Atomistic visualization: Space–time multiresolution integration of data analysis and rendering

Dipesh Bhattarai a, Bijaya B. Karki a,b,*

a Department of Computer Science, Louisiana State University, Baton Rouge, Louisiana 70803, USA
b Department of Geology and Geophysics, Louisiana State University, Baton Rouge, Louisiana 70803, USA

1. Introduction

Atomistic (molecular) visualization is one of the most widely studied scientific applications of visualization. Time-dependent three-dimensional scattered data from molecular dynamics simulations represent snapshots of atomic configurations and they are known to be rich in the structural and dynamical information. To gain insight into such correlated dynamical data, visualization has previously been exploited in different ways depending on the nature of the atomic systems studied. For crystalline solids in which the atoms have well-defined positions, the most of the existing visualization methods are directly applicable. On the other hand, disordered systems such as liquids and amorphous solids show weak or no long-range order. Even the short-range (local) order is temporal to a great extent and transient fluctuations are expected to occur. While many physical properties at macroscopic scale are often obtained as statistical averages, understanding them at microscopic (atomic) scale requires a detailed examination of individual snapshots. Visualization in usual sense tries to render a given dataset using various glyphs representing the atoms and bonds, or other structures already existing in the dataset. A lot of work has been previously done in this context. Some common examples include Molscript [1], VMD [2], XcrysDen [3], Atomeye [4], Atomsviewer [5,6], CrystalMaker (http://www.crystalmaker.com), amiraMol (http://www.amiravis.com/mol), Aviz, gOpenMol, VASP DataViewer, PyMD, etc. In general, the existing visualization systems share many features (see a review by Li [7]):

- A wide variety of representation (rendering and coloring) modes such as balls, points and lines, balls and sticks, space-fill and polyhedra
- Real-time manipulations such as rotation, translation, scaling and selection
- Measurement of distances, angles, and dihedrals
- Crystalline properties (e.g., switching between primitive and conventional cell settings, displaying the Wigner–Seitz cell and Brillouin-zone)

© 2009 Elsevier Inc. All rights reserved.
• Animations (particularly, for MD simulation trajectories) imported either from files or from a direct connection to a running MD simulation
• Support for a variety of database files from a variety of simulation sources.

Offline analyses of the data are used to further understand the atomic position time series [8–10]. How visualization can help in exploring the results derived from such analyses has yet to be addressed in a more complete way. To the best of our knowledge, all previous visualization schemes [7] have a little, if not at all, capability to analyze and visualize the atomic data at multiple length- and time-scales. Our scheme primarily differs from previous schemes in that we integrate data analysis and rendering tasks to support interactive atomistic visualization at space–time multi-resolution. We have previously reported some preliminary ideas and results [11–13]. In particular, the analyses produce various data on the fly that represent a given atomic system at diverse length– (e.g., nearest neighbors to second-nearest neighbors or beyond) and time- (instantaneous or finite intervals or overall averages) scales. The information is thus processed at varying levels of details. Instead of directly rendering the input positional data, additional data that usually have to be extracted by offline analyses are extracted and rendered on the fly. The extracted data are rendered as soon as they are generated to support an interactive visualization. Depending upon the nature of the materials problem, the information that has to be extracted from the simulation data may vary considerably so that one single scheme may not be suitable for the complete information extraction. It warrants that a combined analysis and visualization scheme be developed accordingly to meet domain specific needs.

2. Simulation data

In recent years, studies of complex materials systems like liquids and defect-containing solids using first-principles molecular dynamics simulation methods have become widespread. Unlike perfect crystals in which the symmetry constrains the simulation size (often unit cells with periodic boundary conditions are used) as well as the atomic configuration (a few free positional parameters need to be considered explicitly), liquids and defective systems require much larger atomic systems and the crystal symmetry is violated partially or completely. For example, MgO in its conventional simple cubic structure requires only 8 atoms whose positions are fixed by the cubic symmetry. A liquid MgO uses as many as 216 atoms [17] so that a discrete set of 648 (positional) degrees of freedom need to be explicitly taken into account. The atomic positions of a liquid phase are strongly correlated but without any long-range order. Visualization of the data at different resolutions of length and time allows us to gain insight into the complex structural and dynamical nature of the liquid state.

In first-principles molecular dynamics or more generally, in any molecular dynamics simulation, a system is initially defined by a group of properties that represent atomic positions, atomic types, and constraints that have to be satisfied [8]. Let A be the collection of \( \{R, S, C\} \) describing the system. Here, \( R = \{ r_i \in \mathbb{R}^N \} \) (\( X \) demarcates the simulation box) is the set of the atomic positions in the system, which can be considered as discrete degrees of freedom, \( S = \{ s \in \text{types} \} \) is the set of the atomic types (for instance, Mg, Si and O atoms for a silicate liquid). C is the set of constraints that has to be satisfied by the system. For example, in the FPMD simulation based on canonical NVT ensemble, the number of atoms in the supercell (N), the volume (V), and the temperature (T) are fixed. As the simulation progresses, a new system configuration is generated. Collection of A’s over a given simulation time duration describes the system dynamics, i.e., \( D = \{ \{A_t \} | 0 \leq t \leq 2 \} \). In essence, FPMD computes the trajectories, i.e., positions \( \{ r_i \} | i = 1, ..., N \} \) as a function of time \( t \), of all the atoms by numerically integrating the Newton’s equations of motion:

\[
m_i \frac{d^2 r_i}{dt^2} = F_i
\]

where \( m_i \) is the mass of atom \( i \) and \( F_i = \sum_{\{s\}} f_{is} \) is the force on atom \( i \) due to all the other atoms. In FPMD, these interatomic forces are derived from a fully self-consistent solution of the electronic structure problem within density functional theory [18]. Such quantum mechanical method is computationally more intensive than the methods based on simplified potential models.

The position-time series data used in this study were generated from parallel FPMD simulations of liquids. Our analysis and visualization primarily deals with the simulated silicate liquid containing 10 wt% water [14]. The simulation supercell consists of 168 atoms, i.e., 24 MgSiO2 units and 16 water molecules. The simulation runs range from a few thousands steps to a couple of hundred thousands of steps with 0.5 fs time step. The output results reported in the paper refer to this hydrous silicate system unless other systems (Mgo [17] or silica liquid [19]) are explicitly mentioned. It is important to note that FPMD simulations also produce data related to electronic structures (e.g., [17]) such as charge density distribution and wave functions, however, their visualization is beyond the scope of the present work.
3. Visualizing structural properties

3.1. Radial distribution function

The radial distribution function (RDF) is the simplest but most commonly used form of many distribution functions that can be defined for the atomic positions in a dynamical system [8]. It is also called the pair correlation function. The function gives the probability of finding a pair of atoms a distance \( r \) apart, relative to the probability expected for a completely random distribution at the same density. As such, it represents the average radial packing of the atoms and is used to concisely characterize the structure of complex systems like liquids and colloidal suspensions. In such systems, the particles are in a constant motion and a single snapshot of the system shows only the instantaneous disorder. The structure has meaning, in general, in an average sense and RDF is often considered an effective way of describing such average structure. Another advantage is that it can be measured experimentally using neutron and X-rays diffraction techniques allowing us to compare simulations with experiments. Finally, in the case of the pair-wise potential function, thermodynamic quantities, particularly, the energy and pressure can be expressed directly in terms of RDF.

RDF of a crystalline system in which the constituent atoms wander about their equilibrium positions in a lattice shows sharp peaks at fixed distances and is zero everywhere else. On the other hand, RDF of a disordered system like a liquid shows different patterns (Fig. 1). After some initial distance, it picks a non-zero value up and sharply increases to reach a peak value, which may not be as well defined as it is in the case of crystal. As the radial distance increases, RDF decreases to reach some minimum value (often larger than zero) and thereafter may show another shallow peak before it starts to fluctuate around the unity eventually converging to the unity at larger distances. The atoms too far from each other are statistically uncorrelated so there is no long-range order. For a system with more than one type of atoms, we have to differentiate between the function that describes radial distribution of the particles irrespective of their types (atomic species), called total RDF and those that describe radial distribution of the atoms of a particular type around atoms of another type, called partial RDF. Taking atomic species into consideration enables a detailed analysis of a multi-species system and is useful when distribution of one atomic species around another atomic species is important. For example, in silicate phases, distributions of oxygen atoms around magnesium or silicon atoms are of special interest. The radial distribution function for atomic pair of species \( \alpha \) and \( \beta \) is defined by

\[
g_{\alpha\beta}(r) = \frac{1}{4\pi \rho_\beta r^2} \left[ \frac{\Delta N_{\alpha}(r)}{\Delta r} \right]
\]

where \( \rho_\beta = N_\beta / V \) is the number density of species \( \beta \). For a finite-size supercell used in the simulation, the RDF computation has to take the imposed periodic boundary conditions into account.

We use the binning method for the RDF computation. This involves constructing a series of concentric spheres set at a fixed distance \( \Delta r \) apart around each atom of species \( \alpha \) in the system and counting the atoms of species \( \beta \) in each shell. The total number of species \( \beta \) in the shell of radius \( r \) and width \( \Delta r \) is defined as:

\[
\Delta N_{\beta}(r) = \sum_{i=1}^{N_\beta} \sum_{j \neq i} I(r < d(p_i, p_j) \leq r + \Delta r)
\]

Here, \( d(p_i, p_j) \) is a distance function that computes Euclidean distance between two points, and the indicator function for a predicate \( p, I(p) \), is 1 if \( p \) is true and 0 otherwise. There are \( N_\alpha \) and \( N_\beta \) particles of these species. The calculated number of atoms in the shell needs to be normalized by the number that would be observed if the atoms were uncorrelated. A chosen maximum distance \( d_{\text{max}} \) up to which we want to compute RDF is divided into a number of bins, \( n_{\text{max}} \), so that \( \Delta r = d_{\text{max}} / n_{\text{max}} \). Note that these bins correspond to the spherical shells of width \( \Delta r \) of increasing radius. The atom pairs whose inter-atomic distances fall within individual bins are counted. All the atoms at a distance \( 0 < r < \Delta r \) are put in \( b_{in} \), atoms at a distance \( \Delta r < r < 2\Delta r \) are put in \( b_{in} \), in general, atoms at a distance \( i\Delta r < r < (i + 1)^2 \Delta r \) are put in \( b_{in} \). Thus computed RDF is for an atomic configuration at a particular time instant. An instantaneous RDF fluctuates too rapidly without conveying much useful information. Also as the system is assumed to be ergodic, time averaged distribution is supposed to represent a meaningful distribution. We perform an average over an extended simulation period:

\[
\Delta N_{\beta}(r) = \int_0^r \frac{\Delta N_{\beta}(r) \, dr}{\int_0^r \, dt}
\]

Here, \( \Delta N_{\beta}(r) \) is time-indexed version of \( \Delta N_{\beta}(r) \) and its time average \( \Delta N_{\beta}(r) \) is used in the calculation of RDF. Being a probabilistic function, a physically significant RDF requires the system be simulated for a sufficiently long time.

There are several ways for speeding up the RDF computation. First, the atomic positions are time-correlated and from one time step to another time step they do not change substantially. This allows step skipping during the RDF computation without significantly affecting the accuracy. Second, unlike crystalline solids in which the structure is periodic and well defined, in disordered systems RDF shows well-defined peaks only at relatively short distances (Fig. 1). This suggests that we can choose a small value for \( d_{\text{max}} \) such that only the first and second peaks are covered. Third, we can increase the size of bin or equivalently decrease the number of bins. The curves are then smoothed if needed. Smoothing is, in general, useful to precisely compute the positions of the peaks and minima since the calculated RDF is subject to statistical noise, which can be relatively large in some cases due to poor sampling. Smoothing is achieved by a local fitting in \( r \). Finally, one can perform partial

Fig. 1. Si–O radial distribution function for the hydrous silicate liquid. The vertical lines (red and blue) are used to mark the critical distances.
RDF computation selectively since often a small subset of partial RDFs is of particular importance and frequent use.

RDF can be considered as a starting point for a detailed structural analysis of disordered systems. The presence of one or more peaks at relatively short distances means that the system exhibits a short- to mid-range order. To fully explore such structures, information on critical distances indicating the positions of peaks and minima such as the distance to the first minimum \( r_{\text{min}} \) is needed. Also, a cutoff window of arbitrary width can be selected to further examine structural correlation. For instance, one can calculate the bond length as a weighted average of all distances covered by the first peak by integrating \( g_{ab}(r) \) up to \( r_{\text{min}} \). Similarly, critical distances are used in computing bond angles, coordination, clusters, etc. The RDF plot allows the user to interactively pick up and adjust as necessary the critical distances or window widths used by subsequent analysis. Moreover, the user can vary in real time the range of steps, the size of skipping, the number of bins, the maximum distance, and atomic species to achieve desired performance. A square matrix plot is used to simultaneously render the complete set of partial RDFs for a multi-component system (Fig. 2). However, we can also render only a subset; a row plot rendering RDFs of all species about one selected species whereas a column plot rendering RDFs of a given species about all species. Finally, a single plot renders only one RDF corresponding to a selected pair of species exploiting a full display space so that its details can be better explored. In this mode, RDF can be decomposed into multiple components by treating structurally inequivalent atoms of the same species differently (Fig. 3).

### 3.2. Coordination environment

Atomic coordination, which is an important property used for characterizing the local structure, can be calculated for a given species \( \alpha \) with respect to another species \( \beta \) as:

\[
C_{\alpha \beta} = 4\pi \rho_\beta \int_{r_1}^{r_2} r^2 g_{\alpha \beta}(r) \, dr
\]

Normally, we choose \( r_1 = 0 \) and \( r_2 = r_{\text{min}}^{\alpha \beta} \), a distance window which covers the first peak in RDF (Fig. 1). In this case, the Eq. (5) gives the average number of atoms of species \( \beta \) that lie within spherical regions around atoms of species \( \alpha \) and of radius \( r_c = r_{\text{min}}^{\alpha \beta} \), called the coordination distance. \( C_{\alpha \beta} \) represents the nearest neighbor coordination number and corresponds to the first coordination shell. The atomic distances, which fall between \( r_1 = r_{\text{min}}^{\alpha \beta} \) and \( r_2 = \) the minimum after the second peak, determine the second coordination shell. Since the second peak is poorly defined generally in disordered system, our coordination analysis only deals with the first shell. Coordination represents the structure in a

![Fig. 2. RDF matrix (symmetric) plot rendering all partial radial distribution functions (considering both like-atom pairs and unlike-atom pairs) for the hydrous silicate liquid. The vertical lines mark the critical distances, the first peak position (red line) and the minimum position (blue line).](image-url)
Here, each term $c_{e_{αβ}}$ represents the coordination environment of species $α$ with respect to species $β$ in terms of a single number ($c_{e_{αβ}}$) in the average sense. The diagonal terms represent like-atom coordination whereas the off-diagonal terms represent unlike-atom coordination. For example, with $s = 4$ for hydrous magnesium silicate liquid, there are 16 different partial environments. Both the RDF and mean coordination have limitations as being isotropic so it is important to supplement interpretation of these functions with other structural information by exploring individual snapshots.

One way of doing this is to decompose each coordination environment ($c_{e_{αβ}}$) into a variety of coordination species, which depend on space (i.e., vary from atom to atom) and time (i.e., vary from snapshot to snapshot).

### 3.2.1. Partial coordination environment

The coordination environment of an individual atom in the system at a given time instant represents the distribution of other particles around it. In particular, we are interested to know in the immediate neighborhood of each atom $i$ of type $α$ how many other atoms (labelled as $j$) of type $β$ can be found and how they are arranged. The neighborhood chosen for coordination is simply a spherical region of radius $r_C = r^α_{\text{min}}$, and all particles of type $β$ falling in this neighborhood are taken into account. The partial coordination environment of an atom (labelled as $i$) at time $t$ is thus defined by

$$nn^α_β(t) = \{1 \leq j \leq N : d(i,j) \leq r^α_{\text{min}} \land \text{type}(j) = β\}$$

where $N$ represents the total number of atoms in the system. Thus in a multi-component system, each term in the matrix representation of an instantaneous coordination environment becomes

$$c_{e_{αβ}}(t) = \{nn^α_β(t) \lor i \in N_α\}$$

For each pair of species, a unique coordination environment exists. The corresponding snapshot contains coordination information of all atoms of species $α$ with respect to the atoms of species $β$ at time $t$ (Fig. 4). In the ball-and-stick representation, each coordination snapshot is visualized by displaying atoms of both species under all of their coordinated atoms ($f$'s) by lines. The numbers of lines emanating from the central atoms represent the corresponding coordination numbers $|nn^α_β(t)|$'s. The central atoms are rendered with colors selected from a color map in which the color represents the coordination number (Fig. 5, left).

In some cases, the local coordination environment can be better expressed as polyhedral units. For instance, four-fold coordination means a tetrahedron with the type $α$ atom being at the centre and surrounding type $β$ atoms at the vertices of the tetrahedron. The coordination polyhedra are computed using Quhill algorithm [20]. The number of vertices in the polyhedron is equal to the coordination number of the atom, which is assumed to be at the centre of polyhedron. The color of the polyhedron encodes the coordination number (Fig. 5, right). The links between the central atom and the vertex atoms can be also drawn with an option of the length information encoded. An advantage of the polyhedral representation is that we can study other properties of this structural unit. Polyhedra are rarely perfect (regular). The degree of polyhedral distortion can be characterized in terms of quadratic elongation and bond-angle variance [21,22], which are defined, respectively by

$$λ^α_β = \sum_{k=1}^{mn} \left(\frac{l_k}{l_0}\right)^2$$

and

$$σ^α_β = \sum_{k=1}^{mn} (\theta_k - \bar{θ}_β)^2$$

Here $l_0$ and $θ_β$ are bond lengths and angles in regular polyhedra, respectively, $l_0$ is computed as the average of all lengths forming a polyhedron. There are fixed angles only for a few regular polyhedra; for instance, $θ_β = 110.947$° for $nn_β = 4$ (tetrahedron) and $θ_β = 90°$ for $nn_β = 6$ (octahedron) so the bond-angle variance can be calculated only for some cases. Fig. 6 shows that Mg–O...
polyhedra are more distorted than Si–O polyhedra in the silicate liquid. Another relevant quantity is the polyhedral volume, which varies between polyhedra of the same or different types. It is important to note that coordination can be also represented using other types of polyhedra obtained by using the Voronoi construction method [10]. Such polyhedra are computed by finding the planes, which enclose the set of points that are closer to the central atom than to any other atom. Unlike the coordination polyhedra studied in this study, the Voronoi construction may include the contributions from the second coordination shell and does not directly represent the coordination number. For example, a six-fold Si–O coordination state maps to a cubic Voronoi polyhedron rather than to an octahedron (six O atoms at the vertices and Si atom at the center). A detailed analysis of Voronoi polyhedra was previously carried out for amorphous silica [10].

3.2.2. Mixed coordination environment

A coordination matrix (or row or column) plot renders simultaneously more than one partial coordination environment using separate viewports (Fig. 4). While it allows us to compare different environments with each other, it is also desirable to render the coordination environments among multiple pairs of species in the same viewport. Such a mixed plot is expected to be useful in exploring how different coordination environments complement with each other to define the overall structure of the system in question. For instance, visualization snapshot consisting of color-coded Si–O coordination polyhedra, and O–H and H–O coordination spheres can assist in the identification of different forms of water speciation (Fig. 7). However, using a single viewport to simultaneously render multiple coordination environments has a limitation. An element in the coordination matrix specifies coordination environment of the species defined by the row with respect to the species defined by the column. When multiple environments are visualized together in the same viewport, only one coordination environment from a row can be rendered because one row corresponds to one species, and spheres representing the atoms of this species can be color-coded in only one way to convey the coordination information. Conversely, multiple coordination
environments from one entire column can be selected for a mixed coordination snapshot. The limitation can be modelled using a directed graph, $D$ in which vertices are the atomic species and the edges are the coordination relationships between the corresponding pairs of species (Fig. 8). The graph can also contain a loop in it when coordination of a species with respect to the same species is selected. For the graph to be practically useful, each vertex must have only one outgoing edge although it can have many incoming edges. In Fig. 8, each vertex has only one outgoing edge so this is a valid graph for mixed coordination representation.

### 3.2.3. Coordination stability

Coordination varies from one snapshot to another. For a given snapshot, the atomic configuration is composed of various coordination species since coordination varies from one atom to another of the same species. It is important to explore the time dependence of these coordination states. We define the stability of a coordination environment for a particle $i$ as the fraction of time the environment exists in the system. The stability consists of two components: the stability of the coordination number and the stability of the coordinated atoms. An atom can have the same coordination number even when it gets coordinated with different set of neighbors at different instants. Let $c_i = \sum_{j=1}^{N} n_i^j(t)$ be an instantaneous coordination number for atom $i$ at time $t$. This number is an integer for individual atoms because at any instant, either one atom is coordinated with another atom or it is not coordinated. Two atoms cannot be partially coordinated. So, over $N$ snapshots, we track atom $i$ to find the fraction of the time it is coordinated with $c_i(t)$ atoms. Let this fraction be $f(c_i).$ Then, $c_i$ and $f(c_i)$ are visually represented, respectively, by the color and size of the sphere representing the central atom (Fig. 9). For an atom $i$ of species $\alpha$, the nearest neighbors of species $\beta$ at time $t$ are given by
Then, the set of all the atoms of species $b$ that get coordinated to the atom $i$ during an extended period of $N$ steps is thus:

$$\text{NN}_{i}^{ab}(t) = \bigcup_{j=0}^{N} \text{nn}_{i}^{ab}(j \Delta t)$$

(10)

Let $f_{i}(k)$, where $k \in \text{NN}_{i}^{ab}$, is the fraction of time the atom $k$ is coordinated with atom $i$. Then, when the atom $k$ gets coordinated with atom $i$, $f_{i}(k)$ is used to represent the stability of this coordination relationship, which is encoded by the width of the bonds (Fig. 9).

The set of atoms, which are coordinated with the central atom under consideration during a selected temporal interval are referred to as a coordination cluster (Fig. 10). In any snapshot, this cluster comprises of those atoms that are currently coordinated to the central atom and those atoms that have contributed to the coordination at some other time (before or after the current instant). Visualization differentiates the two types of contributions by using different color or thickness for the connecting lines between the central atom and contributing atoms. Each atom of species $a$ forms its own coordination cluster, which can be rendered separately or together with all other such clusters.

The stability of a polyhedral surface is harder to visualize than the stability of individual bonds. This is because we are actually trying to visualize the effect of more than one bond and somehow represent the cumulative effect of all those bonds in one object at the same time. For this purpose, we first define the stability of the surface as $s_{p} = \prod s_{\text{bond}}$, where $s_{\text{bond}}$ is the stability of individual bonds defined as the fraction of the time it exists in the dataset. As each of the bonds has to exist simultaneously to form the polyhedral structure, we multiply the stabilities of individual bonds to compute the stability of the polyhedron as a whole. One way to represent the polyhedral stability is to modulate the polyhedral surface itself using $s_{p}$ as the modulating quantity and the other way is to vary the color intensity accordingly (Fig. 11). Such modulation of shape or intensity variation tells us about the

---

**Fig. 7.** Mixed coordination plot displaying the Si–O, O–H and H–O coordination environments together. Hydroxyls, water molecules, polyhedral bridging, edge decoration, and four-atom sequences are present. The coordination numbers of 0, 1, 2, 3, 4, 5, 6 and 7 are, respectively, represented by black, red, yellow, green, cyan, blue, magenta and white colors.

**Fig. 8.** Graph model of mixed coordination environment.

**Fig. 9.** Stability of Si–O (left) and Mg–O (right) coordination. Large spheres and thick bonds in left suggest that the four-fold state (cyan) for Si–O coordination is highly stable and dominates the coordination environment. A five-fold coordination state (blue) hardly exists (marked). The color and size of bonds encode the length and stability of the bond, respectively. On the other hand, the Mg–O coordination (right) consists of various states with a wide range of stabilities.
overall stability of the polyhedral object as a whole but it does not
tell us why the object is stable or unstable. Also, as only size
variation is quantitative, change in shape will not be able to convey
the quantitative information that we want to convey. It also does
not tell us anything about which atom(s) is (are) more likely to
detach from the surface than the others or which atoms are likely
to be bound with the surface more strongly than the others. Instead
of modulating the shape, we modulate the size of the coordinated
atom to visualize the probability of their coordination \[13\].

Probability of coordination of atom \(k\) with another atom \(i\) is given
by \(f_i(k)\).

3.3. Cluster structures

3.3.1. Nearest neighbor cluster

An atom \(i\) is said to be a nearest neighbor of another atom \(j\), if
\(d(i,j) < r_c\), where \(r_c\) is the cutoff distance. If we can traverse through
a group of atoms following a path from one atom to another atom
such that the distance between any two consecutive atoms on the
path is within the cutoff, we obtain a cluster which connects
successive nearest neighbor atoms. This is referred to as a nearest
neighbor (NN) cluster. Let \(\text{nn}_i = \{1 \leq j \leq N_a; d(i,j) \leq r_c\}\) be the set of
the nearest neighbors of the atom \(i\). The NN cluster structure
formed by the atom \(i\) can be defined as \(i \cup K\), where \(K\) is recursively
defined as \(i \cup (\text{nn}_i \forall j \in K)\). The breadth-first or depth-first
traversal of a graph data structure \(G(V,E)\) in which each atom is
represented as a vertex and the atomic pair which are within the
cutoff distance is represented as an edge yields the desired cluster.
The cluster contains all the atoms that are reachable from any atom
in it. The NN cluster is rendered as the new atoms are found instead
of first saving the cluster separately.

3.3.2. Common neighbor cluster

A common neighbor cluster for a given pair of atoms is the set of
all the atoms that are the nearest neighbors of both the atoms in
the pair. This method represents a way of decomposing the first
and second peaks of RDF according to the local environment of
each atomic pair that contributes to the peak structures \[23,24\]. An
atom \( k \) is said to be a common neighbor to \( i \) and \( j \), if \( d(i,k) \leq r_c \) and \( d(j,k) \leq r_c \), where \( r_c \) is the cutoff distance. For a pair of atoms \( i \) and \( j \), let \( cn = \{ k \mid d(i,k) \leq r_c \land d(j,k) \leq r_c \} \) be the set of common neighbors. Then, \( cn \subseteq \{i,j\} \) with all the coordination relationship among these atoms within the critical distance \( r_c \) is called the common neighbor cluster for atoms \( i \) and \( j \). In particular, we are interested to calculate the number and properties of common nearest neighbors of the pair under consideration. These information helps us to identify atoms in particular environment such as face-centered cubic, hexagonal close-packed, body-centered cubic or icosahedron. Different types of pairs are associated with different types of local order. There exist other methods (not considered here) for characterizing the state of a local crystal structure in a material system such as one based on computation of common atomic neighborhood parameter \([25]\).

We find the common neighbor clusters using a graph data structure \( G(V,E) \) in which each atom is represented as a vertex and the atomic pair which are within the cutoff distance is represented as an edge. Note that if two vertices are separated by more than two edges, then they cannot share the same neighbors. Cluster computation involves two steps. In the first step, the graph is traversed to find all the pairs of vertices, which are not farther than two edges. This also finds those pairs that have at least one neighbor common to each other. For all vertices, \( i \), in the adjacency list of every vertex \( v \), it checks whether we already know some of its other nearest neighbors. If we do, then we have found a common neighbor to some pairs of vertices and this vertex is added to the common neighbor of each of those pairs. The vertex, \( v \), is also added to the nearest neighbor list of vertex, \( i \). At the end we have common neighbor lists for all pairs that have at least one neighbor common between them. The second step then finds the subgraph spanned by the set of common neighbors. The subgraph and the corresponding vertex-pair together constitute the common neighbor cluster. So the common neighbor cluster is the intersection of the coordination environments of the two vertices. Fig. 12 shows the computed clusters of different sizes for MgO liquid.

3.4. Ring structures

Ring structures are used to understand the medium range order. The ring statistics are widely used to analyze the structural properties of amorphous solids, liquids and nanomaterials \([9,16,26,27]\). The arrangements of the rings formed by the atoms can describe the topology of these systems. There are different definitions of rings available in the literatures and the ring statistics differ according to the definitions \([9,28–31]\). Software systems are also available for offline ring analysis \([32]\). Finding all rings in a system is computationally intensive because the ring computation contains Hamiltonian cycle problem as a special case and there are possibly an exponential number of rings in the system. Considering all rings in the system is not practical for relatively large systems. The maximum ring size can be restricted to a physically relevant size instead of trying to compute all the possible rings. A finite-size supercell used in the simulation is first replicated to find the largest possible ring. This process makes multiple copies of an atom and assigns each of them a unique \( id \). Thus, rings are kept as lists of atoms arranged in parent-child relationship where each atom is accompanied by its position in the replicated system and its original atomic \( id \). The atomic \( id \)'s in the

- **Fig. 12.** Common neighbor clusters containing two, three (mixed), four and five common neighbors for cutoff distances of 2.5, 2.7, 2.9 and 3.1 Å in MgO liquid.
original system have to be mapped from the replicated id’s. Let \( r_i \) be a new index for an atom \( i \) in the replicated system and \( f \) is the replication factor. Function to convert the original index to a set of replicated indices, \( R_i \), is:

\[
R_i = \{ rf \times i + j | 0 \leq j < rf \}; \quad r_i \in R_i
\]  

(11)

So, one index is replicated to \( f \) unique indices and the atomic id \( (i) \) in the original atomic system is computed from \( r_i \) as: \( i = r_i / f \).

Suppose \( G = (V, E) \) is the graph with vertex set \( V \) and edge set \( E \). For any two vertices \( x \) and \( y \) in \( V \), if there is a sequence of vertices, \( p = v_1, v_2, \ldots, v_n \), such that \( (v_i, v_{i+1}) \in E \) for all \( 1 \leq i \leq n-1 \), \( x = v_1 \), and \( y = v_n \), then \( p \) is a path between the vertices \( x \) and \( y \). The length of the path is equal to the number of edges in the path. A path between two vertices \( x \) and \( y \) is a cycle if vertices \( x \) and \( y \) are the same. If every vertex in the cycle except one vertex appears exactly once and the one that appears more than once appears exactly twice, then the cycle is called a simple cycle. There are two distinct paths (of not necessarily equal lengths) between any two vertices in the cycle. The shortest path between two vertices can be defined with respect to the graph, \( G \), or with respect to a cycle. If the shortest path between \( v_i \) and \( v_j \) with respect to the simple cycle is not greater than the shortest path between those two vertices with respect to the graph \( G \), then the cycle \( c \) is a primitive ring. Thus, a ring is primitive if another shorter path between any two vertices in the ring other than the shortest path in the ring itself between those two vertices does not exist. Local ring structure for a vertex \( v_i \) is defined as a set of primitive rings of which \( v_i \) is also a part: \( R(v_i) = \{ c | v_i \in c \} \). This is the typical definition used to calculate ring statistics in a material system. Thus, only the primitive rings are of interest and large rings are less likely to be primitive as they usually have shortcuts.

We first give a brute force approach [9] for the ring computation. A logical cluster data structure for each atom in the system is constructed from the graph of the complete system (Fig. 13). It stores the information about all the atoms reachable at different levels from the atom. The main idea behind the brute force approach is that whenever there is one vertex at the same level in two different places in the cluster structure, and if this is the case, then there must be two unique paths to that vertex starting from the root. A ring can be formed joining the two paths. For large rings, there are many different paths between the vertex and the root, and the number of possible paths in a graph increases exponentially with the ring size. When there are \( n \) different paths, then any pair of paths can make a ring so there are \( n(n - 1)/2 \) different possible rings. However, all the rings may not satisfy the primitiveness criterion. A large number of rings have to be checked for their primitiveness when the ring size is large.

Due to an exponential increase in the number of possible paths and also the size of the cluster structure itself, the brute force algorithm becomes computationally intensive for large systems. At each level, there will be an exponential number of atoms and information that needs to be stored also becomes exponential for large rings. However, all the data need not be stored for efficient computation of the primitive rings set. To reduce the redundant information, the cluster data structure is explicitly split into two separate data structures: a graph and a shortest distance map. Such restructuring of the data structure results in a better algorithm [30]. The improved algorithm first computes shortest distance maps, \( S_{\text{map}} \), for all the vertices in the graph. Then all the vertices \( (m_i) \) at a distance \( r_{\text{size}}/2 \) from vertex \( i \) are computed. If any of the vertices in \( m_i \) has more than one distinct path to the vertex \( i \), then there is possibly a primitive ring. So, all the paths from each middle vertex to the vertex \( i \) are found for each pair of vertex \( i \). Each possible ring is formed and checked for its primitiveness. The computational cost of checking whether a ring is primitive is reduced by limiting the number of pair of vertices checked. We use the mid-node theorem [33] that states, “For any primitive even 2\( r \)-ring, between any check-node and its mid-node, the two equal paths with \( n \) links along the ring must be the shortest.” If the ring is primitive, then it is added into the set of primitive rings. The algorithm described is valid for finding even-sized rings.

The rings are stored as lists of atom indices and positions. They are visualized using the standard ball-and-stick representation (Fig. 14, left). Superimposed on this is rendered a surface generated by a set of triangles for each ring (Fig. 14, right). Consider a ring, \( R \{ r_0, r_1, \ldots, r_{|R|} \} \), for vertex \( i \), such that \( r_0 \) is the parent of \( r_1 \), for \( k > 1 \) and \( |R| \), is the parent of \( r_1 \). The triangle is calculated by taking a set of three vertices, where one vertex is \( r_1 \) and remaining two vertices are \( r_k \) and \( r_{k+1} \), where \( 2 < k < |R| - 1 \). The normal for each triangle is calculated separately. The triangulation algorithm treats individual rings independently. Being computationally intensive, the ring algorithm is implemented to run in separate thread from the main application thread. To achieve interactive visualization of ring structures, the rings are computed for the selected atom for the current time step only. However, finding the ring statistics involves computation of rings for all the atoms averaging over multiple steps. Fig. 15 shows the Si–Si ring structure of silica liquid as the maximum ring size used in the computation is varied from 2 to 12. The ring structure for the cutoff size of 12 consists of eight rings, which include those found for the smaller cutoff sizes.

4. Visualizing dynamical properties

The information on atomic configuration from a MD simulation is time ordered and is commonly visualized using the methods like animation and pathlines. The animation method renders atomic configuration at each time step at a time whereas the pathlines method displays the complete trajectories of all atoms by rendering the atomic positions at all successive time steps. While such representations do not discard any information, the animation presents an instantaneous picture whereas highly entangled trajectories are difficult to interpret. There are other ways of understanding the dynamics as presented here. Various kinds of displacement data are extracted to understand the extent and nature of the movement of the atoms.

4.1. Diffusion coefficients

Assuming that the simulation dataset in question consists of only atomic positions, we can use the Einstein relation to compute
self-diffusion coefficient for each atomic species [8] as follows:

\[ D_a = \lim_{t \to \infty} \frac{\langle |r(t)|^2 \rangle}{6t} \]

where \( \langle |r(t)|^2 \rangle \)

is the mean square displacement (MSD). The positions of the \( i \)th atom at time origin \( t_0 \) and then after time \( t \) are \( r_i(t_0) \) and \( r_i(t + t_0) \) respectively. Translational symmetry of the system is accounted for so that the MSD value is not restricted by the size of the supercell. The partial MSD was calculated by averaging over atoms of a given species. To get a better statistics, MSD is averaged over different time origins, \( t_0 \)’s. MSD curves are extracted and rendered. One could plot these curves for individual atoms separately or the total MSD for different time origins. Alternatively, the integrals of the time-correlation functions for velocities can be used to compute diffusion coefficients. Similarly, other types of the time-correlation functions can be computed depending on the simulation outputs.

4.2. Pathlines

A pathline or trajectory represents the actual movement of atoms in space during a finite time of simulation. In a solid system,
each atom remains confined within a small region around its equilibrium position and the trajectories of different atoms do not overlap. In a dynamical system on the other hand, atoms move a lot resulting in dispersed and overlapping trajectories. As an atom moves within the supercell satisfying the periodic boundary conditions, it can be traced using a color-coded line. The pixel color at a given position is varied according to the elapsed time or the distance travelled by the atom relative to some reference position. The pathlines are used to visualize the raw movement of individual atoms (Fig. 16) and those forming a well-defined structural unit such as a polyhedron and its stability [12].

4.3. Atomic displacements

Various sets of displacement data are generated to characterize the extent and pattern of the atomic movement during simulation. In general, displacements are defined as the extent and pattern of the atomic movement during simulation. 4.3. Atomic displacements

4.4. Principal component analysis

The atoms do not move equally in each direction. Principal component analysis (PCA) [34] is used to analyze the overall extent and anisotropy of each atom movement. PCA usually assumes that data points are independent of each other. In strict sense, this is not the case for the position-time series as they remain highly correlated for long durations. By relaxing the requirement of probabilistic independence among the data points, PCA is expected to provide valuable information about the aggregate motion of individual atoms by reducing the dimensionality of a dataset that consists of a large number of interrelated variables. We define the covariance matrix for each atom, i, for a given dataset by

\[
\Sigma_i = \begin{pmatrix}
\sigma_{xx}^i & \sigma_{xy}^i & \sigma_{xz}^i \\
\sigma_{xy}^i & \sigma_{yy}^i & \sigma_{yz}^i \\
\sigma_{xz}^i & \sigma_{yz}^i & \sigma_{zz}^i
\end{pmatrix}
\]  

(14)

The diagonal terms of the covariance matrix are the variances along three orthonormal axes (x-axis, y-axis and z-axis) and off-diagonal terms are the covariance between the two axes on the superscript. To compute the covariance matrix, the mean position

Fig. 16. Si–O coordination polyhedra (four-fold, cyan; five-fold, blue), hydroxyls (large red–small red pairs) and water molecules (small red–large yellow–small red triplets) with H atom pathlines (time-encoded over 10 ps, green to red) and Mg atoms (green spheres) in the background. Trajectories of two randomly selected H atoms are highlighted with bold pathlines.
of the particle is computed as,

$$\mu_l = \frac{\sum_{j=1}^{n_{\text{steps}}} p_i(j \Delta t)}{n_{\text{steps}}}$$  \hspace{1cm} (15)

and the covariance and variances of the positional components are then computed as,

$$\sigma_{kl}^2 = \frac{\sum_{j=1}^{n_{\text{steps}}} (p_k(j \Delta t) - \mu_k^2)(p_l(j \Delta t) - \mu_l^2)}{n_{\text{steps}}}$$  \hspace{1cm} (16)

where \( k \) and \( l \) are two of the three positional components, \( x \), \( y \), and \( z \). The covariance matrix is a symmetric matrix. Its eigenvalues and eigenvectors give the amount and direction, respectively, of the maximum variation in the data. For a positive semi-definite symmetric matrix, eigenvalues are real. Let the three eigenvalues and corresponding three eigenvectors be \( \lambda_1, \lambda_2, \) and \( \lambda_3, \) and \( v_1, v_2, \) and \( v_3, \) respectively. Then we can define a principal component matrix, \( A, \) as:

$$A = \begin{pmatrix}
\lambda_1 v_1^1 & \lambda_1 v_1^2 & \lambda_1 v_1^3 \\
\lambda_2 v_2^1 & \lambda_2 v_2^2 & \lambda_2 v_2^3 \\
\lambda_3 v_3^1 & \lambda_3 v_3^2 & \lambda_3 v_3^3 
\end{pmatrix}$$  \hspace{1cm} (17)

Now, let’s transform the position set, \( p_i(t) \), by \( A \) as \( p'_i(t) = A p_i(t) \). The transformed positions are defined with respect to the frame reference specified by the three orthogonal eigenvectors. These positions have the maximum variation along the \( v_1 \) axis and successively decreasing variations along \( v_2 \) and \( v_3 \) axes. The ellipsoids are used to visualize the eigenvalues and eigenvectors of the covariance matrices (Fig. 18). We use \( \lambda_1, \lambda_2, \) and \( \lambda_3, \) as the major, medium and minor axes, respectively, of the ellipsoid and use eigenvectors, \( v_1, v_2, \) and \( v_3, \) to construct a rotation matrix to orient the ellipsoid. The principal components are uncorrelated with each other so they form an orthogonal frame of reference. When the set of eigenvectors are not unit vectors, their values are normalized to provide an orthonormal frame of reference for the ellipsoids. The mean position of the set of positions is the center of each ellipsoid.

5. System design and implementation

A number of analysis and rendering methods used for extracting structural and dynamical information from a given position-time series have been presented in previous sections. We now discuss the design decisions taken while integrating the analysis and rendering tasks to develop the atomistic visualization system.

5.1. Layered structure

Our visualization system is designed in a layered fashion (see below). At the bottom, there are rendering functions to render lines, spheres, cylinders and ellipsoids, and data structures to handle various kinds of data processed in real time. The layers above this provide functionalities for analysis and visualization. The visualization algorithms have direct access to the underlying rendering algorithms and low-level data structures, and also to the analytical algorithms on the same layer.

![Diagram of the visualization system](image)
programmability of modern GPU to render quadric objects such as spheres and cylinders. Hardware accelerated algorithm for quadric rendering [35] is implemented as the vertex shader and fragment shader programs. The vertex shader receives the center position of the quadric, size of the quadric, projection matrices and viewport information from the application and computes the size of the point that completely covers the quadric. The fragment shader then computes per pixel normal information and performs lighting computation for each pixel. This results in a high quality output, which is screen-resolution independent (i.e., the quality remains high irrespective of the viewport size) because the algorithm uses implicit function for the rendering.

**Data Structures:** The individual modules maintain their own data structures to manage the module specific information. This way provides higher-level interfaces to the underlying data that other algorithms can use to provide higher-level functionalities and to reduce the unnecessary recomputation. We present three important data structures: First, the coordination computation time is linearly dependent on the number of time steps. Also the number (s) of species plays a role in the computation time since one needs to deal with a square coordination matrix of order \( s \). A special data structure is designed to enable exploration of various coordination properties without extensive overhead of recomputation. The coordination environments for all pairs of species, \( \alpha \) and \( \beta \), are stored in separate data structures, \( ce_{\alpha \beta}^{\text{v}}(t_1, t_2, r_1, r_2) \), where \( i \in N_s \). The data structure contains complete information about coordination environment for a pair of species including the coordination states, stabilities, distribution and cluster. The contents depend on the range of time steps (\( t_1 \) and \( t_2 \)) and the range of cutoff distances (\( r_1 \) and \( r_2 \)). They are used to compute and render the information in going from one visualization session to another session. Only when either the time or distance range is changed, the contents of the data structure changes and have to be updated. Now, we formally define the data structure:

\[
\text{ce}_{\alpha \beta}^{\text{v}}(t_1, t_2, r_1, r_2) = (J, D)
\]  

where

\[
J = \{(j \in N_b, t) \mid \exists \text{st} t, r_1 \leq d(p_i(t), p_j(t)) \leq r_2, T := \{t_1 \leq t \leq t_2 \mid r_1 \leq d(p_i(t), p_j(t)) \leq r_2\}
\]  

and

\[
D = \{(m, t_m)\}.
\]  

Here \( J \) is the set of pair of atom \( j \) of species \( \beta \) that is coordinated with the atom \( i \) of species \( \alpha \), at least, once within the selected time range and \( T \) is the set of times where the atom \( i \) was coordinated with atom \( j \). This set contains the coordination cluster of the atom \( i \). \( D \) is the set of pairs of numbers where first element, \( m \), is the coordination number and the second element, \( t_m \), is the total time the atom \( i \) had \( m \) coordination number. So, \( D \) effectively contains the coordination stability information for the atom \( i \). The bond stability between atom \( i \) and atom \( j \) is given by \( |T|/\tau \) for the total selected time range \((\tau = t_2 - t_1)\). Note that \( |T| \) is the total time the atom \( i \) was bonded with atom \( j \) given by the corresponding \( T \) set in \( J \). We can also calculate the coordination number of atom \( i \) for the current time, \( t \), as follows:

\[
C^i(t) = ||\{j | j \in T \text{ where } T \in (ce^i J)\}||
\]  

Here, \( (ce^i J) \) is the set of the atom \( j \).

The common neighbor cluster computation maintains an extensive data structure. The computed CN clusters are stored in a separate data structure so that the clusters can be explored according to different criteria such as the number of common neighbors and the number of bonds between the common neighbors. \( \text{ClusterList} \) holds the information about the number of common neighbors in the cluster. As more than one pair can have the same number of common neighbors, all the clusters having the same number of common neighbors are added to the \( \text{ClusterInfoList} \), which holds the information about the number of bonds in the cluster among the common neighbors. All the clusters having the same number of bonds among the common neighbors are added to \( \text{ClusterInfo} \), which contains the detail information about the cluster. It contains the pair of atoms (\( \text{startAtom} \) and \( \text{endAtom} \)), the number of common neighbors between them (\( \text{numCommonNeighbors} \)), the number of bonds among the common neighbors (\( \text{numBonds} \)), the list of common neighbors (\( \text{listCN} \)) and the actual cluster formed by the common neighbors (\( \text{cluster} \)). The cluster simply stores the information about the connection among the common neighbors.
If two common neighbors are bonded, then the corresponding entry in the cluster is 'true', otherwise the entry is 'false'.

Finally, the ring structure computation algorithm maintains a shortest distance map apart from the graph data structure to store the shortest distance between selected point and all the other points. The shortest distance maps among all vertices in the graph are computed using Dijkstra's shortest distance algorithm for undirected graphs. The graph is stored in an adjacency list representation for reduced memory consumption. As the rings are computed, the algorithm also maintains the list of currently computed rings.

**Analytical Algorithms:** The analysis algorithms operate on the data maintained by the data structures to provide a higher-level picture of the underlying dataset. For example, information maintained by the coordination data structure is used to compute coordination stability, coordination cluster, and polyhedral surface among other things. These algorithms have access to the data structures but not have access to the rendering functionalities. Such separation allows for an easy addition of other analytical functionalities in future.

**Visualization Algorithms:** The visualization algorithms utilize all the other modules to provide an interface to the underlying dataset. They utilize rendering functions to present the information generated by the analytical algorithms. The visualization algorithms also have access to the underlying raw dataset so that the data points can be rendered in their raw form without any extra information attached to them.

**User Interface Management:** The user interface represents an integrated view of multiple visualization approaches and enables active interaction with the system. It provides options to translate, scale and the scene, to select a group of atoms and analyze relevant attributes of the set of atoms. The user can use the two markers provided in the RDF plot to select the minimum and maximum cutoff distances for any pair of species. Any change in a parameter results immediate visual feedback facilitating visual exploration. Push buttons, selection boxes and textboxes are used to design a prototype user interface. The user interface provides access to underlying analyses and rendering parameters. These parameters control information extraction from the dataset and final graphical output generation. The complete set of the parameters for the currently selected module is readily available from the currently shown interface. There is no need to navigate multiple levels of menu hierarchy to change one parameter. We provide solution to occlusion problem by supporting selective dynamic manipulation scheme, which combines various options such as selective rendering, multiple rendering and transparency [36]. Browsing and highlighting options are added to relate various objects on the scene. The user interface is changed according to which visualization module is selected. The parameter set for the selected module can be changed immediately.

A spreadsheet-like interface is implemented to have a comparative analysis among different outputs in a very primitive form; a more complete interface is supposed to provide the typical features such as a tabular layout, operator and dependency between the cells [37]. The interface makes frequent use of multiple viewports. For instance, if there are $s$ species, the screen will have $s^2$ viewports showing coordination between all possible pairs of species. The user can select any particular viewport to further explore the coordination environment in more detail; currently selected viewport is indicated by a red-rectangle around the viewport. It is also possible to select a subset of pairs of species from the set of all pairs of species in the system. Suppose the user selects $\lambda$ pairs of species from $s^2$ pairs of species. Let $w$ and $h$ are the width and height of the window screen on which we want to map $\lambda$ pairs of coordination environments. If $\lambda$ is a perfect square, the screen is split along both width and height into $\lfloor \lambda + 0.5 \rfloor$ parts of $w/\sqrt{\lambda} + 0.5$ and $h/\sqrt{\lambda} + 0.5$. Then $\lambda$ selected pairs of species are mapped into $\lambda$ individual viewports.

**5.2. Information representation**

Our scheme can be viewed as a special case of hybridization of scientific visualization and information visualization. The atomic position-time series dataset represents a real material system and when the dataset is rendered, the physical relevance (3D atomic arrangement) is to be reproduced/preserved. Each snapshot representation consists of displaying the constituent atoms as spheres of varying color and size according to their species. A periodic table is integrated into the system to make a consistent representation of atomic species across different systems. Subsequent analyses generate additional data that represent information that does not have any direct physical relevance. Such information can be superimposed with the physically relevant data and rendered in a more flexible way. Our system uses color and intensity, size, thickness, shape and geometrical objects to represent various kinds of information extracted during the visualization and analysis process.

**5.2.1. Color and intensity**

For discrete data like atomic species or coordination numbers, we can use an indexed color map. But for a continuous quantity like time or distance, we design a transfer function, $t(v)$, that assigns a color $c$ according to the value $v$. For example,

$$
\epsilon \equiv t(v) = \left[ \frac{v - v_{\min}}{v_{\max} - v_{\min}}, 0, 1 \right]
$$

Here, $v_{\max}$ is the maximum value that the quantity can take. The output is in RGBA format. The R and G components vary between 0.0 and 1.0 where as the B and A components always remain fixed. Variation in the bond length, $r$, of a given bond between $r_{\min}$ and $r_{\max}$ is quantified using

$$
\lambda_a = \frac{r - r_{\min}}{r_{\max} - r_{\min}} \quad \text{for } r_{\min} \leq r \leq r_{\max}
$$

which varies from 0 (when two atoms are separated by the minimum distance) to 1 (when the atoms are separated by the maximum distance). Now we can define a transfer function like one defined above for $v = \lambda_a$ and also use an index color map by finding the index number as $i = \lfloor \lambda_a \sqrt{n} \rfloor$.

The intensity of the color depends on the viewpoint so it may not represent the information uniquely. We have used the intensity to encode the stability of the coordination polyhedron for which the color represents the coordination number. This is effective in conveying relative stabilities among different polyhedra within a snapshot or between two snapshots (Fig. 11).

**5.2.2. Size and thickness**

The size of a sphere in the ball-and-stick representation of coordination encodes the stability of the coordination state (which is represented by the sphere’s color) of the central atom. The size is also used to encode the extent of atomic displacement. The thickness of a bond is used to visualize the stability of an atomic bond. A minimum size or thickness is always assigned to make an atom or a bond visible.

**5.2.3. Geometric objects**

Points, lines, spheres, cylinders, polyhedra and ellipsoids are used. Atoms are usually displayed as spheres but those atoms, which are not the focus in the current snapshot, are drawn as points for the context. The atom trajectories are obtained either by
drawing points at the atomic positions as a function of time or by
drawing line segments between two successive atomic positions as a
function of time. Bonding and coordination relationships between
atoms are displayed as lines or cylinders. Polyhedra are used to
represent the coordination environment and structural unit. Finally,
spheres and ellipsoids are used to represent the atomic movements.

5.3. Information presentation

There is simply too much information to present on the display
screen of a finite size. The data points, in our case the atoms, have
many attributes; some of the attributes are specified in the dataset
and several others are computed on the fly. Different types of
information often need to be presented together. Our information
presentation scheme exploits the essence of overview, details-on-
demand, focus + context, and zoom-and-filter techniques. Anima-
tion, pathlines and RDF provide a complete overview of the data.
Initially we do not know which temporal region can be of interest
and animation gives some idea in this context. The displacement
spheres and ellipsoids also provide aggregate information about
the system dynamics. Once a temporal interval is selected, one
particular atom or a set of atoms can be selected for further
exploration. Spatial region around an atom or a group of atoms can
be zoomed and explored further using the selective display feature.
The RDF provides a complete spatial map of the data. The
interpretation of the specific features of RDF requires a detailed
structural analysis. The coordination environment and coordina-
tion distribution, different types of cluster structures, ring
structures, bond-lifetimes, structural stability can be explored on
doing for different cutoff distances and cutoff windows. The
computed quantitative information can also be saved to files for
future analysis by other external tools.

5.4. Programming language and libraries

The C/C++ programming language is used for the system
development. The standard template library in C++ provides
performance guarantee of the data structures and algorithms in the
library. OpenGL (www.opengl.org) is used for graphics rendering
supplemented by GLUT and GLUI libraries. OpenGL provides the
algorithms for drawing geometric primitives such as points, lines
and triangles and defining lighting environment. GLUT library
abstracts away the underlying graphics context initialization and
management, and presents a cleaner interface useful for applica-
tion development. GLUI is a user interface library built upon GLUT
and OpenGL. We also take advantage of hardware accelerated
graphics rendering; a shader program written in GLSL (OpenGL
shading language) is executed directly in GPU and is also faster
compared to a normal program written in high-level languages
executing in CPU. The shader improves the rendering performance
and enables highly flexible rendering algorithm to be implemen-
ted. Programs written using GLSL are compiled during the runtime
and OpenGL drivers come with built-in compiler for GLSL. Qhull
[13] algorithm is used to compute coordinate polyhedra. It uses
the “beneath and beyond” algorithm to compute coordinate
polyhedra. MATLAB$^{\text{TM}}$ C/C++ libraries are used to compute
eigenvalues and eigenvectors of the covariance matrices required
during the principal component analysis for diffusion ellipsoid.

5.5. Performance

Our visualization system is expected to enable user to explore
dataset quickly by providing higher frame rates. We took various
optimization measures to ensure that the visualization system is
interactive and responsive. A selected subset of the complete
positional dataset is stored in the main memory and a separate
coordinator module coordinates the access to the dataset. Data
structures are constructed dynamically from the positional dataset
for different analytical and visualization modules. These modules
cache their previously computed results until some parameter is
changed avoiding the unnecessary computation. Some analytical
algorithms take too long for an interactive visualization. Such
intensive algorithms are implemented in separate threads so that
they do not deteriorate the interactivity of the system and also do
not corrupt the data. Benefit of increased responsiveness out-
weighs the extra overhead associated with multithreads. Outputs
from these algorithms are made available for exploration as soon
as new data is computed. As discussed in previous sections, we
have adopted various optimization schemes to accelerate RDF,
coordination and ring computation. It is shown that the interactive
frame rates are achievable for systems consisting of up to a
thousand atoms and a couple of hundred thousand time steps in a
normal desktop environment. Such system sizes are typical in the
case of today’s common first-principles molecular dynamics
simulations of a wide range of materials problems including
liquids, which are used to justify the effectiveness and usefulness
of our atomistic visualization scheme.

6. Conclusions

In this paper, we have proposed an efficient scheme to gain
insight into the position-time series data generated by molecular
dynamics simulations by going beyond direct rendering of the data.
Our scheme integrates various analysis and rendering tasks together
in order to support interactive visualization of the data at space-
time multiresolution. It starts by processing the original positional
data to extract additional data representing information about
several relevant structural (radial distribution functions, coordina-
tion environments, clusters and rings) and dynamical (diffusion
coefficients, atomic displacements and covariance matrices) pro-
erties. These data are generated on the fly and rendered interactively at
varying levels of details. It is the integration of analytical and
rendering methods that makes our study different from the most of
the existing visualization systems that are applicable for atomic and
molecular systems. While a number of analysis methods have been
supported in this study, additional methods can be incorporated as
needed. We have achieved the interactive frame rates for systems
consisting of up to a thousand atoms and a couple of hundred
thousand time steps in a normal desktop environment. Making our
visualization system scalable with system size is important since
ever larger supercells and longer runs are expected to be amenable
to the first-principles MD method in coming days. Although only the
FPMD simulation data of moderate size were visualized in this work,
one can also visualize the results from pair potentials-based MD
simulations of relatively much larger size. Another reason is that a
better understanding of the physical system or problem in the
question often requires a simultaneous processing of multiple data
sets in the same visualization. One feasible approach to addressing
the scalability issue is to implementation in multi-processor
(parallel) computing and multi-display environments. Besides,
faster algorithms for compute-intensive tasks such as ring-analysis
algorithm designed for large graphs [31] need to be developed/implemented. Due to the distributed nature of relevant data sources
and potential users, it is also desirable to develop support for remote
(web-based) atomistic visualization. Finally, the proposed scheme
only deals with the visualization of geometric (atomic) structure and
dynamics of the simulated systems. The FPMD simulations produce
a variety of data related to electronic structures in addition to
position-time series. They include electron (charge) distribution,
electronic wave functions, density of states, bond overlap popula-
tions, etc. Visualization of these data needs to be integrated with the
visualization of atomic configurations as was done in a few previous studies, e.g., [3], for instance, superimposing electron density isosurfaces on atomic configurations.

Acknowledgement

This work is supported by National Science Foundation (EAR-0809489 and EAR-0347204).

References