

Implementation of Cellular Automata using a Graphics Processing Unit

Johannes Singler
Student, IAKS, University of Karlsruhe
johannes@singler.name

Motivation

Cellular Automata are used in Natural Science as well as in Social Sciences to simulate various processes. Since simulation of cellular automata is a calculation-intensive task, parallelization is desired and promises to be highly scalable this way due to the high locality of CAs.

By comparing the performance of various kinds of cellular automata, some of them more of theoretical, some of them more of practical interest, the class of CA that is adequate for calculation on a GPU is explored.

Results

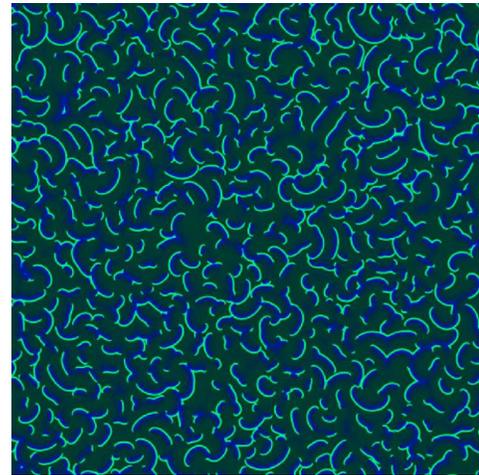
As a result, automata that only have a small number of states like the Game of Life or WireWorld (which proves the Turing completeness of this approach) do not perform well. The simulation on a CPU benefits even more from massive parallelization through bit vectors and logical operations, which are not available on the GPU.

However, some cellular automata break the paradigm of having a small number of states. In fact, their state is represented by a tuple of floating-point numbers. The implementation of probabilistic CA simulating a reaction diffusion process after Fitz-Hugh and Nagumo uses two 32-bit floating point numbers for the state representation and achieves a speedup of 10 in comparison to the CPU version.

Poster

This poster gives an overview of the implementation of several cellular automata using a modern Graphics Processing Unit. It primarily presents screenshots of some of the implementations as well as sample code for the Game of Life and the Fitz-Hugh-Nagumo implementation.

Also, a couple of subproblems are addressed, e. g. effective provision of pseudo-random numbers for probabilistic CAs as well as encoding states and state change tables in pixmaps. Another interesting feature is the use of certain texture-filtering modes for summing up or averaging values. Also, the drawbacks of simulating CAs on a GPU are discussed. There is no possibility of further interaction between cells or any global knowledge.



Fitz-Hugh-Nagumo CA

E. g., CAs that require a global status to detect stability cannot be efficiently implemented this way. A negative example is brought up with the Sandbox CA.

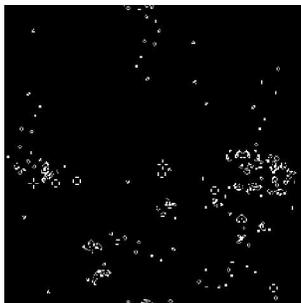
Serving as eye candy, implementations of the mandelbrot fractal and the propagation of waves are shown.

Platform

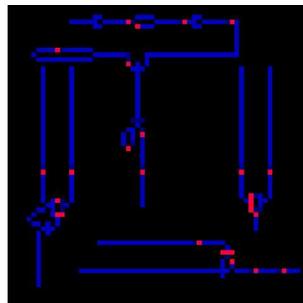
The used platform consists of a Pentium-4-PC running Windows XP. A graphics card based on the ATI 9700 chipset equipped with 128 MB graphics memory serves as GPU. The program utilizes DirectX 9 and restricts itself to the operations of the Pixel Shader 2.0 specification.

Useful Future Enhancements

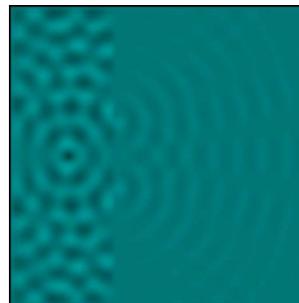
If current GPUs supported binary logic operations, one pixel could contain the state of a large number of cells. But this interferes with the GPU's floating-point preference. Also, global variables that at least can be accumulated would help to establish some kind of global knowledge and statistics.



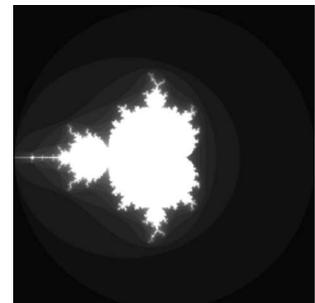
Game of Life



Wire World



Wave Propagation



Mandelbrot Fractal