Visualizing Molecular Dynamics

Two Components

- **MD Simulation Algorithm:**

  Simulate a real material system at molecular or atomic level

  - ➢ How to perform atomistic simulation
  - ➢ What are the output data

    Atomic coordinates, other properties such as temperature, energy, stress distribution

- **MD Visualization Algorithm:**

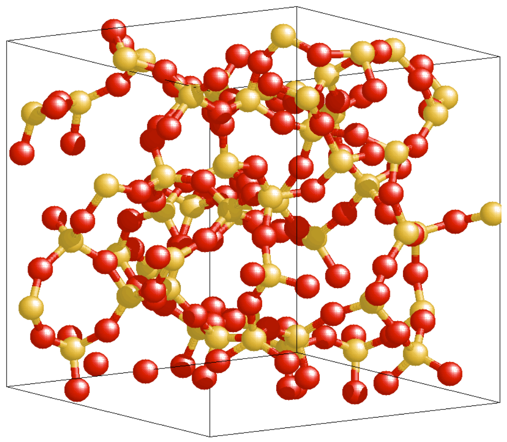
  Visualize a real material system at molecular or atomic level

  - ➢ How to visualize the simulated data
  - ➢ Particle sampling and rendering technique

MD Simulation Algorithm

N-Body Problem

- MD simulation:
  - solving a problem of *N* interacting atoms



GaAs System

- Calculate interatomic interactions (energy and forces) for a given configuration

- Optimize the configuration until the total energy of the system becomes minimum

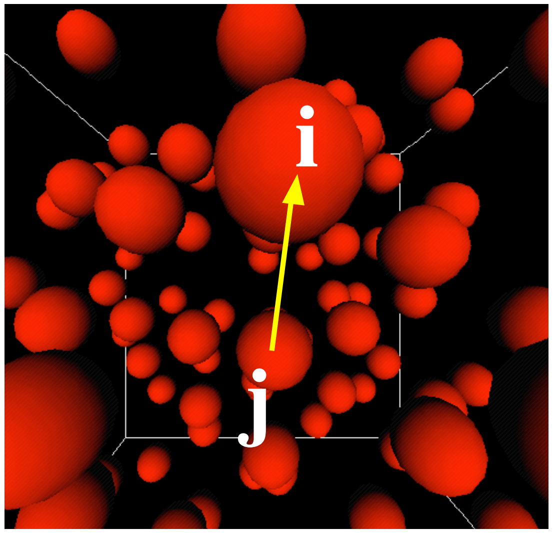
Interatomic Interaction

- For a pair of atoms *i* and *j*, the Lennard-Jones (LJ) potential energy is

$$u(r_{ij}) = 4\varepsilon\left[\left(\frac{\sigma}{r_{ij}}\right)^{12} - \left(\frac{\sigma}{r_{ij}}\right)^{6}\right]$$

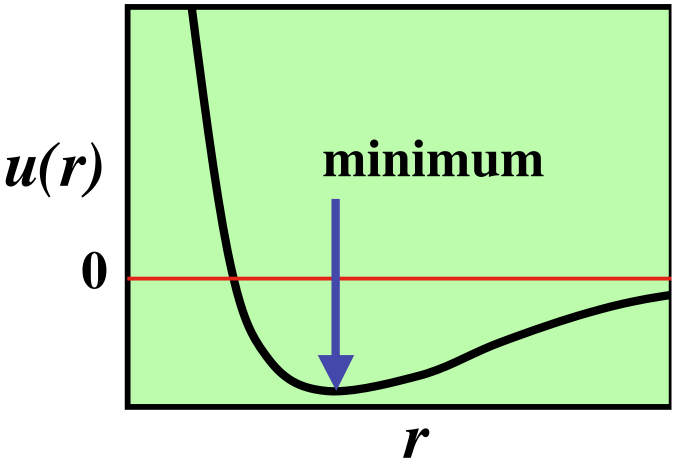
Truncated potential at cutoff $\boldsymbol{r_c}$

$u(r_{ij}) = 0$ for $r_{ij} \geq r_c$



- Force that atom *j* exerts on atom *i* is

$$f_{ij} = -\nabla u = \left(\frac{48\varepsilon}{\sigma^2}\right)\left[\left(\frac{\sigma}{r_{ij}}\right)^{14} - \frac{1}{2}\left(\frac{\sigma}{r_{ij}}\right)^{8}\right] r_{ij}$$



Equations of Motion

- Newton's second law of motion gives

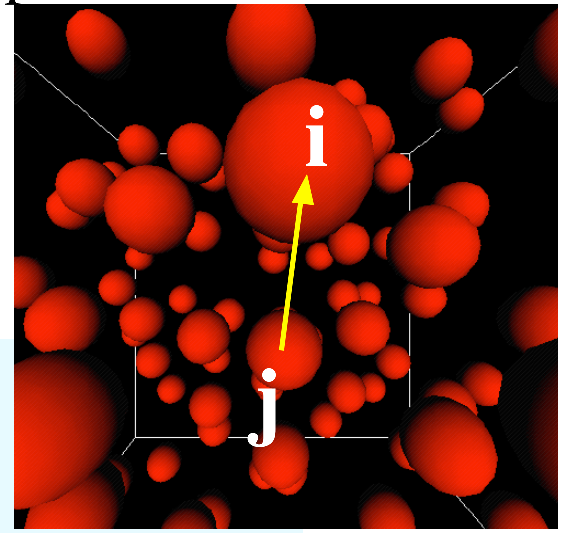
$$ma_i = F_i = \sum_{\substack{j=1 \\ (j \neq i)}}^{N} f_{ij}$$

Mass x Acceleration = Force

$$m\frac{\partial^2 r_i}{\partial t^2} = \frac{48\varepsilon}{\sigma^2}\sum_{\substack{j=1 \\ j \neq i}}\left[\left(\frac{\sigma}{r_{ij}}\right)^{14} - \frac{1}{2}\left(\frac{\sigma}{r_{ij}}\right)^{8}\right]\vec{r}_{ij}$$

$O(N^2)$ Complexity

- Computation involves $O(N^2)$ pairs or operations



```
for $i \leftarrow 1$ to $N$
  $F_i \leftarrow 0$
  for $j \leftarrow 1$ to $N$
    if $(i \neq j)$ $F_i \leftarrow F_i + f_{ij}$
```

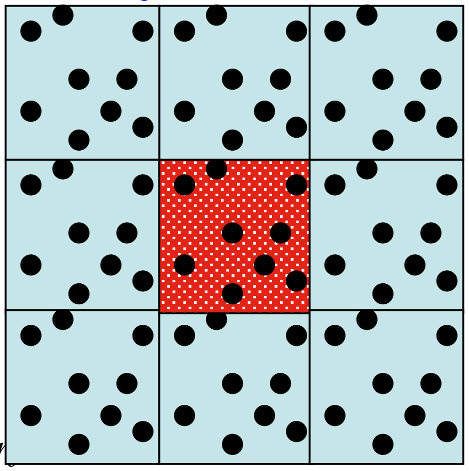
Periodic Boundary Condition

- A real material system is infinite
  - Typical atomic size is a few Angstroms (~10^{-8} cm)
  - 1 cm^3 sample ~ 1 / (10^{-8} x 10^{-8} x 10^{-8}) = 10^{24} atoms

- Simulation region is of finite size (few hundred Angstroms)

- Introduction of periodic boundaries
  - Infinite space filling array of identical copies of the simulation region

    Wraparound effect: Particle leaving the simulation region through a particular boundary reenters the region through the opposite side

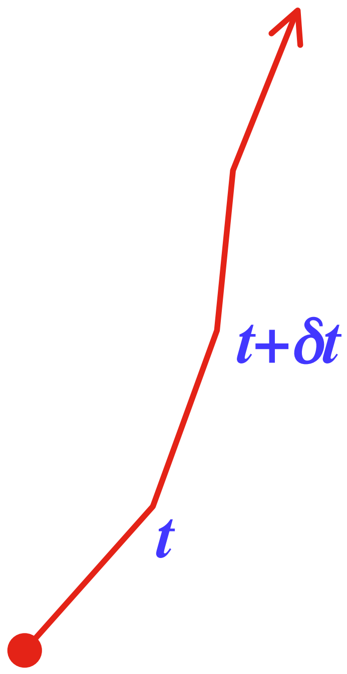


Periodic boundary conditions in 2D case

Integration

- Velocity Verlet algorithm
  - Numerical technique for integrating the equation of motion
  - Size of the time step used for the numerical integration is ***δt***

Position and velocity of ***i***th atom at (***t+δt***) are



$$r_i(t+\delta t) = r_i(t) + \delta t v_i(t) + \frac{1}{2}\delta t^2 a_i(t)$$

$$v_i(t+\delta t) = v_i(t) + \frac{1}{2}\delta t[a_i(t) + a_i(t+\delta t)]$$

Program Organization

```
int main(int argc, char **argv) {
   InitParams();      /* Initialize MD parameters */
   InitConf();        /* Initial atomic configuration */
   ComputeAccel();    /* Compute initial accelerations */
   moreCycles = 1;             /* activates MD step */
   while (moreCycles) {        /* MD loop continues */
     SingleStep();             /* until stepLimit */
     fprintf();                /* print output */
     if (stepCount >= StepLimit) moreCycles = 0;
   }
   return 0;
}
```

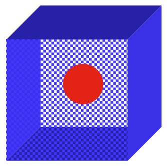
Input Control Parameters

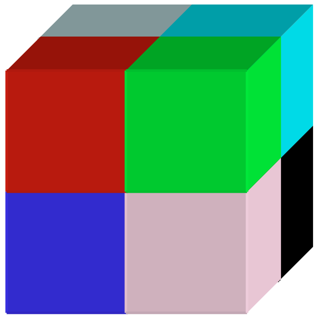
- `InitParam()` reads all the data needed to specify the simulation

Unit cell dimensions:	`InitUcell[3]`
Density:	`Density`
Temperature	`InitTemp`
Time step	`DeltaT`
Number of MD steps	`StepLimit`

- Input file ‘md.in’ contains

```
2 2 2
0.5
10.0
0.005
100
```


cubic unit cell (one atom)


2 x 2 x 2 cubic lattice (8 atoms)

Output: Atomic Configuration

```
Step: 1
8                                    no. of atoms
0.000000 4.000000                    simulation box
0.000000 4.000000
0.000000 4.000000
0.971958 1.000586 0.994289           atomic x, y, z-coordinates
3.015601 1.001716 1.022176
0.996467 2.990148 0.975188
2.996066 2.999772 1.028197
1.005322 0.978026 2.985069
3.018493 0.991320 2.983990
1.010259 3.016432 3.018755
2.985834 3.021999 2.992337
```

Bigger Picture of MD

- A wide variety of materials properties
  - ➢ Structure, transport, fracture, melting, chemical reactions
  - ➢ Physics, biology, chemistry

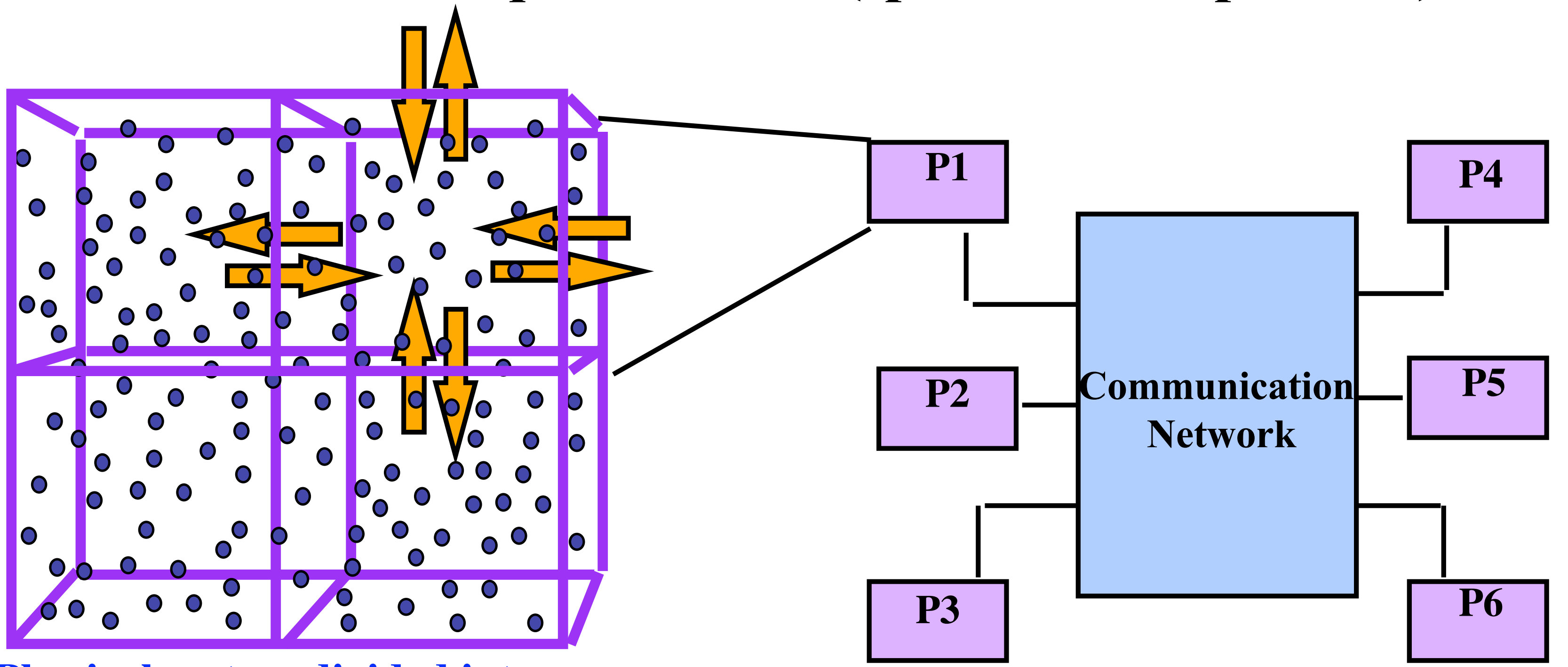
- Under a variety of conditions
  - ➢ Temperature, pressure, electric and magnetic fields

- Large-scale simulations
  - ➢ Billion-atom simulations

- Massively parallel machines
  - ➢ Thousands of processors

Implementation on Parallel Machines

Divide-and-Conquer Scheme (spatial decomposition)



Physical system divided into domains of equal volumes

Each domain assigned to individual processors, P1, P2, ...

Interprocessor communication using MPI

Structured 6-step message passing

MD Visualization Algorithm

Particle Rendering

- Display individual data that define particles (atoms) by spheres -- so called ball representation

- Static visualization and animation

Basic Steps

- Read atomic data from the input file or during the MD run
- Specify color and lighting environment
- Set viewpoint and perspective projection
- Draw a solid sphere at each atom position
- Draw a box to define the simulation region
- Enable depth test for hidden-surface removal
- Use single or double buffered window mode

Data Structures

- **Number of atoms**

```
Int natoms;
```

- **Atom data type.** `crd[0|1|2]` **is the x|y|z coordinate of atoms**

```
Typedef struct {
   float crd[3]}
} AtomType;
```

- **Array of atoms (dynamically allocated)**

```
AtomType *atoms;
```

- **Range of x|y|z coordinate of atoms (system's size)**

```
float min_ext[3], max_ext[3];
```

readConf();

The atomic configurations are input from file ‘md.conf’

```
fscanf(fp,"%d", &natoms);

atoms = (AtomType *) malloc(sizeof(AtomType)*natoms);

for (l=0; l<3; l++)
    fscanf(fp,"%f%f",&min_ext[l],&max_ext[l]);

for (j=0; j<natoms; j++)
    fscanf(fp,"%f %f %f", &(atoms[j].crd[0]),
    &(atoms[j].crd[1]), &(atoms[j].crd[2]));
```

Lighting Environment

```
GLfloat light_diffuse[] = {1.0, 1.0, 1.0, 1.0};
GLfloat light_position1[] = {0.5, 0.5, 1.0, 0.0};

GLfloat mat_specular = {1.0, 1.0, 1.0, 1.0};
GLfloat shininess[] = {100.0};

glLightfv(GL_LIGHT0,GL_DIFFUSE,light_diffuse);
glLightfv(GL_LIGHT0,GL_POSITION,light_position1);

glEnable(GL_LIGHTING);
glEnable(GL_LIGHT0);
```

Viewpoint Selection

```
gluLookAt(eye[0], eye[1], eye[2], center[0],
center[1], center[2],  up[0], up[1],up[2]);
```

- **position:**

  ```
  eye[0] = center[0]
  eye[1] = center[1]
  eye[2] = center[2] + body diagonal length
  ```

  `(center[0],center[1],center[2])` is the center of the simulation region or box

- **Direction:** Towards the center of the simulation region

- **Up vector:** Default setting (0.0,1.0,0.0)

Perspective Projection

```
gluPerspective(fovy,aspect,near_clip,far_
   clip);
```

```
fovy = 1.0 atan[180a/((d-0.5c)M_PI)]
aspect = w/h
near_clip = (d - 0.5c)
far_clip = 2(d + 0.5c)
```

Where**, a, b** and **c** are the three sides of the simulation box, and **d** is the body diagonal length

Drawing Atoms

Display list `atomsid` to draw a sphere at each atom position

```
glNewList(atomsid, GL_COMPILE);
   for (i=0; i < natoms; i++) {
   glPushMatrix();
      glTranslatef(atoms[i].crd[0],atoms[i].crd[1],
                   atoms[i].crd[2]);
      glColor3f(r,g,b);
      glutSolidSphere(radius,20,16);
   glPopMatrix();
   }
glEndList();
```

Mark Simulation Region

```
void display()
{
    glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);

    glCallList(atomsid);

    glTranslatef(eye[0],eye[1],0.0);
    glLineWidth(4.0);
    glColor3f(1.0,1.0,1.0);
    glutWireCube(2.0*center[2]);

    glFlush();
}
```

Register Callback Functions

```
int main(int argc, char **argv)
{
    readConf();
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE|GLUT_RGBA|GLUT_DEPTH);
    glutInitWindowSize(winx, winy);
    glutCreateWindow("Visualizing Molecular Dynamics");
    initView(min_ext, max_ext);
    glutDisplayFunc(display);
    glutReshapeFunc(reshape);
    makeAtoms(atom_radius);
    glutMainLoop();
    return 0;
}
```

Header: atom_vis.h

```
#define Ratom 1.0
#define Gatom 0.0
#define Batom 0.0

typedef struct {
     float crd[3];
} AtomType;

float atom_radius = 0.2;
int winx=640, winy=640;
float min_ext[3], max_ext[3];
int natoms;
AtomType *atoms;
float eye[3];
float center[3];
float up[3];
```

MD Animation

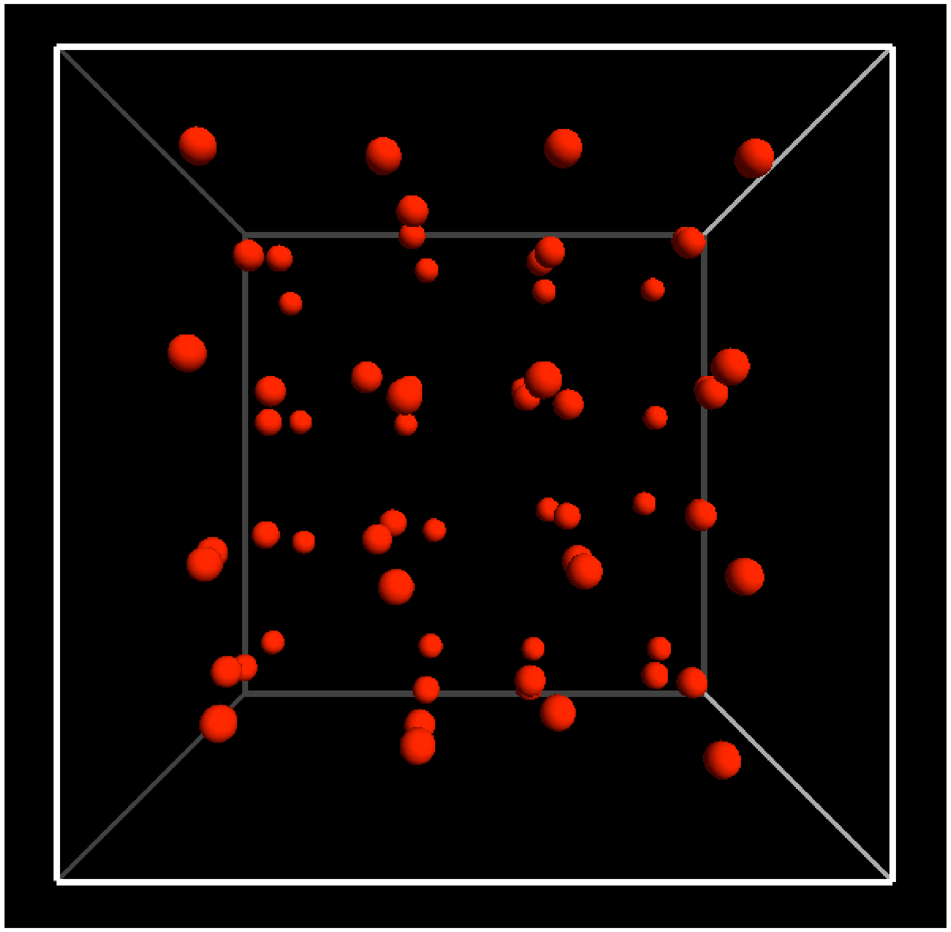
- Combine `md.c` and `atom_vis.c`
- Merge header files `md.h` and `atom_vis.h`
- Check consistency in data structures

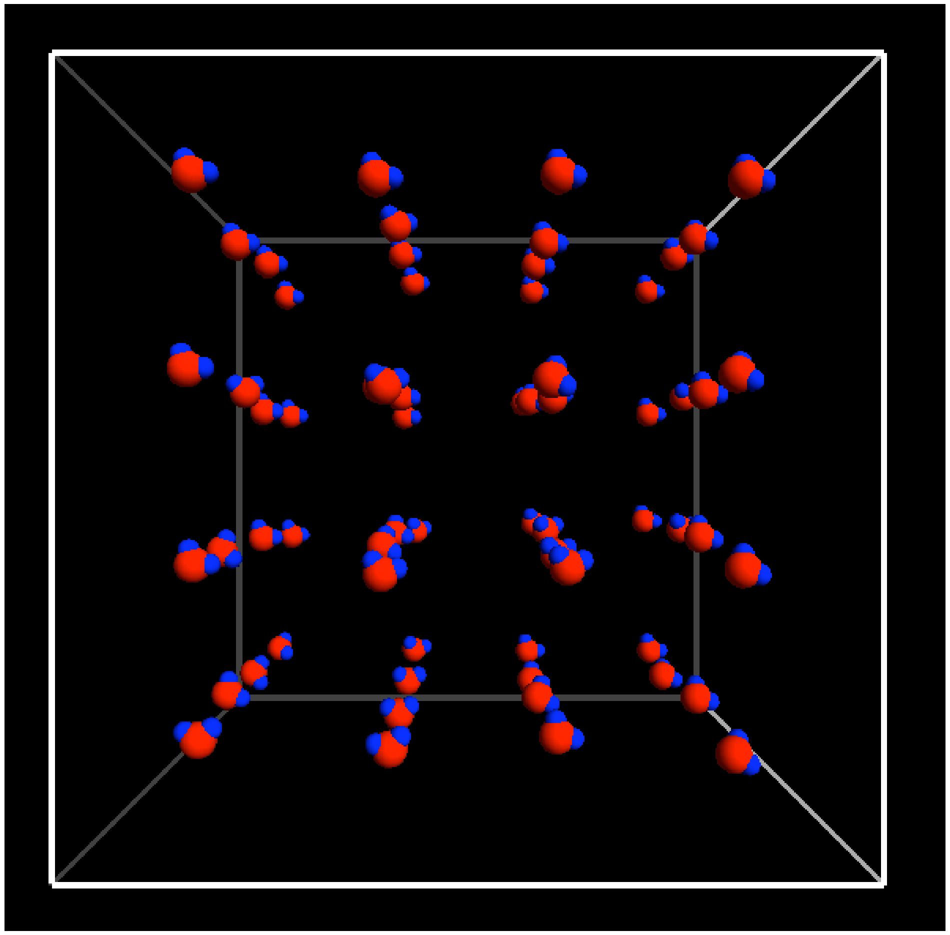
  AtomType *atoms; — `atom_vis.c`

  double r[NMAX][3]; — `md.c`
- Define a function to do animation and pass it

  glutIdleFunc()

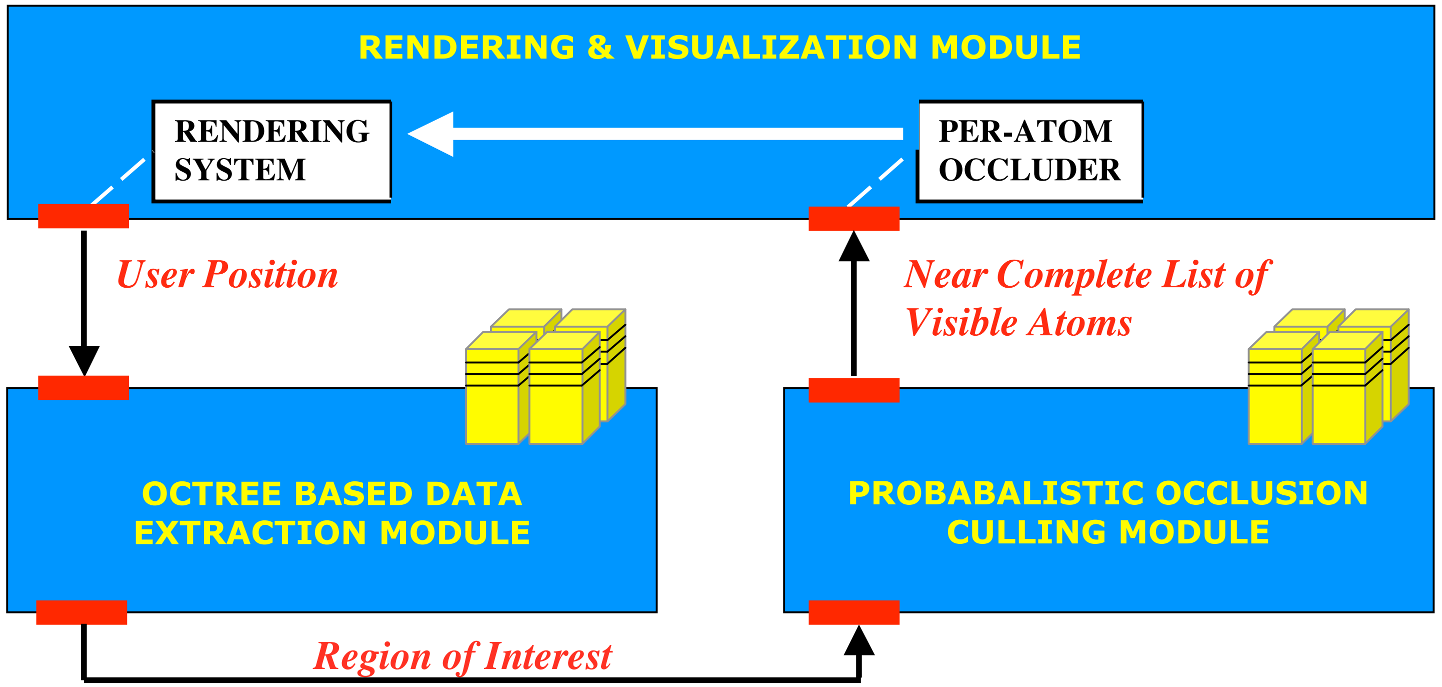
A Typical MD Snapshot


Argon


H_2O

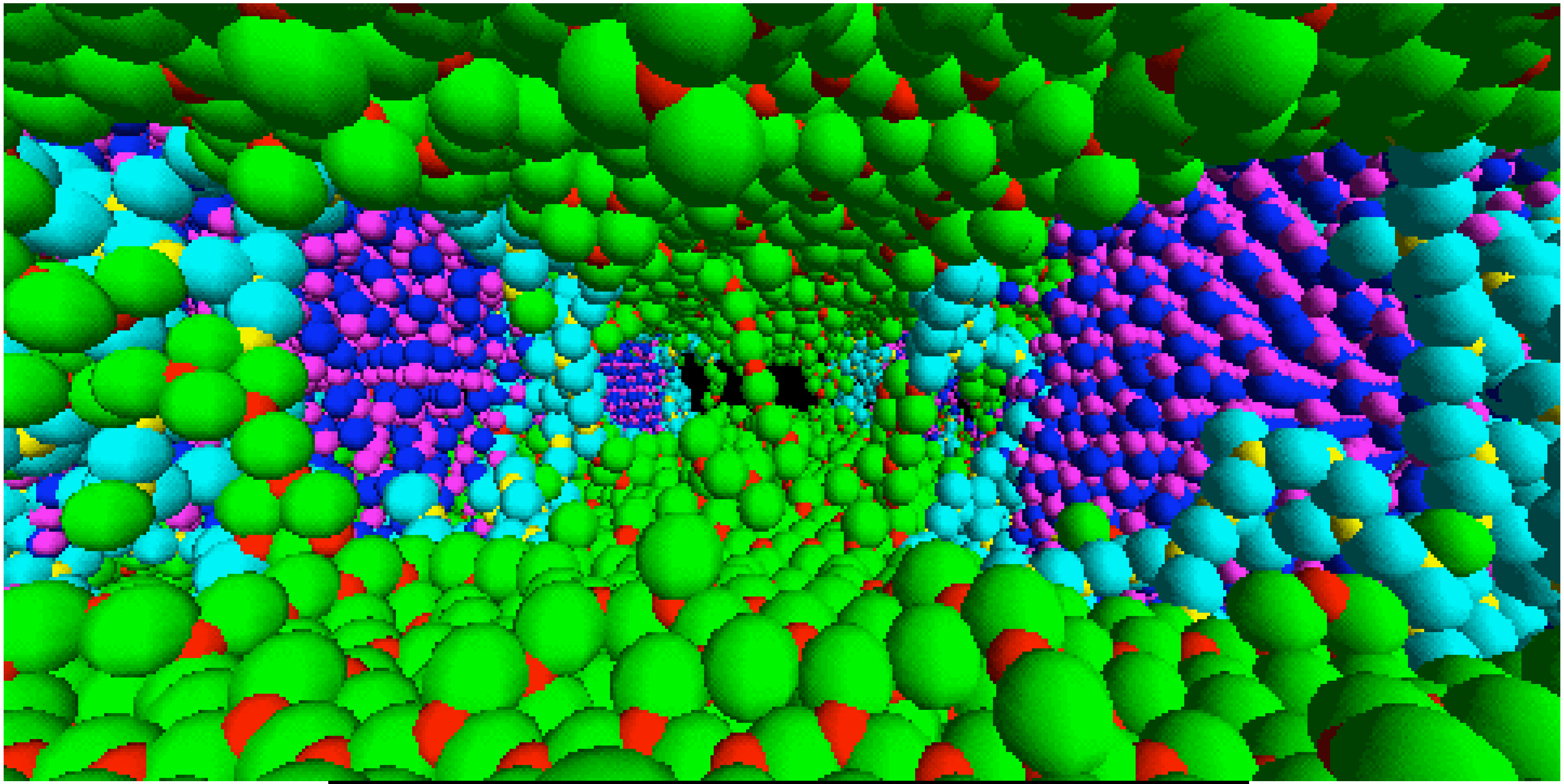
Processing of Large MD Datasets

- Enormous challenge to achieve an interactive frame rate
- Parallel processing of data in a PC cluster before rendering on graphics server


RENDERING & VISUALIZATION MODULE
RENDERING SYSTEM
PER-ATOM OCCLUDER
User Position
Near Complete List of Visible Atoms
OCTREE BASED DATA EXTRACTION MODULE
PROBABALISTIC OCCLUSION CULLING MODULE
Region of Interest

Sharma et al., IEEE Virtual Reality 2002

A Billion-Atom Visualization



A billion-atom MD simulation of fracture in SiN-matrix SiC-fiber composite

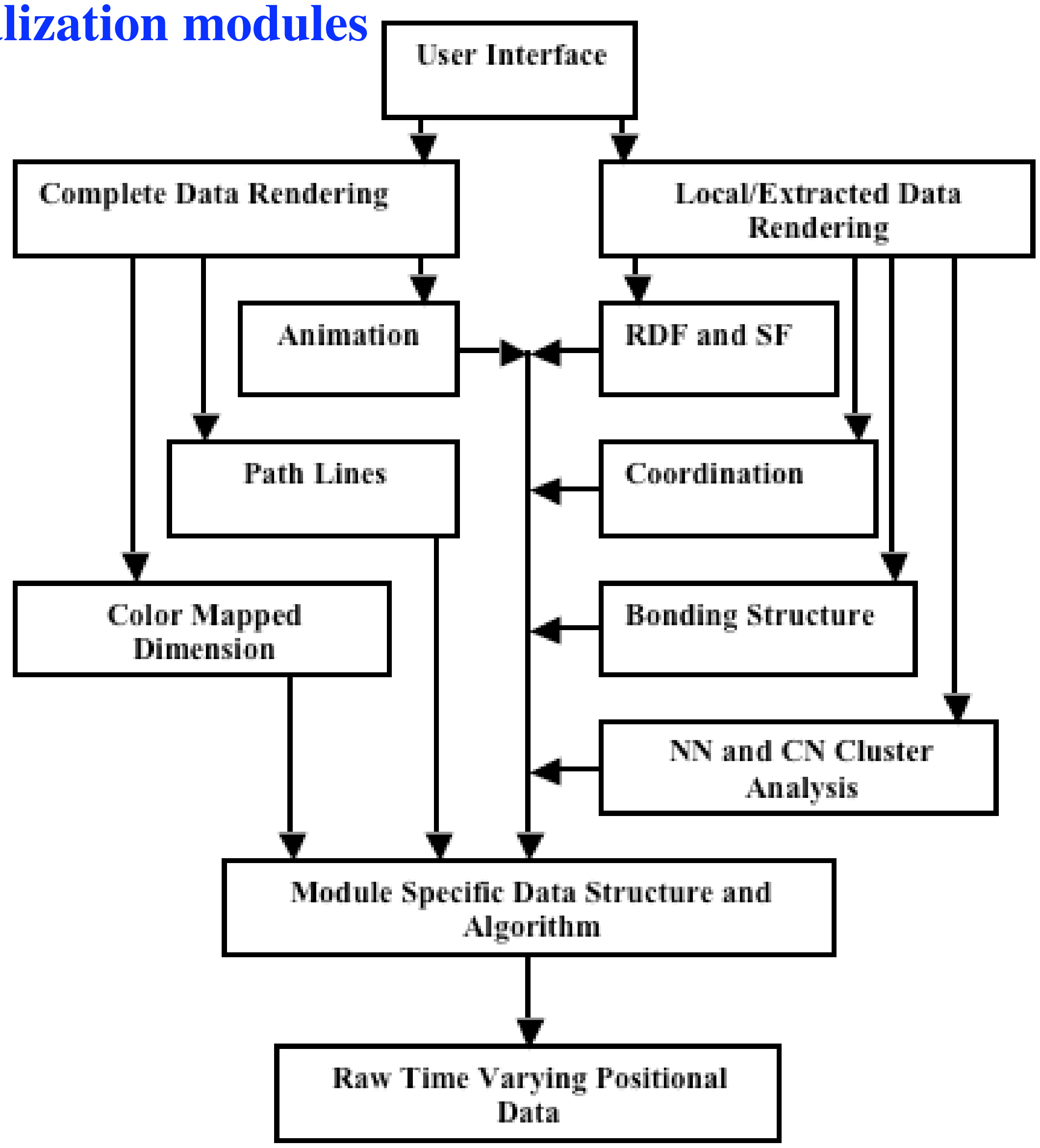
Nakano et al., 2000

Space-Time Multiresolution Atomistic Visualization

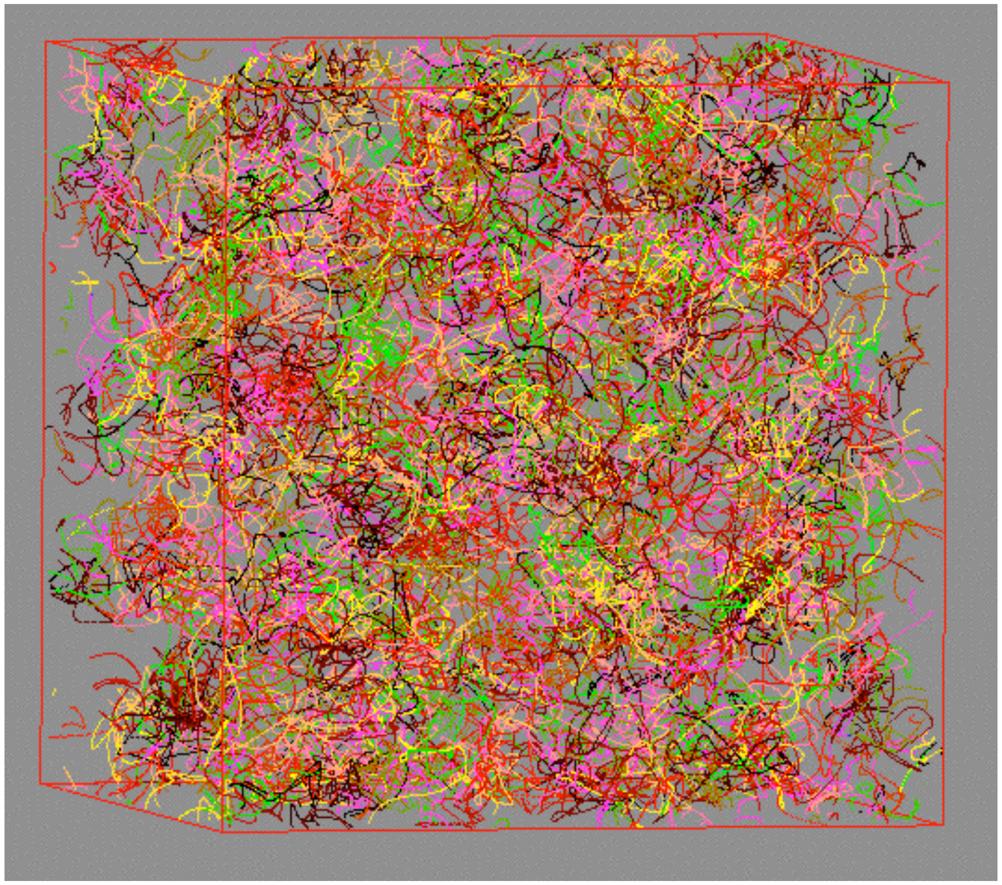
- Simulation data
  - Discrete sets of degrees of freedom

    atomic positions in 3D space
  - Time-varying
  - Correlated data
- Approach
  - Spatial proximity
  - Temporal proximity
  - Spatio-temporal analysis
- Model
  - Complete data rendering
  - Local/extracted data rendering

Visualization modules


User Interface
Complete Data Rendering
Local/Extracted Data Rendering
Animation
RDF and SF
Path Lines
Coordination
Color Mapped Dimension
Bonding Structure
NN and CN Cluster Analysis
Module Specific Data Structure and Algorithm
Raw Time Varying Positional Data

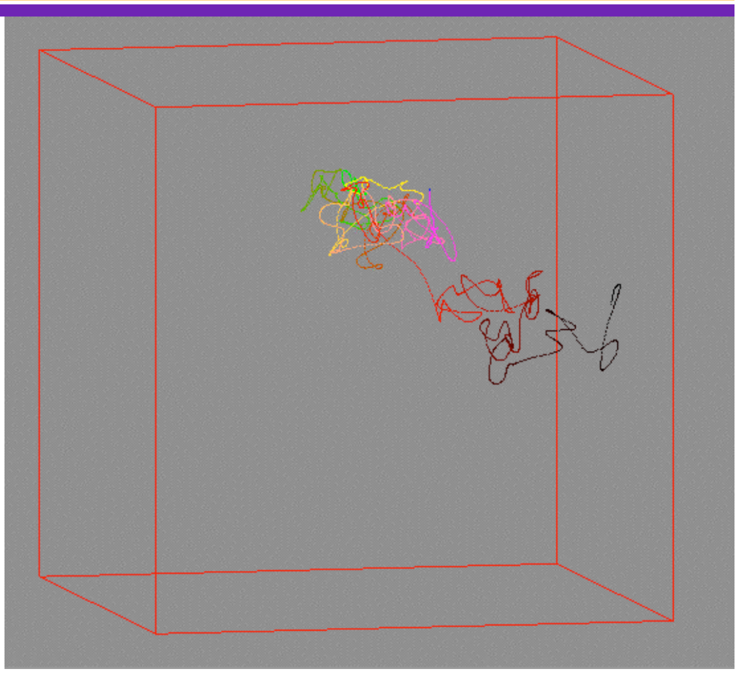
Pathlines



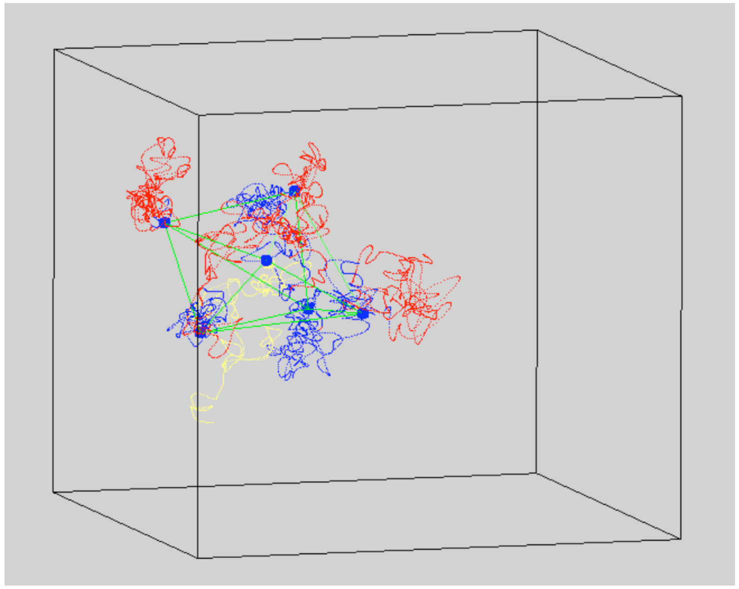
Pathlines of 64 atoms in liquid MgO

Color: time elapsed or distance from some reference point

Single atom

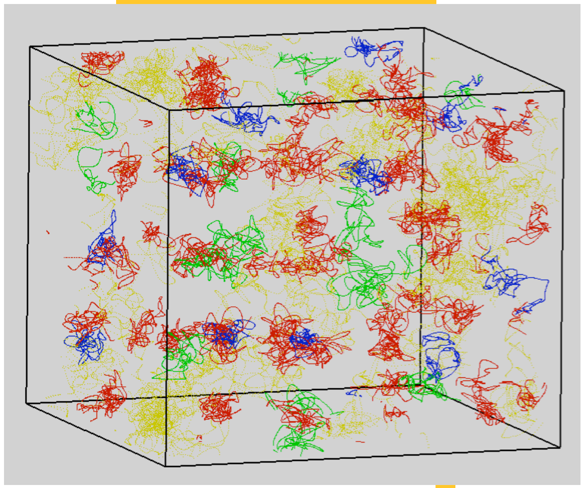


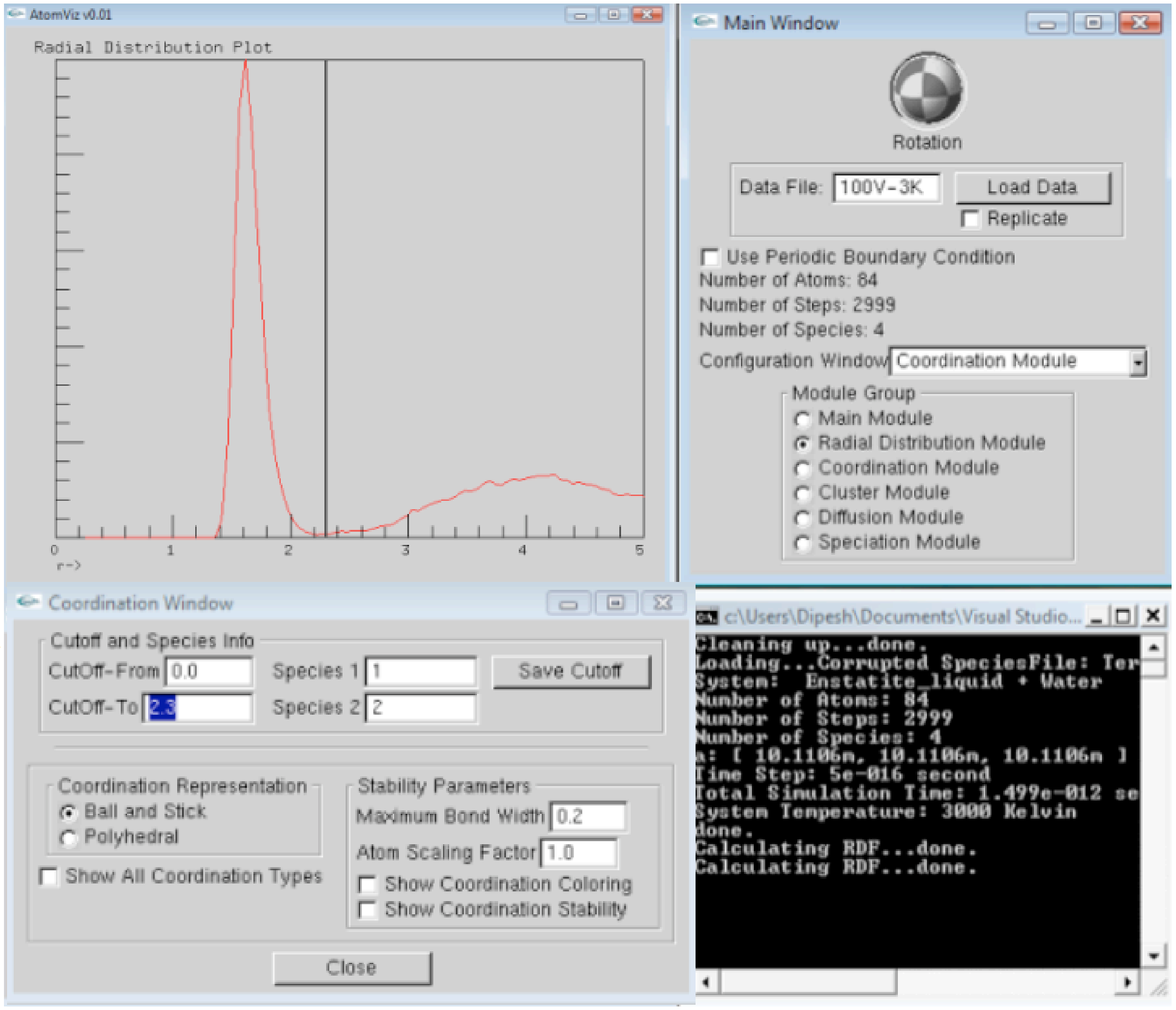
SiO_6 unit



Radial Distribution Function

$D = \{P(t)\}$





Data File: 100V-3K Load Data
☐ Replicate
☐ Use Periodic Boundary Condition
Number of Atoms: 84
Number of Steps: 2999
Number of Species: 4
Configuration Window: Coordination Module

Module Group
- Main Module
- [✓ Radial Distribution Module]
- Coordination Module
- Cluster Module
- Diffusion Module
- Speciation Module

Cutoff and Species Info
CutOff-From 0.0 Species 1 1 Save Cutoff
CutOff-To 2.3 Species 2 2

Coordination Representation
- [✓ Ball and Stick]
- Polyhedral

☐ Show All Coordination Types

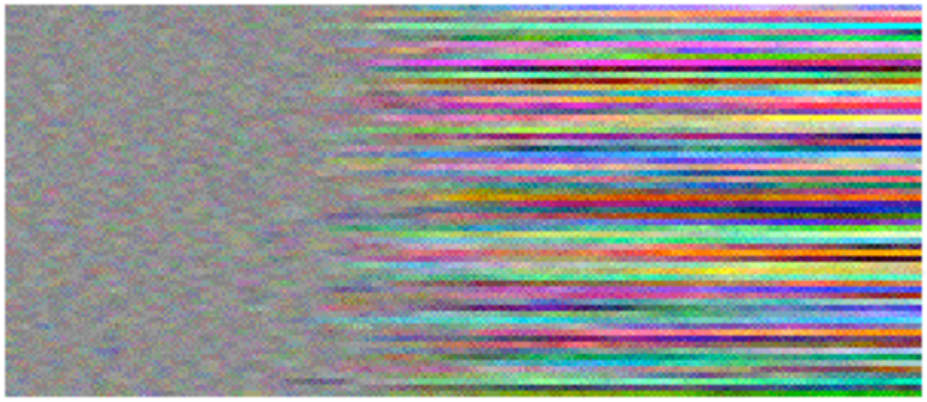
Stability Parameters
Maximum Bond Width 0.2
Atom Scaling Factor 1.0
☐ Show Coordination Coloring
☐ Show Coordination Stability

c:\Users\Dipesh\Documents\Visual Studio...

```
Cleaning up...done.
Loading...Corrupted SpeciesFile: Ter
System:  Enstatite_liquid + Water
Number of Atoms: 84
Number of Steps: 2999
Number of Species: 4
a: [ 10.1106m, 10.1106m, 10.1106m ]
Time Step: 5e-016 second
Total Simulation Time: 1.499e-012 se
System Temperature: 3000 Kelvin
done.
Calculating RDF...done.
Calculating RDF...done.
```

Pick up a cutoff window interactively

Color-Mapped Dimension



Atomic displacement relative to perfect crystal over 3000 time steps

Pixel color at screen position S_{ij}:

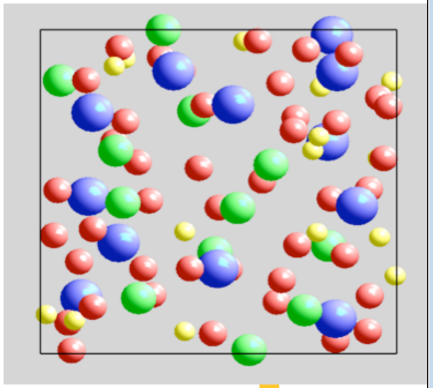
$$C_{ij} = \left[R_{ij}, G_{ij}, B_{ij}\right]$$

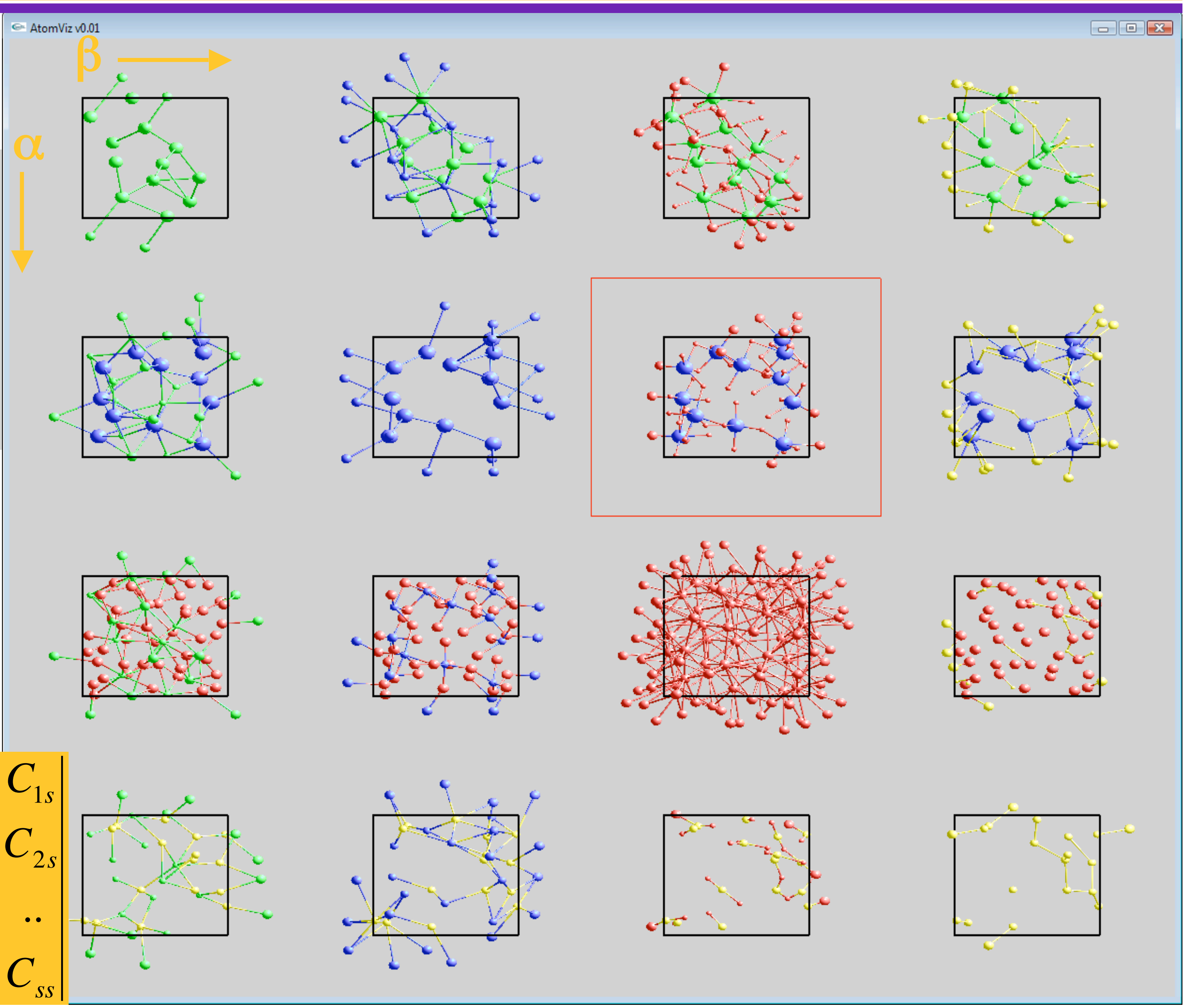
$$R_{ij} = \left(0.5 + f\left[x_{ij} - x_i^0\right]\right) \quad G_{ij} = \left(0.5 + f\left[x_{ij} - x_i^0\right]\right) \quad B_{ij} = \left(0.5 + f\left[x_{ij} - x_i^0\right]\right)$$

Nearest Neighbor Pairs Matrix

$\alpha = 1 \quad 2 \quad 3 \quad 4$

Mg	Si	O	H





NN Clusters

Coordination Matrix

$$C = \begin{vmatrix} C_{11} & C_{12} & .. & C_{1s} \\ C_{21} & C_{22} & .. & C_{2s} \\ . & .. & .. & .. \\ C_{s1} & C_{s2} & .. & C_{ss} \end{vmatrix}$$

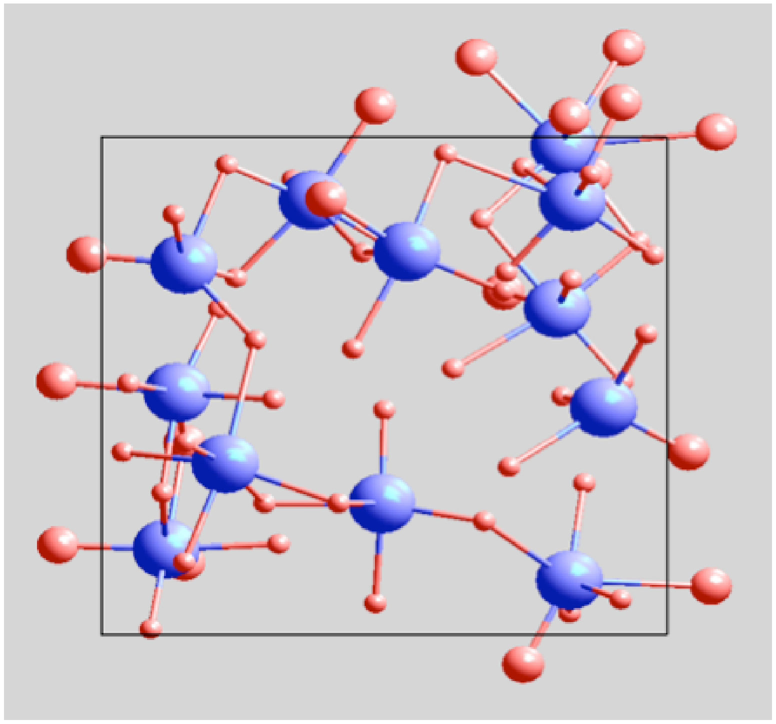
Coordination-Encoding

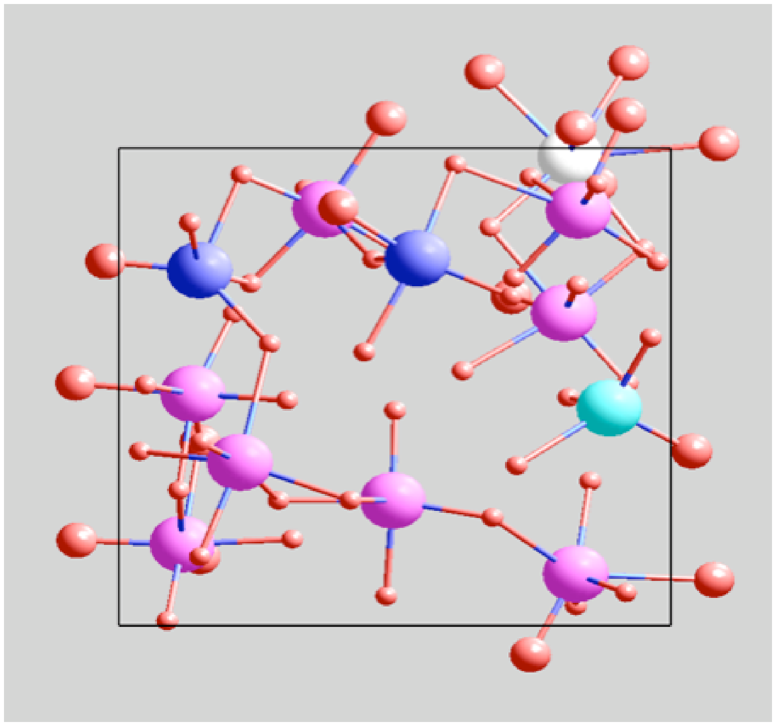
Partial coordination environment (per atom basis) is defined by

$$nn_i^{\alpha\beta}(t) = \left\{1 \le j \le n : d(i,j) \le r_{\min}^{\alpha\beta} \wedge type(j) = \beta\right\} \quad \text{for} \quad i = 1 \ldots.\ n^{\alpha}$$

Coordination number $C_i^{\alpha\beta}(t) = \left|nn_i^{\alpha\beta}(t)\right|$

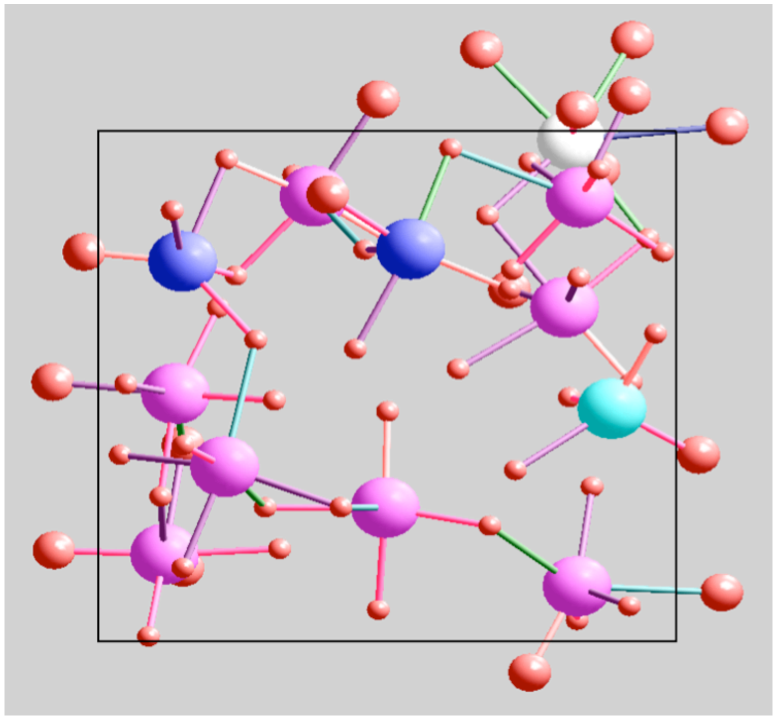
0	1	2	3	4	5	6	7





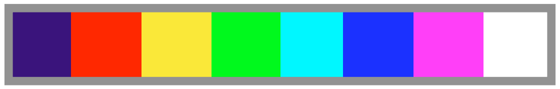
Four-, five-, six- and seven-fold coordination states

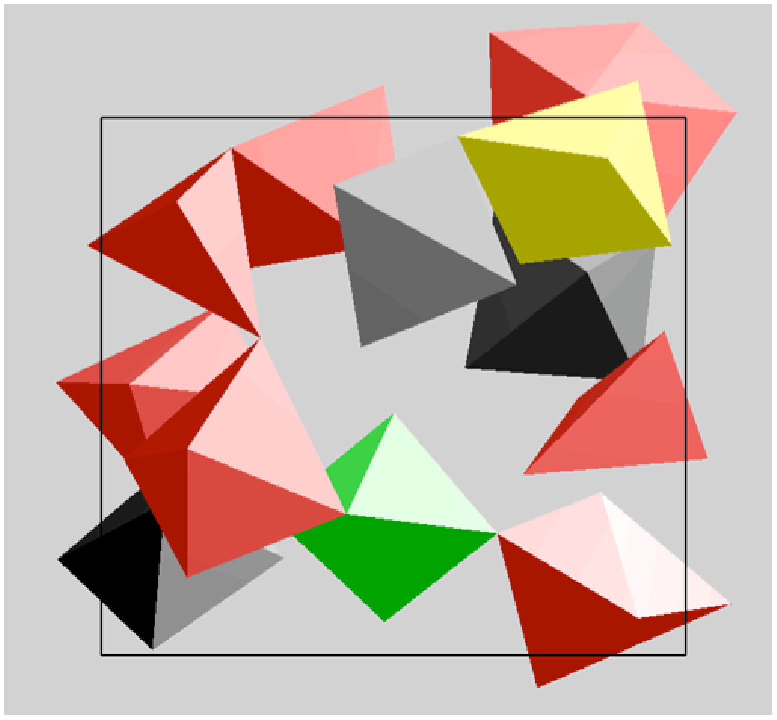
Coordination Distortion



Bond-length distribution:

$$\lambda_B = \frac{r - r_{MIN}}{r_{MAX} - r_{MIN}}$$





Polyhedral distortion:

Quadratic elongation

$$\lambda_P = \sum_{i=1}^{n} \left(l_i / l_0 \right)^2 \Big/ n$$

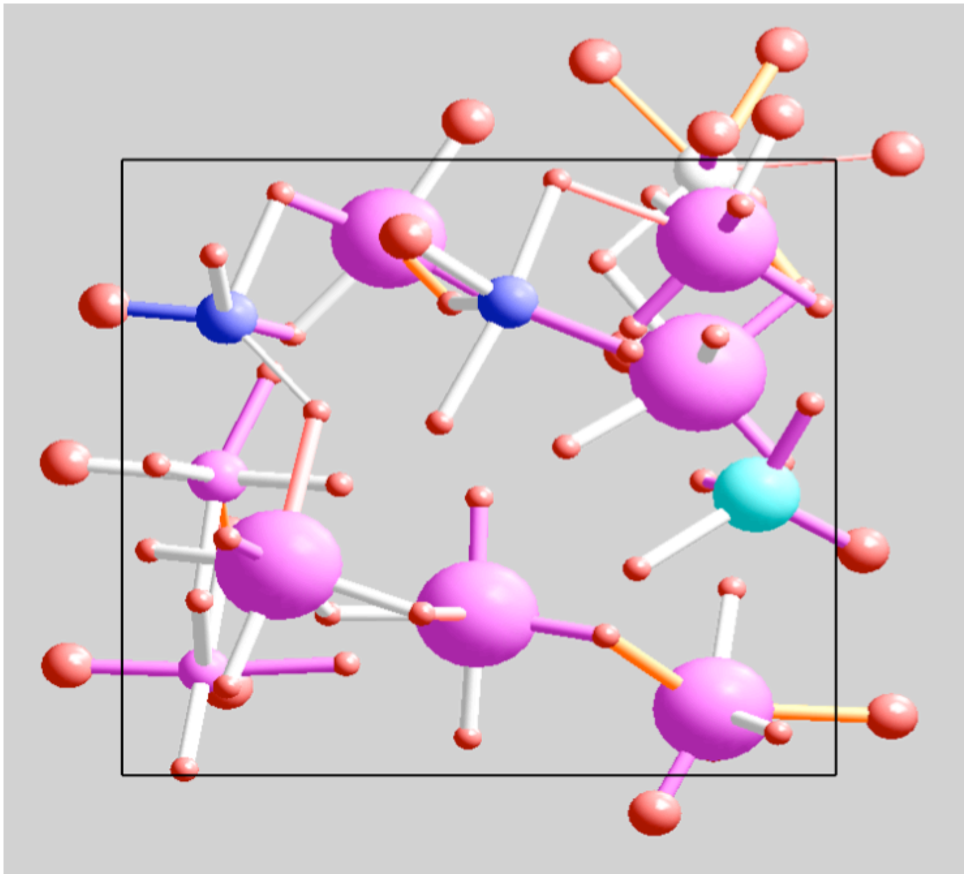
Angle-variance

$$\sigma = \sum_{i=1}^{n} \left(\theta_i - \theta_0 \right)^2 \Big/ (n-1)$$

Coordination Stability

$C_i^{\alpha\beta}(t),\ i = 1 \ldots n^{\alpha}$

4	5	6	7



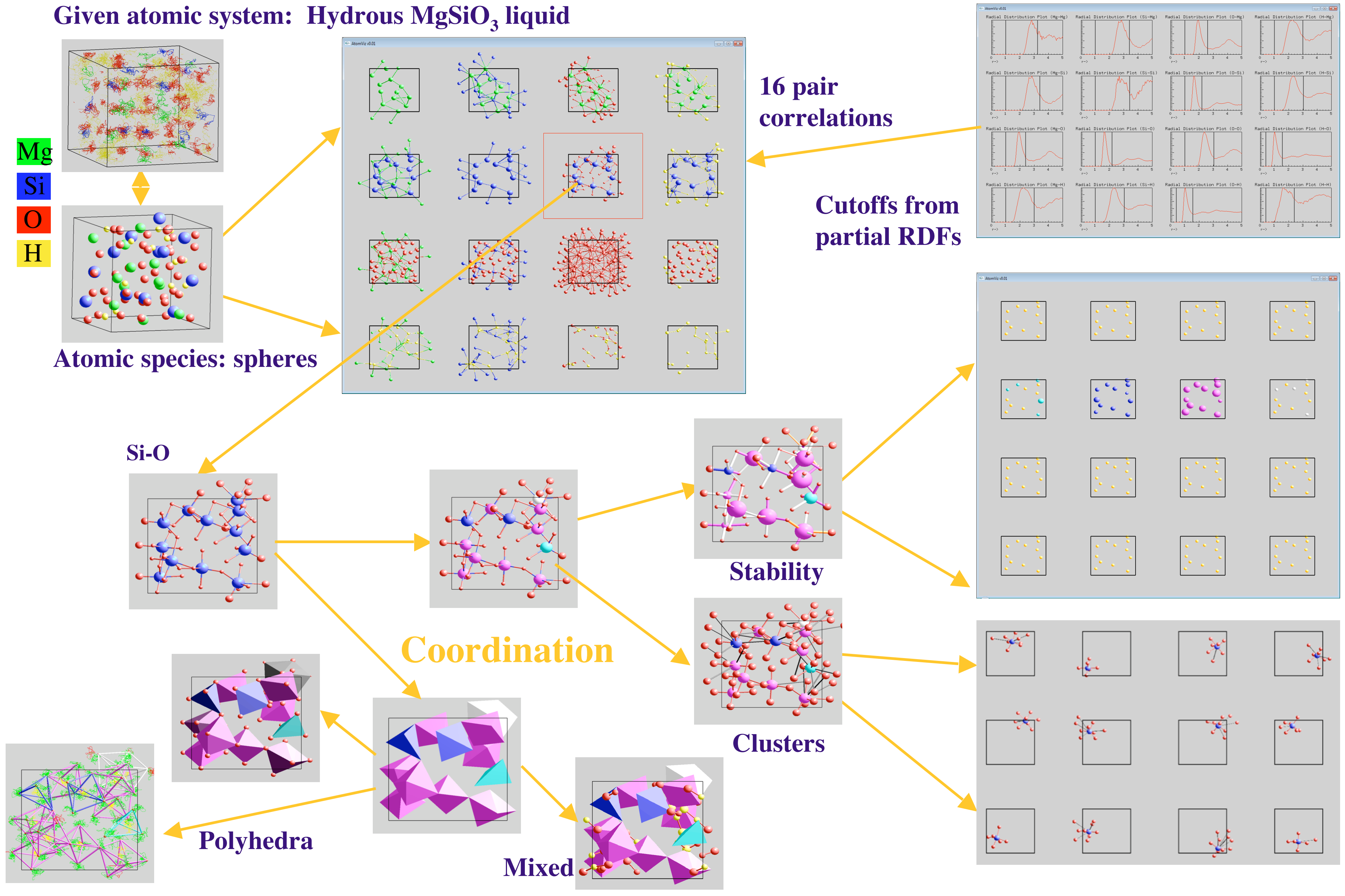
Center atoms: Size encodes the coordination stability
Lines: Thickness encodes the bond stability

The stability represents the fraction of the total simulation time over which a given coordination state or bond exists:

$$f'[C_i^{\alpha\beta}(t)],\ i = 1 \ldots n^{\alpha}$$

$$f_i(k),\ k \in NN_i^{\alpha\beta},\ i = 1 \ldots n^{\alpha}$$

Visualizing Coordination Environment


Given atomic system: Hydrous $MgSiO_3$ liquid
Mg
Si
O
H
Atomic species: spheres
AtomViz v0.01
16 pair correlations
Radial Distribution Plot (Mg-Mg)
Radial Distribution Plot (Si-Mg)
Radial Distribution Plot (O-Mg)
Radial Distribution Plot (H-Mg)
Radial Distribution Plot (Mg-Si)
Radial Distribution Plot (Si-Si)
Radial Distribution Plot (O-Si)
Radial Distribution Plot (H-Si)
Radial Distribution Plot (Mg-O)
Radial Distribution Plot (Si-O)
Radial Distribution Plot (O-O)
Radial Distribution Plot (H-O)
Radial Distribution Plot (Mg-H)
Radial Distribution Plot (Si-H)
Radial Distribution Plot (O-H)
Radial Distribution Plot (H-H)
Cutoffs from partial RDFs
Si-O
Coordination
Stability
Clusters
Polyhedra
Mixed

Atomic Movement

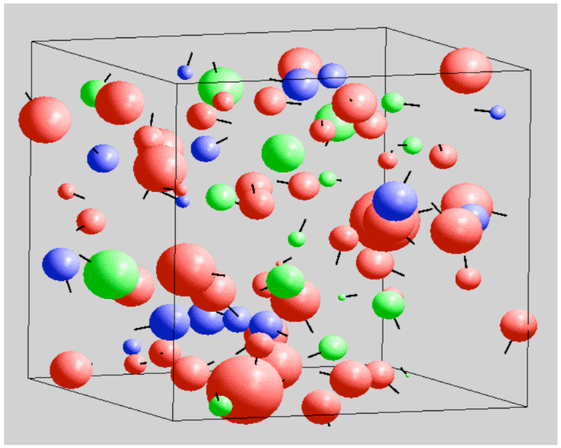
- A variety of displacement data: $\Delta r_i = r_i^t - r_i^0$
  - Reference configuration: Initial, Previous, Next or Mean configuration
  - Centroid spheres
  - Crystalline

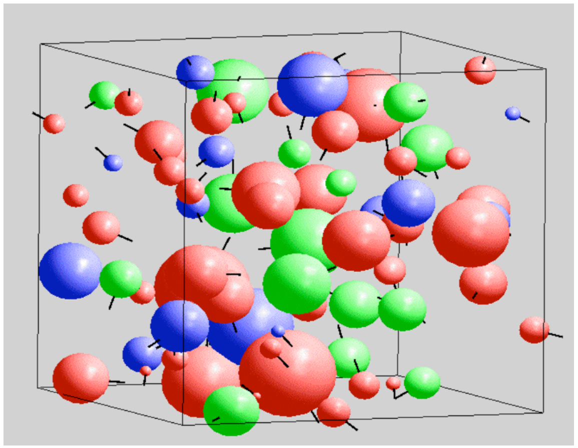
Sphere-and-line representation

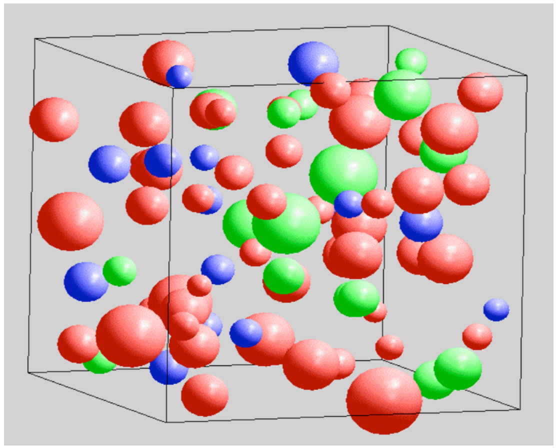
$$r_i^s = f\left|\Delta r_i\right| + a$$

$$line\left(\left|\Delta r_i\right| p_i, \left|\Delta r_i\right| p_i + l\frac{\Delta r_i}{\left|\Delta r_i\right|}\right)$$

Mg Si O

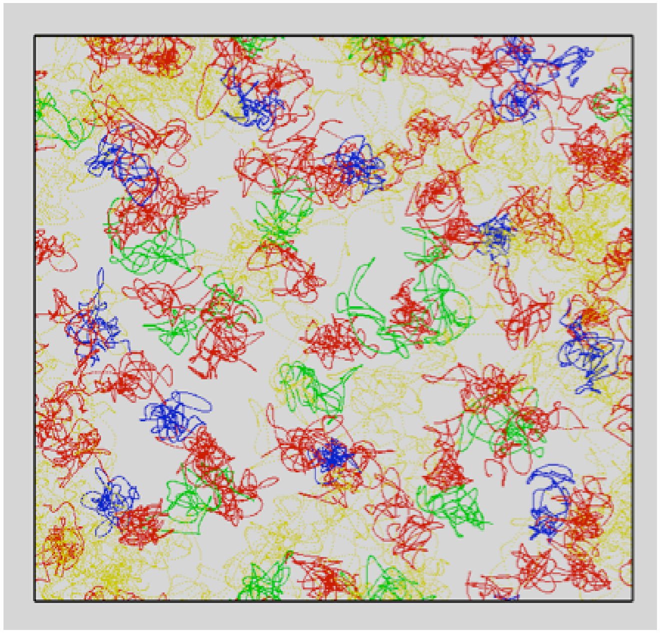

Instantaneous displacement ($f = 20$)


Displacement over 500 steps ($f = 0.4$)


Centroid spheres over 2000 steps ($f = 0.2$)

Ellipsoids

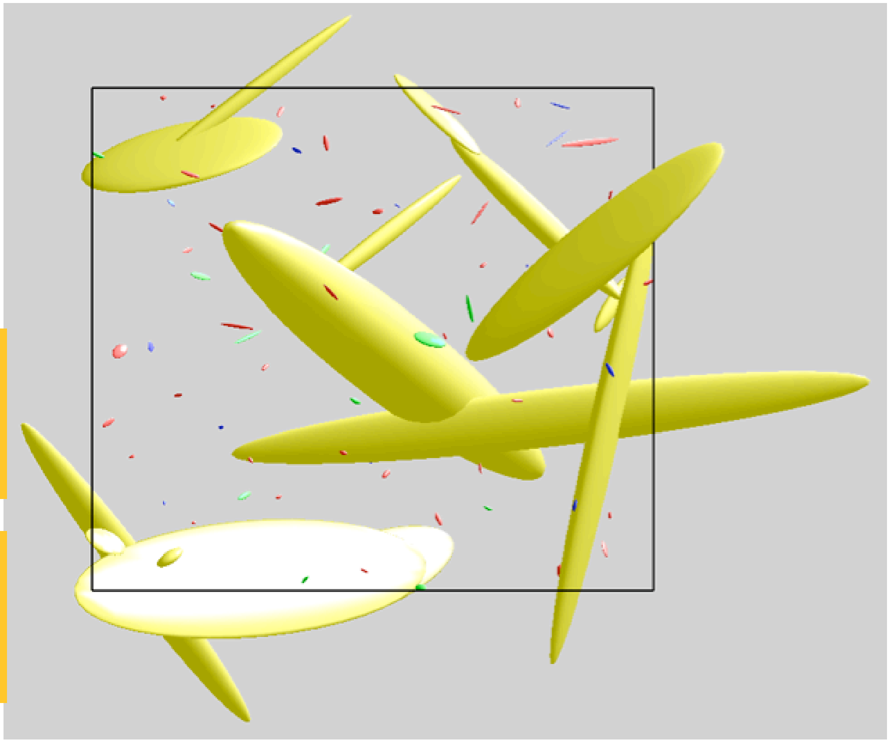
Trajectories



Mg Si O H

- Principal component analysis
  - Tensor point data

$$\mathrm{cov}_i = \begin{pmatrix} \sigma_i^{xx} & \sigma_i^{xy} & \sigma_i^{xz} \\ \sigma_i^{yx} & \sigma_i^{yy} & \sigma_i^{yz} \\ \sigma_i^{zx} & \sigma_i^{zy} & \sigma_i^{zz} \end{pmatrix}$$



$$\sigma_i^{kl} = \frac{1}{N_{STEP}} \sum_{j=0}^{j=N_{STEP}} \left(p_i^k(j\Delta t) - \mu_i^k\right)\left(p_i^l(j\Delta t) - \mu_i^l\right)$$

$$\mu_i^k = \frac{1}{N_{STEP}} \sum_{j=0}^{j=N_{STEP}} p_i^k(j\Delta t)$$

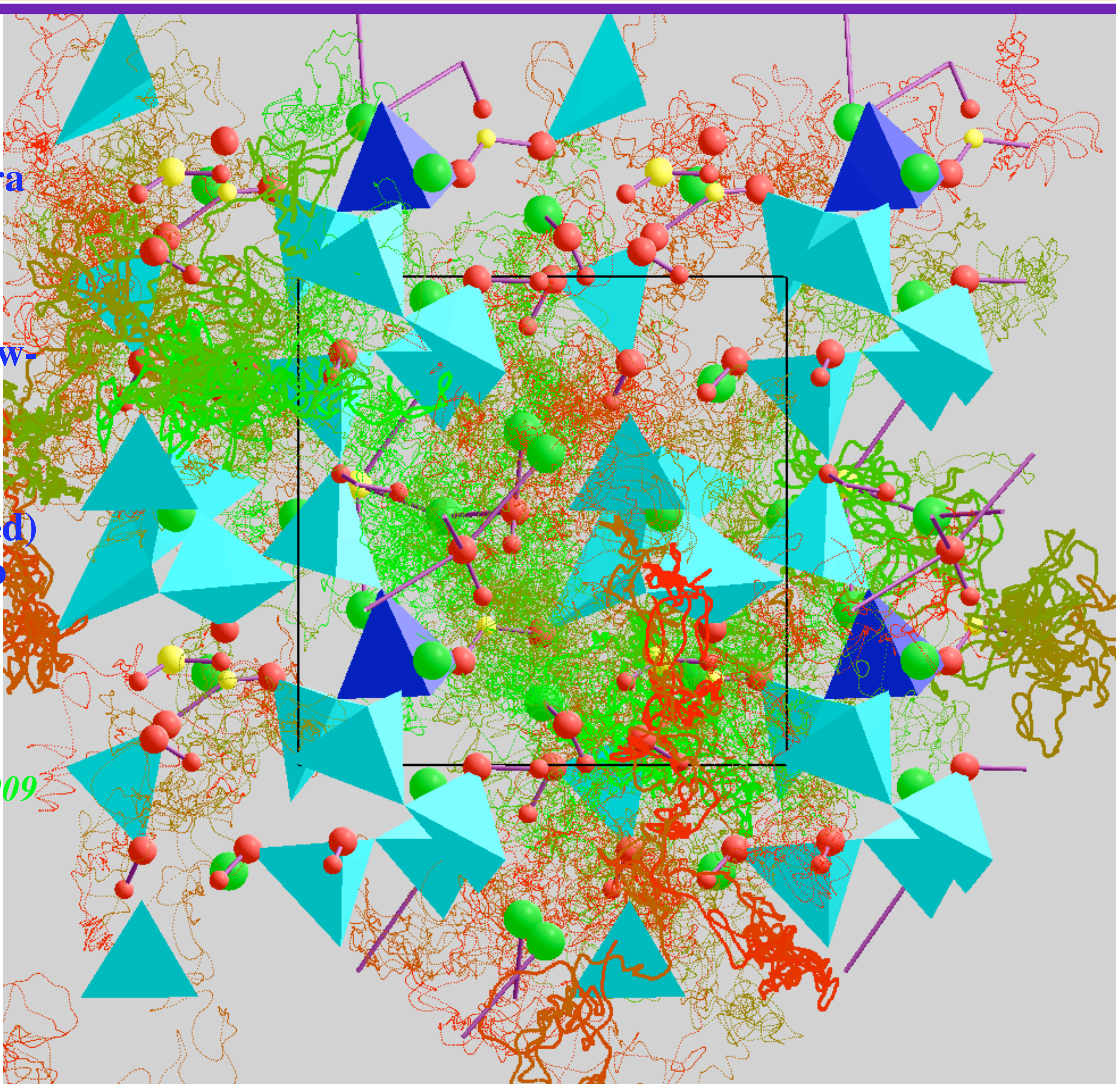
Visualizing Hydrous Silicate Liquid

First-principles MD data:

Si-O coordination polyhedra (4-fold, cyan; 5-fold, blue)

Hydroxyls (red-red pairs)
Water molecules (red-yellow-red triplets) bound to Mg (green)

H trajectories (time-encoded) in the background with two trajectories (thick) highlighted.



Bhattarai and Karki, JMGM 2009