Using GPUs for Live Data Analysis

GPGPUs *FTW!*

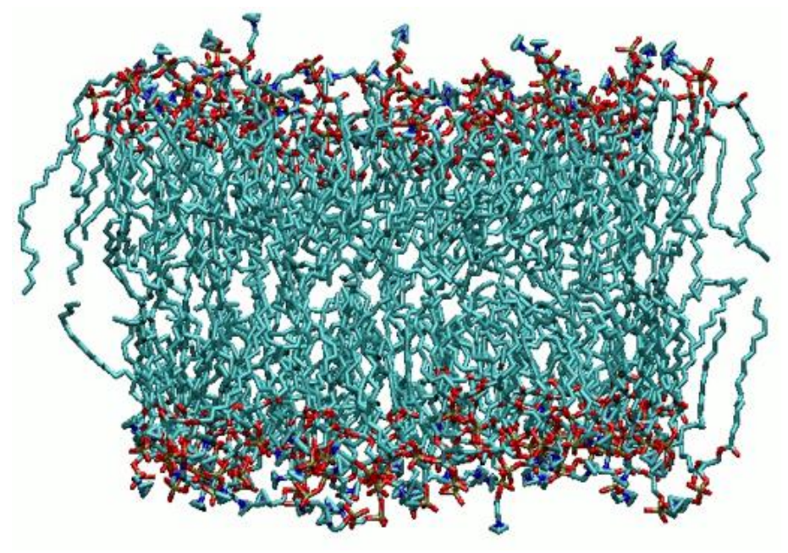
Chris McClanahan | Dr. Matthew Wolf

College of Computing, Georgia Institute of Technology, Atlanta GA

Introduction

- Problem
  - Scientific simulations are computationally expensive and generate masses of data that needs to be filtered and analyzed
- Goal
  - Perform variety of statistical tests, analysis, and visualizations of live streaming simulation data
  - Provide interactive analysis and visualization tools
  - Enable lightweight thin-client visualizations
- Solution
  - Graphics Processing Units provide an inexpensive way to add additional numeric processing power to a machine
  - Use GPUs for opportunistic and fast data analysis and visualization before/while writing simulation data to disk
  - Use nVidia's CUDA GPU parallel programming language

Related Works



- VMD
  - Visual Molecular Dynamics
  - Provides a nice visualization suite
  - Supports multiple views of data
  - Partly uses the GPU - only for rendering
  - Doesn't operate on live streaming data
  - Can only load certain file types
- PyMol
  - Python Molecular Dynamics Visualization
  - Similar to VMD, but in Python instead of C++
- The "Old" SmartPointer
  - Molecular Dynamics Visualization Service
  - Starting point for this research
  - Re-work code to map to the GPU

CUDA Acceleration

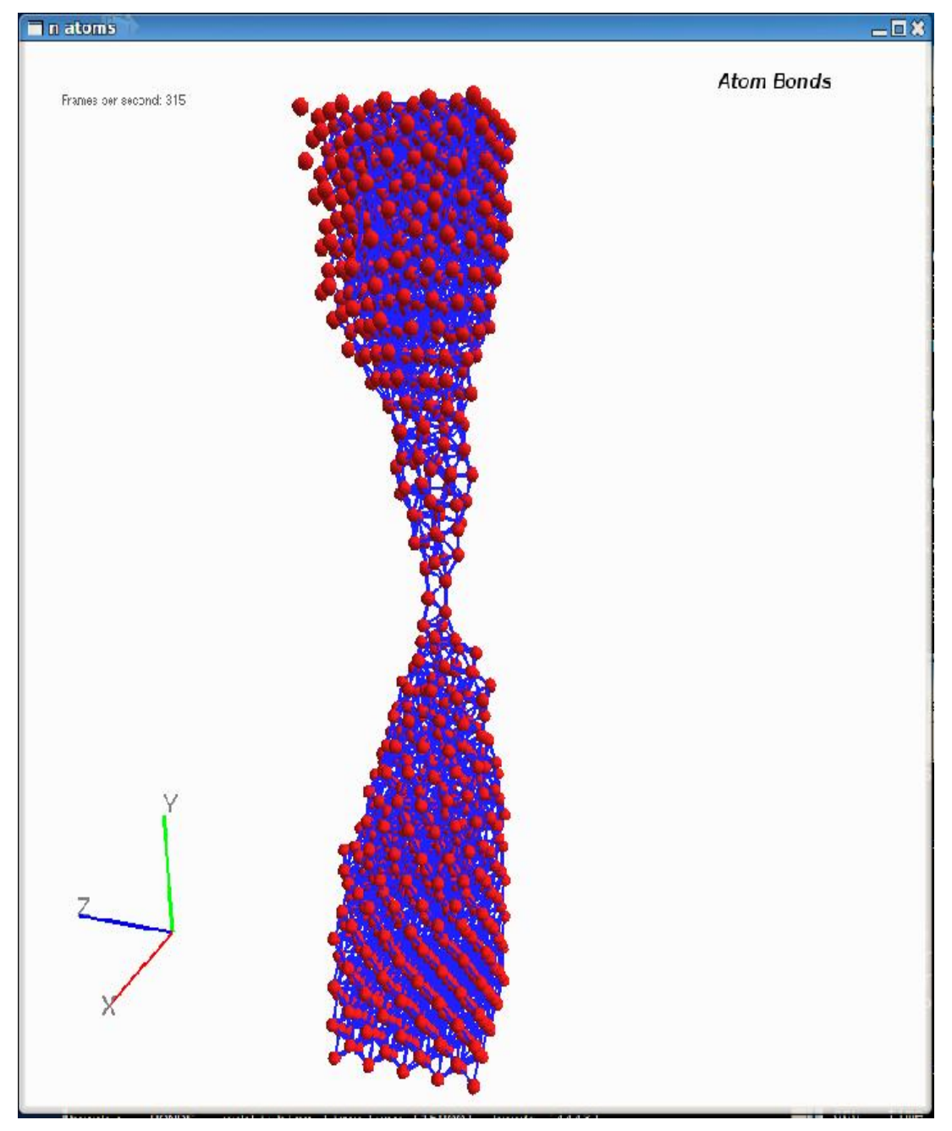
- Compute Unified Device Architecture
  - Harness the GPU's "many-core" architecture
  - Each core running thousands of threads simultaneously
- Atomic Bonds Calculation
  - Find all atoms between *Rmin* and *Rmax* radius
  - CPU ~ $O(n^2)$
    - For *N* atoms, find the distance to each *N* neighbors
  - GPU ~ $O(n)$
    - Spawn one GPU thread per atom to find neighbors
  - Save neighbor distances in a histogram
    - Useful for displaying properties of elements in simulation
- Autocorrelation Calculation
  - Leverage the "Thrust" CUDA template library
  - Vectorize statistics calculations
  - Support for various statistics gathering (Currently 2)
    - Correlation Coefficient (Pearson's r)
    - Dissimilarity Coefficient
  - Track changes to atom positions over time

Our Approach

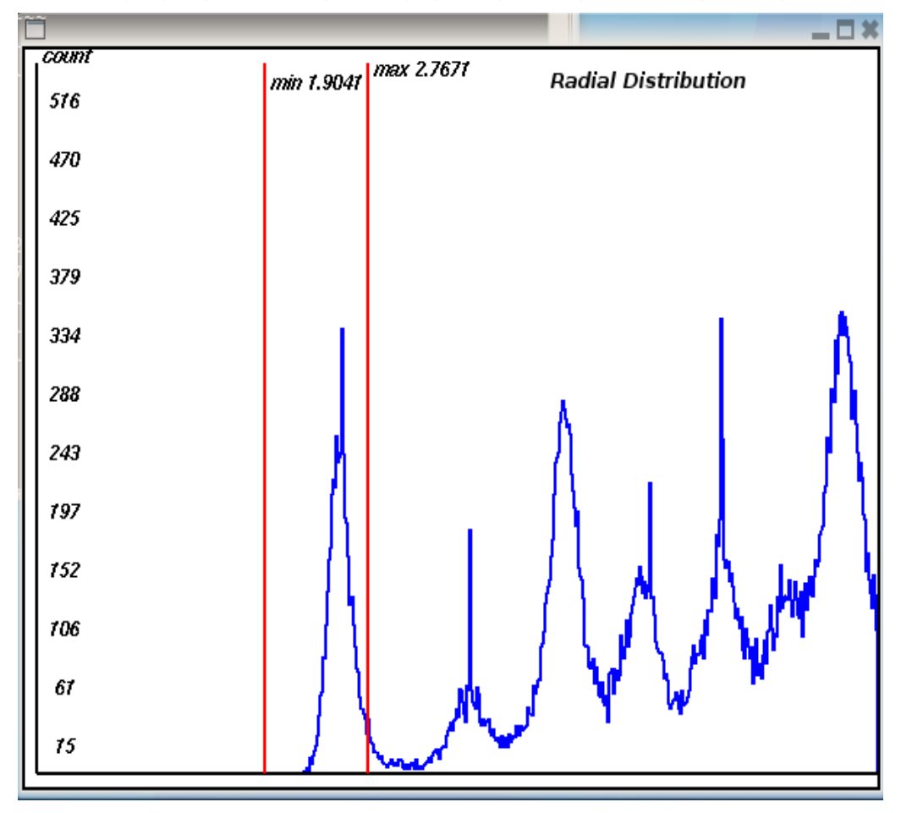
- Provide a modular monitoring service
  - Build off existing SmartPointer code base
  - Leverage existing FFS/LGS message passing framework
  - Can use in existing scientific work-flows as an additional service
  - Use GPU acceleration for scientific data analysis and visualization
  - Process data in-line - before / while writing to disk
- Example: Molecular Dynamics Systems
  - Monitor live streaming simulation data
    - 3D coordinates of atom positions
  - Enable real-time interactive:
    - *2D Radial distribution* - Visualize and control the bonds calculation radius
    - *3D Bonds Visualization* - OpenGL rendering of atoms and computed neighbors
    - *Autocorrelation plot* - Graph changes to XYZ positions over time
    - *Mobile Client* - Android client port of the 3D bonds visualization

Our System

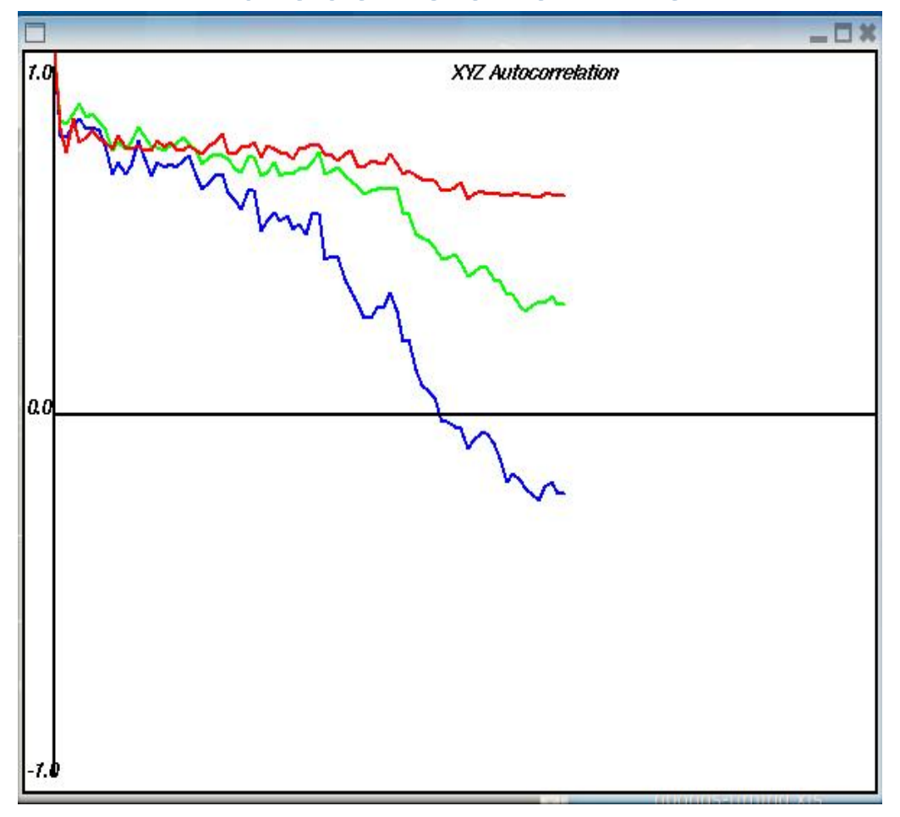
3D Atomic Bonds Visualization



Radial Distribution of Atoms

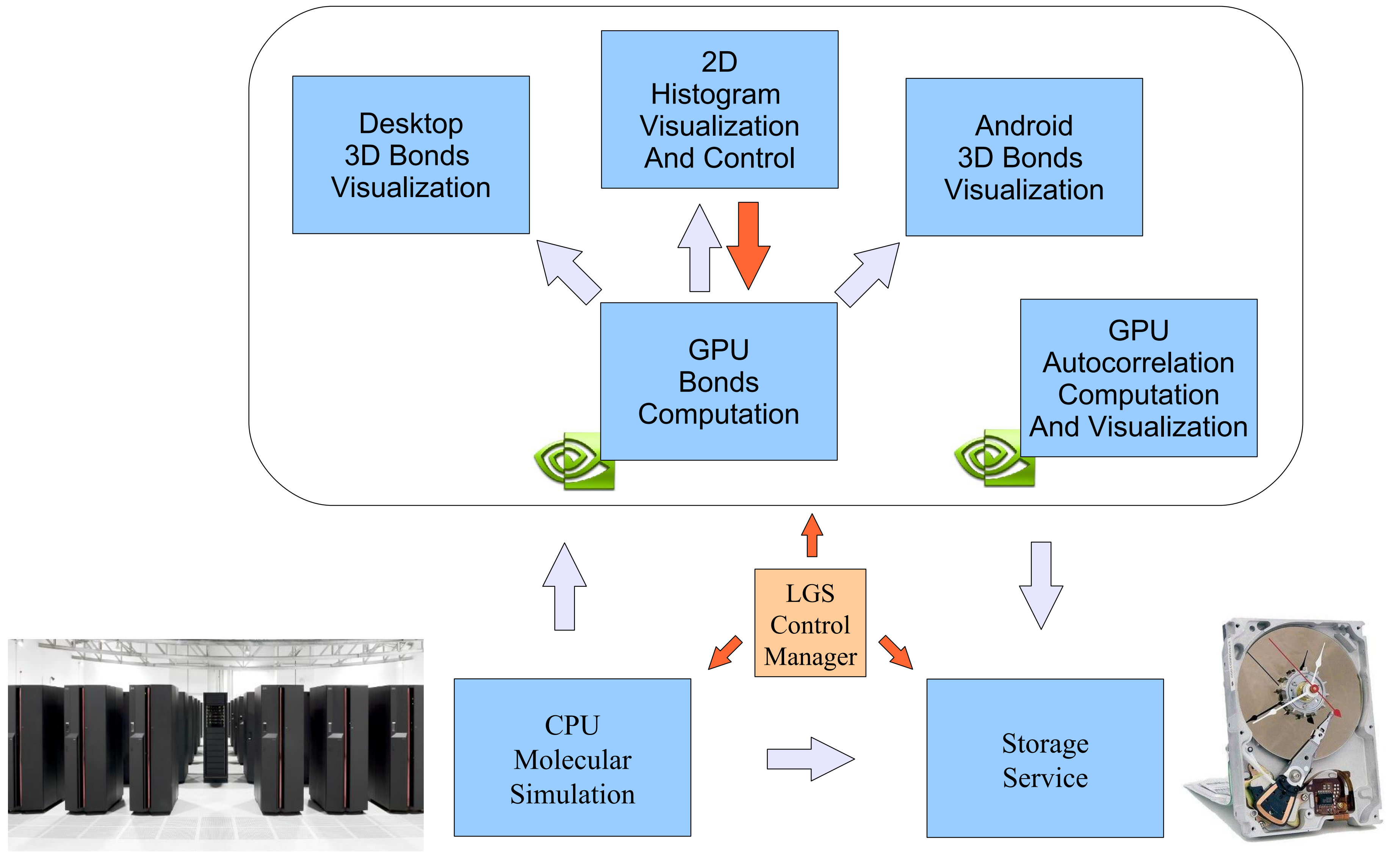


Autocorrelation Plot



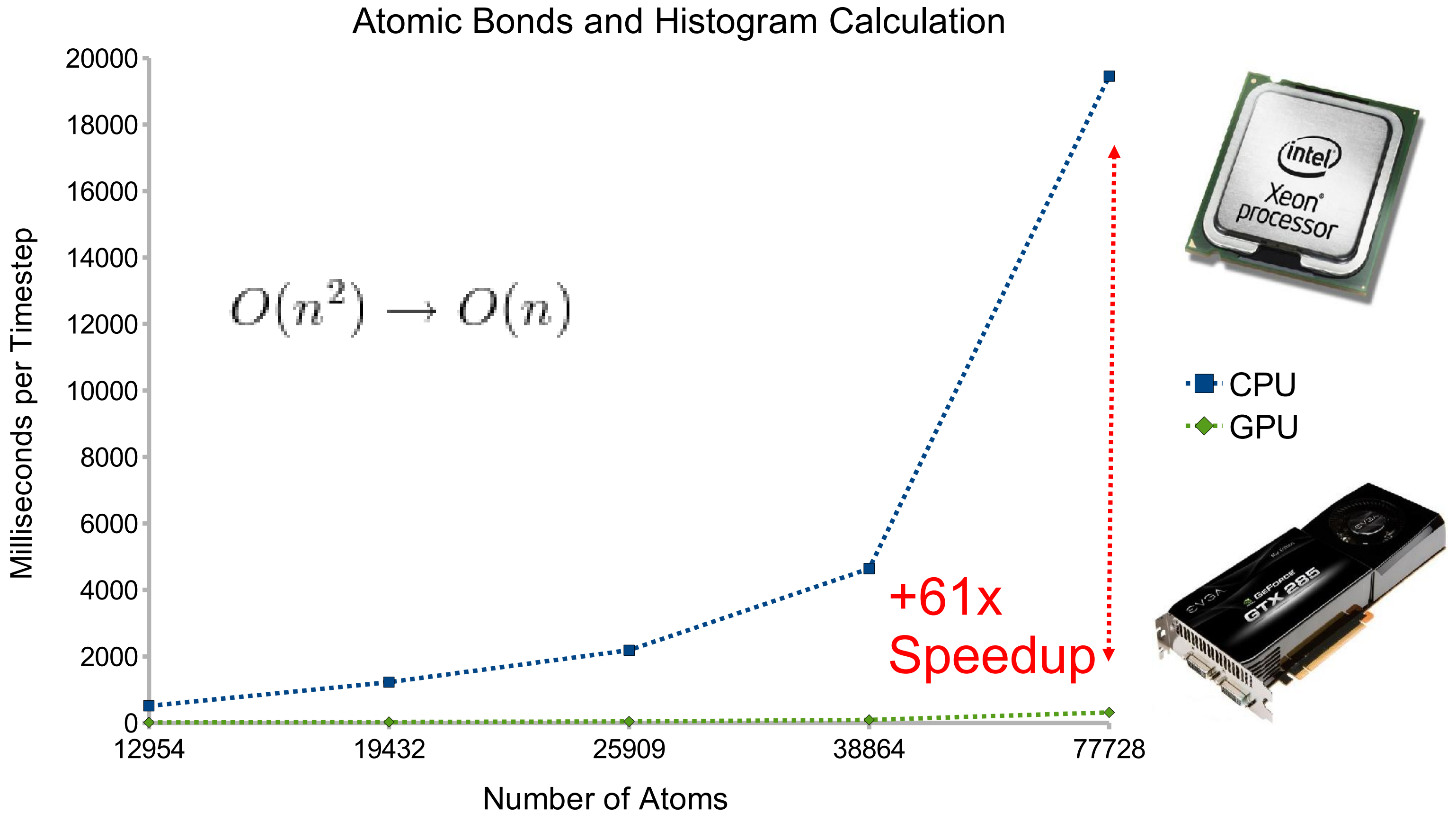
- All computation performed on the GPU
  - Minimal CPU overhead for rich analysis and visualizations
  - Provide insight into simulation without slowing it down
- Service Oriented
  - Each module can be on a separate machine

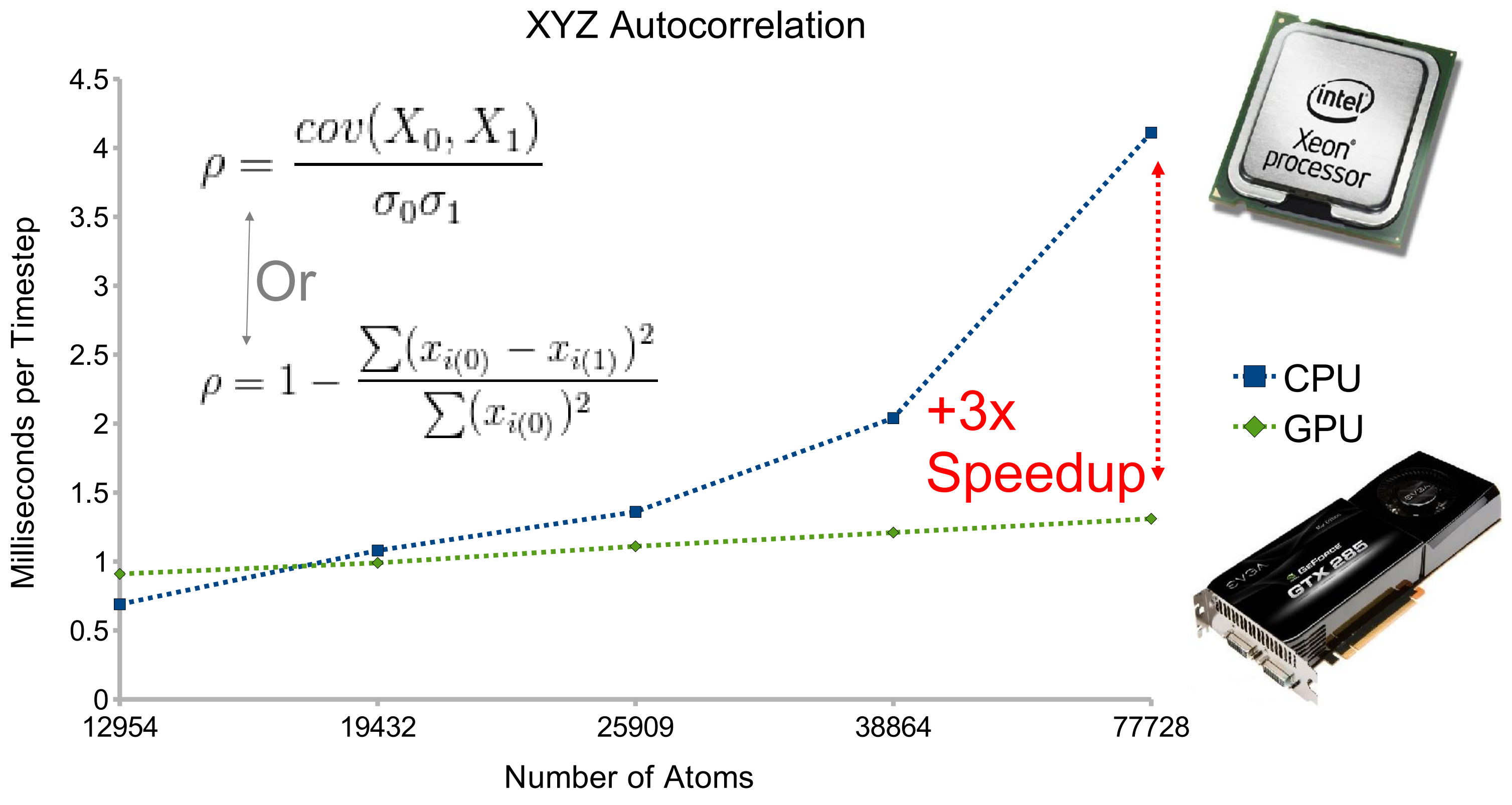
Our Framework


Desktop 3D Bonds Visualization
2D Histogram Visualization And Control
Android 3D Bonds Visualization
GPU Bonds Computation
GPU Autocorrelation Computation And Visualization
LGS Control Manager
CPU Molecular Simulation
Storage Service

Results

- Bonds Computation on the GPU
  - Process each atom in a separate GPU thread
  - Calculate all-pairs distances between atoms
  - Compare thresholds and generate a distance histogram
  - More than 61x speedup over the CPU!
- Autocorrelation Computation on the GPU
  - Parallel calculations: mean / variance / standard deviation / correlation
  - Two types of correlation functions implemented
  - More than 3x speedup over the CPU!





Future Work

- Add more analysis modules
  - Entropy calculations
  - Other autocorrelation functions
  - Data compression and hashing
- Explore CUDA performance tweaks
  - Use more shared memory and 'float4' optimizations
  - Support multi-GPU configurations