Abstract

General purpose graphic processing unit (GPGPU) provides high performance resource for computing. CUDA (Compute Unified Device Architecture) and OpenCL (Open Computing Language) permit writing of parallel computing programs that utilize multiple central processing units (CPU) and GPGPUs. The image processing library, OpenCV (Open Source Computer Vision library), may benefit greatly from parallel use of multiple GPGPUs, however, its CUDA implementation is restricted to benefiting from a single GPGPU only. This research develops an abstraction layer above OpenCV single GPU module that enables multiple GPUs for single instruction multiple data (SIMD) architecture. This approach has a controller/parent thread which generates various worker threads to operate on several GPU devices, to handle balancing of work load on GPUs, as the task allocation is dynamic for any number of GPUs. The experiments on running bilateral filtering, color to gray conversion, fast Fourier transform, and convolution on homogeneous and heterogeneous sized images of scenery, objects, and faces, indicate that: (1) threading reduces computation time by half of sequential operation for GPU; (2) tuned static load balanced GPU threading reduces computation time by up to a fourth
when compared to CPU threading; (3) performance of dynamic load balancing approaches that of manually iteratively balanced static operation.

References

13. OpenCV Test data. [online] Available at: https://github.com/itseez/opencv_extra.

Index Terms

Computer Science  Information Systems

Keywords
Efficient Dynamic Multiple GPGPU Layer for OpenCV

GPGPU, OpenCV, SIMD, CUDA, OpenCL, Multiple GPU, Load Balancing, Threading.