Hough Transform
4) Hough Lines
5) Forward/Inverse Discrete Wavelet Transform (Haar)
6) Color Space Conversions
   a) RGB to Grayscale (Tripple Channel 8bit uchar to Greyscale 8bit uchar)
   b) RGBA to Grayscale (Four Channel 8bit uchar to Greyscale 8bit uchar)
   c) RGB2YUV
   d) Image Gamma Correction

The following functions will be released in the upcoming releases:

Image Segmentation
   1) 2D & 3D Levelsets
   2) 2D & 3D Graphcuts
   3) FloodFill
   4) Watershed

Feature Detectors
   1) SIFT
   2) SURF
   3) MSERs
   4) KLT/Harris Corners

SAMPLES