# ZFP and CUDA

Mark Kim, Peter Lindstrom, and Charles Hansen

**Abstract**—High performance computing has seen a remarkable shift over the last ten years as GPGPU has become intrinsic in the design of large supercomputers to achieve improved scaling performance over traditional CPU-only systems. Unfortunately, as GPU accelerator performance has increased non-linearly, memory bandwidth over the PCI bus has stagnated which has resulted in increased latency. Further, the size of GPU RAM has increased without a corresponding increase in memory bandwidth, which increases the fixed time cost of moving data between main memory and the GPU.
Without increasing the physical memory bandwidth, software solutions are required if we are to achieve exascale computing. Previously, a fixed rate floating point compressor, zfp, was introduced. This lossy scheme usually compresses to an accuracy within machine epsilon delta. Unfortunately, a GPU implementation was unavailable. Therefore, we introduce a GPGPU implementation of zfp.

**Index Terms**—Compression, GPGPU

## 1 Introduction

High performance computing has seen a remarkable shift over the last ten years as GPGPU has become intrinsic in the design of large supercomputers to achieve improved scaling performance over traditional CPU-only systems. Unfortunately, as GPU accelerator performance has increased non-linearly, memory bandwidth over the PCI bus has stagnated which has resulted in increased latency. Further, the size of GPU RAM has increased without a corresponding increase in memory bandwidth, which increases the fixed time cost of moving data between main memory and the GPU.

Without increasing the physical memory bandwidth, software solutions are required if we are to achieve exascale computing. Previously, a fixed rate floating point compressor, zfp, was introduced. This lossy scheme usually compresses to an accuracy within machine epsilon. Unfortunately, a GPU implementation was unavailable to alleviate the bottleneck on the GPU. Therefore, we introduce a GPGPU implementation of zfp.

## 2 Previous Works

Something about previous works.

### 2.1 Nebo

Some stuff about Nebo.

### 2.2 ZFP on the CPU

ZFP on CPU.

## 3 GPU

## 4 Results

## 5 Future Works

1.

2. Test performance with and without zfp.

- *Mark Kim is with SCI Institute at the University of Utah. E-mail: mbk@cs.utah.edu.*
- *Peter Lindstrom is with Lawrence Livermore National Laboratory. E-mail: lindstrom@llnl.gov*
- *Charles Hansen is with SCI Institute at the University of Utah. E-mail: hansen@cs.utah.edu.*

*Manuscript received 31 Mar. 2015; accepted 1 Aug. 2015; date of publication xx Aug. 2015; date of current version 25 Oct. 2015.*
*For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.*

## Acknowledgments

The authors wish to thank A, B, C. This work was supported in part by a grant from XYZ.

## References

[1] G. Grinstein, D. Keim, and M. Ward. Information visualization, visual data mining, and its application to drug design. IEEE Visualization 2002 Course #1 Notes, October 2002.
[2] G. Kindlmann. Semi-automatic generation of transfer functions for direct volume rendering. Master's thesis, Cornell University, 1999.
[3] Kitware, Inc. *The Visualization Toolkit User's Guide*, January 2003.
[4] M. Levoy. *Display of Surfaces from Volume Data*. PhD thesis, University of North Carolina at Chapel Hill, 1989.
[5] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Computer Graphics (Proceedings of SIGGRAPH 87)*, volume 21, pages 163–169, July 1987.
[6] N. Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, June 1995.
[7] G. M. Nielson and B. Hamann. The asymptotic decider: Removing the ambiguity in marching cubes. In *Visualization '91*, pages 83–91, 1991.
[8] C. Ware. *Information Visualization: Perception for Design*. Morgan Kaufmann Publishers, second edition, 2004.