GPU-BASED SIGNAL PROCESSING SCHEME FOR BIOINSPIRED OPTICAL FLOW

Fermin Ayuso, Carlos García, Guillermo Botella, Manuel Prieto, Francisco Tirado

Dept. of Computer Architecture and Automation, Complutense University of Madrid

ABSTRACT

The aim of this work contribution is the neuromorphic low-power GPU implementation of the processing stages for robust and multichannel optical flow estimation that permits highly parallel real-time filtering.

Key Words: GPUs, Digital Signal Processing, Vision

1. INTRODUCTION

The graphic processor units (GPU), are available at low cost due the evolution of the entertainment industry. This devices are designed as multicore systems, with a complex memory hierarchy. These platforms are designed to use the high data parallelization degree when rendering or constructing 3D scenes.

Nevertheless these systems can be used nowadays as parallel processors executing a high number of threads simultaneously. As example a NVIDIA chip Tesla C2070 reaches up a maximum throughput of 1,28 Tera FLOPs in comparison with a processor Intel i7-975 which only completes 55 GigaFLOPs. This notably processing throughput has got attention from scientific and technical community belong to many areas, using actually GPUS for self accelerating applications.

Our starting point is evaluating the possibilities of GPUs in computer visión, specifically focusing in motion estimation. The aim of this work will be design three pre-processing motion steps common to many gradient optical flow models such with the final goal to develop the Multichannel Gradient Model [1]. This scheme will be designed in a optimal and efficient way taking advance of the specific architecture and the memory model of GPUs platforms [2].

2. TEMPORAL, SPATIAL AND STEER FILTERING

2.1 Temporal domain filtering

The experiments carried out by Hess and Snowden in 1992 evidenced that the human visual cortex has three different channels for temporal filtering: a low-pass and two band-pass (centered around 10 and 18 Hz).

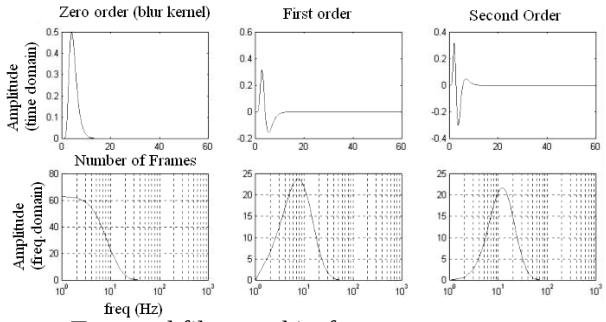

Fig. 1. Temporal filters and its frequency response.

2.2 Spatial filtering

Early experiments have shown that receptive fields of the cells in the primary visual cortex can be modeled as derivatives of Gaussians of several orders. The result is a range of independent spatial channels working in parallel to enhance spatial variations of different magnitudes.

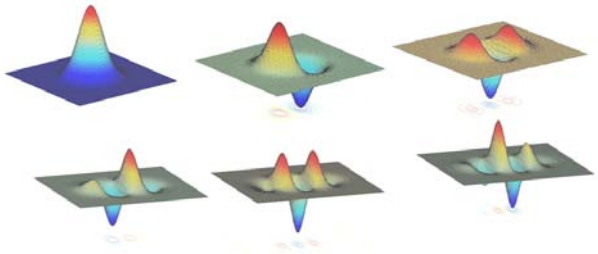

Fig. 2. Range of spatial filters used in this work.

2.3 Steering filters

Steering algorithms can synthesize filters at arbitrary orientations from a linear combination of other filters in small “basis” set. If the number of filters in the basis set is relatively small then the computation can be highly efficient compared to direct convolution with many oriented filters. Oriented filtering is typically avoided in real-time systems due to the computational expense of performing multiple convolutions with a bank of oriented filters, nevertheless we have developed faster implementations for oriented filters that have been shown to improve the efficiency considerably.

$$s \cdot \cos(\theta)^2 + s \cdot \sin(\theta)^2 + s \cdot 2\cos(\theta)\sin(\theta) = Steer(s) \qquad (1)$$

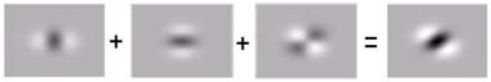

Fig. 3. Scheme of first order steering filter (45º).

A further benefit of the steering algorithm can be gained from the linearity of the convolution operation. Therefore, we can interchangeably discuss the synthesis of oriented filter from a basis set of filters, and the synthesis of oriented filters responses from a basis set of responses. Furthermore, forming a weighted addition of the spatially sampled basis functions is equivalent to spatially sampling the weighted sum of continuous basis functions.

3. SIMULATION ENVIRONMENT

The system used is based on Tesla technology, it has 2 processors Intel Xeon E5530 with 4 cores (2.40 GHz with 8MB cache memory and Hyperthreading technology, connecting to 4 GPUs Tesla C1060). The operating system is Debian 2.6.38 kernel, the compiler is g++ GNU v.4.5.2 using the compilation options *-O3 -m64* and is used also the CUDA v.2.3. This graphic card has a CUDA capability 1.3 which means it has 240 processing nucleus with 1024 *threads* for each multiprocessor. The hierarchy of memory is 4GB for global memory, 16KB for shared memory and 64KB for memory of constants.

4. RESULTS

There are shown the throughput gained using GPUS as accelerators.

First analysis is to measure the speedup in the temporal filtering. We have performed several implementations:

- Base: starting point where all input frames and info information are saved in global memory.
- Global: all input frames are saved in global but filter is kept in memory of constants.
- Shared: The *nL* frames to be filtered are saved in shared memory and filter remain in memory of constants.
- Shared-optimized: The same as previous case but constructing a circular buffer structure. a) In t=0 is initialized a set of frames from 1 to *nL*+1 and b) in t=1 and next frames only is overwritten the content.

As shown in Fig. 4 the best results are measured when temporal filters using Global configuration. Fig. 5 shows up the throughput with spatial filters of 4th order and a size 7, 9, 15, 31 points. The fig. 6 remarks the results varying the rotation every 60º, 30º y 15º angles. The orientations' number and the performance obtained are directly proportional, resulting the best configuration the most complete one (24 angles and greatest frame size).

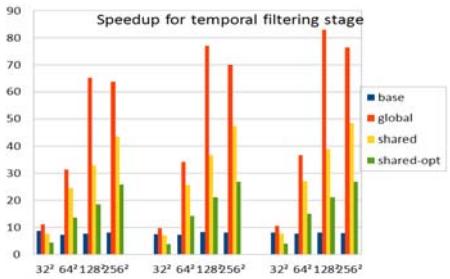

Fig. 4. Throughput of the Temporal Filtering.

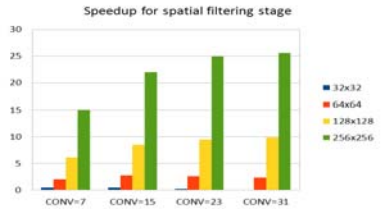

Fig. 5. Throughput of the Spatial Filtering .

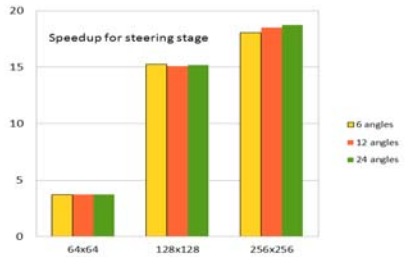

Fig. 6. Throughput of the Steering Filtering.

5. CONCLUSIONS & FUTURE LINES

At the present time, it has been built the architecture for a bio-inspired kernel of space-temporal filtering in a massively parallel implementation. This module is especially useful for robust processing of information required for artificial vision, especially optical flow. The future goal is to extend the number of stages processed in GPU considering color distinction and binocular disparity to compare with previous FPGA-based works [3].

6. REFERENCES

[1] A. Johnston, P.W. McOwan, C.P. Benton, "Biological computation of image motion from flows over boundaries". J Physiol. Paris, vol. 97, pp. 325-334, 2003

[2] [on-line] http://developer.nvidia.com/cuda-downloads

[3] G. Botella, A. García, M. Rodríguez, E. Ros, U. Meyer-Baese, M.C.Molina. "Robust Bioinspired Architecture for Optical-Flow Computation". IEEE Trans. VLSI Syst. 18(4): 616-629 (2010)