Stereographic Visualization of Molecular Configurations in a CAVE

G'Nita R. Wright[1], Sanjay Kodiyalam[2], Amitava Jana[2]
Department of [1]Biological Sciences / [2]Mechanical Engineering
Southern University, Baton Rouge, Louisiana.

OUTLINE

- Motivation
- Objectives
- The CAVE at SUBR’s College of Engineering (CoE)
- Methodology
- Results
- Conclusion
- Future Work
- Acknowledgements
- Questions

Motivation

- Parallel atomistic simulations produce large scale (~million atom) configurations
- Need visual inspection at widely varying length scales to motivate and determine subsequent analysis
- CAVE is ideal for such inspection due to enhanced depth perception and data navigation ability

The Cave at SUBR

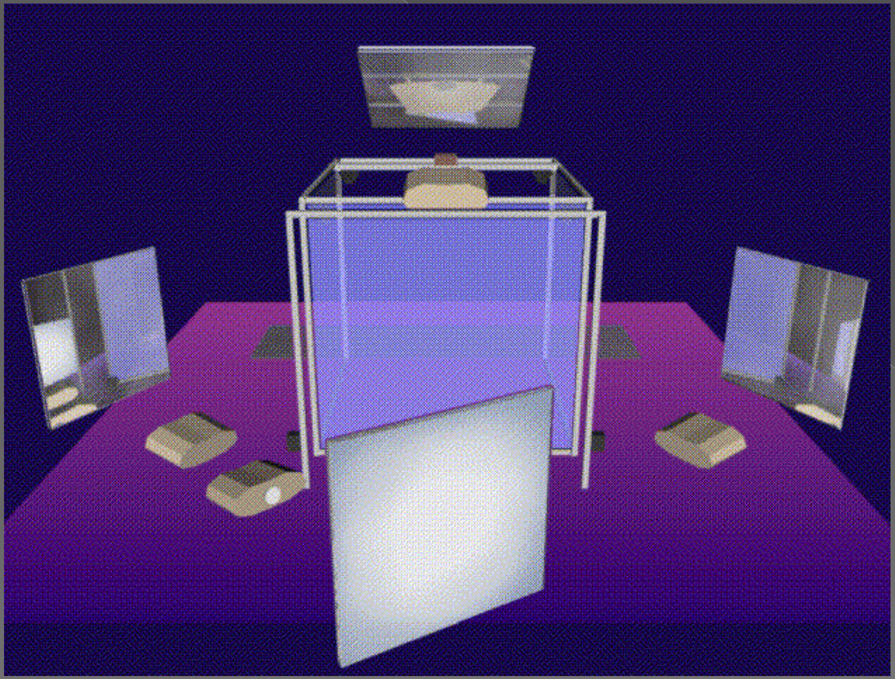


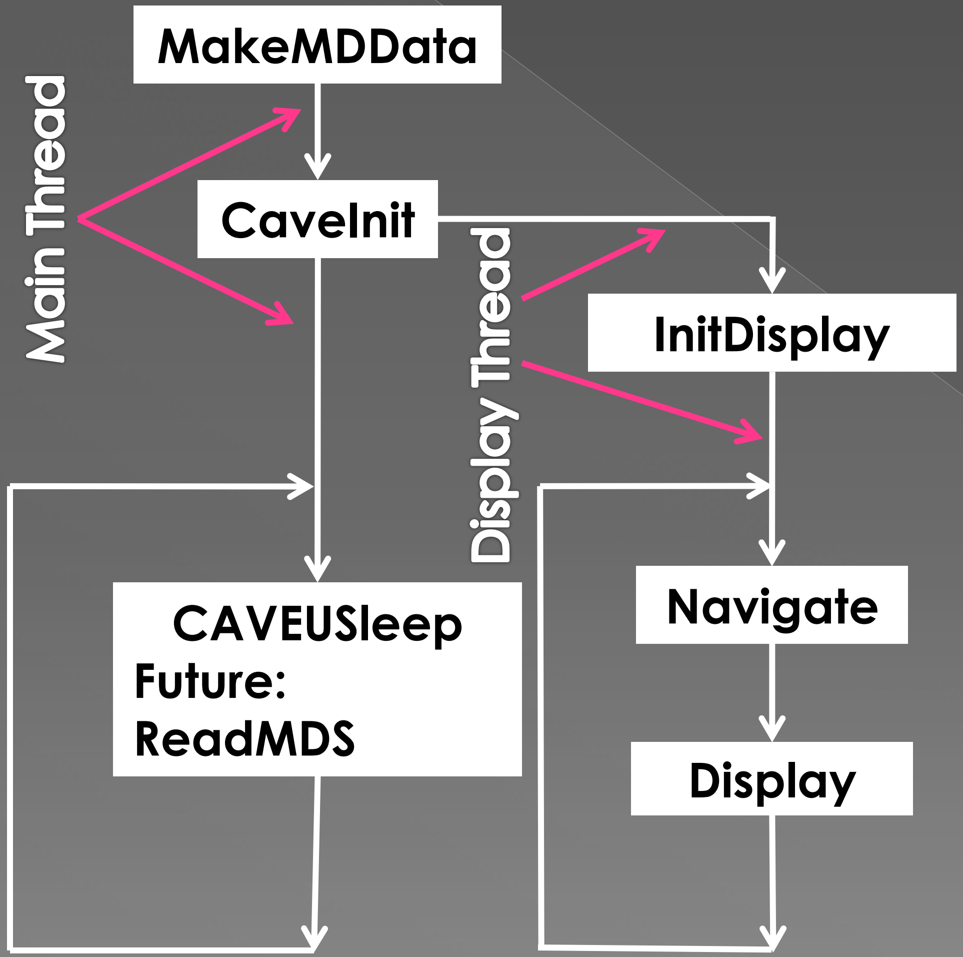
Figure 1. Schematic of the CAVE at SUBR showing the projectors, mirrors, and screens. (Fig. From ~/kenyon/conferences)

- 8ft cubed space with three screen-walls & floor serving as displays
- Images projected via mirrors setting optical distance = projector's throw
- Active stereo: Separate left & right eye images synchronized with eye-ware
- Position and orientation of two sensors (eye-ware and wand) tracked
- CAVE driven by 2-node cluster: Master (collects sensor info) & Display (drives projectors)
- Perspective transformations required for displays automatically carried out by CAVE-library using eye-ware sensor information
- All sensors' information accessible to the visualization application for use in updating the display

Objectives

- In the absence of simulation data construct Simple Cubic (SC) and Body Centered Cubic (BCC) lattice configurations
- Visualize configurations: Non-stereo on desktop development platform, stereo in CAVE
- On desktop: Compute the CPU Time required for the display versus the total # of atoms
- Identify future tasks to improve code's execution efficiency

Methodology: CAVE Library Use

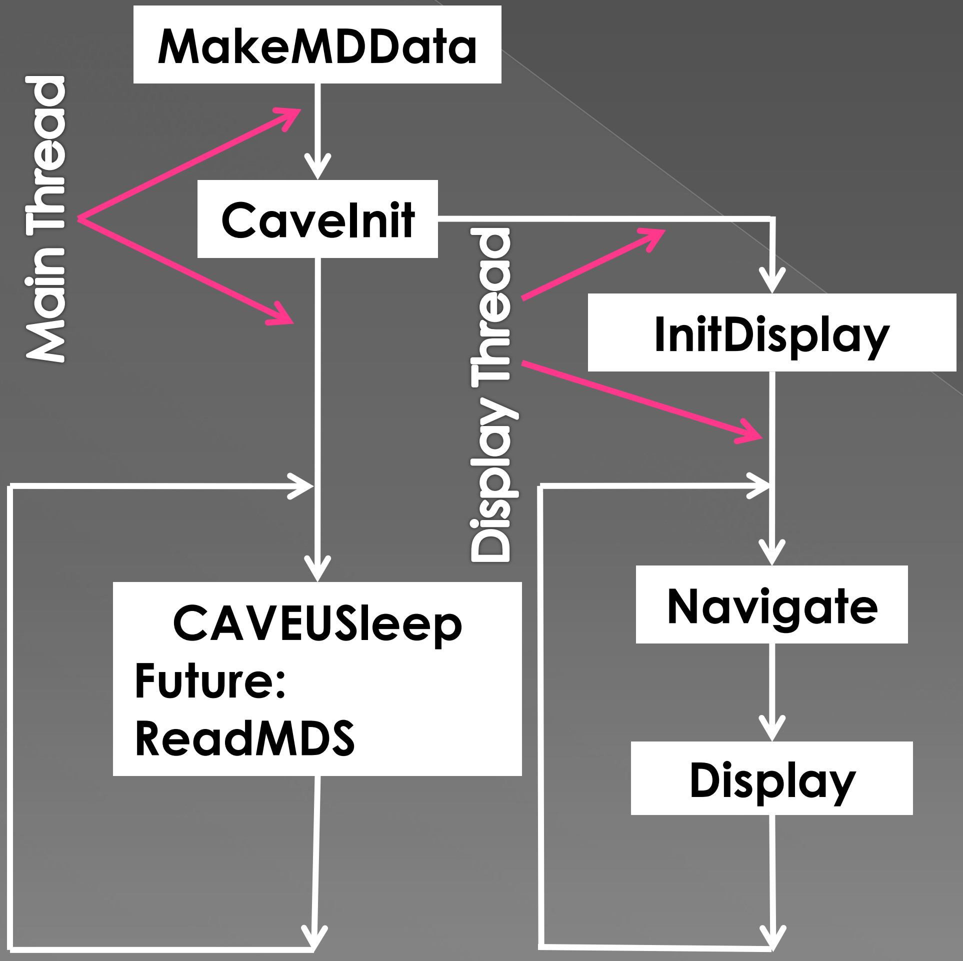


- Project structure determined by CAVE-Lib operation: Spawns one additional thread for each display (all synchronized); Main thread continues asynchronously

In the main thread:

- A function (MakeMDdata) creates linear arrays corresponding to atom type(integer), position and velocity (both double precision)
- Subsequently an infinite loop allows CPU to sleep and prevents termination; Reading in data from steps of molecular dynamics simulations (ReadMDS) or its analysis can be attempted in future

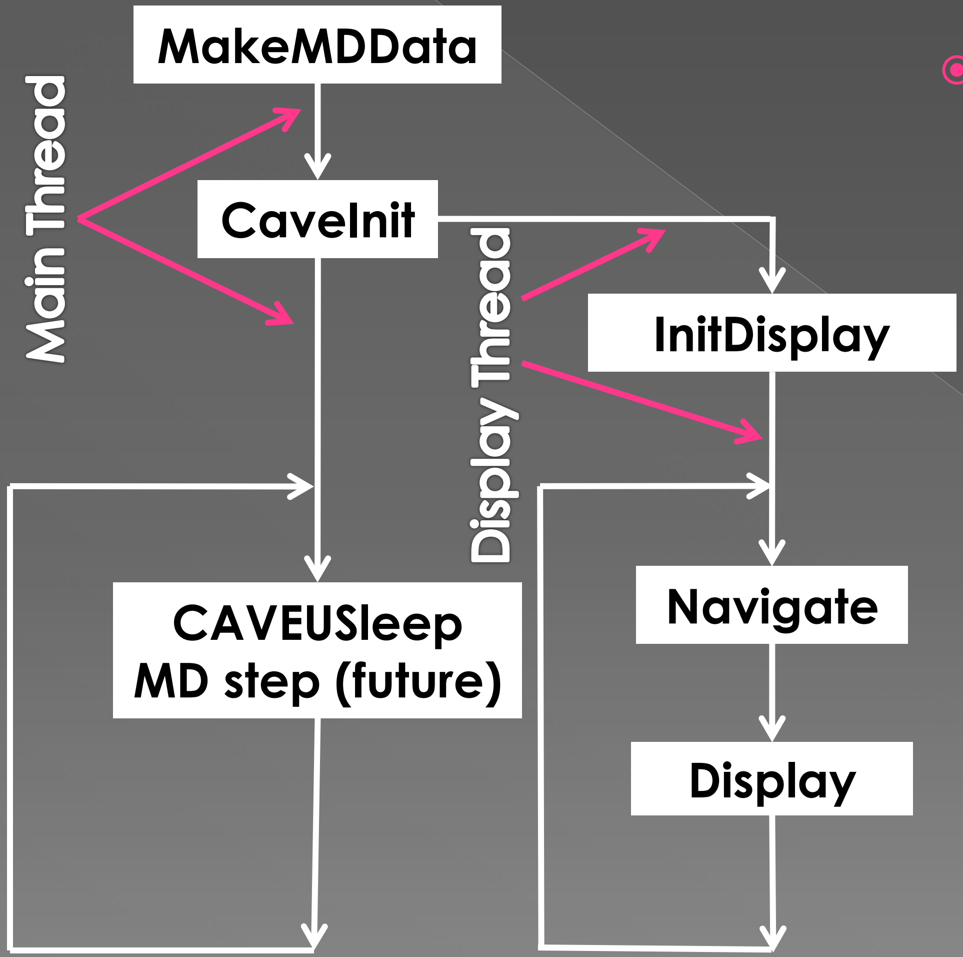
Methodology: CAVE Library Use



In each display thread: (Open GL & GLU library use)

- An initialization function (InitDisplay) sets a single directional light with ambient, diffuse, and specular qualities to enhance depth perception
- In (infinite) display loop a navigation function enables the translation, rotation, and scaling of entire virtual world using the wand [D. Driggs *et. al.* (2011)]

Methodology: CAVE Library Use



In each display thread: (Open GL & GLU library use)

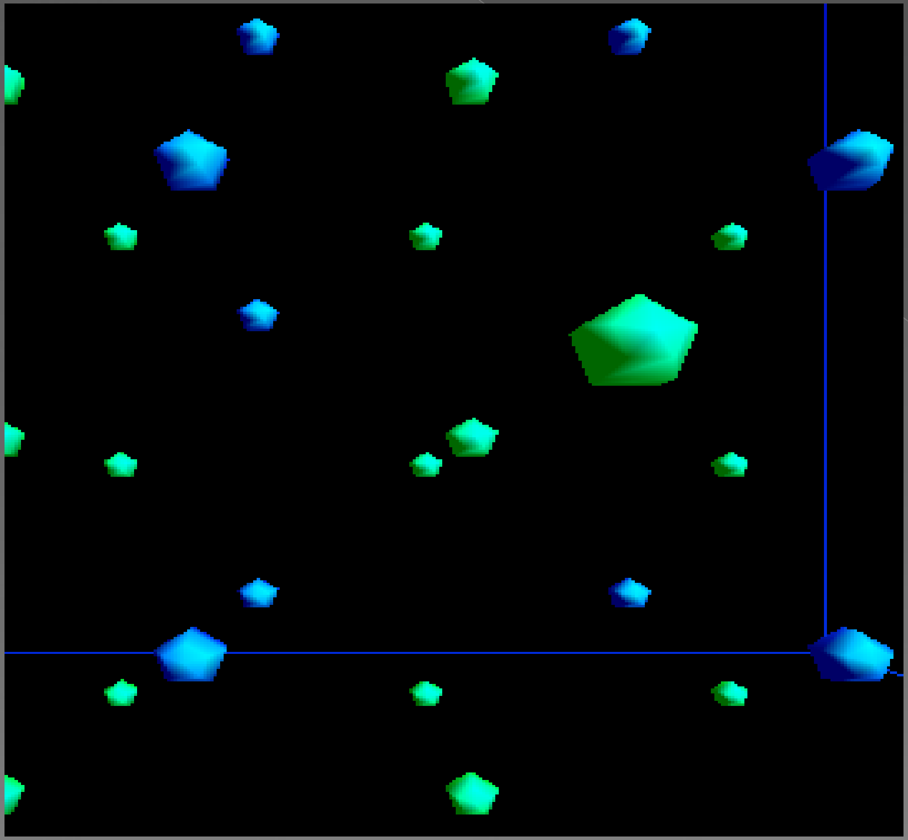
- Display function:
  - Color is set by glMaterial calls
  - Atoms displayed in a loop using gluSphere that creates sphere at current origin
  - Smoothness of sphere set by number of slices(longitudes) and stacks(latitudes
  - Positions are set using glTranslate before the call to gluSphere
  - glPushMatrix-glPopMatrix pair over each atom's display to enable independent setting of positions
  - Bonds are displayed as lines in double loop by setting GL_LINES
  - Double loop requires to determine nearest-neighbor atoms via inter-atomic distance compared to expected bond length
  - Only nearest-neighbor atoms of different type are bonded: Therefore none in SC lattice, 8/atom in BCC lattice

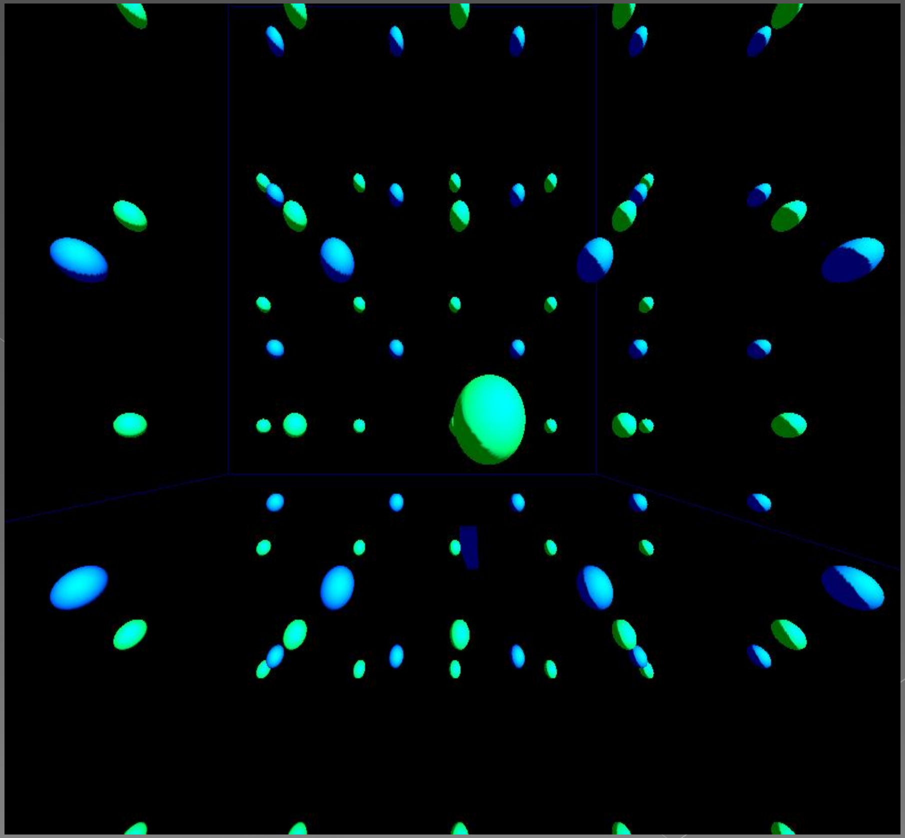
Methodology: Execution Timing

- Tests on the desktop development platform: single display in non-stereo “CAVE simulator” mode
- Windows-XP desktop having 2 GB of RAM and 4 CPU (2 x dual core Opterons, 2.4 GHz).
- Study to determine upper limit on the number of atoms while having *interactive* frame rate of 10 per second (= 0.1 seconds of (CPU) time per frame)
- Time per frame determined for display of atoms alone or bonds alone as different trends with increasing number of atoms are expected

Results: Only Atoms in BCC Lattice

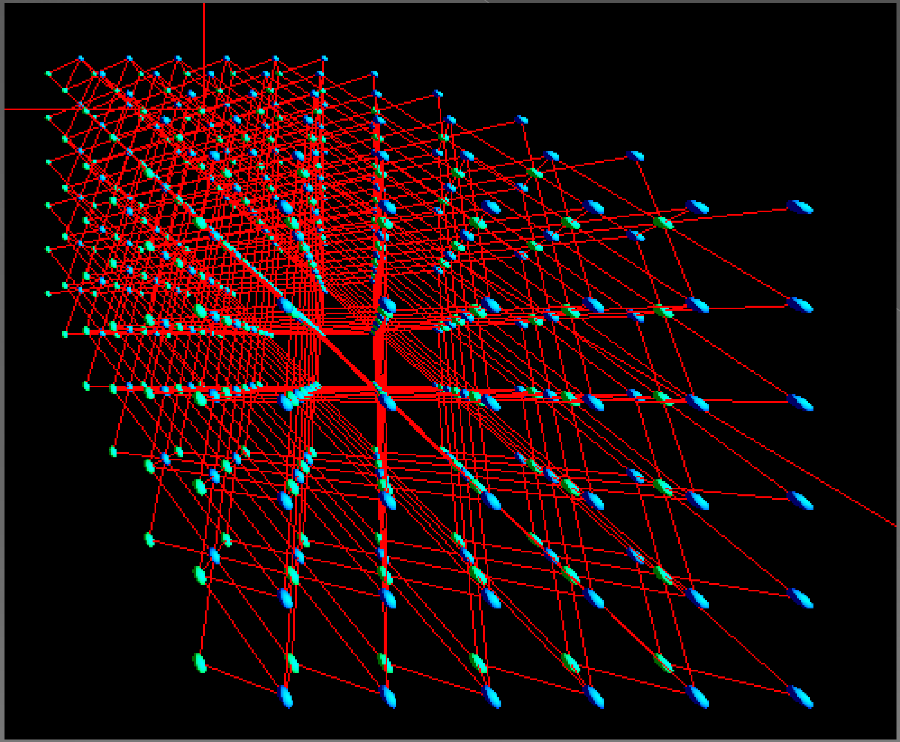
- Color Guide: Type 1- Blue; Type 2- Green


Atoms as coarse spheres – each composed out of 5 slices and 5 stacks Smoothness = 25

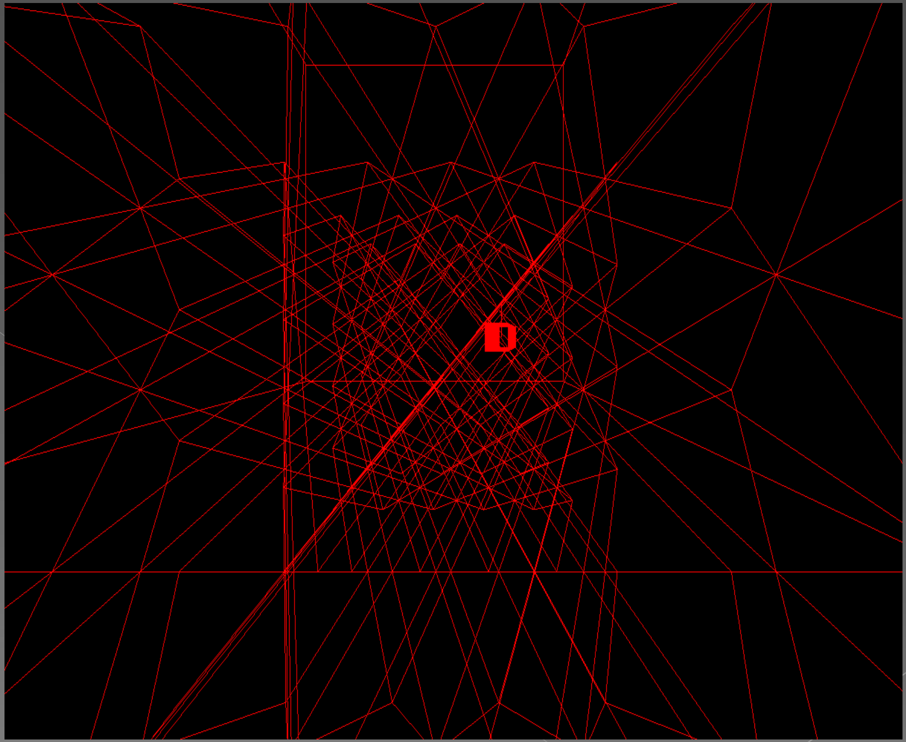

Atoms as smooth spheres – each composed out of 50 slices and 50 stacks Smoothness = 2500

Results: Bonds±Atoms BCC Lattice

- Color Guide: Type 1- Blue; Type 2- Green; Bonds-Red

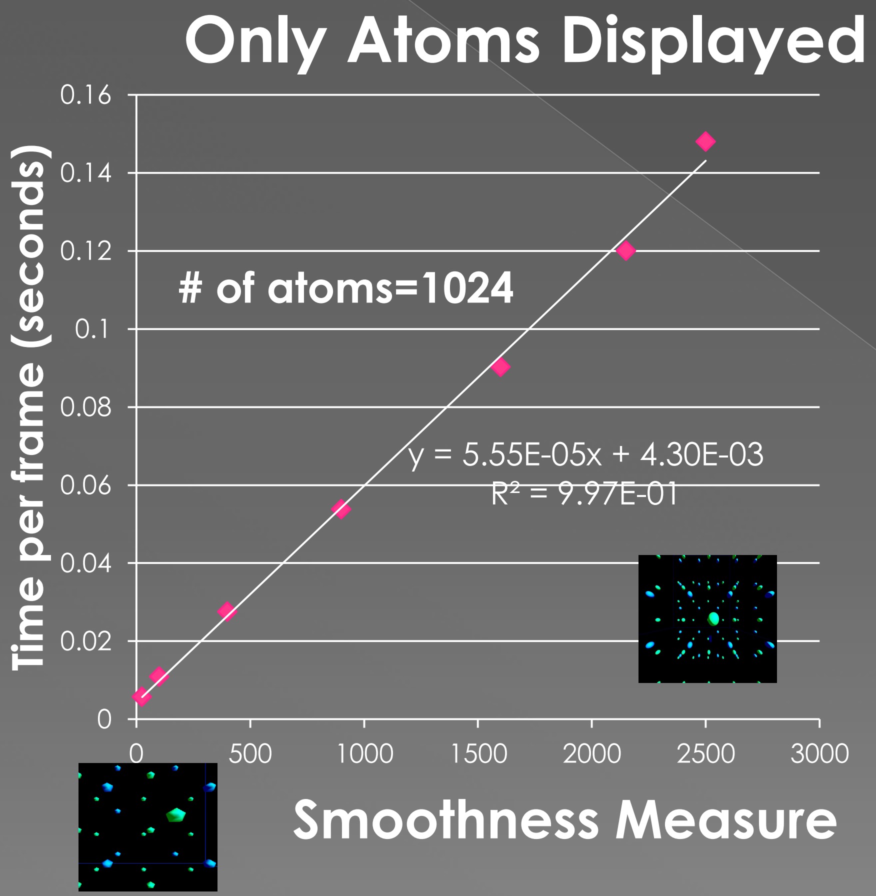


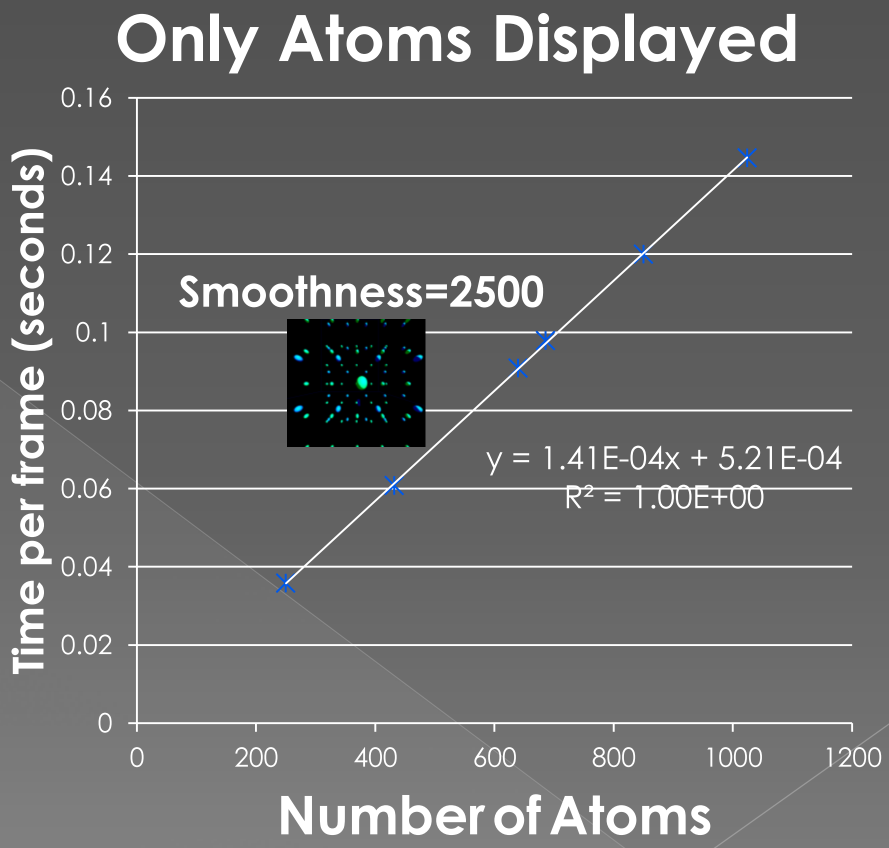
Each atom is bonded to 8 nearest neighbors



Bonds only configurations for execution timing study

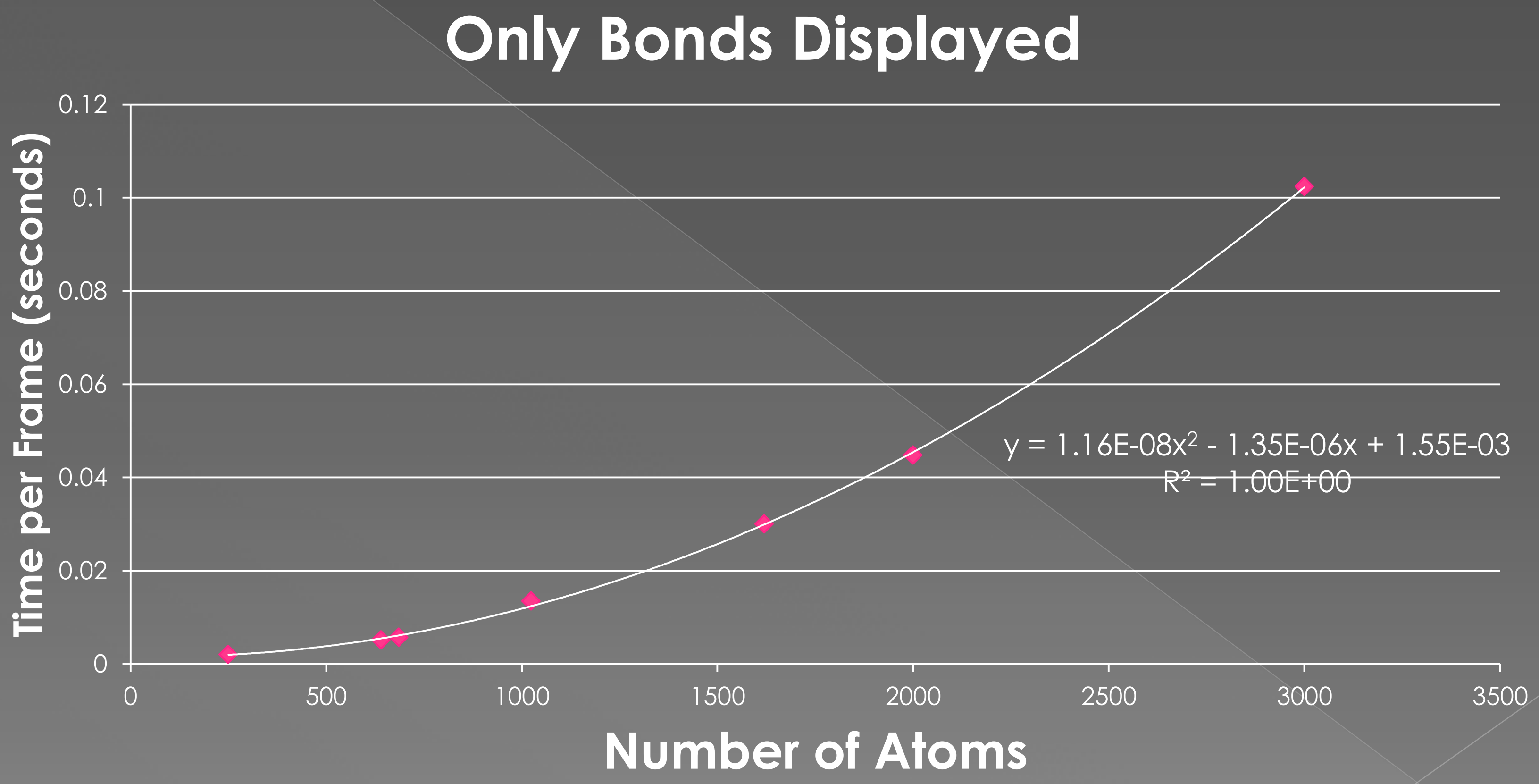
Results: Execution Timing





Execution Time □ Total number of OpenGL primitives

Results: Execution Timing



Execution Time □ (Number of atoms)^2 : Double loop over atoms to determine bonded neighbors

Conclusion

The current version of the project:

- CPU time required for display of atoms alone scales linearly with the total number of atoms / OpenGL primitives
- CPU time required for display of bonds alone is quadratic in the total number of atoms due to double loop over atoms to determine bonded neighbors
- From curve fits to timing graphs: Upper bound to number of atoms with interactive frame rate (10 per second): 706 (2913) for the display of atoms (bonds) alone.

Future Work

- Increase display rate of atoms by having variable smoothness for sphere display depending on perspective angular width of atom [A. Sharma *et. al*. (2003)]
- Bond display made to scale linearly with the number of atoms by using linked and neighbor lists
- Robust display of bond using cylinders

Acknowlegements

This work was funded by Louisiana Board of Regents: LASIGMA Award Nos.

EPS-1003897, NSF (2010-15)-RII-SUBR, HRD-1002541

Thanks for your attention

???QUESTIONS???