ATPESC Track 7: Scalable Molecular Visualization and Analysis Tools in VMD

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/vmd/

Argonne Training Program on Exascale Computing (ATPESC)
3:30pm-4:30pm, St. Charles Amphitheater, Q Center,
St. Charles, IL, Thursday August 9th, 2018
VMD – “Visual Molecular Dynamics”

- 100,000 active users worldwide
- Visualization and analysis of:
  - Molecular dynamics simulations
  - Lattice cell simulations
  - Quantum chemistry calculations
  - Cryo-EM densities, volumetric data
- User extensible scripting and plugins
- http://www.ks.uiuc.edu/Research/vmd/
VMD Hands-On Tutorials

- [http://www.ks.uiuc.edu/Training/Tutorials/#vmd](http://www.ks.uiuc.edu/Training/Tutorials/#vmd)
  - Main VMD tutorial
  - QwikMD simulation preparation and analysis plugin
  - VMD images and movies tutorial
  - Structure check
  - VMD quantum chemistry visualization tutorial
  - Visualization and analysis of CPMD data with VMD
  - Parameterizing small molecules using ffTK
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

• Incorporates tools for simulation preparation, visualization, and analysis
VMD Interoperates with Mainstream Research Tools

- Provides tools for simulation preparation, visualization, and analysis
- Interprets and processes multi-modal structural information
- Connects with key software tools to enable state-of-the-art simulations
- Openness, extensibility, and interoperability are VMD hallmarks
- Uses advanced algorithms and hardware technologies to address data size challenges posed by cutting-edge experimental imaging and simulation

Structure

<table>
<thead>
<tr>
<th>Parameterization</th>
<th>Analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Refinement with MDFF</td>
<td>Preparation with QwikMD</td>
</tr>
</tbody>
</table>

X-ray, cryo-EM, cryo-ET, NMR
Goal: A Computational Microscope
Study the molecular machines in living cells

Ribosome: target for antibiotics

Poliovirus
VMD Approach to Visualization

- Molecular scene is composed of “graphical representations”
- Each representation encapsulates a group of selected atoms, a drawing style, coloring style, and other parameters
- Representations are independent of each other, can be toggled on/off easily, allowing molecular scenes to be built-up incrementally
- VMD atom selection language is shared with its analytical and scripting interfaces
Selection, Filtering

- Most viz tools allow interactive visual picking, menu-driven selections of structure components to display or operate on.
- VMD also extensively uses a text-based selection language (think google):
  - water within 10 of protein and z > 0
  - nucleic or protein or ions
  - segname BR
  - name “C.*”
  - Allows selection on user-defined data fields
  - **Promotes synergy between interactive and scripting interfaces, visualization and quantitative analysis tasks**
  - Works well with huge time-varying structures
Structure Visualization

Molecular representations provide different levels of abstraction, atomic detail vs. higher level organizational information

- Atoms, VdW spheres, bonds, ball-stick, …
- Coarse-grained “beads”
- Ribbons, secondary structure, “cartoon” reps, RNA/DNA
- Molecular surfaces
- Molecular orbitals (quantum chemistry)
Computed Properties

- Smoothing of thermal noise
- Secondary structure
- Hydrogen bonds, salt bridges
- Forces, energies, stress, strain
- Time averaging of electrostatic fields, occupancy maps
- Quality-of-fit cross correlation with cryo-EM density maps
- Normal modes, principal component analysis, essential dynamics
- Cluster simulation trajectory timesteps by structural similarity

Chemoreceptor trimer-of-dimers analysis with Bendix plugin in VMD
Display of Computed Properties on Structures

Per-residue solvent-accessible surface area of Ubiquitin

PME electrostatic potential contour for a helicase on a volumetric slice plane
CheA kinase PCA: first principal component porcupine plot
Computing Molecular Properties

Compute properties, e.g., density, distance, occupancy, electrostatic potential over thousands of MD simulation trajectory frames

Example: display binding sites for diffusively bound ions as probability density isosurfaces

tRNA magnesium ion occupancy
Visualization of Molecular Dynamics

- Molecular dynamics simulations save trajectories of atomic coordinates as simulated time progresses.
- Researchers study trajectories by analyzing force profiles, energies, structural changes, etc.
- Visualization selections, graphics, structure properties recomputed for each trajectory timestep!
Cryo-EM / Cryo-ET Density Map Segmentation

Evaluate 3-D volumetric electron density maps and segment them, to identify key structural components.

Index/label components so they can be referred to, colored, analyzed, and simulated…
Density Map Segmentation

VMD GPU-accelerated density map segmentation of GroEL

All-Atom Molecular Dynamics Today

- HIV Capsid
- ATP Synthase
- Aquaporin
- Lysozyme
- STMV
- Photosynthetic Chromatophore (100 nm)^3
- Ribosome

Number of Atoms

10^8
10^7
10^6
10^5
10^4
10^3
10^2
10^1
1

Year


(NIH)
NAMD on Summit, May 2018

NAMD simulations can generate up to 10TB of output per day on 20% of Summit
Molecular Dynamics Trajectory Analysis

- MD simulations sample femtosecond timescales
- Millions of timesteps stored per trajectory
- Dynamics of biomolecular complexes are main interest, but solvent often accounts for half or more of the simulation content
  
Skip I/O for regions of bulk solvent where possible [1]
- Modern MD tools, e.g., VMD, NAMD, LAMMPS, HOOMD, employ extensive embedded scripting (Python, Tcl, etc) to permit simulation preparation, custom simulation protocols, analysis, and visualization
- Unified collective variables module allows identical analytical computations to be performed within LAMMPS, NAMD, and VMD, during pre-simulation modeling, in-situ, and post-hoc [2]


Petascale Molecular Dynamics I/O and Storage Challenges

- NAMD simulations can produce up to 10TB/day @ 1024 nodes (~20%) of ORNL Summit, more as optimizations raise NAMD performance further
- Petascale science campaigns require months of simulation runs
- Long-term storage of large-fractional petabytes impractical
- Historical “download output files for analysis and visualization” approach is a non-starter at this scale
- Demands visualization and analysis operate on the data in-place on the HPC system, whether post-hoc, in-transit, or in-situ
- Analyses must identify salient features of structure, dynamics, cull data that don’t contribute to biomolecular processes of interest
VMD Petascale Visualization and Analysis

- Analyze/visualize large trajectories too large to transfer off-site:
  - User-defined parallel analysis operations, data types
  - Parallel rendering, movie making

- Supports GPU-accelerated Cray XK7 nodes for both visualization and analysis:
  - GPU accelerated trajectory analysis w/ CUDA
  - OpenGL and GPU ray tracing for visualization and movie rendering

- Parallel I/O rates up to 275 GB/sec on 8192 Cray XE6 nodes – can read in 231 TB in 15 minutes!

Parallel VMD currently available on:

ORNL Titan, NCSA Blue Waters, Indiana Big Red II, CSCS Piz Daint, and similar systems
VMD supports EGL for in-situ and parallel rendering on clouds, clusters, and supercomputers

- Eliminate dependency on windowing systems
- Simplified deployment of parallel VMD builds supporting off-screen rendering
- Maintains 100% of VMD OpenGL shaders and rendering features
- Support high-quality vendor-supported commercial OpenGL implementations in HPC systems that were previously limited to Mesa
VMD EGL Rendering: Supports full VMD GLSL shading features
Vulkan support coming soon...

Swine Flu A/H1N1 neuraminidase bound to Tamiflu

64M atom HIV-1 capsid simulation

High Performance Molecular Visualization: In-Situ and Parallel Rendering with EGL.
VMD EGL Performance on Amazon EC2 Cloud

<table>
<thead>
<tr>
<th>MPI Ranks</th>
<th>EC2 “G2.8xlarge” GPU Instances</th>
<th>HIV-1 movie rendering time (sec), (I/O %) 3840x2160 resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>626s (10% I/O)</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>347s (19% I/O)</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>221s (31% I/O)</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>141s (46% I/O)</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>107s (64% I/O)</td>
</tr>
<tr>
<td>32</td>
<td>8</td>
<td>90s (76% I/O)</td>
</tr>
</tbody>
</table>

Performance at 32 nodes reaches ~48 frames per second

Next Generation: Simulating a Proto-Cell

- Emulate aspects of the *Mycoplasma mycoides* bacterium
- 200nm diameter
- ~1 billion atoms w/ solvent
- ~1400 proteins in membrane

Cryo-ET image of ultra-small bacteria (scale bar 100nm)
Proto-Cell Data Challenges

- 1B-atom proto-cell requires nodes with more than TB RAM to build complete model...
- 1B-atom proto-cell binary structure file: 63GB
- Trajectory frame atomic coordinates: 12GB, 1.2TB/ns of simulation (1 frame per 10ps)
- Routine modeling and visualization tasks are a big challenge at this scale
  - Models contain thousands of atomic-detail components that must work together in harmony
  - Exploit persistent memory technologies to enable “instant on” operation on massive cell-scale models – eliminate several minutes of startup during analysis/visualization of known structure
  - Sparse output of results at multiple timescales will help ameliorate visualization and analysis I/O
  - Data quantization, compression, APIs like ZFP
High Fidelity Ray Tracing

- **Interactive RT** on laptops, desk, cloud
- Large-scale parallel rendering: in situ or post hoc visualization tasks
- AO, DoF, instancing, ....
- Stereoscopic panorama and full-dome projections
- Omnidirectional VR: YouTube, HMDs

- Built-in ray tracing engines:
  - **Tachyon**: cross-platform RT
  - **NVIDIA OptiX**: GPU-accelerated and remote RT on VCA clusters
  - **Intel OSPRay**: CPU x86/Phi-optimized parallel rendering w/ MPI

VMD/OptiX all-atom Chromatophore
Lighting Comparison, STMV Capsid

Two lights, no shadows

Ambient occlusion + two lights, 144 AO rays/hit
VMD w/ OptiX

- Interactive RT on laptops, desktops, and cloud
- Large-scale parallel rendering: in situ or post hoc visualization tasks
- Remote RT on NVIDIA VCA clusters
- Stereoscopic panoramic and full-dome projections
- Omnidirectional VR for YouTube, VR HMDs
- GPU memory sharing via NVLink


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

New VMD TachyonL-OptiX on XK7 vs. Tachyon on XE6: K20X GPUs yield up to twelve times geom+ray tracing speedup

<table>
<thead>
<tr>
<th>Ray Tracer Version</th>
<th>Node Type and Count</th>
<th>Script Load</th>
<th>State Load</th>
<th>Geometry + Ray Tracing</th>
<th>Total Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>New TachyonL-OptiX</td>
<td>64 XK7 Tesla K20X GPUs</td>
<td>2 s</td>
<td>39 s</td>
<td>435 s</td>
<td>476 s</td>
</tr>
<tr>
<td>New TachyonL-OptiX</td>
<td>128 XK7 Tesla K20X GPUs</td>
<td>3 s</td>
<td>62 s</td>
<td>230 s</td>
<td>295 s</td>
</tr>
<tr>
<td>TachyonL-OptiX [1]</td>
<td>64 XK7 Tesla K20X GPUs</td>
<td>2 s</td>
<td>38 s</td>
<td>655 s</td>
<td>695 s</td>
</tr>
<tr>
<td>TachyonL-OptiX [1]</td>
<td>128 XK7 Tesla K20X GPUs</td>
<td>4 s</td>
<td>74 s</td>
<td>331 s</td>
<td>410 s</td>
</tr>
<tr>
<td>TachyonL-OptiX [1]</td>
<td>256 XK7 Tesla K20X GPUs</td>
<td>7 s</td>
<td>110 s</td>
<td>171 s</td>
<td>288 s</td>
</tr>
<tr>
<td>Tachyon [1]</td>
<td>256 XE6 CPUs</td>
<td>7 s</td>
<td>160 s</td>
<td>1,374 s</td>
<td>1,541 s</td>
</tr>
<tr>
<td>Tachyon [1]</td>
<td>512 XE6 CPUs</td>
<td>13 s</td>
<td>211 s</td>
<td>808 s</td>
<td>1,032 s</td>
</tr>
</tbody>
</table>

20 M atom chromatophore patch
Omnidirectional Stereoscopic Ray Tracing

- Ray trace 360° images and movies for Desk and VR HMDs: Oculus, Vive, Cardboard
- Stereo spheremaps or cubemaps allow very high-frame-rate interactive OpenGL display
- AO lighting, depth of field, shadows, transparency, curved geometry, …
- Summit 6x Tesla V100 GPU nodes:
  - Render many omni-stereo viewpoints, no acceleration structure rebuilds, tens of frames/sec per-node!
  - OptiX multi-GPU rendering, NVLink compositing and data distribution, etc…
  - Future: AI for warping between views


Goal: Intuitive interactive viz. in crowded molecular complexes

Results from 64 M atom, 1 μs sim!

Close-up view of chloride ions permeating through HIV-1 capsid hexameric centers
Planetarium Dome Master Projections
Technology Opportunities and Collaborations

- **Supercomputer Centers, Cray, IBM**
  - Remote visualization
  - Performance, power profiling and optimization
- **NVIDIA**
  - **GPU computing**
  - Ray tracing
  - Remote visualization
  - ARM, Tablets, power profiling and optimization
- **Intel**
  - x86, Xeon Phi optimization
  - Ray tracing
- **Amazon**
  - Cloud deployment of VMD/NAMD, related tools
  - Remote visualization
- **Universities:**
  - G. Fiorin, J. Henin, Toni Giorgino, collective variables
  - T. Ertl, U. Stuttgart: visualization algorithms
  - M. Kuttel, U. Cape Town: visualization and analysis
  - W. Sherman, Indiana U.: VR HMDs, visualization

Energy efficiency: ARM+GPU
VMD is a Platform for Developing Research Tools
Over 110 VMD Plugins, Half Developed by Users

- VMD user-extensible scripting w/ Tcl/Tk, Python
- User-developed plugins:
  - Alanine Scanning
  - Collective Variable Analyzer
  - Clustering Tool
  - Carbon Nanostructure Builder
  - TorsionPlot
  - RMSD Trajectory Tool
  - Many others…
QwikMD: Guided MD Simulation and Training

- Smooths initial learning curve (non-expert users)
- Training: used in 4 Center workshops to-date
- Speed up tedious simulation preparation tasks (expert users)
- Reproducibility: detailed log of all steps
- Interactive preparation, simulation, and analysis

## Selected VMD Plugins: Center Developed, and **User Developed**

<table>
<thead>
<tr>
<th>Analysis</th>
<th>Modeling</th>
<th>Visualization</th>
<th>Collaboration</th>
</tr>
</thead>
<tbody>
<tr>
<td>APBSRun, CatDCD, Contact Map, GoFrgui, HeatMapper, ILSTools, IRSpecGUI, MultiSeq, NAMD Energy, NAMD Plot, NetworkView, NMWiz, ParseFEp, PBCTools, PMEpot, PropKa GUI, RamaPlot, RMSD Tool, RMSD Trajectory Tool, RMSD Visualizer Tool, Salt Bridges, Sequence Viewer, Symmetry Tool, Timeline, TorsionPlot, VolMap</td>
<td>AutoIonize, AutoPSF, Chirality, Cionize, Cispeptide, CGTools, Dowser, fftK, Inorganic Builder, MDFF, Membrane, Merge Structs, Molecufacture, Mutator, Nanotube, Psfgen, RESPTool, RNAView, Solvate, SSRestrains, TopoTools</td>
<td>Clipping Plane Tool, Clone Rep, DemoMaster, Dipole Watcher, Intersurf, Navigate, NavFly, MultiMolAnim, Color Scale Bar, Remote, Palette Tool, ViewChangeRender, ViewMaster, Virtual DNA Viewer, VMD Movie Maker</td>
<td>Remote Control, Data Import and Plotting, Data Import, Multiplot, PDBTool, MultiText, Externally Hosted Plugins and Extensions, Check sidechains, MultiMSMS, Interactive Essential Dynamics, Mead Ionize, Clustering Tool, iTrajComp, Swap RMSD, Intervor, SurfVol, vmdICE</td>
</tr>
</tbody>
</table>

### Simulation

- AlaScan, AutoIMD, IMDMenu, NAMD GUI, NAMD Server, QMTool

### 75 MolFile I/O Plugins:

- structure, trajectory, sequence, and density map

http://www.ks.uiuc.edu/Research/vmd/plugins/
Example VMD Visualization and Analysis Plugins

**Bendix**
Dahl ACE, Chavent M and Sansom MSP  

**Normal Mode Wizard**
Bakan A, Meireles LM, Bahar I  
Making Our Research Tools Easily Accessible

• Cloud based deployment
  – Full virtual machines (known as “AMI” in Amazon terminology)
  – Amazon AWS EC2 GPU-accelerated instances:
    http://www.ks.uiuc.edu/Research/cloud/

• Docker “container” images available in NVIDIA NGC registry
  – Users obtain Docker images via registry, download and run on the
    laptop, workstation, cloud, or supercomputer of their choosing
  – https://ngc.nvidia.com/registry/

Molecular dynamics-based refinement and validation for sub-5 Å cryo-electron microscopy
maps. Abhishek Singhary, Ivan Teo, Ryan McGreevy, John E. Stone, Jianhua Zhao, and Klaus
QwikMD-integrative molecular dynamics toolkit for novices and experts. Joao V. Ribeiro, Rafael C.
Reports, 6:26536, 2016.
High performance molecular visualization: In-situ and parallel rendering with EGL. John E. Stone,
Peter Messmer, Robert Sisneros, and Klaus Schulten. 2016 IEEE International Parallel and Distributed
Molecular Dynamics Flexible Fitting (MDFF)

X-ray crystallography

APS at Argonne

MDFF

Electron microscopy

FEI microscope

ORNL Titan

Structural Route to the all-atom HIV-1 Capsid


Crystal structures of separated hexamer and pentamer

High res. EM of hexameric tubule, tomography of capsid, all-atom model of capsid by MDFF w/ NAMD & VMD, NSF/NCSA Blue Waters computer at Illinois

Evaluating Quality-of-Fit for Structures Solved by Hybrid Fitting Methods

Compute Pearson correlation to evaluate quality-of-fit between a reference cryo-EM density map and a simulated density map from an all-atom structure.
MDFF Density Map Algorithm

- Build spatial acceleration data structures, optimize data for GPU

- Compute 3-D density map:

\[
\rho(\vec{r}; \vec{r}_1, \vec{r}_2, \ldots, \vec{r}_N) = \sum_{i=1}^{N} e^{-\frac{|\vec{r} - \vec{r}_i|^2}{2\alpha^2}}
\]

- Truncated Gaussian and spatial acceleration grid ensure linear time-complexity

3-D density map lattice point and the neighboring spatial acceleration cells it references
3-D density map decomposes into 3-D grid of 8x8x8 tiles containing CC partial sums and local CC values.

Small 8x8x2 CUDA thread blocks afford large per-thread register count, shared memory.

Each thread computes 4 z-axis density map lattice points and associated CC partial sums.

Padding optimizes global memory performance, guaranteeing coalesced global memory accesses.

Fusion of density and CC calculations into a single CUDA kernel!!!

Spatial CC map and overall CC value computed in a single pass.

Threads producing results that are used.

Inactive threads, region of discarded output.

Single Pass MDFF GPU Cross-Correlation.
Parallel MDFF Cross Correlation Analysis on Cray XK7

Rabbit Hemorrhagic Disease Virus (RHDV)

<table>
<thead>
<tr>
<th></th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traj. frames</td>
<td>10,000</td>
</tr>
<tr>
<td>Structure component selections</td>
<td>720</td>
</tr>
<tr>
<td>Single-node XK7 (projected)</td>
<td>336 hours (14 days)</td>
</tr>
<tr>
<td>128-node XK7</td>
<td>3.2 hours 105x speedup</td>
</tr>
<tr>
<td>2048-node XK7</td>
<td>19.5 minutes 1035x speedup</td>
</tr>
</tbody>
</table>

Calculation of 7M CCs would take **5 years** using serial CPU algorithm!

**VMD Tesla V100 Cross Correlation Performance**

Rabbit Hemorrhagic Disease Virus: 702K atoms, 6.5Å resolution

Volta GPU architecture almost 2x faster than previous gen Pascal:

<table>
<thead>
<tr>
<th>Application and Hardware platform</th>
<th>Runtime, Speedup vs. Chimera, VMD+GPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chimera Xeon E5-2687W (2 socket) [1]</td>
<td>15.860s, 1x</td>
</tr>
<tr>
<td>VMD-CUDA IBM Power8 + 1x Tesla K40 [2]</td>
<td>0.488s, 32x 0.9x</td>
</tr>
<tr>
<td>VMD-CUDA Intel Xeon E5-2687W + 1x Quadro K6000 [1,2]</td>
<td>0.458s, 35x 1.0x</td>
</tr>
<tr>
<td>VMD-CUDA Intel Xeon E5-2698v3 + 1x Tesla P100</td>
<td>0.090s, 176x 5.1x</td>
</tr>
<tr>
<td>VMD-CUDA IBM Power8 “Minsky” + 1x Tesla P100</td>
<td>0.080s, 198x 5.7x</td>
</tr>
<tr>
<td><strong>VMD-CUDA Intel Xeon E5-2697Av4 + 1x Tesla V100</strong></td>
<td><strong>0.050s, 317x 9.2x</strong></td>
</tr>
<tr>
<td><strong>VMD-CUDA IBM Power9 “Newell” + 1x Tesla V100</strong></td>
<td><strong>0.049s, 323x 9.3x</strong></td>
</tr>
</tbody>
</table>


Clustering Analysis of Molecular Dynamics Trajectories: Requires I/O+Memory for All-Pairs of Trajectory Frames

Use of Node-Local Burst Buffers and Non-Volatile Memory DIMMs

- Perform viz+analysis in-transit in node-local SSDs, persistent memory NVDIMMs
- ORNL Summit I/O:
  - Parallel FS: 2.5 TB/s
  - Node-local PCIe “burst buffer” SSDs: 10+ TB/sec, 7PB capacity
- Plenty of capacity for full-detail MD trajectories, **could enable ~100x increase in temporal resolution** in cases where it would be valuable to the science
- Enable all-pairs trajectory clustering analyses and resulting visualizations
- Future systems with NVDIMMs (3D Xpoint, phase change memory) could eventually provide bandwidths approaching DRAM
- Use NVDIMMs w/ `mmap()`, **APIs like PMDK** to perform formerly-out-of-core calculations using persistent memory:
  - [https://github.com/pmem/pmdk](https://github.com/pmem/pmdk)
- Imagine future Summit-like machines w/ NVLink-connected GPUs w/ access to high-bandwidth persistent memory on each node
Trade FLOPS for Reduced I/O

ORNL Summit compute node:

- 6x Tesla V100 GPUs, 2x POWER9 CPUs
- GPUs Peak: ~46 DP TFLOPS, ~96 SP TFLOPS
- Peak IB rate per node: ~23GB/sec
- **Ratio of FLOPS vs. I/O:**
  - ~2,000 DP FLOPS/byte, ~4000 SP FLOPS/byte
  - ~16K FLOPS per FP word

Unconventional approach: Recompute to avoid I/O
Computing+Visualizing Molecular Orbitals

- Movies of simulation trajectories provides insight into results
- QM, and hybrid (QM/MM) MO visualizations historically done from huge “cube” files, impractical
- Store QM wavefunctions + Gaussian basis set, only 10s of KB per stored timestep compared to 100s of MB
- Recompute MO grid on-the-fly from QM basis set, huge decrease in RAM+I/O in exchange for heavy FP arithmetic

http://dx.doi.org/10.1038/nmeth.4638

MO Kernel for One Grid Point (Naive C)

for (at=0; at<numatoms; at++) {
    int prim_counter = atom_basis[at];
    calc_distances_to_atom(&atompos[at], &xdist, &ydist, &zdist, &dist2, &xdiv);
    for (contracted_gto=0.0f, shell=0; shell < num_shells_per_atom[at]; shell++) {
        int shell_type = shell_symmetry[shell_counter];
        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];
            contracted_gto += contract_coeff * \( \exp(-exponent \times \text{dist}^2) \);
            prim_counter += 2;
        }
        for (tmpshell=0.0f, j=0, zdp=1.0f; j<=shell_type; j++, zdp*=zdist) {
            int imax = shell_type - j;
            for (i=0, ydp=1.0f, xdp=pow(xdist, imax); i<=imax; i++, ydp*=ydist, xdp*=xdiv)
                tmpshell += wave_f[i &func++] \times xdp \times ydp \times zdp;
        }
        value += tmpshell \times \text{contracted}_\text{gto};
        shell_counter++;
    }
} ....
MO GPU Parallel Decomposition

MO 3-D lattice decomposes into 2-D slices (CUDA grids)

Small 8x8 thread blocks afford large per-thread register count, shared memory

Each thread computes one MO lattice point.

Padding optimizes global memory performance, guaranteeing coalesced global memory accesses

Grid of thread blocks

Threads producing results that are used

Threads producing results that are discarded

Lattice computed using multiple GPUs

GPU 2
GPU 1
GPU 0
VMD C\textsubscript{60} MO Viz. Perf, 516x519x507 Grid: @ .13s/frame, avoids 3.8GB/s I/O per-node

<table>
<thead>
<tr>
<th>Hardware platform</th>
<th>Runtime,</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM Power8 (ORNL ‘crest’) + 1x Tesla K40 [1]</td>
<td>3.49s,</td>
<td>1.0x</td>
</tr>
<tr>
<td>Intel Xeon E5-2697Av4 + 1x Tesla V100</td>
<td>0.610s,</td>
<td>5.7x</td>
</tr>
<tr>
<td>Intel Xeon E5-2697Av4 + 2x Tesla V100</td>
<td>0.294s,</td>
<td>11.8x</td>
</tr>
<tr>
<td>Intel Xeon E5-2697Av4 + 3x Tesla V100</td>
<td>0.220s,</td>
<td>15.9x</td>
</tr>
<tr>
<td>IBM Power9 “Newell” + 1x Tesla V100</td>
<td>0.394s,</td>
<td>8.8x</td>
</tr>
<tr>
<td>IBM Power9 “Newell” + 2x Tesla V100</td>
<td>0.207s,</td>
<td>16.8x</td>
</tr>
<tr>
<td>IBM Power9 “Newell” + 3x Tesla V100</td>
<td>0.151s,</td>
<td>23.1x</td>
</tr>
<tr>
<td>IBM Power9 “Newell” + 4x Tesla V100</td>
<td>0.130s,</td>
<td>26.8x</td>
</tr>
</tbody>
</table>

MO Kernel Structure, Opportunity for NRTC JIT…
Data-driven execution, but representative loop trip counts in (...)

Loop over atoms (1 to ~200) {
  Loop over electron shells for this atom type (1 to ~6) {
    Loop over primitive functions for this shell type (1 to ~6) {
      Small loop trip counts result in significant loop overhead. Runtime kernel generation and NVRTC JIT compilation can achieve in a large (1.8x!) speed boost via loop unrolling, constant folding, elimination of array accesses, …
    }
    Loop over angular momenta for this shell type (1 to ~15) {}
  }
}


**Molecular Orbital Computation and Display Process**

**Runtime Kernel Generation, NVRTC Just-In-Time (JIT) Compilation**

**One-time initialization**

- Initialize Pool of GPU Worker Threads

**Read QM simulation log file, trajectory**

- Preprocess MO coefficient data eliminate duplicates, sort by type, etc…

**Generate/compile basis set-specific CUDA kernel**

**For each trj frame, for each MO shown**

- For current frame and MO index, retrieve MO wavefunction coefficients

**Compute 3-D grid of MO wavefunction amplitudes using basis set-specific CUDA kernel**

- Extract isosurface mesh from 3-D MO grid

- Render the resulting surface
for (shell=0; shell < maxshell; shell++) {
    float contracted_gto = 0.0f;

    // Loop over the Gaussian primitives of CGTO
    int maxprim = const_num_prim_per_shell[shell_counter];
    int shell_type = const_shell_symmetry[shell_counter];
    for (prim=0; prim < maxprim; prim++) {
        float exponent = const_basis_array[prim_counter];
        float contract_coeff = const_basis_array[prim_counter + 1];
        contracted_gto += contract_coeff * expf(-exponent*dist2);
        prim_counter += 2;
    }
}

contracted_gto = 1.832937 * expf(-7.868272*dist2);
contracted_gto += 1.405380 * expf(-1.881289*dist2);
contracted_gto += 0.701383 * expf(-0.544249*dist2);

General loop-based data-dependent MO CUDA kernel

Runtime-generated data-specific MO CUDA kernel compiled via CUDA NVRTC JIT...

1.8x Faster
Vectors of wavefunction amplitudes are computed using hardware SIMD instructions.

Each CPU thread computes 1, 4, 8, 16 MO lattice points per loop iteration: C, SSE, AVX2 or AVX-512ER.

MO 3-D lattice decomposes into 2-D slices.

Lattice decomposed across many CPU threads.

Padding: Inactive SIMD lanes or region of discarded output used to guarantee aligned vector loads+stores.

SIMD lanes producing results that are used.
AVX-512ER MO CGTO Loop

int maxprim = num_prim_per_shell[shell_counter];
int shelltype = shell_types[shell_counter];
for (prim=0; prim<maxprim; prim++) {
    float exponent = basis_array[prim_counter];
    float contract_coeff = basis_array[prim_counter + 1];

    // contracted_gto += contract_coeff * exp(exponent*dist2);
    __m512 expval = _mm512_mul_ps(_mm512_set1_ps(exponent * MLOG2EF), dist2);

    // expf() approximation required, use (base-2) AVX-512ER instructions…
    __m512 retval = _mm512_exp2a23_ps(expval);

    m512_ctmp = _mm512_mul_ps(_mm512_set1_ps(contract_coeff), retval);
    contracted_gto = _mm512Add_ps(contracted_gto, m512_ctmp);
    prim_counter += 2;
}
Performance of AVX-512ER Instrinsics vs. Autovectorization on KNL: Small 172x173x169 Grid

- Intel C++ ‘15 autovectorization (fail): 220+ sec
- Hand-coded SSE2 w/ existing thread scheme: 48.5 sec
- Hand-coded AVX-512ER w/ existing thread scheme: 6.3 sec
- Hand-coded AVX-512ER, refactoring thread pool: 0.2 sec
- Hand-coded AVX-512ER tuned thread pool: 0.131 sec
- Hand-coded AVX-512ER+FMA tweaks: 0.107 sec

Further improvement will require attention to details of cache behaviour and further tuning of low-level threading constructs for Xeon Phi/KNL.
Acknowledgements

- Theoretical and Computational Biophysics Group, University of Illinois at Urbana-Champaign
- CUDA Center of Excellence, University of Illinois at Urbana-Champaign
- NVIDIA CUDA and OptiX teams
- Intel OSPRay team
- Funding:
  - DOE INCITE, ORNL Titan: DE-AC05-00OR22725
  - NIH support: 9P41GM104601, 5R01GM098243-02
“When I was a young man, my goal was to look with mathematical and computational means at the inside of cells, one atom at a time, to decipher how living systems work. That is what I strived for and I never deflected from this goal.” – Klaus Schulten
Related Publications
http://www.ks.uiuc.edu/Research/gpu/


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


- **Winner of the SC’14 Visualization and Data Analytics Showcase**


- **Unlocking the Full Potential of the Cray XK7 Accelerator.** M. D. Klein and J. E. Stone. Cray Users Group, Lugano Switzerland, May 2014.


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


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


Related Publications

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


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

- **Adapting a message-driven parallel application to GPU-accelerated clusters.**

- **GPU acceleration of cutoff pair potentials for molecular modeling applications.**

- **GPU computing.**

- **Accelerating molecular modeling applications with graphics processors.**

- **Continuous fluorescence microphotolysis and correlation spectroscopy.**
Related Publications
http://www.ks.uiuc.edu/Research/gpu/


Related Publications

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

• **Adapting a message-driven parallel application to GPU-accelerated clusters.**

• **GPU acceleration of cutoff pair potentials for molecular modeling applications.**

• **GPU computing.**

• **Accelerating molecular modeling applications with graphics processors.**

• **Continuous fluorescence microphotolysis and correlation spectroscopy.**