Alias Cloth Technology Demonstration for the Cell Processor

Overview

This technology demonstration shows a prototype of a next generation cloth solver algorithm. It is a proof-of-concept that demonstrates the potential for interactivity and realistic display of cloth behaviors. The demonstration shows independent simulations being added, one by one, with no degradation in performance. Each cloth tumbles within a cube, in some cases with pinned constraints or objects, to demonstrate natural folding and correct behaviour in response to forces such as gravity and collisions. Each simulation can be manipulated interactively while the others continue to tumble. The demonstration runs in a client-server configuration where the cloth simulation computations are executed on one or more Cell processors running the Linux® operating system (the server) and the results of those computations are sent to an Apple® Power Mac® G5 client for display.

The cloth solver algorithm is a completely novel approach under development by researchers at Alias. It can provide a stable and fast simulation of cloth behavior with full handling of self-intersection, collisions with moving objects, and links constraints. The algorithm allows for a wide range of cloth types to be simulated at near interactive rates even on traditional CPUs.

Porting to the Cell Architecture

The most notable features of the Cell processor architecture are the IBM® Power-based RISC CPU core (PPE), the 8 SIMD vector units (SPEs) and the impressive bandwidth that can be leveraged to obtain maximum performance from the architecture.

In order to port the prototype cloth simulator to the Cell processor, the code and data structures had to be restructured. First, changes had to be made to allow for parallelization of the algorithm. Even for more conventional multi-core targets, this is a challenging process. With the Cell architecture, there is the additional requirement to manage memory transfers between the main processor (the PPE) and the SPEs. Second, where feasible, scalar code was re-written to take advantage of the vector support in both the PPE and the SIMD units.

We were able to organize the code such that multiple instances of the cloth solver are able to run on the server-side with each instance running several simulations concurrently. Each of the simulations approximates the physically correct movement of a 3-dimensional mesh with collision with flat surfaces, self-collision and forces acting on it such as gravity [Figure 1].
Each simulation then produces a stream of data that encodes the updated positions of the mesh vertices and sends it to the rendering client. The client-side of the application receives the streams of data coming from one or more Cell processors, reconstructs the mesh and generates the display.

![Figure 1](image)

**Observations on Cell Processor Performance**

In order to extract the maximum performance from the Cell processor, the vector units should be leveraged extensively. Although SPEs can run generic code at a comparable speed or faster than the PPE, the SIMD nature of their architecture performs at its best when the code it executes is structured to use vector intrinsics that process multiple integer or floating point operands with every instruction.

**Considerations on bandwidth and usage of DMA engine**

Each of the 8 SPE units can access 256KB of local uncached high-speed RAM (local storage), which is shared between code and data. All accesses to and from main RAM go through DMA transfers, which either the PPE or the SPEs can initiate.

The size of local storage calls for an intensive use of the Cell processor’s DMA engine (the MFC). Our tests show that memory can be moved to and from local storage at truly impressive rates. In one experiment, a series of blocking DMA transfers approximately 4KB each was replaced with interleaved double-buffered DMA transfers and the overall performance difference
was negligible. For larger volume transfers the benefit of using double or multi-buffering schemes becomes more apparent and is recommended.

The MFC uses several DMA request queues to avoid stalling the various units: the PPE DMA request queue size is 8, and each SPE DMA request queue size is 16. The difference in the size of the queues and the fact that the SPEs can request DMA transfers without PPE supervision makes it more convenient to initiate most of the transfers from the SPE side. The MFC handles DMA transfers up to 16KB in size, but can process several chained transfers from non-contiguous memory to local storage and vice-versa using a DMA list concept where various DMA commands are linked together. This enables usage of very effective scatter-gather algorithms.

The PPE cache hierarchy can interfere with the DMA speed to maintain coherency, so it is good practice to flush the cache lines that have to be transferred from main RAM.

Performance Results

The cloth simulation demonstration takes advantage of the scaleable parallel processing capabilities of the Cell processor by running eight instances of the simulation per Cell processor simultaneously, one on each of a Cell processor’s eight SPEs. In addition, since the cloth simulation algorithm requires complex mathematics and processes large matrices of data, we were able to leverage the vector support of the Cell processor and achieve even greater performance by vectorizing the code that performs arithmetic functions on the mesh data structures.

At the current stage of our experiments, a prototype 2.4 GHz Cell processor runs approximately five times as many simulation frames per second as a 3.6 GHz Pentium 4 class processor [Figure 2].

The demonstration executables were built using IBM’s XLC compilers for the PPE and the SPE.
Acknowledgements

Alias would like to acknowledge and thank IBM Research for providing access to the Cell Processor Based Blade prototype hardware and development tools that enabled the development of this technology demonstration.

Contact Information

For additional information about this technology demonstration, please contact Joyce Janczyn, jjanczyn@alias.com. For information about Alias products and services, please visit http://www.alias.com.