Agent Based GPU, a Real-Time 3D Simulation and Interactive Visualisation Framework for Massive Agent Based Modelling on the GPU

Contribution

* GPU is “an ideal solution to simulate and visualise the behaviour of high population ABM” - Agent Based Modelling.
* CPU is primarily effective with a small set of population and when agents need to communicate with one another in the system, as the population grows performance is showing.
* GPU allows parallel processing
* Significant performance increases take place due to the avoidance of data transfers between the simulation and rendering (execution) stages.
* GPU popularity growing
* Programmers face difficulties since they need to know how the underlying hardware works and designed in detail, before being able to achieve performance increases.
* At its best the GPU would become useful if we can hide the underlying graphical concept and be able to directly access memory variables without the explicit knowledge of the underlying storage mechanisms.
* Future work to be extended on the current work by applying the findings to multiple GPUs on a single machine or a high performance GPU grid.

Framework

* Previous GPU work done by other scientists such as Reynolds and Erra
* Trying to minimise processing agent communication by not having to communicate with every other agent in the system
* Understand each interaction radius
* Reduce bottlenecks transfers from CPU to GPU

Use-Case

This defines a list of actions and interactions between a role and a system, to achieve a goal. The actor can be a human or other external system.

In the case of this paper, the authors present how the Graphics Processing Unit (GPU) is the ideal way to simulate and visualise the behaviour of high population Agent Based Modelling.

Thus, Use-Case are not applicable in this.

If however, the paper showed GPU architecture vs. the CPU’s and its role in assisting Agent Based Modelling, then the ‘actor’ would have been the GPU itself.

Tools / Techniques

* APIs to the GPU similar to libraries already existing for the CPU.
* The ‘experiments’ of the Boids Model and analysis presented have been done on a single PC with an AMD Athlon 2.51 Ghz Dual Core Processor with 3GB of RAM and GeForce 8800 GT.
* Simulate agents communication using a set of C++ classes and an agent update script, which is a “C like syntax and can be compiled with a C header file which contains placeholders for key communication functions”.

Theories

* Spatial partitions of the environment are created present how agents can be mapped to the GPU and how they can interact and communicate well with one another - this saves time during the communication
* The spatial partitions are compared to other work done by Erra who looked at sorting algorithms taking place on the CPU, e.g. GPUSort
* Calculating the communication radius of the agents helps understand the overall processing time.
* Significant performance increases take place by reducing the “unnecessary communication by distant agents” (same principal as building high speed Ethernet network).
* Raynolds looked at PS3 architecture to see if distributed parallel systems (although different to the GPU) can benefit agents communication in similar environments.
* Boids Model demonstrates the functionality of the ABGPU based on Reynolds *goal rule* - global variables that set the agents positions after each simulation provided they are above a threshold – otherwise, they will move to new positions within the boundaries.
* The problem with Reynolds approach is that his design uses Global variables. The use of Global Variables is inefficient and causes copying of large objects from one area of memory to another. One of my questions would be will an increase in performance take place in case he used Pointers when setting the number of weights which control each of the Boids rules ?

Critic

* Paper assumes a great deal of background knowledge by the reader
* Aimed at very specific audience.
* Paper discusses areas which are not necessary for the purpose of the article causing the paper to be longer than necessary. For example, the comparisons to Reynolds and Erra work on Boids Model and PS3 are explained in too much details – there is no need to explain the architecture of PS3 going into detail about NVIDIA RSX card and the additional IBM Cell Microprocessor which is capable of scheduling 8 parallel Synergistic Processing Units (SPU’s). A short explanation would have been sufficient.
* Repetition - references to Flocks, Fish, Birds – the principal is the same – all of those are groups of agents – a description of one group and results on the GPU would have made the areas covered less overwhelming than three (or four when later referring to humans).

New Terminology

Bitonic GPU

Render Targets (MRTs)

Frame-Buffer Object (FBO)

Level Of Detail (LOD)

Programmable Fragment Processor - most useful stage of the GPU rendering pipeline to GPGPU programmers.

Scattering Matrix vs. scatter matrix

Synergistic Processing Units (SPU’s)

Spatial Partitioning

Power Processing Unit (PPU)

Agent Specification – determine the GPU texture space required for agent data storage.

Citations

This paper addresses scientists specialises in GPU and ABM. Thus, their papers will most likely be relevant for my research :-

* Reynolds, C. W. (1987), Flocks, Herds, and Schools: A Distributed Behavioural Model, in Computer Graphics, 21(4) (SIGGRAPH '87 Conference Proceedings) pages 25-34 **(cited by 8018)**
* Minar, N. Burkhart, R. Langton, C. Askenazi, M. (1996), The Swarm simulation system: **a** toolkit for building multi-agent simulations, Working Paper 9606-042, Santa Fe Institute, Santa Fe **(cited by 897)**
* Purcell, T. Donner, C. Camarano, M. Jensen, H. Hanrahan, P. (2003), Photon mapping on programmable graphics hardware, in Proceedings ACM SIGGRAPH/Eurographics Workshop on Graphics Hardware 2003, pages 41-50 **(Cited by 431)**
* Kipfer, P. Segal, M. Westermann, R. (2004), UberFlow: a GPU-based particle engine, In Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Conference on Graphics Hardware (Grenoble, France, August 29 - 30, 2004). HWWS '04. ACM, New York, NY, pages 115-122 **(cited by 272)**
* Collier, N. (2002), RePast: An Extensible Framework for Agent Simulation **(Cited by 235)**
* Reynolds, C. (2006), Big fast crowds on PS3, In Proceedings of the 2006 ACM SIGGRAPH Symposium on Videogames (Boston, Massachusetts, July 30 - 31, 2006). sandbox '06. ACM, New York, NY, pages 113-121 **(cited by 167)**
* Latta, L. (2004), Building a Million Particle System, In proceedings of Game Developers Conference, San Francisco, CA **(Cited by 108)**

The remaining articles no more than 65 citations which leads me to believe that they are less relevant to my research area.