Visualization and Analysis of Petascale Molecular Dynamics Simulations

John E. Stone

Theoretical and Computational Biophysics Group

Beckman Institute for Advanced Science and Technology

University of Illinois at Urbana-Champaign

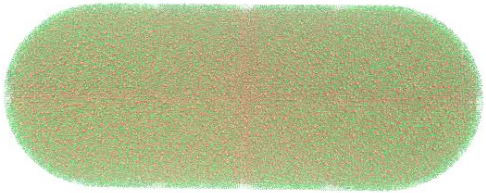
http://www.ks.uiuc.edu/Research/gpu/

GPU Computing Symposium

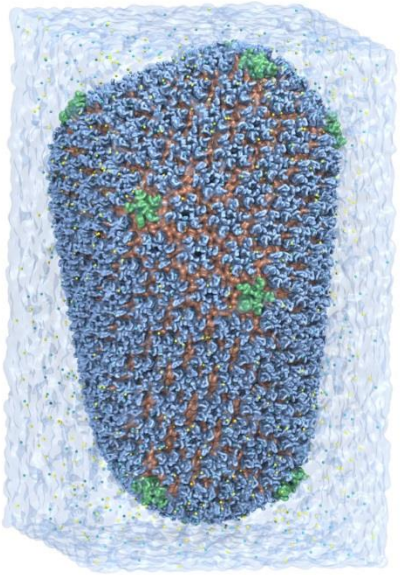
University of California San Diego, November 5, 2013

VMD – “Visual Molecular Dynamics”

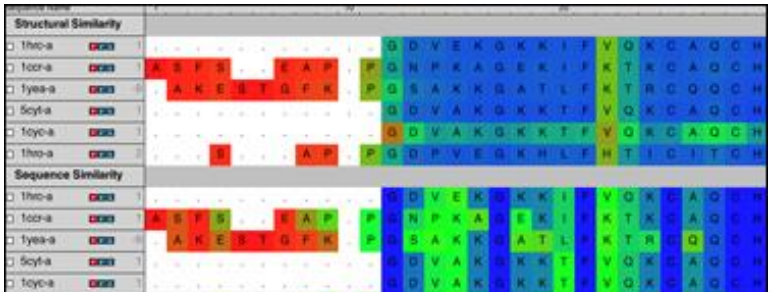
- Visualization and analysis of:
  - molecular dynamics simulations
  - particle systems and whole cells
  - cryoEM densities, volumetric data
  - quantum chemistry calculations
  - sequence information
- User extensible w/ scripting and plugins
- http://www.ks.uiuc.edu/Research/vmd/



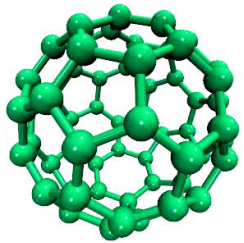
Whole Cell Simulation



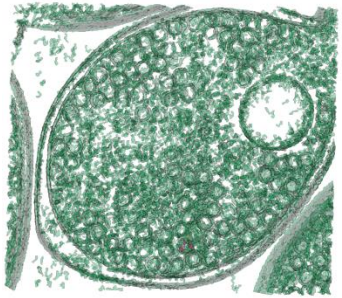
MD Simulations



Sequence Data



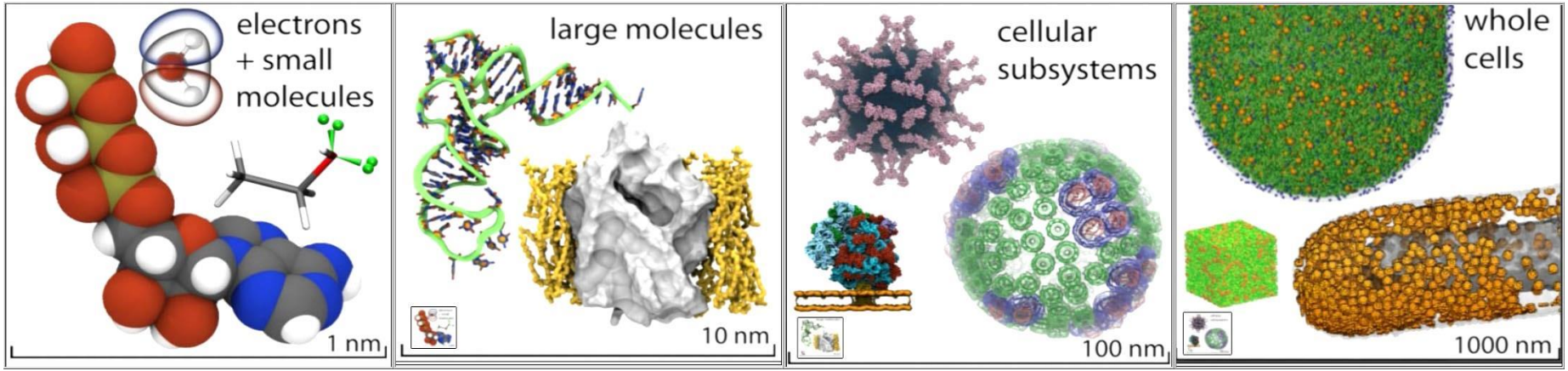
Quantum Chemistry



CryoEM, Cellular Tomography

VMD Interoperability Serves Many Communities

- Uniquely interoperable with a broad range of tools: AMBER, CHARMM, CPMD, DL_POLY, GAMESS, GROMACS, HOOMD, LAMMPS, NAMD, and many more …
- Supports key data types, file formats, and databases, e.g. electron microscopy, quantum chemistry, MD trajectories, sequence alignments, super resolution light microscopy
- Incorporates tools for simulation preparation, visualization, and analysis



GPU Computing

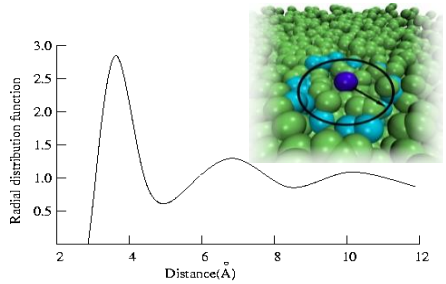
- Commodity devices, omnipresent in modern computers (over **one million** sold per **week**)
- Massively parallel hardware, thousands of processing units, **throughput oriented architecture**
- Standard integer and floating point types supported
- Programming tools allow software to be written in dialects of familiar C/C++ and integrated into legacy software
- GPU algorithms are often multicore friendly due to attention paid to **data locality** and **data-parallel** work decomposition

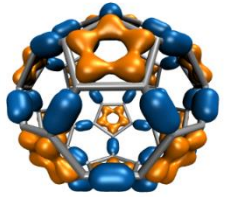
What Speedups Can GPUs Achieve?

- Single-GPU speedups of **3x** to **10x** vs. multi-core CPUs are common
- Best speedups can reach **25x** or more, attained on codes dominated by floating point arithmetic, especially native GPU machine instructions, e.g. expf(), rsqrtf(), …
- **Amdahl's Law** can prevent legacy codes from achieving peak speedups with shallow GPU acceleration efforts

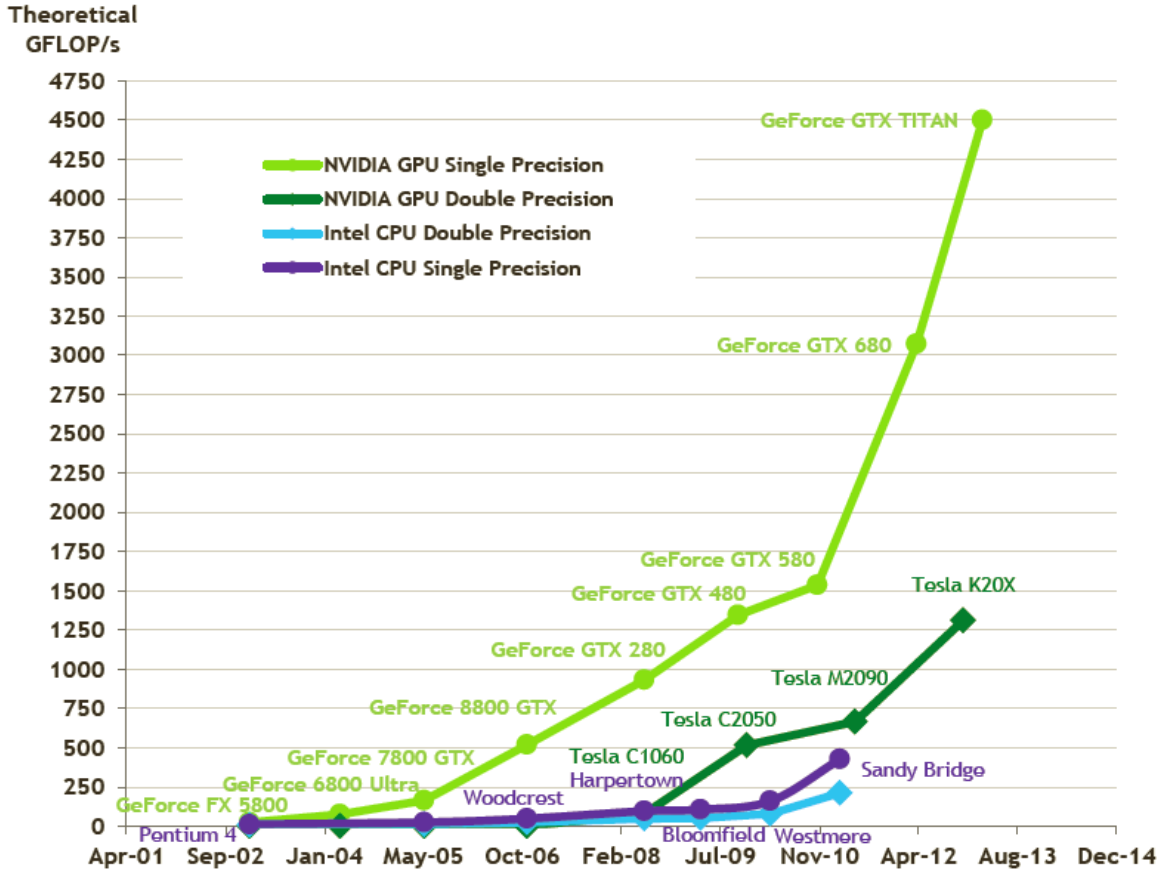
CUDA GPU-Accelerated Trajectory Analysis and Visualization in VMD

VMD GPU-Accelerated Feature or Kernel	Typical speedup vs. multi-core CPU (e.g. 4-core CPU)
Molecular orbital display	**30x**
Radial distribution function	**23x**
Molecular surface display	**15x**
Electrostatic field calculation	**11x**
Ray tracing w/ shadows, AO lighting	**8x**
Ion placement	6x
MDFF density map synthesis	6x
Implicit ligand sampling	6x
Root mean squared fluctuation	6x
Radius of gyration	5x
Close contact determination	5x
Dipole moment calculation	4x

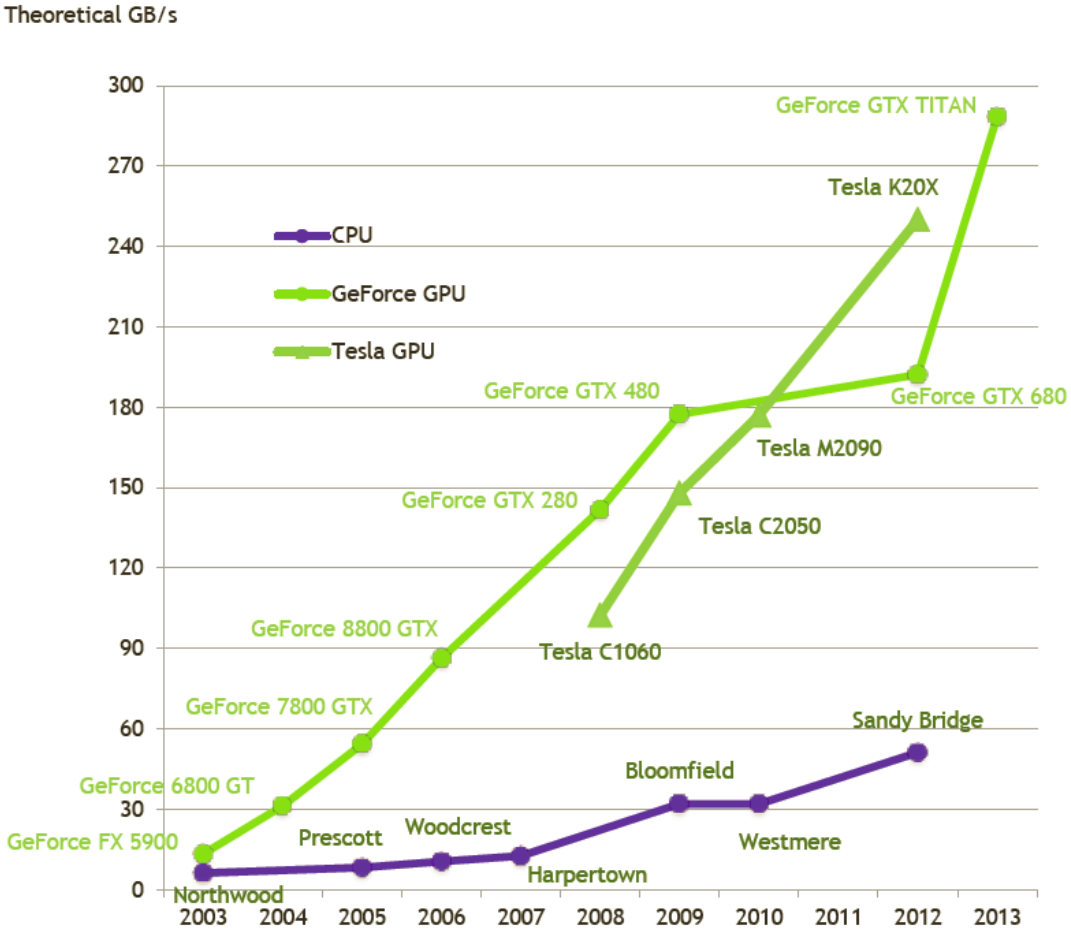




Peak Arithmetic Performance: Exponential Trend

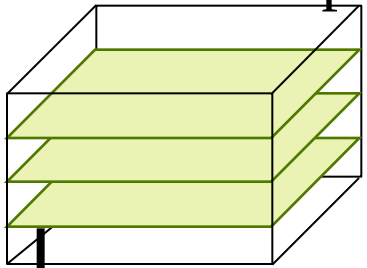


Peak Memory Bandwidth: Linear*ish* Trend



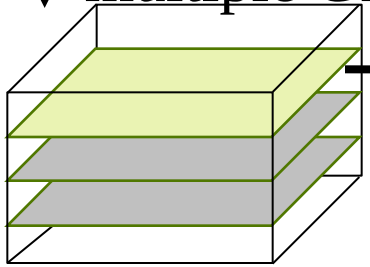
GPU Solution: Computing C_{60} Molecular Orbitals

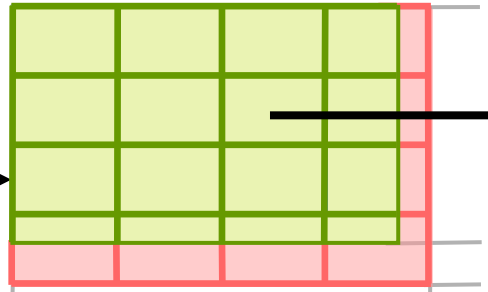
3-D orbital lattice: millions of points



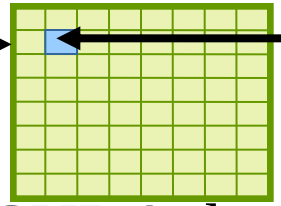
Device	CPUs, GPUs	Runtime (s)	Speedup
2x Intel X5550-SSE	8	4.13	1
GeForce GTX 480	1	0.255	16
GeForce GTX 480	4	0.081	51

Lattice slices computed on multiple GPUs





2-D CUDA grid on one GPU

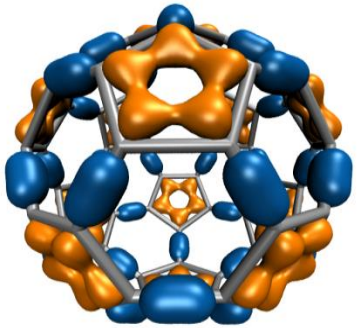


CUDA thread blocks

GPU threads each compute one point.

Molecular Orbital Inner Loop, Hand-Coded x86 SSE
Hard to Read, Isn't It? (And this is the "pretty" version!)

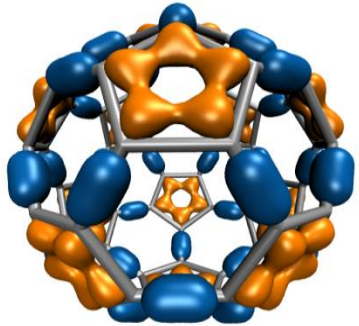
```
for (shell=0; shell < maxshell; shell++) {
   __m128 Cgto = _mm_setzero_ps();
   for (prim=0; prim<num_prim_per_shell[shell_counter]; prim++) {
      float exponent         = -basis_array[prim_counter      ];
      float contract_coeff =  basis_array[prim_counter + 1];
      __m128 expval = _mm_mul_ps(_mm_load_ps1(&exponent), dist2);
      __m128 ctmp = _mm_mul_ps(_mm_load_ps1(&contract_coeff), exp_ps(expval));
      Cgto = _mm_add_ps(contracted_gto, ctmp);
      prim_counter += 2;
   }
   __m128 tshell = _mm_setzero_ps();
   switch (shell_types[shell_counter]) {
      case S_SHELL:
         value = _mm_add_ps(value, _mm_mul_ps(_mm_load                    _[ifunc++]), Cgto));    break;
      case P_SHELL:
         tshell = _mm_add_ps(tshell, _mm_mul_ps(_mm_load_ps1(&wave_f[ifunc++]), xdist));
         tshell = _mm_add_ps(tshell, _mm_mul_ps(_mm_load_ps1(&wave_f[ifunc++]), ydist));
```



Writing SSE kernels for CPUs requires assembly language, compiler intrinsics, various libraries, **or** a really smart autovectorizing compiler **and lots of luck...**

Molecular Orbital Inner Loop in CUDA

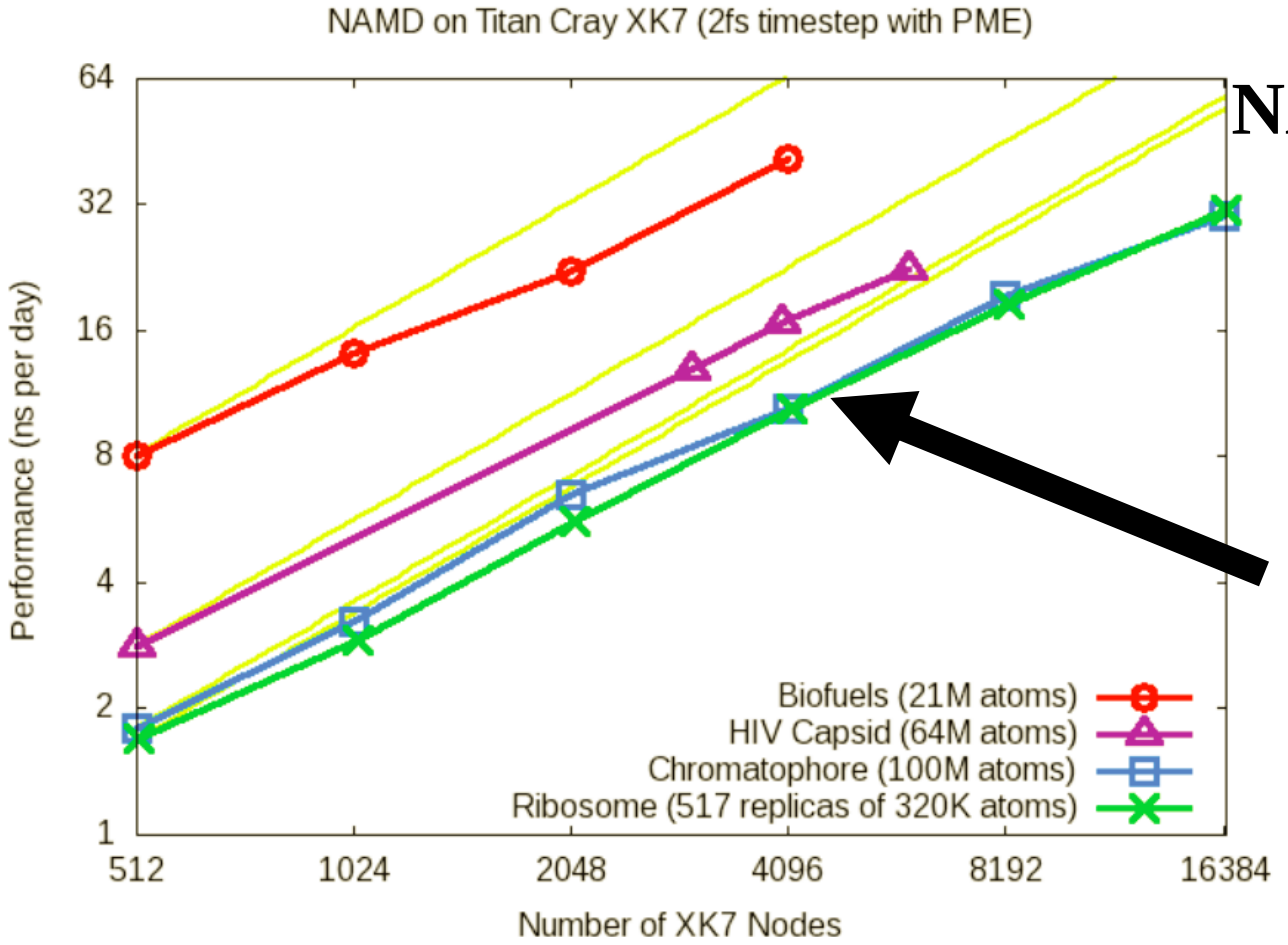
```
for (shell=0; shell < maxshell; shell++) {
  float contracted_gto = 0.0f;
  for (prim=0; prim<num_prim_per_shell[shell_counter]; prim++) {
    float exponent          = const_basis_array[prim_counter     ];
    float contract_coeff = const_basis_array[prim_counter + 1];
    contracted_gto += contract_coeff * exp2f(-exponent*dist2);
    prim_counter += 2;
  }
  float tmpshell=0;
  switch (const_shell_symmetry[shell_counter]) {
    case S_SHELL:
      value += const_wave_f[ifunc++] * contracted_gto;    break;
    case P_SHELL:
      tmpshell += const_wave_f[ifunc++] * xdist;
      tmpshell += const_wave_f[ifunc++] * ydist
      tmpshell += const_wave_f[ifunc++] * zdist;
      value += tmpshell * contracted_gto;   break;
```



Aaaaahhhh….

Data-parallel CUDA kernel looks like normal C code for the most part….

NAMD Titan XK7 Performance August 2013



NAMD XK7 vs. XE6

Speedup: 3x-4x

HIV-1 MDFF Trajectory: ~1.2 TB/day @ 4096 XK7 nodes

VMD Supports Petascale Biology

- **Where to put the data?**
  - Trajectories too large to download
  - **Analyze 231 TB trajectory set in 15 min**, parallel I/O @ **275 GB/sec** on 8,192 nodes
- **Supports GPU-accelerated Cray XK7 nodes for both visualization and analysis tasks**
  - GPU electrostatics, RDF, density quality-of-fit
  - OpenGL Pbuffer off-screen rendering support
  - GPU ray tracing w/ ambient occlusion lighting
- VMD analysis calculations and movie renderings use dynamic load balancing, tested with up to 262,144 CPU cores
- Available on: NCSA Blue Waters, ORNL Titan, Indiana Big Red II

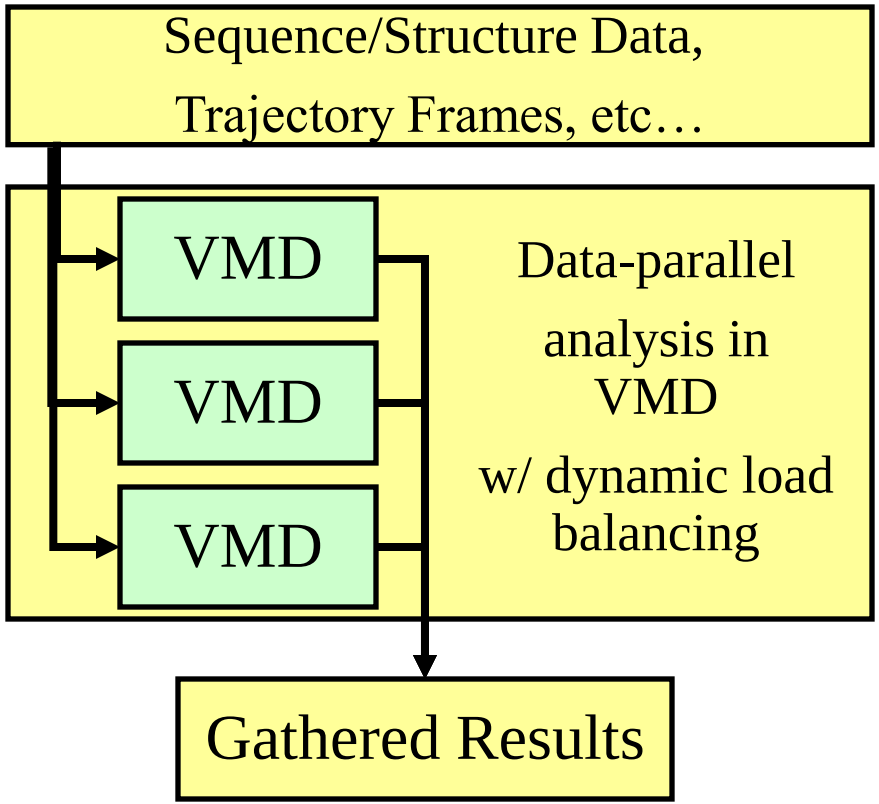
NCSA Blue Waters
Cray XE6 / XK7 Supercomputer

22,640 XE6 CPU nodes

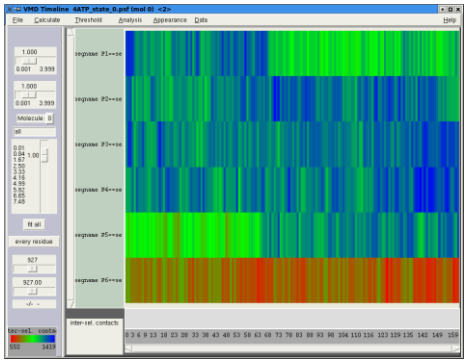
4,224 XK7 nodes w/ GPUs enable fast VMD analysis and visualization

VMD for Demanding Analysis Tasks
Parallel VMD Analysis w/ MPI

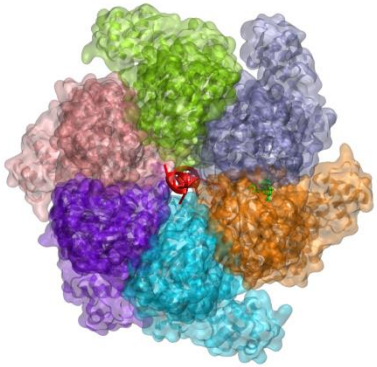
- Compute time-averaged electrostatic fields, MDFF quality-of-fit, etc.
- Parallel rendering, movie making
- User-defined parallel reduction operations, data types
- **Parallel I/O on Blue Waters:**
  - **109 GB/sec on 512 nodes**
  - **275 GB/sec on 8,192 nodes**
- **Timeline per-residue SASA calc. achieves 800x speedup @ 1000 BW XE6 nodes**
- **Supports GPU-accelerated clusters and supercomputers**



Analyze large, long simulations with TimeLine



TimeLine 2D plot



Rho hexameric helicase 3D structure

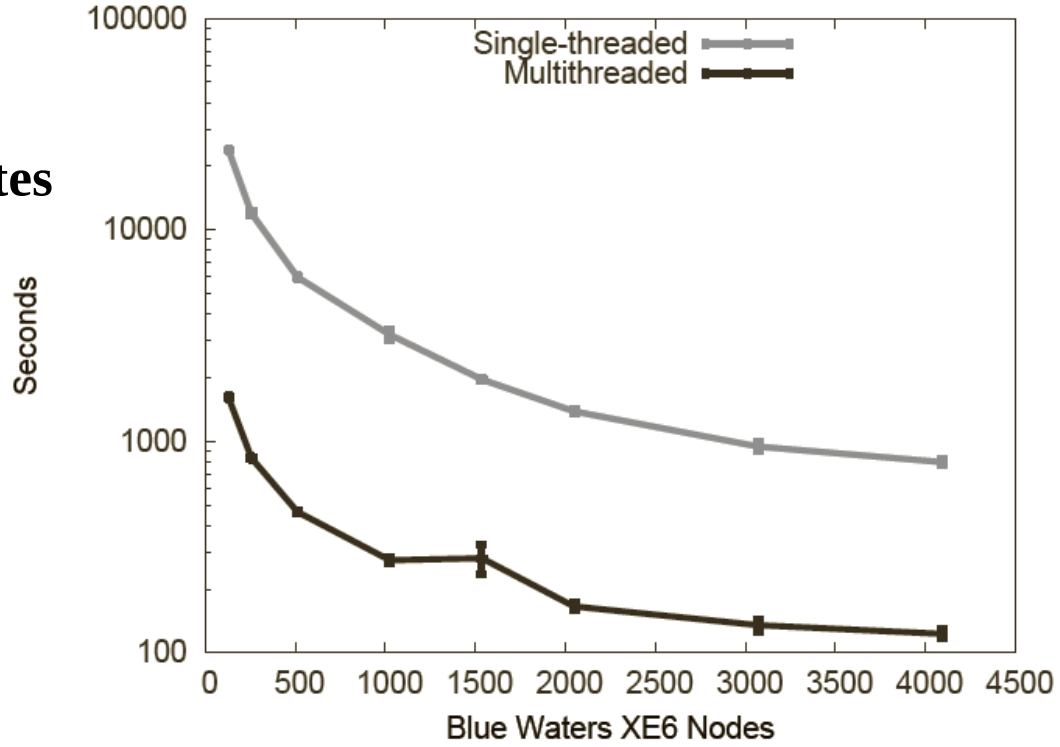
TimeLine:

- graphing and analysis tool to **identify events** in an MD trajectory
- live 2D whole-trajectory plot linked to 3D structure
- user-extendable

- Perform analysis faster
  - High-performance parallel trajectory analysis on supercomputers and clusters
  - Current development versions show up to **3500x speedup** on Blue Waters
- Analysis types: filtering, time series analysis, sorting (e.g. bond energies)
- Remote interactive analysis: data at supercomputer center; view in office

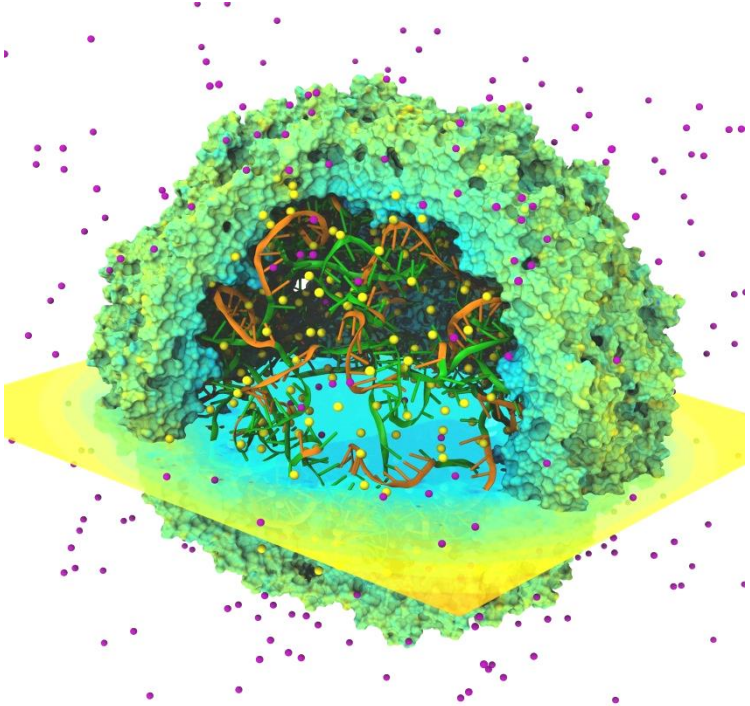
Parallel TimeLine Calc. for Per-Residue Solvent-Accessible Surface Area

- Single-node single-thread runtime projected to be **35 days**!
- 2,048 node runtime under **4 minutes**



Time-Averaged Electrostatics Analysis on Energy-Efficient GPU Cluster

- **1.5 hour** job (CPUs) reduced to **3 min** (CPUs+GPU)
- Electrostatics of thousands of trajectory frames averaged
- Per-node power consumption on NCSA "AC" GPU cluster:
  - CPUs-only: 448 Watt-hours
  - CPUs+GPUs: 43 Watt-hours
- GPU Speedup: **25.5x**
- Power efficiency gain: **10.5x**



Quantifying the Impact of GPUs on Performance and Energy Efficiency in HPC Clusters.
J. Enos, C. Steffen, J. Fullop, M. Showerman, G. Shi, K. Esler, V. Kindratenko, J. Stone, J. Phillips.
The Work in Progress in Green Computing, pp. 317-324, 2010.

Time-Averaged Electrostatics Analysis on NCSA Blue Waters

NCSA Blue Waters Node Type	**Seconds per trajectory frame for one compute node**
Cray XE6 Compute Node: 32 CPU cores (2xAMD 6200 CPUs)	9.33
Cray XK6 GPU-accelerated Compute Node: 16 CPU cores + **NVIDIA X2090 (Fermi) GPU**	2.25
Speedup for GPU XK6 nodes vs. CPU XE6 nodes	**XK6 nodes are 4.15x faster overall**
Tests on XK7 nodes indicate MSM is CPU-bound with the Kepler K20X GPU. **Performance is not much faster (yet) than Fermi X2090** **Need to move spatial hashing, prolongation, interpolation onto the GPU…**	**In progress….** **XK7 nodes 4.3x faster overall**

Preliminary performance for VMD time-averaged electrostatics w/ Multilevel Summation Method on the NCSA Blue Waters Early Science System

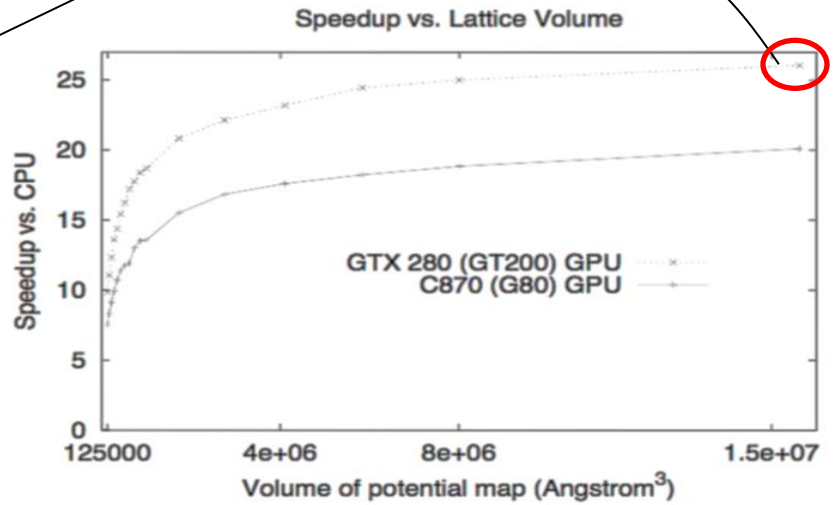
Multilevel Summation on the GPU

Accelerate **short-range cutoff** and **lattice cutoff** parts

Performance profile for 0.5 Å map of potential for 1.5 M atoms.

Hardware platform is Intel QX6700 CPU and NVIDIA GTX 280.

Computational steps	CPU (s)	w/ GPU (s)	Speedup
Short-range cutoff	480.07	14.87	32.3
Long-range anterpolation	0.18		
restriction	0.16		
lattice cutoff	49.47	1.36	36.4
prolongation	0.17		
interpolation	3.47		
Total	533.52	20.21	26.4



Multilevel summation of electrostatic potentials using graphics processing units.
D. Hardy, J. Stone, K. Schulten. *J. Parallel Computing*, 35:164-177, 2009.

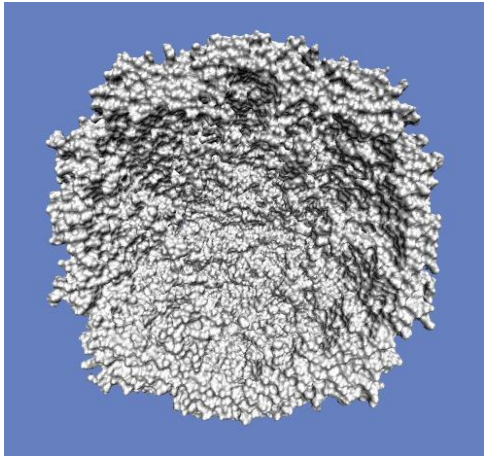
VMD "QuickSurf" Representation

- Displays continuum of structural detail:
  - All-atom models
  - Coarse-grained models
  - Cellular scale models
  - Multi-scale models: All-atom + CG, Brownian + Whole Cell
  - Smoothly variable between full detail, and reduced resolution representations of very large complexes
- GPU acceleration yields 10x-15x speedup vs. multi-core CPUs

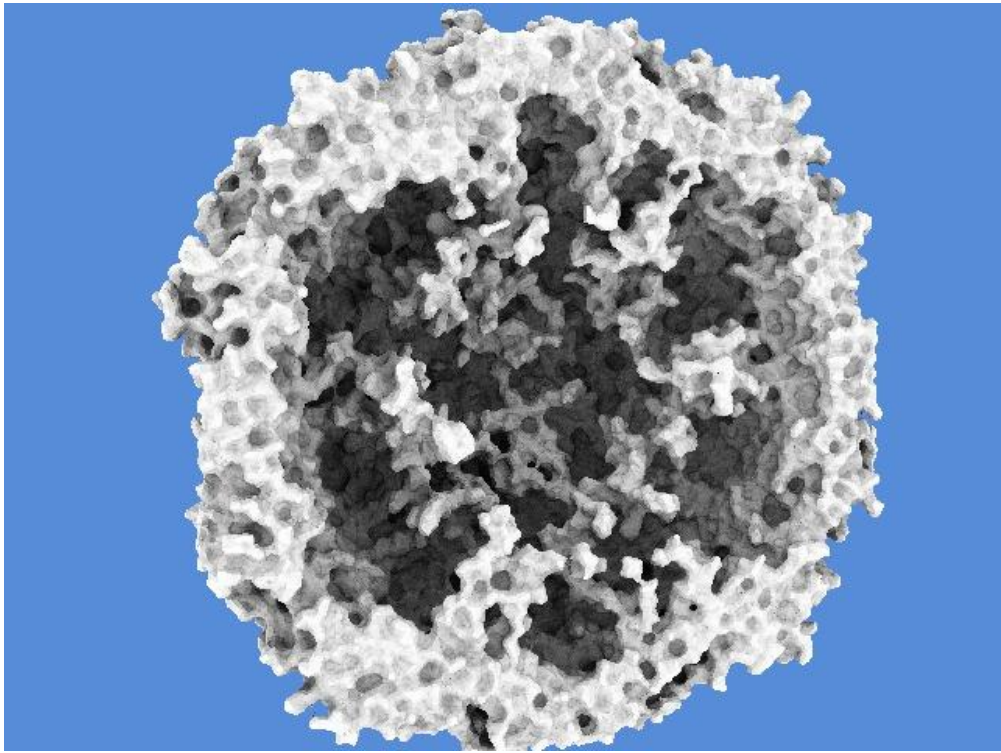
Fast Visualization of Gaussian Density Surfaces for Molecular Dynamics and Particle System Trajectories.
M. Krone, J. E. Stone, T. Ertl, K. Schulten. *EuroVis Short Papers*, pp. 67-71, 2012

Ray Tracing of VMD Molecular Graphics

- Ambient occlusion lighting, shadows, reflections, transparency, and much more…
- Satellite tobacco mosaic virus capsid w/ ~75K atoms

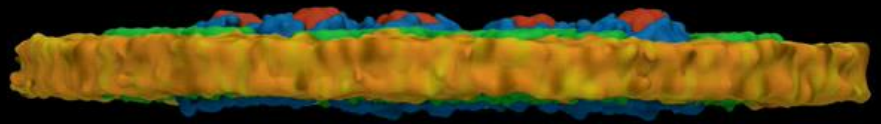


Standard OpenGL rasterization



VMD/Tachyon/CUDA/OptiX GPU ray tracing w/ ambient occlusion lighting

BW VMD/Tachyon Movie Generation



20 M atom chromatophore patch

360 XE6 nodes for 3h50m @ 4096x2400

HIV-1 Parallel Movie Rendering Results

- Unexpected I/O overhead from sourcing scripts!
- XK7 CUDA algorithms reduce per-frame surface and other geometry calculation times by a factor of ~15 vs. multithreaded SSE CPU code on XE6 nodes
- OpenGL rasterization is so fast it is essentially “free” – I/O time dominates OpenGL test cases currently... (XK7 partition had no I/O nodes)
- For CPU-only Tachyon, XE6 nodes render almost exactly 2x faster than XK7 nodes
- All test cases start to be penalized at >= 512 nodes due to increased I/O contention for common input files, reading of scripts, etc – need broadcast scheme for this data

VMD Movie Rendering on Blue Waters

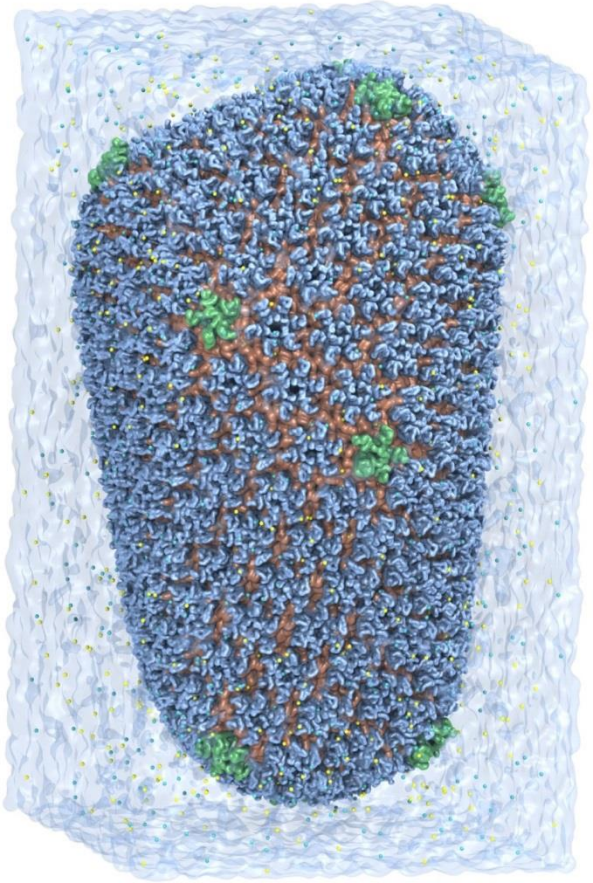
Movie Resolution	Rendering Mode	Node Type	Nodes	Wall Clock Execution Time			
				Script Loading	State Loading	Geometry and Rendering	Total
"PowerPoint" 1057 × 652 689,164 pixels	OpenGL rasterization	XK7	16	2 s	152 s	99 s	253 s
		XK7	32	2 s	158 s	45 s	205 s
		XK7	64	2 s	167 s	20 s	189 s
		XK7	128	2 s	191 s	11 s	205 s
		XK7	512	7 s	302 s	2.5 s	312 s
	In-place Tachyon ray tracing w/ ambient occlusion (AO) lighting	XK7	256	4 s	225 s	918 s	1,147 s
		XK7	512	9 s	292 s	532 s	834 s
		XE6	128	2 s	83 s	943 s	1,029 s
		XE6	256	4 s	125 s	560 s	692 s
		XE6	512	7 s	221 s	330 s	560 s
	Combined OpenGL rasterization and Tachyon ray tracing w/ AO	XK7	256	4 s	214 s	913 s	1,170 s
		XK7	512	9 s	300 s	531 s	848 s
4K UltraHD 3840 × 2160 8,294,400 pixels	OpenGL rasterization	XK7	512	9 s	300 s	3.1 s	314 s
	Combined OpenGL rasterization and Tachyon ray tracing w/ AO	XK7	512	9 s	295 s	5,828 s	6,133 s
No Image Output	Tesla K20X CUDA Geometry Calc.	XK7	512	7 s	188 s	1.5 s	197 s
	CPU Geometry Calc.	XE6	512	7 s	214 s	23 s	244 s

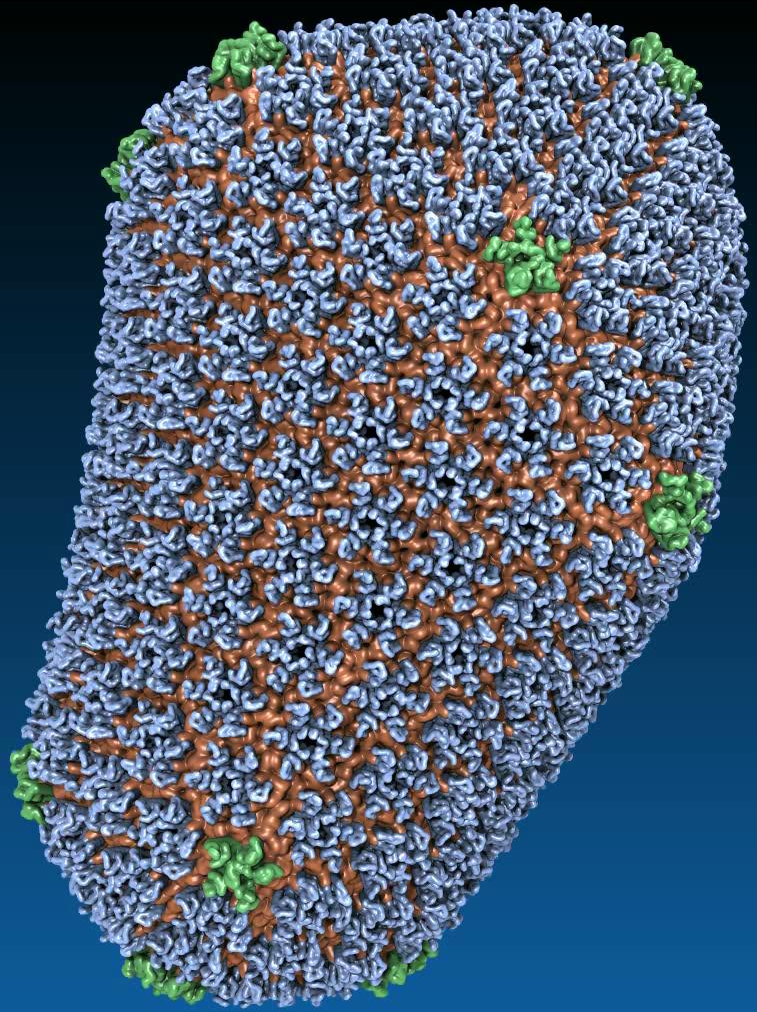
TABLE II. VMD PARALLEL MOVIE RENDERING PERFORMANCE TESTS.

Early Experiences Scaling VMD Molecular Visualization and Analysis Jobs on Blue Waters. J. E. Stone, B. Isralewitz, and K. Schulten. In proceedings, Extreme Scaling Workshop, 2013. (In press)

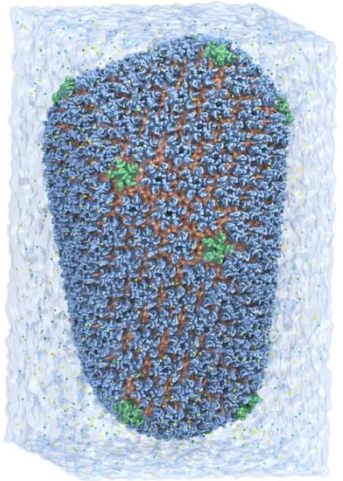
GPU Ray Tracing of HIV-1 on Blue Waters

- 64 million atom virus simulation
- 1079 movie frames
- Surface generation and ray tracing stages each use >= 75% of GPU memory
- Ambient occlusion lighting, shadows, transparency, antialiasing, depth cueing, 144 rays/pixel minimum





HIV-1 Parallel HD Movie Rendering on Blue Waters Cray XE6/XK7



New “TachyonL-OptiX” on XK7 w/ K20 GPU vs. Tachyon on XE6 GPUs yield up to **eight times** geom+ray tracing speedup

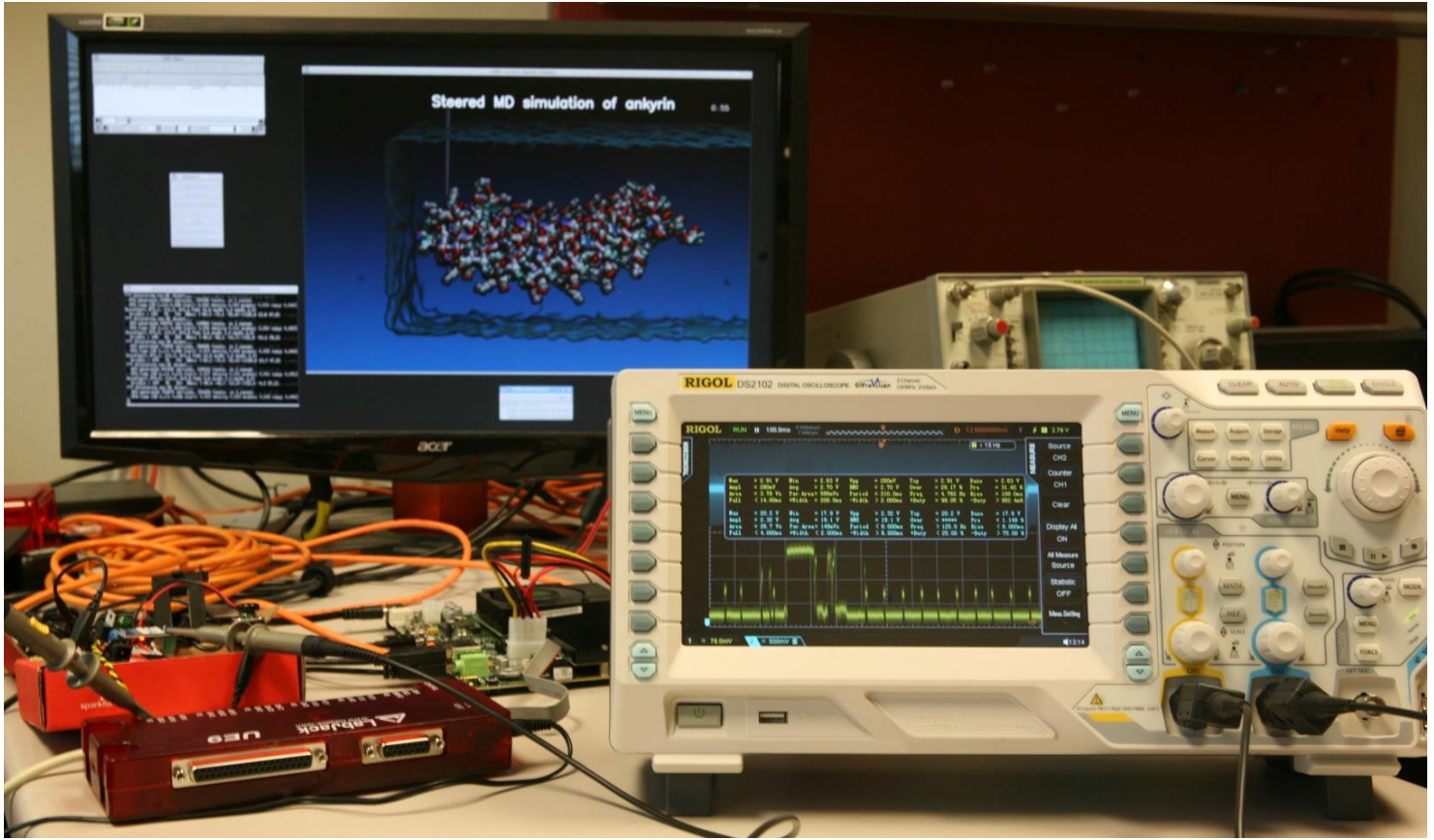
Node Type and Count	Script Load Time	State Load Time	Geometry + Ray Tracing	Total Time
256 XE6 CPUs	7 s	160 s	**1,374 s**	**1,541 s**
512 XE6 CPUs	13 s	211 s	808 s	1,032 s
64 XK7 Tesla K20X GPUs	2 s	38 s	655 s	695 s
128 XK7 Tesla K20X GPUs	4 s	74 s	331 s	410 s
256 XK7 Tesla K20X GPUs	7 s	110 s	**171 s**	**288 s**

GPU-Accelerated Molecular Visualization on Petascale Supercomputing Platforms, UltraVis'13: Eighth Workshop on Ultrascale Visualization Proceedings, 2013. (In press)

Optimizing GPU Algorithms for Power Consumption

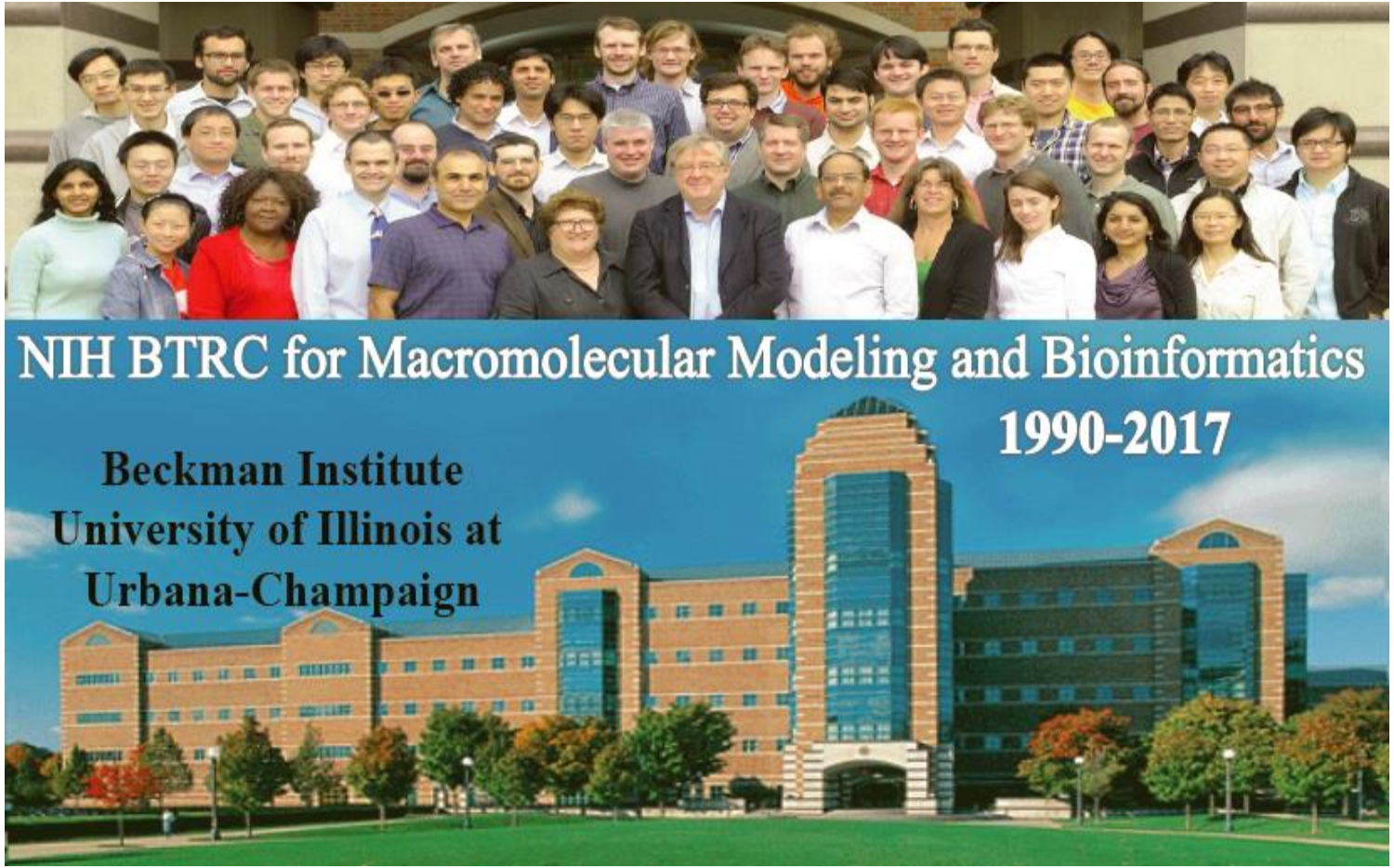
NVIDIA "Carma" and "Kayla" single board computers

Tegra+GPU energy efficiency testbed



Acknowledgements

- Theoretical and Computational Biophysics Group, University of Illinois at Urbana-Champaign
- NCSA Blue Waters Team
- NVIDIA CUDA Center of Excellence, University of Illinois at Urbana-Champaign
- Many of the staff at NVIDIA and Cray
- Funding:
  - NSF OCI 07-25070
  - NSF PRAC “The Computational Microscope”
  - NIH support: 9P41GM104601, 5R01GM098243-02


NIH BTRC for Macromolecular Modeling and Bioinformatics
1990-2017
Beckman Institute
University of Illinois at
Urbana-Champaign

GPU Computing Publications

http://www.ks.uiuc.edu/Research/gpu/

- **Early Experiences Scaling VMD Molecular Visualization and Analysis Jobs on Blue Waters.** J. E. Stone, B. Isralewitz, and K. Schulten. In proceedings, Extreme Scaling Workshop, 2013. (In press)
- **Lattice Microbes: High-performance stochastic simulation method for the reaction-diffusion master equation.** E. Roberts, J. E. Stone, and Z. Luthey-Schulten. J. Computational Chemistry 34 (3), 245-255, 2013**.**
- **Fast Visualization of Gaussian Density Surfaces for Molecular Dynamics and Particle System Trajectories.** M. Krone, J. E. Stone, T. Ertl, and K. Schulten. *EuroVis Short Papers*, pp. 67-71, 2012.
- **Immersive Out-of-Core Visualization of Large-Size and Long-Timescale Molecular Dynamics Trajectories.** J. Stone, K. Vandivort, and K. Schulten. G. Bebis et al. (Eds.): *7th International Symposium on Visual Computing (ISVC 2011)*, LNCS 6939, pp. 1-12, 2011.
- **Fast Analysis of Molecular Dynamics Trajectories with Graphics Processing Units – Radial Distribution Functions.** B. Levine, J. Stone, and A. Kohlmeyer. *J. Comp. Physics*, 230(9):3556-3569, 2011.

GPU Computing Publications

http://www.ks.uiuc.edu/Research/gpu/

- **Quantifying the Impact of GPUs on Performance and Energy Efficiency in HPC Clusters.** J. Enos, C. Steffen, J. Fullop, M. Showerman, G. Shi, K. Esler, V. Kindratenko, J. Stone, J Phillips. *International Conference on Green Computing*, pp. 317-324, 2010.
- **GPU-accelerated molecular modeling coming of age. J. Stone, D. Hardy, I. Ufimtsev, K. Schulten.** *J. Molecular Graphics and Modeling*, 29:116-125, 2010.
- **OpenCL: A Parallel Programming Standard for Heterogeneous Computing. J. Stone, D. Gohara, G. Shi.** *Computing in Science and Engineering*, 12(3):66-73, 2010.
- **An Asymmetric Distributed Shared Memory Model for Heterogeneous Computing Systems**. I. Gelado, J. Stone, J. Cabezas, S. Patel, N. Navarro, W. Hwu. *ASPLOS '10: Proceedings of the 15th International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 347-358, 2010.

GPU Computing Publications

http://www.ks.uiuc.edu/Research/gpu/

- **GPU Clusters for High Performance Computing**. V. Kindratenko, J. Enos, G. Shi, M. Showerman, G. Arnold, J. Stone, J. Phillips, W. Hwu. *Workshop on Parallel Programming on Accelerator Clusters (PPAC)*, In Proceedings IEEE Cluster 2009, pp. 1-8, Aug. 2009.
- **Long time-scale simulations of in vivo diffusion using GPU hardware**. E. Roberts, J. Stone, L. Sepulveda, W. Hwu, Z. Luthey-Schulten. In *IPDPS'09: Proceedings of the 2009 IEEE International Symposium on Parallel & Distributed Computing*, pp. 1-8, 2009.
- **High Performance Computation and Interactive Display of Molecular Orbitals on GPUs and Multi-core CPUs**. J. Stone, J. Saam, D. Hardy, K. Vandivort, W. Hwu, K. Schulten, *2nd Workshop on General-Purpose Computation on Graphics Pricessing Units (GPGPU-2), ACM International Conference Proceeding Series*, volume 383, pp. 9-18, 2009.
- **Probing Biomolecular Machines with Graphics Processors**. J. Phillips, J. Stone. *Communications of the ACM*, 52(10):34-41, 2009.
- **Multilevel summation of electrostatic potentials using graphics processing units**. D. Hardy, J. Stone, K. Schulten. *J. Parallel Computing*, 35:164-177, 2009.

GPU Computing Publications

http://www.ks.uiuc.edu/Research/gpu/

- **Adapting a message-driven parallel application to GPU-accelerated clusters**.
  J. Phillips, J. Stone, K. Schulten. *Proceedings of the 2008 ACM/IEEE Conference on Supercomputing*, IEEE Press, 2008.
- **GPU acceleration of cutoff pair potentials for molecular modeling applications**.
  C. Rodrigues, D. Hardy, J. Stone, K. Schulten, and W. Hwu. *Proceedings of the 2008 Conference On Computing Frontiers*, pp. 273-282, 2008.
- **GPU computing**. J. Owens, M. Houston, D. Luebke, S. Green, J. Stone, J. Phillips. *Proceedings of the IEEE*, 96:879-899, 2008.
- **Accelerating molecular modeling applications with graphics processors**. J. Stone, J. Phillips, P. Freddolino, D. Hardy, L. Trabuco, K. Schulten. *J. Comp. Chem.*, 28:2618-2640, 2007.
- **Continuous fluorescence microphotolysis and correlation spectroscopy**. A. Arkhipov, J. Hüve, M. Kahms, R. Peters, K. Schulten. *Biophysical Journal*, 93:4006-4017, 2007.