LoD PLI: Level of Detail for Visualizing Time-Dependent, Protein-Lipid Interaction

Naif Alharbi[1], Michael Krone[2], Matthieu Chavent[3] and Robert S. Laramee[1]
[1]*Swansea University, UK*
[2]*University of Tbingen, Germany*
[3]*IPBS, Toulouse, France*

Keywords: Scientific visualization, Molecular Dynamics, Protein-lipid Interaction

Abstract: In Molecular Dynamics (MD) Visualization, representative surfaces of varying resolution are commonly used to depict protein molecules while a variety of geometric shapes, ribbons, and spheres are used to represent residues and atoms at different levels of detail (LoD). The focus of the visualization is usually on individual atoms or molecules themselves and less often on the interaction space between them. Here we focus on LoD interaction between lipids and proteins and the space in which this occurs in the context of a membrane simulation. With naive approaches, particles may overlap and significant interaction details can be obscured due to clutter. Furthermore, the spatial complexity of the protein-lipid interaction (PLI) increases over time. Co-developed with an MD domain expert, we address the challenge of visualizing complex, time-dependent interactions between lipids and proteins by introducing two abstract LoD representations with six levels of detail for lipid molecules and protein interaction. We also propose a fast GPU-based projection that maps lipid-constituents involved in the interaction onto the abstract LoD protein interaction space. The tool provides fast LoD, the imagery of PLI for 336,260 particles over almost 2,000 time-steps. The result is a great simplification in both perception and cognition of this complex interaction that reveals new patterns and insight for computational biologists. We also report feedback from the domain expert to our visualization.

1 INTRODUCTION AND MOTIVATION

Visualization of Molecular Dynamics (MD) has received a great amount of attention over the past two decades. Numerous MD visualization tools are used for visualizing MD data from simple to complex molecules [Kozlíková et al. (2017) and Alharbi et al. (2017)]. With recent advances in computer graphics, most of the MD visualization research focuses on optimal rendering quality and performance.

Proteins perform an array of functions such as drug and neurotransmitter transport. Lipids intimately influence the structure and function of membrane proteins by extensive protein-lipid interaction (PLI) Antonny (2011). Here the PLI plays a significant role in understanding the interdependence of membrane proteins with their surrounding lipids. Simulations facilitate structural understanding in the context of a lipid bi-layer. This is the application we focus on here.

In an MD visualization, a number of models are used to depict molecules at the molecular and atomistic scale. Protein molecules, for example, are often represented by surfaces. This representation provides an overview of the protein shape at the molecular scale. At the atomistic scale, small molecules including atoms' bounds are often represented by models such as ball-and-stick. These representations are commonly used to visualize the MD data at the desired scale. However, depending on the intended analysis task, these same representations might also become part of the challenge due to their complexity.

The visualization of PLIs in time-dependent systems involves a number of challenges. Complex protein surfaces might not be an ideal solution in this case. The dynamics of the system may cause many particles to be occluded or to overlap. The PLI occurs at the atomistic level which presents computation and visualization challenges. Every single interaction holds information that might explain the lipid's influence on a protein's functions. At the very beginning of the simulation, a lipid might interact with one protein at most. Over time, the complexity of the system increases due to the evolution of the system particles. Figure 1 illustrates the motivation behind this work where lipid particles interact with more than one protein. Special techniques we can use to enable users to investigate the PLI at different levels of abstrac-

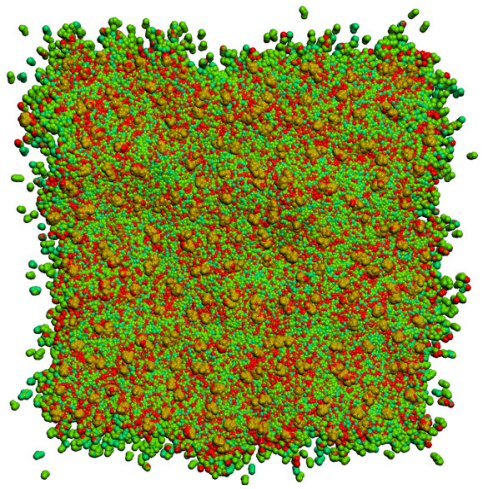

Figure 1: A snapshot of a naive Protein-Lipid interaction representation. Protein particles are yellow, and lipid particles are green initially and turn red after an interaction. Perceptual difficulties stem from complexity and occlusion.

tion. We extend Alharbi et al. (2018) by design and implement a LoD PLI in close cooperation with domain experts to help computational biology scientists to obtain insight into PLI for time-dependent systems. Our contributions are:

- Six levels of detail for the protein-lipid interaction and an enhanced implementation of the PLI space utilizing a tree-structure.
- A smooth transition between the six LoDs utilizing lipid-scale and particle-scale effects. The lipid-scale effect provides a smooth transition for particle positions between levels. The particle-scale effect provides a smooth change of the radius of the particles between levels.
- While the previous work harnesses the GPU to accelerate the computation of PLI here we propose a fast GPU-based approach to accelerate representing the PLI LoD in the abstract interaction space.
- A number of PLI filtering and selecting techniques as well as a custom color-map feature to enhance the visualization.

The paper is organized as follows: In Section 2, we discuss related work. In Section 3, we describe the simulation data. Section 4 discusses the visual-design aspects. In Section 5 we present the results of our method and discuss the reaction of a domain expert in computational biology to our novel LoD PLI representation. Conclusions and directions for future work are given in Section 6. The implementation is described in the supplementary file.

2 RELATED WORK

There has been a fair amount of recent as well as useful review papers on the theme of molecular visualization. Readers can be referred to O'Donoghue et al. (2010). The report covers web-based and stand-alone tools and discusses the advantages and disadvantages of the most common molecular structure acquisition techniques for biological data. Kozlíková et al. (2017) in another of their state-of-the-art review, classify recent advances in visualization undertaken within the area of molecular simulation particularly focussed on structural biology applications. Alharbi et al. (2017) present the first survey of surveys on molecular dynamics visualization involving survey papers from the computational biology community as well as computer graphics. The chosen surveys (in total 11 surveys) discuss a diverse range of research topics in molecular visualization spanning the long history of advances in molecular dynamics visualization (MDV) spanning 1980 to 2017.

In the molecular visualization literature, metaphor and levels of detail (LoD) are often used to enhance software performance or reduce visual complexity, and enhance perception. Table 1 classifies related work into LoD for performance and perception and their complexity.

LoD for Performance And Computational Complexity Preceding literature often cites spatial data structures which rely on 1) the assembly of atoms within a molecule, or 2) modifying the surface resolution of a molecule concerning the LoD. The aim is to allow a user to interactively visualize larger molecular datasets by minimizing the computational complexity needed for data representation.

Sharma et al. (2004) implement a hierarchical frustum-culling algorithm on the basis of an octree data structure. This algorithm is efficient for removing atoms which are outside the field-of-view window. This is through the application of probabilistic and depth-based occlusion culling as well as the utilization of a multi-resolution algorithm for rendering the chosen subset of visible atoms at different levels of detail.

Bajaj et al. (2004) introduce a biochemically sensitive level-of-detail hierarchy for molecular representations. The hierarchy is constructed from the base upwards in such a manner that each member of this hierarchy includes all members which are related to the level immediately preceding it. For example, an atom is the lowest level of the hierarchy. A residue contains all constituent atoms, while a secondary structure contains all constituent residues. Each level of the hierarchy is associated with a geometric representation. A single sphere represents an atom with

Table 1: A classification of the related work with respect to: performance and perception and their complexity. The fourth column presents the focus elements. (S= Surfaces, P= Protein, M= Molecules, R= Residues, A= Atoms, Ast= Astrocytes, N= Neurites).

Paper	Performance and Computational Complexity	Perception and Perceptual Complexity	Focus Objects
Bajaj et al. (2004)	✓	✗	S
Sharma et al. (2004)	✓	✗	A
Lee et al. (2006)	✓	✗	S
Lampe et al. (2007)	✓	✗	P
Weber (2009)	✗	✓	M
Van Der Zwan et al. (2011)	✗	✓	M, A
Lindow et al. (2012)	✓	✗	M, A
Krone et al. (2012)	✓	✗	S
Falk et al. (2013)	✓	✗	M, A
Parulek et al. (2014)	✗	✓	M, A, S
Le Muzic et al. (2014)	✓	✗	M, A
Mohammed et al. (2018)	✗	✓	N, Ast
Alharbi et al. (2018)	✓	✗	M, A

a Van der Waal radius, while a residue is depicted by a minimal single bounding sphere which includes all its constituent atoms. This approach considerably reduces visual clutter by presenting the suitable volume occupancy and shape. Additionally, the hierarchical image-based rendering enables the mapping of derived physical characteristics onto the molecular surfaces.

Lee et al. (2006) propose an algorithm for real-time surface visualization on the basis of a view-dependent LoD method. This technique comprises two stages: a pre-processing stage and a real-time rendering stage. This technique offers high-quality rendering and also displays critical components of molecular models. In the preprocessing stage, the mesh of the molecular surface is simplified and classified by different levels of detail. In the real-time rendering stage, hierarchical LoD models are constructed which result in accelerating the rendering performance. Lampe et al. (2007) introduce a two-stage approach for visualizing large as well as time-dependent protein complexes. In the first stage, each residue is substituted with a single vertex based on the residue's rigid transformation. In the second level, the atoms which are contained in the residues are dynamically generated within the geometry shader on the basis of the initial simulation vector and the transformation matrix.

Lindow et al. (2012) introduce a 3D voxel grid based on the GPU, for storing the atomic data and utilizing a fast ray-voxel traversal through which just the spheres within the existing voxel are tested for intersection. This algorithm enables for the interactive revelation of biological structures comprising of billions of atoms. Comparable to results obtained by Lindow et al. (2012), Falk et al. (2013) store every molecule in its own supportive grid, which is further traversed in detail via a ray. Here the grid is constructed on the basis of each atom's size while in Lindow et al. (2012) the grid is constructed based on the number of atoms. Krone et al. (2012) propose a novel fast approach for molecular surface extraction that can be used for depicting structural details on a constant scale. By modifying the resolution of the grid as well as the density kernel function the scale may vary from atomic detail to lower resolution detail in visual representations.

Le Muzic et al. (2014) present a LoD approach which envelopes adjacent atoms within a single sphere dependent on the distance from the camera. They utilize texture for storing the atomic positions of the whole molecule. Tessellation and geometry shaders are used for constructing the atom's position based on a rotational quaternion which represents the molecule's present orientation. Their approach is similar to Lampe et al. (2007) but uses the tessellation shader instead of the geometry shader. Alharbi et al. (2018) introduce a novel abstract protein space and provide a solution to address PLI computations and visualizations challenges. They focus on the space between lipids and proteins and visualize the PLI at the atomistic level. The PLI details are maintained via a uniform cylindrical grid. The paper mainly focuses on the abstract protein space as a proposed solution for PLIs in complex MD systems.

LoD for Perception and Perceptual Complexity In the context of molecular visualization, abstraction

typically refers to structural abstraction. The structure of a molecule is commonly depicted via various representations. e.g., a space-filling diagram, the ball and sticks model, and the ribbon model.
Generally, every representation is associated with a particular molecular structure, and several LoD techniques are applied for the provision of a smooth transition within the structures.

Van Der Zwan et al. (2011) propose a novel continuous abstraction space for illustrative molecular visualization. This approach is GPU-based for depicting continuous transitions between various stages of structural abstraction of a molecular system (i.e. space-filling, licorice, ball-and-stick, ribbon, and backbone). A seamless structural transition is applied from space-filling to ribbon. Parulek et al. (2014) propose a novel LoD approach for visualizing larger protein complexes. Its application to individual levels is based on the distance to the camera. They utilize three distinct surface representations (such as solvent-excluded surface (SES), Van der Waals spheres and Gaussian kernels). Shading abstractions, as well as hierarchical representations, are used for creating smooth transitions between the three representations. Mohammed et al. (2018) present a novel tool for visualization of astrocytes and neurons at varied levels of abstraction. This tool employs a novel abstraction space to provide a set of visualizations ranging from detailed 3D surface images of segmented structures (realistic images) to simplified abstractions (like skeletons and graphs). Previous work, in general, does not focus on the interaction space between lipids and proteins.

In this work we extend Alharbi et al. (2018) by providing six dynamic, levels of detail to address visual complexity and perception. We provide a simplified representation of the PLI at different levels. Here, the protein's interaction space is also used to enhance performance during detection of protein-lipid intersections on-the-fly. The LoD is computed based on the zoom value. Each level represents the molecule by increasing the number and the size of its particles. Our approach enables the user to smoothly zoom between the levels of detail without losing the overall structure of the molecule. Also, our visualization conveys historical information about the dynamic protein-lipid interaction overall simulation time steps. It differs from the previous one in three significant ways. First, Alharbi et al. (2018) focus on the PLI space at the atomistic level while we here introduce 1) six LoD representations for an abstract protein space and 2) six LoD representations for lipid molecules. Second, Alharbi et al. (2018) utilize a uniform 2D grid to maintain the PLI detail. We employ a custom quad-tree structure to exploit the properties of a tree structure in the PLI LoD representations. We also implement a fast GPU-based method to accelerate representing the PLI on the PLI space. Thirdly, this work introduces a number of additional features which enable the user to 1) filter the frequency of the PLI, 2) filter the PLI based on the participating molecule type, 3) map the PLI frequency to different color maps utilizing a custom color map method that provide an enhanced color mapping with respect to the distribution of the PLI along the color scale.

3 SIMULATION DATA DESCRIPTION

Data-sets characterize biological dynamics of lipids and proteins within the perspective of a membrane model. The simulation domain's dimensions are 116 $\times$ 116 $\times$ 10 nanometers (x, y, and z respectively). Individual trajectories reflect the evolution of 336,260 particles over 1,980 nanoseconds (ns). The system comprises three molecule types: one protein, and two lipid types (POPE and POPG). The protein type simulates 256 protein molecules while the collective lipid POPE and the POPG types comprise of 14,354 and 4,738 molecules respectively. The collection of the structures is described below:

1- Each protein has 171 residues. Each residue consists of 1 to 3 particles. A single protein consists of 344 particles which result in (256 $\times$ 344) 88,064 particles in total.

2- The lipid collection consists of 19,092 lipids. Each lipid contains 3 groups: i) a head group (2 particles), ii) a tail group (6 particles), and iii) a second tail group (5 particles). Each lipid molecule has 13 particles which result in (19,092 $\times$ 13) 248,196 lipid particles in total.
The models discussed are on the basis of the MARTINI coarse-grained forcefield Marrink and Tieleman

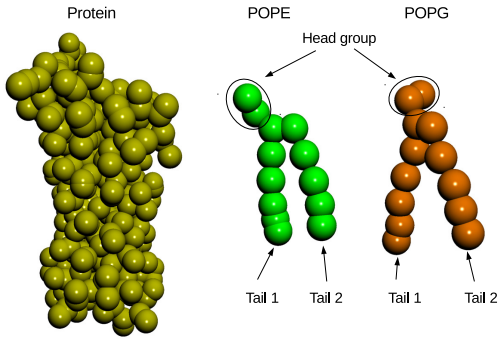

Figure 2: Representation of molecules: Protein, POPE and POPG types (left to right). The Protein and the Lipid type images are generated with our tool.

(2013) which does not signify all the atoms, however, simplifies four heavy atoms through one coarse-grained particle. Figure 2 provides a depiction of the structure of a protein, lipid POPE type, and POPG type. Regarding the size of the simulation, the dataset contains more than 666 million space-time positions that occupy 8 Gigabytes of memory.

4 VISUALIZATION OF PROTEIN-LIPID INTERACTION

This section discusses the main aspects of the visualization of protein-lipid interaction. It describes the LoD representation of lipids and discusses the relation between lipid LoD representations and the PLI detail at different levels. It also discusses the protein interaction space and describes the LoD representations of the protein-lipid interaction space. The last two subsections discuss the construction of our IBQT (index-based quad-tree) and the projection of the PLI onto the protein LoD space.

4.1 Lipid LoD Representation

The hierarchy for lipid is simplified based on the MARTINI coarse-grained forcefield Marrink and Tieleman (2013). Generally, lipids are represented either by an abstraction such as Van der Waals spheres, CPC, licorice Humphrey et al. (1996) and hyperballs Chavent et al. (2011) or by atomistic representations where every individual particle is depicted by a single sphere (sticks might be used to represent bonds between atoms). We choose the atomistic representation over the abstract representations as the latter lends itself to a smooth LoD representation, especially with dynamic systems. We depict the lipid particles as spheres and cylinders representing bonds between particles. The rendering is accelerated by GPU-based ray-casting techniques for both spheres and cylinders.

Lipid Base-Structure: Our lipid LoD representation starts with the lipid base-structure. As described in section 3 lipids have a well-defined structure: a head group and two tails. The three groups are connected via a linker particle in such a way that their final representation reflects the Lipid model in Figure 2. We employ the geometry of the base-structure to build five derivative structures such that each extends the previous one and represents the lipid at the given LoD.

Derived Structure: The first derived structure consists of only one relatively large particle, i.e., the linker particle which is the representative of the lipid at LoD 1. The second derived structure involves the linker and its closest neighbors which represent the lipid at LoD 2. New structures for LoD 3, 4 and 5 are derived by smoothly extending neighbors of existing particles until a lipid reaches the most detailed representation and smallest particle size at LoD 6. See Figure 3.

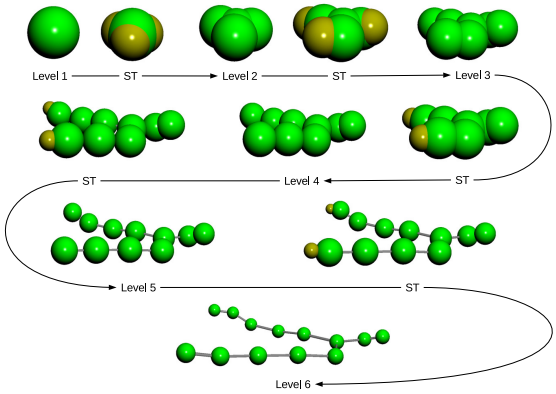

Figure 3: Six levels of detail for Lipids. A smooth transition (ST) is applied between the six levels. The size of the sphere shrinks between level as well. The yellow spheres represent the transition stage. The LoD is calculated based on the zoom value.

Smooth LoD Transition: By convention we use the term ”terminal particle” to describe relatively small particles that occupy leaf nodes of the given structure. We utilize a smooth dynamic transition between levels to avoid flickering caused by sudden transitions. The smooth transition is achieved by applying two continuous effects, a lipid-scale, and a particle-scale effect. The lipid-scale effect is responsible for providing a smooth, forward/backward translation for terminal particles between levels. The particle-scale effect is responsible for providing a smooth increasing/decreasing of each terminal particles' radius during the transition. The speed of the transition is a user option, and it can be configured by increasing or decreasing the transition time. The longer the time, the smoother the transition. Figure 3 shows the transition between levels.

4.2 LoD Lipid Interaction

One of the properties of the PLI is that it occurs at the atomistic level. This property poses a LoD computational and visualization challenge. The property implies that for every lipid molecule each of its particles needs to be included in the PLI proximity test. For example, if the interaction test is performed at LoD 1 then, for every lipid, only one particle is involved

in the interaction test while some of its other particles might pass the interaction test if they were involved. On the other hand, not all the particles of the given lipid necessarily pass the interaction test at the maximum LoD. For example, if the interaction is caused by a particle p that exists only at LoD n we will not be able to reveal its interaction detail at LoD $n-1$ as particle p is not represented at this level downward. These challenges might result in unexpected approximations at all the LoDs except LoD 6 as all the lipid particles are included in the structure. Hence the PLI test is performed on the GPU for LoD 6 as the lipid structure in this level involves all the lipid particles. The result is maintained in full detail. For every LoD, we inherit the interaction details from the upper level in such way the visualization reveals the interaction detail for the present LoD and higher approximations. It is clear from Figure 4 that adding lipid LoD helps reduce occlusion and clutter. However further innovation is required for protein LoD.

4.3 Protein Interaction Space

Cylindrical PLI Representation: The description of the cylindrical PLI representation is given by Alharbi et al. (2018). See Section 4.

4.4 LoD Protein-Lipid Interaction space

PLIs can occur at any position within a protein space. By utilizing an abstract cylinder to represent a protein space, a PLI can be directly depicted on the surface of the abstract protein space using a tile. Essentially, the surface of the abstract protein space is divided into 1024 tiles. Each tile is associated with an IBQT (index-based quad-tree) node (see Section 4.5). The IBQT nodes are responsible for maintaining the PLI detail while tiles are used to depict the nodes' information. We exploit the properties of IBQT to apply LoD techniques to a PLI space. The root of the IBQT represents the first LoD. The PLI frequency is accumulated from the nested levels before it is projected on the cylinder. The LoD level 1 helps the user obtain an overview of the PLI. The leaf nodes of level 6 represent the sixth LoD. Each node is mapped to a tile on the cylinder. LoD 6 provides the user with the PLI at the highest resolution where each tile depicts the smallest interaction area on the abstract surface. The user can investigate the PLI at different resolutions in between LoD 1 and LoD 6. The LoD is calculated based on the camera zoom level. The zoom interval between levels is parametrized enabling the user to control the switching time between levels. The user also is provided with a manual LoD slider. The manual LoD slider helps the user to focus on the PLI at particular LoD while zooming in and out or at a fixed camera position. Figure 5 illustrates six LoDs for a single PLI.

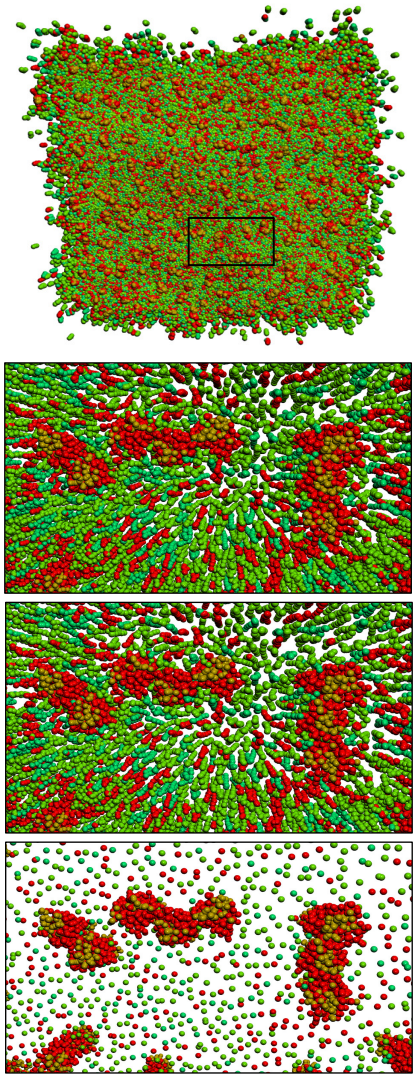

Figure 4: An overview of protein-lipid interaction. The top image shows the default representation of the lipid molecules (no level of detail). The middle and bottom images show the representation of the same lipid after enabling the level of detail (level 4 and 1 respectively). Protein particles are yellow, and lipid particles are green initially and turn red after an interaction. In the first image, more interaction details are occluded, due to overlapping particles.

4.5 Constructing the Interactive Index-Based Quad-Tree (IBQT)

Taking into account 256 simulated proteins, we maintain a dynamic LoD interaction representation i.e each interaction position, time and frequency over 1981 time-steps. In terms of storing and retrieving the in-

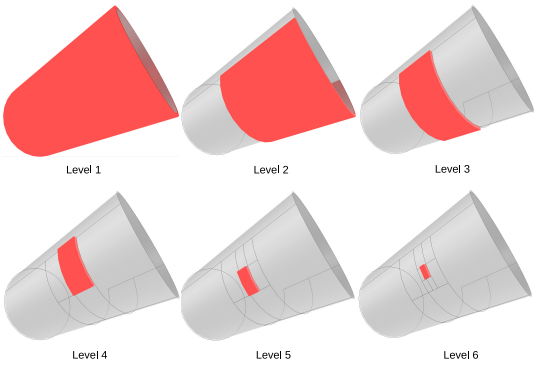

Figure 5: Six levels of detail to represent protein interaction. The color of each tile is mapped to the number of interactions with each tile's extent. The interaction detail is maintained by passing the number of particle interactions down to the next level. The LoD is dynamically updated based on the camera zoom value.

teraction data in a spatial data structure, there are a variety of solutions discussed by Samet (2006). However, the key challenge here is posed by the dynamics of the simulation. The interaction position of every individual particle needs to be identified, translated, and rotated, per time step with the given protein. Another challenge is that the PLI occurs in a three-dimensional space represented by an abstract cylinder. A naive solution might require conversion between a cylindrical coordinate system to a 3D Cartesian system. The conversion is necessary to map the PLI from/to the cylindrical space to a 3D Cartesian space which can be represented by a quad-tree. We address these challenges by proposing a fast quadtree-based mapping approach using the given protein's local coordinate space. Our approach eliminates the need for storing the global interaction position of each individual particle or updating its translation or rotation. Also, it eliminates any explicit conversion between cylindrical and Cartesian coordinate systems. See Section 4.6.

We utilize a custom index-based quad-tree (IBQT) to capture and store the protein-lipid interactions. The IBQT is designed to accommodate a GPU-based projection onto each protein's local cylinder. Because the PLI is represented with an abstract cylindrical shape, the local polar system of each protein can be exploited in the IBQT design. The IBQT is inspired by the Samet's sector tree Samet (2006). Samet (2006) discusses a sector tree structure that utilizes a polar coordinate system to maintain an object's spatial data in two dimensions. In the sector tree, the angle of the intersection between the boundary of the two regions participates in its construction. The common facet between the sector tree and the IBQT is that in both angles play a key role in defining a tree structure. Our IBQT differs from the sector tree in two aspects. First, our structure is a point-based quad-tree structure while the sector tree is region-based. Second, our IBQT encodes a three-dimensional position to represent a PLI in 3D space. Each IBQT node has a key index (called a base index). The base index of a node is used to identify the node with respect to a given LoD. The node index consists of two components; the first component encodes the interaction position along the z axis while the second component encodes the angle of the interaction. Every interaction node stores the interaction frequency, a list of particles involved in the interaction, and the interaction time-step. Figure 6 depicts our IBQT and illustrates the PLI encoding. The index components are derived from the interaction position by applying the following:

$$u1 = \frac{h}{2^{LoD-1}} \tag{1}$$

Where h indicates the height of the cylinder used to represent a protein's interaction space.

$$u2 = \frac{2\pi}{2^{LoD-1}} \tag{2}$$

With respect to a given LoD, Equation 1 calculates the size of a unit scale that span the z axis while 2 calculates the size of unit scale spanning 2π.

$$i_z = floor(\frac{z}{u1}) \tag{3}$$

Equation 3 calculates i_z which represents the PLI along the z axis with respect to a given LoD. The following calculates the angular value i_{xy} of the PLI with respect to a given LoD:

$$i_{xy} = floor(\frac{DoublePI(arctangent(y,x))}{u2}) \tag{4}$$

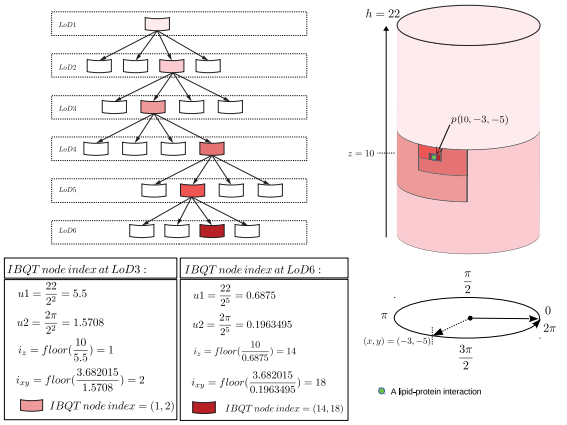

Figure 6: This image illustrates the computation of the IBQT node's index. An IBQT node index is encoded on the GPU by applying Equations 1, 2, 3, 4 to an interaction position.

The function *DoublePI()* returns a positive *radian* $\in [0, 2\pi]$, *arctangent* returns an angle confined to the interval $[-\pi, \pi]$ and positive for $y > 0$. Equations 3 and 4 encode the PLI position into an IBQT node index. The index consists of two components. The first and second components are represented by i_z and i_{xy} respectively. Finally, we construct the IBQT by utilizing the derived node index instead of the global interaction coordinates. Figure 6 illustrates process of encoding the PLI position into IBQT node index. The following equation is used to find the index of the parent node index (PNIndex) of any given node:

$$d_f = \frac{2^{LoD_{max}-1}}{2^{LoD-1}} \tag{5}$$

LoD_{max} indicates the maximum level of detail. The d_f is used to calculate the parent node's index based on the first and second components of the given node index. It ensures that the node index $\in [0, 31]$.

$$PNIndex = (floor(\frac{i_z}{d_f}) \times d_f, floor(\frac{i_{xy}}{d_f}) \times d_f) \tag{6}$$

For example, let us take a node n from level 6 and let us say its index is $(17, 23)$. We can compute the index of the parent of node n at level 5 as follows:
The node index at level 6 is given by i_z =17 and i_xy =23. First, we compute $d_f = \frac{2^5}{2^4} = 2$ via Equation 5. Then Equation 6 can be used to compute the index of the parent node:
$PNIndex = (floor(\frac{17}{2}) \times 2, floor(\frac{23}{2}) \times 2) = (16, 22)$

4.6 Projecting the Interaction on to the LoD Protein Space

The interaction mapping process is performed on the GPU utilizing a geometry shader. The advantage of this approach is that it is no longer necessary to store, update, translate or rotate the interaction positions for every time-step because they are based on the given protein's local coordinate system. First we retrieve the interaction data from the IBQT using three parameters: the time step, the LoD and a vector of a *TileData* object. The time step and the LoD parameters are used to specify the IBQT query. The *TileData* vector returns a list of *TileData* objects. Each *TileData* contains an interaction node index, a molecule type identifier and interaction frequency. Then, based on the chosen level, every non-empty node is projected and visualized by a tile on the surface of the corresponding abstract protein space. The projection involves three steps. The first step is to project the left edge of the interaction node on the surface. The second step to identify the right edge of the interaction before finally forming a tile by constructing a curved,

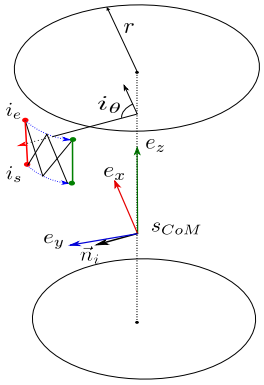

Figure 7: This image illustrates our GPU-based LoD projection. A protein interaction space and the index of a non-empty node are used to construct the associated tile on the surface of the cylinder. The red and green segments represent the left and right edges of the interaction node respectively. The blue dotted arrows represent two curves that connect the left and right edges to construct a tile.

triangular mesh between the two edges. To identify the left edge, we derive two 3D points from the interaction node index. For a given LoD, we calculate the z_{start} and the z_{end} of the left edge of the interaction node.

$$z_{start} = float(i_z) \times u1 \tag{7}$$

$$z_{end} = \frac{2^{Lod_{max}-1}}{2^{LoD}} \times u1 \tag{8}$$

The interaction angle is computed and used to construct the *interaction normal* $\vec{\boldsymbol{n}}_i$

$$i_\theta = \frac{i_{xy}}{2^{Lod_{max}-1} \times 2\pi} \tag{9}$$

The interaction angle i_θ of a node can be computed by utilizing the node index. For example, given the node (17, 23). Its interaction angle can be computed as follows: $i_\theta = \frac{i_{xy}}{2^{Lod_{max}-1} \times 2\pi} = \frac{23}{32 \times 2\pi} = 0.11439\ Rad$. The $\vec{\boldsymbol{n}}_i$ components are computed as:

$$\vec{\boldsymbol{n}}_{i(x)} = cos(i_\theta) \times \vec{\boldsymbol{x}}_x + sin(i_\theta) \times \vec{\boldsymbol{y}}_x \tag{10}$$

$$\vec{\boldsymbol{n}}_{i(y)} = cos(i_\theta) \times \vec{\boldsymbol{x}}_y + sin(i_\theta) \times \vec{\boldsymbol{y}}_y \tag{11}$$

$$\vec{\boldsymbol{n}}_{i(z)} = cos(i_\theta) \times \vec{\boldsymbol{x}}_z + sin(i_\theta) \times \vec{\boldsymbol{y}}_z \tag{12}$$

Where $\vec{\boldsymbol{x}}$ and $\vec{\boldsymbol{y}}$ are the *eigenvectors* in the *x-direction* and *y-direction* of the protein interaction space respectively. The z_{start}, the z_{end} and $\vec{\boldsymbol{n}}_i$, then, can be used to find the left edge start and end points:

$$i_s = s_{CoM} + (z_{min} \times \vec{\boldsymbol{z}}) + (z_{start} \times \vec{\boldsymbol{z}}) + \boldsymbol{r} \times \vec{\boldsymbol{n}}_i \tag{13}$$

$$i_e = s_{CoM} + (z_{min} \times \vec{\boldsymbol{z}}) + (z_{end} \times \vec{\boldsymbol{z}}) + \boldsymbol{r} \times \vec{\boldsymbol{n}}_i \tag{14}$$

Where z_{min} is the min z value of the cylinder. The terms s_{CoM} and $\boldsymbol{r}$ are the center of mass and the radius of the abstract protein space respectively and $\vec{\boldsymbol{z}}$ is the

eigenvector in the z direction of the protein interaction space.

Similarly, we can obtain the starting and ending points of right edge by adding the *tile segment count* to the *second component* and apply Equations 13 and 14. See Figure 7.

5 EXPERIMENTAL RESULTS AND DOMAIN EXPERT FEEDBACK

The new visual design enables the user to investigate the system and obtain insight into the PLI. See the accompanying video for further visualization results at: `https://youtu.be/VTh-IOk7hWo`

Figure 9 shows three different LoD representations for the PLI. The top image depicts lipid particles and a PLI space at high LoD (LoD 6). The visualization at this resolution reveals more PLI detail. i.e., each tile represents the number of PLI that take place within its extent. The middle image shows lipid particles and PLI at LoD 4. The PLI frequencies are spatially aggregated. The user can easily observe regions that receive the highest number of interactions. The bottom image shows lipid particle and PLI space at low LoD. This visualization is useful for an overview. It can guide users to identify proteins that receive the highest frequency of interaction for further investigation.

Color Map Schemes and Adjusting the Color Map Legend: We provide the user with two-color maps. A continuous color map based on Telea (2014) and a categorised color map using Color Brewer Harrower and Brewer (2003). The PLI frequency is mapped to the selected color schema. However, the PLI histogram shows an uneven distribution which leads some colors to dominate. A naive approach to address this challenge is to clamp and rearrange the underlying data to obtain a better color distribution. However,

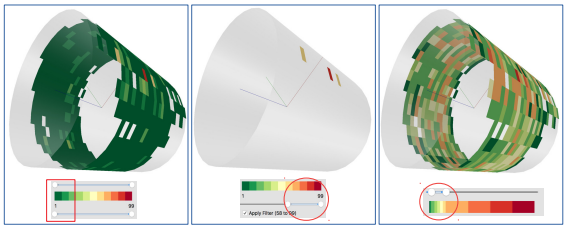

Figure 8: PLI filtering and color map steering. The left image shows no filtering and no modification to the color map algorithm. The middle image shows highest frequency PLI using the filtering slider. The right image shows the effect of steering the underlying map to enhance color distribution.

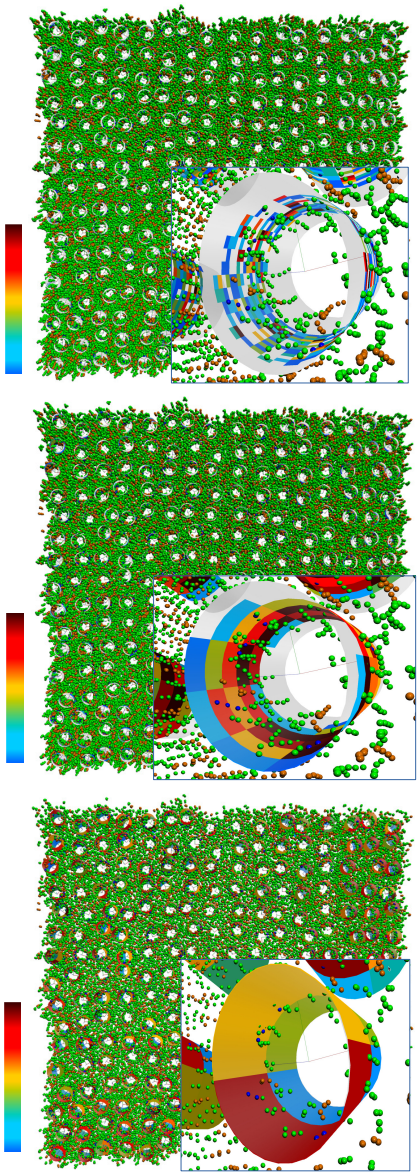

Figure 9: Three LoD PLI representations. PLI frequency is mapped to color. From top to bottom, PLI at high resolution, medium resolution, and low resolution. Lipid particles are green and brown. Blue spheres indicate lipid interaction particles.

this method might result in a mapping error for data that lay outside the lower and upper boundaries. We address this challenge by enabling the user to adjust the color map's underlying look-up table. The user can control the color map's look-up table through a range slider to obtain more color distribution. The right image in Figure 8 shows an example of using the color map slider.

PLI Frequency to Tile Height: A three-dimension

histogram can enhance the visualization. The height of the histogram encodes a property to emphasize it. We enable the user to map the PLI frequency to the tile height. Figure 10 depicts the PLI by a simple histogram by mapping the PLI frequency to tile height.

PLI Space Cluttering Reduction: When two or more protein spaces intersect each other, they cause visual cluttering. The user may reduce cluttering by either omitting the back face of a protein cylinder or rendering the cylinder lids.

Lipid Type and Protein Selection and Filtering: Three types of molecules are involved in the PLI. The PLI molecule filter enables the user to focus on one or more molecule types. Applying the PLI molecule filter on one or more molecule types showing only the PLI of the selected types. The user also can filter the PLI based on the PLI frequency. The PLI frequency filter accepts a range of values between the PLI min and max of the current histogram data.

Focus-and-Context: We apply the focus-and-context technique to reduce visual complexity. It can result in more detailed visualization, especially for an overview. See the accompanying video for a demonstration.

Domain Expert Feedback The following is feedback from a domain expert in computational biology with whom we have collaborated since 2014. "This new representation is particularly useful to better understand the fine interactions in between lipids and membrane proteins in very large membrane models. These models are constituted by millions of particles composing thousands of lipids and hundreds of proteins. These systems are very complex to understand especially if we only visualize them without filtering. Even with some filters (removing some molecules, changing the representation of the proteins), it is still almost impossible for a user to interactively understand how lipids may transiently interact with proteins. Thanks to the new visualization, the user can zoom on a protein and easily understand how (and which) lipids can create contact with this protein. This is possible thanks to the combination of LoD and the new PLI visualization. Mapping the interactions onto a cylinder also helps the user to identify areas where the lipid may preferentially interact with the protein just by visualizing the trajectory. Without this visualization, lipid-protein interaction analysis necessitates long post-processing treatments which give, in general, only a final averaged view. The filtering based on the number of interactions is helping the user to focus on the important proteins. To this expert's knowledge, this feature is unique and has never been available in any other molecular viewer program. Thus, this new depiction helps considerably the users to understand their models and gives new clues to better analyze them. This new representation is also clear enough to be used directly as a figure for publications in a scientific article."

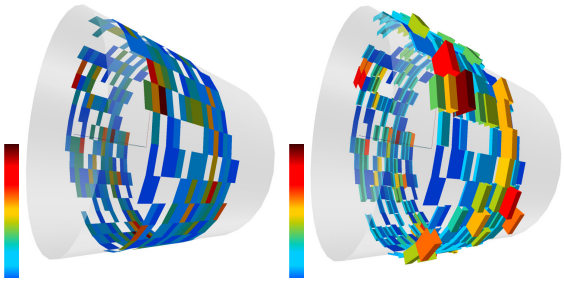

Figure 10: (Left) the number of protein-lipid interactions is mapped to color. (Right) the same data map to the height of the tiled interaction cylinder.

6 CONCLUSIONS AND FUTURE WORK

We describe an abstract protein space in the context of membrane protein-lipid interaction and we propose six levels of detail for both lipid types and the PLI abstract space. The visual design of the abstract space simplifies the PLI challenge and helps users obtain insight into the PLI. The LoD lipid representations reduce clutter caused by lipid particles. The LoD PLI space representations enable users to investigate the PLI space at the desired LoD. We also introduce a number of useful user options to aid the PLI visualization.

In future work, we aim to focus more on analysis. Discrete abstract rings can be employed to visualize protein deformation. Properties of cylinder shape can be exploited to map the PLI from a cylinder onto a 2D plane. Visual analytic techniques can be applied to 2D PLI results to understand the relation between residues and particles involved in the interaction.

ACKNOWLEDGEMENTS

This work was partially funded by the Ministry of Education of Saudi Arabia, the Saudi Cultural Bureau in London, the Department of Computer Science at Swansea University, and the DFG as part of project PROLINT. Finally, we would like to thank D. Rees and L. McNabb for proof-reading the paper.

REFERENCES

Alharbi, N., Alharbi, M., Martinez, X., Krone, M., Rose, A., Baaden, M., Laramee, R. S., and Chavent, M. (2017). Molecular visualization of computational biology data: A survey of surveys. *Eurovis short papers*, pages 133–137.

Alharbi, N., Chavent, M., Krone, M., and Laramee, R. S. (2018). VAPLI: Novel visual abstraction for protein-lipid interactions. In *21-26 October 2018, Berlin, Germany*, pages 133–137. IEEE VIS 2018.

Antonny, B. (2011). Mechanisms of Membrane Curvature Sensing. *Annual review of biochemistry*, 80(1):101–123.

Bajaj, C., Djeu, P., Siddavanahalli, V., and Thane, A. (2004). Texmol: Interactive visual exploration of large flexible multi-component molecular complexes. In *Proceedings of the Conference on Visualization'04*, pages 243–250. IEEE Computer Society.

Chavent, M., Vanel, A., Tek, A., Levy, B., Robert, S., Raffin, B., and Baaden, M. (2011). Gpu-accelerated atom and dynamic bond visualization using hyperballs: A unified algorithm for balls, sticks, and hyperboloids. *Journal of Computational Chemistry*, 32(13):2924–2935.

Falk, M., Krone, M., and Ertl, T. (2013). Atomistic visualization of mesoscopic whole-cell simulations using ray-casted instancing. *Computer Graphics Forum*, 32(8):195–206.

Harrower, M. and Brewer, C. A. (2003). Colorbrewer. org: an online tool for selecting colour schemes for maps. *The Cartographic Journal*, 40(1):27–37.

Humphrey, W., Dalke, A., and Schulten, K. (1996). VMD: visual molecular dynamics. *Journal of Molecular Graphics*, 14(1):33–38.

Kozlíková, B., Krone, M., Falk, M., Lindow, N., Baaden, M., Baum, D., Viola, I., Parulek, J., and Hege, H.-C. (2017). Visualization of biomolecular structures: State of the art revisited. *Computer Graphics Forum*, 36(8):178–204.

Krone, M., Stone, J. E., Ertl, T., and Schulten, K. (2012). Fast visualization of gaussian density surfaces for molecular dynamics and particle system trajectories. *EuroVis Short Papers*, 2012:67–71.

Lampe, O. D., Viola, I., Reuter, N., and Hauser, H. (2007). Two-level approach to efficient visualization of protein dynamics. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1616–1623.

Le Muzic, M., Parulek, J., Stavrum, A.-K., and Viola, I. (2014). Illustrative visualization of molecular reactions using omniscient intelligence and passive agents. In *Computer Graphics Forum*, volume 33, pages 141–150.

Lee, J., Park, S., and Kim, J.-I. (2006). View-dependent rendering of large-scale molecular models using level of detail. In *Hybrid Information Technology, 2006. ICHIT'06. International Conference on*, volume 1, pages 691–698. IEEE.

Lindow, N., Baum, D., and Hege, H.-C. (2012). Interactive rendering of materials and biological structures on atomic and nanoscopic scale. In *Computer Graphics Forum*, volume 31, pages 1325–1334.

Marrink, S. J. and Tieleman, D. P. (2013). Perspective on the martini model. *Chemical Society Reviews*, 42(16):6801–6822.

Mohammed, H., Al-Awami, A. K., Beyer, J., Cali, C., Magistretti, P., Pfister, H., and Hadwiger, M. (2018). Abstractocyte: A visual tool for exploring nanoscale astroglial cells. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):853–861.

O'Donoghue, S. I., Goodsell, D. S., Frangakis, A. S., Jossinet, F., Laskowski, R. A., Nilges, M., Saibil, H. R., Schafferhans, A., Wade, R. C., Westhof, E., et al. (2010). Visualization of macromolecular structures. *Nature Methods*, 7:S42–S55.

Parulek, J., Jönsson, D., Ropinski, T., Bruckner, S., Ynnerman, A., and Viola, I. (2014). Continuous levels-of-detail and visual abstraction for seamless molecular visualization. In *Computer Graphics Forum*, volume 33, pages 276–287. Wiley Online Library.

Samet, H. (2006). *Foundations of Multidimensional and Metric Data Structures*. Morgan Kaufmann.

Sharma, A., Kalia, R. K., Nakano, A., and Vashishta, P. (2004). Scalable and portable visualization of large atomistic datasets. *Computer Physics Communications*, 163(1):53–64.

Telea, A. C. (2014). *Data visualization: principles and practice*. CRC Press.

Van Der Zwan, M., Lueks, W., Bekker, H., and Isenberg, T. (2011). Illustrative molecular visualization with continuous abstraction. In *Computer Graphics Forum*, volume 30, pages 683–690.

Weber, J. R. (2009). Proteinshader: illustrative rendering of macromolecules. *BMC Structural Biology*, 9(1):19.