# GPGP: General Purpose Computations using Graphics Processors

Naga K. Govindaraju, Ming Lin, Dinesh Manocha
University of North Carolina at Chapel Hill
{naga,lin,dm}@cs.unc.edu

## Introduction

In the last decade, high-performance 3D graphics hardware has become as ubiquitous as floating-point hardware. Graphics processors (GPUs) are now a part of almost every personal computer, game console, or workstation. In fact, the two major computational components of a desktop are the CPU and the GPU. Modern GPUs feature programmable vertex and fragment processors. The GPUs can be thought of as a particular kind of stream processors, which operate on data streams consisting of an ordered sequence of attributed primitives like vertices or fragments. As compared to conventional CPUs, GPUs consist of high-bandwidth memories and more floating-point hardware units. For example, current top of the line GPU, such as NVIDIA 6800 Ultra, has peak performance of 45 GFLOPS and memory bandwidth of 36 GB/sec, as compared to 12 GFLOPS and 6 GB/sec, respectively, for a 3 GHz, Pentium IV CPU. Furthermore, the GPUs performance for graphics applications has been growing at a rate of 2.- 3 times a year, which is faster than the Moore's Law for CPUs. For example, the recently announced Sony Playstation 3 console has a programmable GPU known as "Reality Synthesizer" with a peak performance of 1.8 TFLOPS and a cell processor with a peak performance of 218 GFLOPS. Moreover, desktop and laptop systems with multiple commodity GPUs are also becoming available.

Our goal is to exploit the computational power of GPU for many scientific, database and geometric applications. GPUs are primarily designed for the rapid rasterization of geometric primitives to shaded pixels on the screen. In order to utilize the computational power and memory bandwidth available in the GPUs, we have designed novel GPU-based algorithms for a variety of problems that utilize the rasterization capabilities. The set of problems include database and data mining queries, linear algebra computations, sorting and fast fourier transformations, motion planning and navigation, simulations of physical phenomena including fluids, and geometric algorithms. In all these applications, the performance of the underlying algorithms is a direct function of rasterization performance and is growing at a faster than Moore's Law on successive generation of GPUs.

Our GPU-based algorithms use the inherent pipelining and parallelism, single instruction and multiple data (SIMD) capabilities, and the vector processing functionalities of the graphics processors to perform the computations efficiently. Our algorithms account for relatively low bandwidth available between the CPU and the GPU, and perform a large fraction of the computation efficiently on the GPU. We also take into account the poor performance of programming constructs such as branching instructions in the programmable GPU pipeline, and use alternate strategies for efficiently evaluating the computations such as blending-based conditional assignments Finally, we have analyzed the memory access behaviors of these computations on the GPUs and designed cache-efficient algorithms for improved performance.

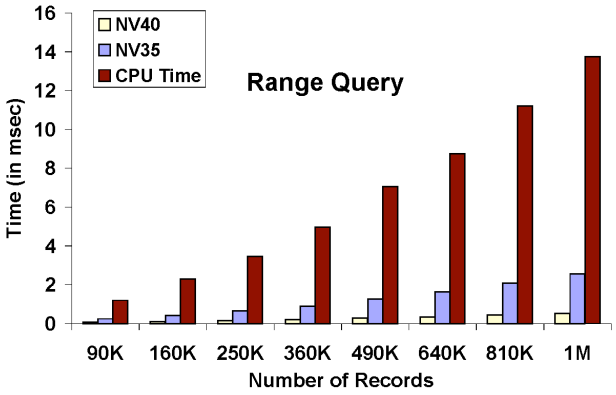


**Figure 1: This graph highlights the performance of a database range query on a NVIDIA GeForce FX 6800 Ultra GPU (NV40), NVIDIA GeForce FX 5900 Ultra GPU (NV35), and a 2.8GHz Intel Xeon processor. We observe that our GPU-based algorithm exhibits nearly 5 times performance improvement between two successive generations of GPUs, NV35 and NV40.**

## Applications

In this section, we briefly describe many of our GPU-based algorithms and their applications.

## Database Algorithms

Database management systems are an integral part of many data warehousing applications and often demand high processing power for fast query execution. We have designed new algorithms to perform fast relational database operations on GPUs [2]. These include predicates, boolean combinations, selectivity and aggregation queries, and join queries. Our algorithms have been implemented using simple fragment programs and applied to databases with up to a million records. We have performed comparisons with SSE optimized CPU algorithms on a 2.8 GHz Xeon processor. Our results indicate a performance improvement of 5-20 times using our GPU-accelerated database queries on a NVIDIA GeForceFX 6800 Ultra GPU.

## Stream Mining Algorithms

Many real-world applications such as sensor networks, network and financial monitors, online transaction trackers analyze large volumes of data streams, usually collected from different sources. In these data streaming applications, each data element has to be processed in real-time, and used for estimating the frequency of the element etc. Due to the limited memory requirements, and the need for computational resources, the underlying CPU is usually resource-limited. We use the fast stream processing capabilities of the GPUs for estimating the frequencies and quantiles in large data streams. We present a fast sorting algorithm on GPUs [3] and use it for histogram computations. Our sorting algorithm uses the blending and texture mapping hardware on GPUs to efficiently perform comparisons and comparator mapping on GPUs. We have obtained a speedup of a factor of 2 over optimized CPU algorithms on high end PCs.

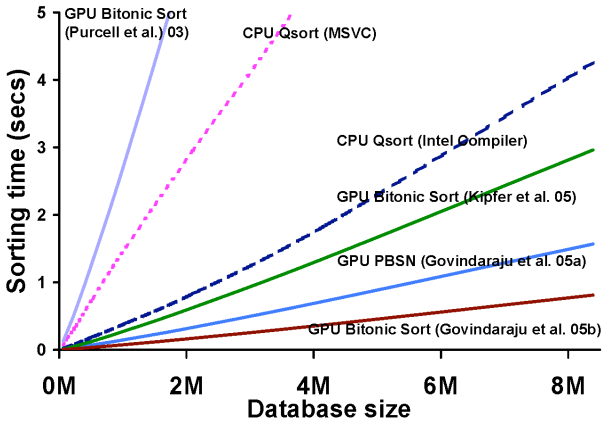


**Figure 2: This graph highlights the performance of CPU and GPU sorting algorithms on a NV40 GPU and a 3.4 GHz Intel Pentium IV CPU. We observe that our bitonic sorting algorithm is nearly 6 times faster than Quicksort on CPUs.**

## Linear Algebra Computations

LINPACK is a popular benchmark used in the TOP500. Dense linear algebra solvers have been studied for several decades and used in many high performance simulations such as fluids. We have mapped two direct linear system solvers on GPUs – LU decomposition and Gaussian elimination using efficient GPU representations and analyzed their performance [1]. Our algorithms achieve the peak memory bandwidth available on GPUs and the performance is comparable with the cache- and SSE-optimized CPU implementations such as ATLAS.

## Geometric Algorithms

Geometric algorithms are fundamental in the design, visualization, and engineering of automobiles, virtual reality and CAD/CAM applications, spatial queries for geological information systems, etc. We have developed new GPU-based algorithms for performing voronoi computations [9], proximity and intersection computations [6, 8], transparency and shadow generation algorithms, distance fields, and line-of-sight computation algorithms. In many cases, we are able to obtain more than one order of magnitude improvement over CPU-based algorithms, e.g. collision detection between deformable models and 3D distance field computations.

## Motion Planning and Navigation

Motion planning is a classis problem in robotics. Its applications include virtual prototyping, surgical simulation, navigation in virtual environments and computer animation. We have designed novel motion plannings algorithms for dynamic environments and deformable robots in complex environments [7,8]. These algorithms compute a discrete approximation of the distance field using the rasterization hardware and significantly outperform prior CPU-based algorithms.

## Scientific Computations

GPU-based algorithms have been designed for a variety of scientific applications including fluid simulation [4], phase field methods [5] as well as solving deformable models. In many of these cases, the underlying numerical computations are reduced to running fragment programs on operands represented in the texture memory.

There is considerable interest in building GPU clusters to solve many high-performance computations such as molecular dynamics, cloth simulation and finite-element simulations.

## References


[1] N. Galoppo, M. Henson, N. Govindaraju, D. Manocha, *LU-GPU: Efficient Algorithms for Solving Dense Linear Systems on Graphics Hardware*, Technical report, UNC Chapel Hill, 2005.

[2] N. Govindaraju, B. Lloyd, W. Wang, M. Lin, D. Manocha, *Fast Database Operations Using Graphics Processors in* Proc. of ACM SIGMOD 2004.

[3] N. Govindaraju, N. Raghuvanshi, D. Manocha, *Fast and Approximate Stream Mining of Quantiles and Frequencies Using Graphics Processors in* Proc. of ACM SIGMOD 2005.

[4] B. Baxter, Y. Liu and M. C. Lin, *A Viscous Pain Model for Interactive Applications,* Journal of Computer Animation and Virtual Worlds, 2004.

[5] T. Kim and M. Lin, *Visual Simulation of Ice Crystal Growth in* Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation, 2003.

[6] N. Govindaraju, M. Lin, D. Manocha, *Fast and Reliable Collision Culling using Graphics Processors in* Proc. of ACM Virtual Reality Software and Technology, 2004.

[7] R Gayle. P. Segars, M. Lin, D. Manocha, *Path Planning for Deformable Robots in Complex Environments,* Proc. of Robotics: Science and System, 2005.

[8] A. Sud, M. Otaduy, D. Manocha, *DiFi: Fast 3D Distance Field Computation using Graphics Hardware in* Proc. of Eurographics, 2004.

[9] K. Hoff, J. Keyser, T. Culver, M. Lin and D. Manocha *Fast Computation of Generalized Voronoi Diagrams using Graphics Hardware*, Proc. Of ACM SIGGRAPH, 1999.