Evaluating the Performance of Processing Medical Volume Data on Graphics Hardware

Matthias Raspe, Guido Lorenz, Stefan Müller
Computer Graphics Working Group, Institute for Computational Visualistics, University of Koblenz-Landau

Motivation
Modern graphics hardware (GPU) is often used only for visualization tasks in medical applications. Although there has been a lot of research in generalizing the high computational power of standard graphics processors, two main problems remain:

- programming the GPU requires in-depth knowledge of computer graphics concepts
- the performance gain for single operations is often outweighed by the additional data transfer

As the performance gap between GPUs and (multi-core) CPUs continues to grow (see figure), using graphics hardware also for processing tasks is of high interest\(^1\). Thus, we have evaluated the performance for basic medical image processing tasks in order to motivate the utilization of graphics hardware for computations in current medical applications. Both single operations and sequences of different complexity have been implemented using two different frameworks, our GPU-based system “Cascada” and the widely used “MeVisLab”\(^4\), and have been tested on typical data sets with commodity PC/graphic hardware systems.

Test setup
- Intel Core2Duo, 2.4 GHz, 2 GB RAM
- Nvidia GeForce 8800 GTS, 640 MB
- Windows XP, with:
  - MeVisLab 1.5.1
  - Cascada 1.0
- Medical data sets of typical size and bit depth:
  - Abdomen CTA: 512 x 512 x 223
  - Head MRI: 256 x 256 x 256

Results
- GPU-only implementations of common kernels clearly outperform software filter
- GPU algorithms are less sensitive to the amount of data and complexity of kernel operations than software implementations
- However, including the transfer from/to the graphics memory often abolishes the GPU’s performance advantage
- Successive computations on the different platforms (both without memory transfer) show better overall performance
- Utilizing the graphics hardware for whole workflows reveals their clear performance superiority

Conclusion
- Clear performance advantage for hardware implementations, if many tasks are executed on GPU
- Also more intricate computations (vesselness filter, watershed transform, etc.) are up to multiple orders of magnitude faster
- With the processed data in graphics memory, (intermediate) results can be visualized directly with negligible performance overhead
- Our approach for GPU programming in “Cascada” is more application-oriented than current low-level APIs such as CUDA or CTM
- Further evaluation of more complex operations and the influence of different optimization strategies towards a classification of algorithms for load balancing during run-time etc.

References