Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

NORBERT LINDOW, DANIEL BAUM, AND
HANS-CHRISTIAN HEGE

Ligand Excluded Surface: A New Type of Molecular Surface

Preprint version

ZIB Report 14-27 (July 2014)

Herausgegeben vom
Konrad-Zuse-Zentrum für Informationstechnik Berlin
Takustraße 7
D-14195 Berlin-Dahlem

Telefon: 030-84185-0
Telefax: 030-84185-125

e-mail: bibliothek@zib.de
URL: http://www.zib.de

ZIB-Report (Print) ISSN 1438-0064
ZIB-Report (Internet) ISSN 2192-7782

Ligand Excluded Surface:
A New Type of Molecular Surface

Norbert Lindow, Daniel Baum, and Hans-Christian Hege

Zuse Institute Berlin, Germany

August 11, 2014

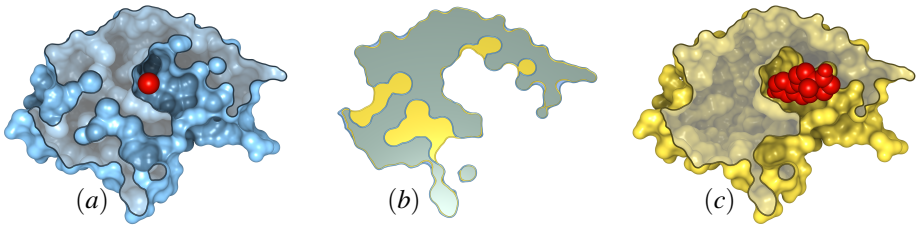

Figure 1: Comparison of a solvent excluded surface (SES) and a ligand excluded surface (LES) on the example of ketosteroid isomerase (PDB-Id: 1OGZ). To simplify the comparison, both surfaces have been cut. (a) SES of isomerase with probe radius 1.4$\mathring{A}$. The spherical probe is depicted in red. (c) LES of isomerase for the ligand equilenine, computed with 0.25$\mathring{A}$ grid spacing and 200 orientations. The ligand (red) is depicted in its active position. (b) Overlaid surface cuts, showing differences between SES and LES.

Abstract

The most popular molecular surface in molecular visualization is the *solvent excluded surface* (SES). It provides information about the accessibility of a biomolecule for a solvent molecule that is geometrically approximated by a sphere. During a period of almost four decades, the SES has served for many purposes – including visualization, analysis of molecular interactions and the study of cavities in molecular structures. However, if one is interested in the surface that is accessible to a molecule whose shape differs significantly from a sphere, a different concept is necessary. To address this problem, we generalize the definition of the SES by replacing the probe sphere with the full geometry of the ligand defined by the arrangement of its van der Waals spheres. We call the new surface *ligand excluded surface* (LES) and present an efficient, grid-based algorithm for its computation. Furthermore, we show that this algorithm can also be used to compute molecular *cavities* that could host the ligand molecule. We provide a detailed description of its implementation on CPU and GPU. Furthermore, we present a performance and convergence analysis and compare the LES for several molecules, using as ligands either water or small organic molecules.

1 Motivation

In molecular sciences, the use of molecular surfaces has a long tradition, starting with the van der Waals (vdW) surface and the solvent accessible surfaces (SAS) described by Lee and Richards in 1971 [26]. A few years later, in 1977, Richards defined the solvent excluded surface (SES), which he simply named *molecular surface* [40]. Only one year later, Greer and Bush [12] proposed the first method to depict parts of the 3D structure of the SES by computing the height field of the molecule with respect to a particular direction. In 1983, Pearl and Honegger [37] proposed a three-dimensional grid method to compute the complete SES, before in 1985 Conolly [6] presented a method to directly compute a triangular surface from the analytical surface description of the SES. Since the first three-dimensional representations of the SES, there has been a large number of publications dealing with the computation and the visualization of the SES. However, the fundamental problem that almost *no* molecule, not even water, has a spherical shape, has so far not been addressed in the computation and visualization of molecular surfaces.

To address this, we propose a new molecular surface, called *ligand excluded surface* (LES). It represents the surface of a receptor that is accessible to a specific individual ligand, which is represented by its spatial configuration of atom spheres (rather then a single 'approximating' sphere). Thus receptor and ligand are geometrically represented in the *same* manner. An example of an LES is shown in Fig. 1, which also demonstrates the difference to the SES. We present an algorithm for computing the LES based on a grid representation. We define the LES as an implicit function of a distance field. In order to approximate this distance field well, we need to take into account a possibly large number of orientations of the ligand. Since a brute-force approach that places the ligand at all positions of the grid with a large number of orientations would be computationally too expensive, we apply a two-phase approach that significantly reduces the computational effort. Moreover, we implement this method on the GPU, which further speeds up the computation.

The algorithm for computing the LES can be easily extended to also compute cavities that are large enough to host the ligand molecule (see, for example, Fig. 5). In addition to the geometry of the cavities, we also obtain information about how the ligand is positioned inside the cavities. This might be of particular interest when applying our method for subsequent docking simulations.

2 Related Work

Since our contributions concern both molecular surfaces and cavity analysis, we have split the related work with regard to these subfields.

2.1 Molecular Surfaces

Since the ligand excluded surface is most closely related to the solvent excluded surface, we will focus here on previous work dealing with SES. As mentioned before, the SES was originally defined by Richards in 1977 [40]. Since then, many approaches have been proposed to compute and visualize this surface. These approaches either compute an implicit representation of the surface or a piecewise parametric description.

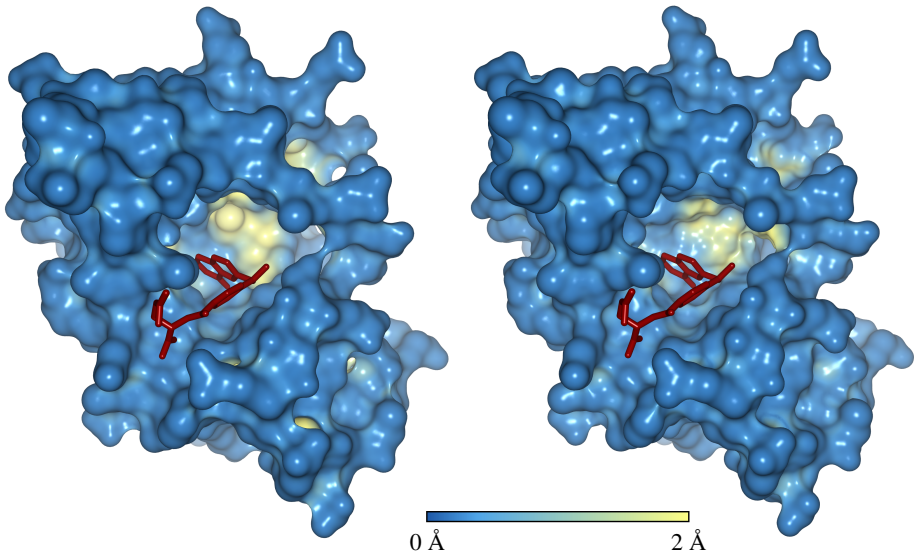

Figure 2: Difference between the SES (left) and the LES (right) for an enzyme (PDB-Id: 4DFR). The LES was computed for methotrexate (red). Each surface point is colored according to its minimal distance to the other surface.

Most implicit representations use a grid structure on which the implicit surface is defined [2, 17, 35, 37]. This is done by marking the grid points as lying either inside or outside the SES or by computing a distance function giving the minimal distance of each grid point to the SES. A triangular mesh of the SES can then be computed using the marching cubes algorithm [31] or it can be rendered directly using ray casting [13]. Instead of using a grid structure to represent the implicit surface, Parulek et al. [34] apply a ray casting approach that computes the distance function to the SES interactively for small molecules. Another way to approximate the SES is to use Gaussian models [21, 33].

The computation of a parametric description of the SES was first addressed by Connolly [6], who also presented an algorithm to triangulate the surface. More efficient algorithms were later proposed by Sanner et al. [41] (reduced surface algorithm) and Totrov and Abagyan [45] (contour-buildup algorithm). Due to the advent of commodity multi-core hardware and the advances in GPUs, it was worthwhile to consider the matter again. In 2009, Krone et al. [20] presented a ray casting approach to render the SES based on the reduced surface algorithm. One year later, we presented a CPU-parallelized version of the contour-buildup algorithm which also applied ray casting for the visualization of the SES [30]. With both approaches, it became possible to compute and render the SES of dynamic molecules with several thousands of atoms at interactive speed. A further acceleration can be achieved by a GPU implementation of the contour-buildup algorithm [23].

Since the LES, as defined in this work, cannot be computed analytically, we approximate the implicit description of the surface using a discrete signed distance function. The surface can then be rendered directly or triangulated using marching cubes.

2.2 Cavity Analysis

Methods for computing molecular cavities sometimes are classified according to the particular type of cavity (void, pocket and tunnel). We do not make this distinction and refer to all of these types as 'cavity'. Instead, we subdivide the methods according to whether the primary structures being computed are volumetric representations of the cavities or molecular paths, which can be considered as centerlines of the cavities. While it is easy to compute a volumetric representation from a path representation, the opposite is generally not true.

One of the first approaches to compute a volumetric representation of cavities was presented by Levitt and Banaszak [27]. Their approach creates a regular grid. A point on this grid is defined as cavity point if it lies outside the molecule and at least for one of the three main axes, the protein is hit in both directions. Finally, neighbored cavity points are clustered to define a cavity. A similar strategy was used by a few other approaches that consider more axes, define the protein grid points differently, or cluster cavity points in a different way [16, 19, 47]. Laskowski [24] identifies cavities by placing spheres between two non-intersecting atoms. If other atoms intersect this sphere, its radius is decreased until no atom is intersected anymore. Spheres of appropriate size describe cavities. Brady et al. [5] also place spheres from which the final cavities are computed. However, they start with an initial coating that is iteratively extended layer by layer, until no more spheres can be placed. Alpha shapes have also been used to compute molecular cavities [3, 9]. Recently, a computation of internal cavities using alpha shapes was presented that is more robust regarding perturbations of atom radii [44]. A meta-approach to cavity analysis was presented by Huang [18], who combined several approaches [5, 19, 24, 25] to improve the prediction quality. Another approach uses volume rendering to determine cavities [22]. Here, the molecule is represented as a continuous density sampled on a grid. The density can be efficiently computed and the cavity analysis is interactive, such that it can be applied to analyze molecular dynamics trajectories.

One of the earliest methods to compute molecular paths was presented by Smart et al. [43]. Starting from a user-defined point inside a cavity, their method computes a path to the outside of the molecule using simulated annealing. Petřek et al. [39] also compute paths from a cavity to the outside by utilizing shortest paths on a weighted graph that is computed from the distance field of the molecule. In a subsequent work [38], they improve their method by using the Voronoi diagram of the atom positions for the graph computation. Further works employing Voronoi diagrams of points include the ones by Yaffe et al. [48] and Medek et al. [32]. More recently [29], we developed an approach based on the Voronoi diagram of spheres, which takes correctly into account the size of the atoms. In a subsequent work [28], we extended this approach to dynamic molecular paths computed from molecular trajectories. Two of the few approaches that compute possible molecular paths based on the correct geometry of a ligand are the methods by Cortes et al. [7, 8] and Haranczyk & Sethian [14]. While Cortes et al. use rapidly exploring random trees, Haranczyk & Sethian sample the ligand orientation space to compute possible molecular paths.

The cavities we compute are volumetric representations given by the ligand structure on a discrete grid. Additionally, at each grid point the possible orientations of the ligand are known. This information can be used to quickly compute and display the cavity surfaces.

3 LES Definition

Informally, the LES of a receptor can be defined as the surface enclosing the space around the receptor that is not accessible to a specific ligand. Below we give a mathematical definition of the LES.

We start with some formalization: Let the receptor molecule r, for which we want to compute the LES, consist of n atoms with positions $p_i^r \in \mathbb{R}^3$ and atomic radii $r_i^r \in \mathbb{R}$, with $i = 1...n$. Furthermore, let the flexibility of the ligand l be given by a set of c conformations and let the ligand consist of m atoms with positions $p_{jk}^l \in \mathbb{R}^3$ and radii $r_j^l \in \mathbb{R}$, with $j = 1...m$ and $k = 1...c$. The state of the ligand is given by a conformation k, a translation T, and an orientation R, which is given by a rotation. We define the state of the ligand as valid if no ligand atom intersects any atom of the receptor molecule. This can be formulated using a binary function $I_{r,l} : (\mathbb{N}, \mathbb{R}^3, \mathbb{SO}^3) \to \{0,1\}$, which is 1 if the state is valid and 0 otherwise.

$$I_{r,l}(k,T,R) = \begin{cases} 1, & \left\| p_i^r - (Rp_{jk}^l + T) \right\| \geq r_i^r + r_j^l, \forall i = 1...n, \forall j = 1...m \\ 0, & \text{else.} \end{cases}$$

Now we can define the ligand excluded surface as the surface that encloses all points in $\mathbb{R}^3$ that are not reachable by a valid ligand state. This can be mathematically described by a ligand-dependent distance function $d_{r,l}$ with

$$d_{r,l}(p) = \max_{\substack{k=1...c, \\ T \in \mathbb{R}^3, R \in \mathbb{SO}^3}} \begin{cases} \max\limits_{j=1...m} r_j^l - \left\| p - (Rp_{jk}^l + T) \right\|, & I_{r,l}(k,T,R) = 1 \\ -\infty, & \text{else.} \end{cases}$$

The ligand excluded surface is then given as implicit function of all points p with $d_{r,l}(p) = 0$. Note that $d_{r,l}$ is not completely equal to the Euclidean distance function of the LES, but they are equal inside the LES and within a local distance outside the LES. Furthermore, for a 'ligand' consisting of a single atom, the LES is equal to the SES; thus the LES is a generalization of the SES.

In contrast to the SES, the ligand excluded surface cannot be computed in an analytical way. We can only approximate the surface geometrically by discretizing the space of ligand states. Since we use a finite set of c conformations, the dynamics of the ligand is already discretized. Additionally we need to discretize the orientations and positions of the ligand. To do so, we introduce two parameters, the number $o \in \mathbb{N}$ of sample orientations and the spacing $g \in \mathbb{R}$ of the cubic grid being used for uniform sampling of the positions.

4 Algorithm

First, we give an overview of the computation of the ligand excluded surface (LES) (Sect. 4.1). Then, the two phases (Sect. 4.2 and Sect. 4.3) for the construction of the discrete distance function defining the LES are described in detail. We conclude with a description about how we can exploit our approach to compute cavities (Sect. 4.4). Note that the distance field we compute only correctly approximates the distance function defined in Sect. 3 in the local vicinity of of the LES.

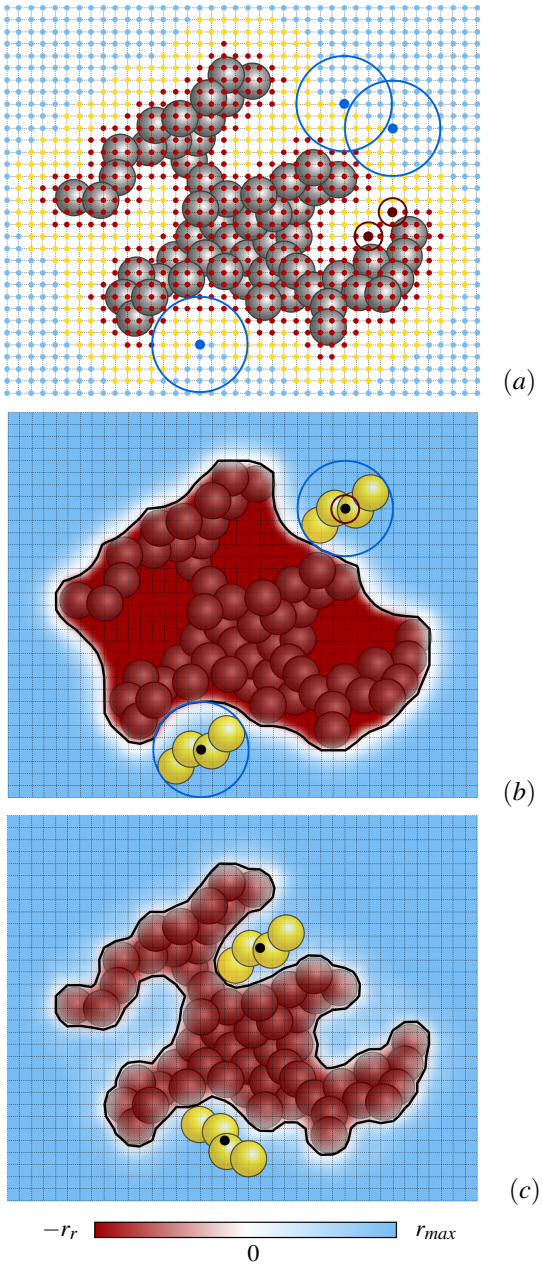

Figure 3: Illustration of our algorithm: (a) The gray spheres depict the receptor molecule and the grid points show the discretization of the space around the molecule. Red dots mark sample positions where the ligand inscribed sphere (red circles) cannot be placed without intersecting the receptor; blue dots mark sample positions where the ligand bounding sphere (blue circles) can be placed without intersecting the receptor. The red and blue dots are the result of phase I of the algorithm. The yellow dots mark the remaining sample positions, which need to be processed in phase II. (b) The distance field after completing phase I. The resulting implicit surface is equal to the SES of the ligand bounding sphere. The yellow spheres depict the ligand molecule enclosed by its ligand bounding sphere. (c) The distance field and resulting LES after completing phase II.

4.1 Overview

Instead of directly computing the LES, we compute a signed distance field that is negative inside the LES and positive outside. Thus, the LES is implicitly defined by the zero level set of this distance field. To compute the LES, we do not need the exact distance field; it needs to be exact only in the vicinity of the LES. We compute the distance field iteratively on a discrete grid. In order to approach the correct distance field values close to the LES, we apply a two-phase approach.

In the first phase, a sphere completely enclosing the ligand is used to update the distance field at positions of the grid where the ligand can rotate around its center without ever intersecting any atom of the molecule. In the following, we call this sphere *ligand bounding sphere*. For the positions at which we can place the ligand bounding sphere (blue points in Fig. 3(a)), we update the distance field for all grid points inside the bounding box of the ligand bounding sphere. The resulting distance field is depicted in Fig. 3(b) together with its implicit surface. This implicit surface is a discrete representation of the SES for a probe sphere equal to the ligand bounding sphere. Note that after computing the distance field, the largest distance value will be equal to the radius of the ligand bounding sphere. This value is set at all positions where the ligand bounding sphere could be placed. In addition to modifying the distance field, we also mark the positions where the ligand bounding sphere could be placed. These positions do not need to be considered any further. In the first phase, a second sphere is used to conservatively mark grid points at which no ligand can be placed without intersecting the molecule, including grid points inside the molecule but also some grid points close to the LES. These grid points are depicted in Fig. 3(a) as red points.

The second phase of our approach completes the computation of the distance field in the vicinity of the LES. It uses all information computed in the first phase, in particular the grid positions that are neither marked blue nor red. These are marked as yellow points in Fig. 3(a) and represent the remaining sampling positions to be considered. At these positions, it will now be tested whether the ligand can be placed without intersecting with any atom sphere of the molecule. To approach the distance field as well as possible, the ligand will be rotated at each of these positions. Furthermore, if the ligand is flexible, more than one conformation can be considered. For each rotation and each conformation for which the ligand does not intersect any atom of the molecule, the distance field of all grid points inside the bounding box of the ligand will be updated. Note, however, that the distance values will be updated only if the values increase. The result of phase II is illustrated in Fig. 3(c).

4.2 Phase I

In phase I, we sample the discrete grid with two spheres. The first one is a small sphere that is inscribed in the ligand atom spheres and is therefore referred to as *ligand inscribed sphere* (see red spheres in Fig. 3(b)). The second sphere is usually much larger than the first one and completely contains all ligand atom spheres. In the following, we refer to it as *ligand bounding sphere* (see blue spheres in Fig. 3(b)). These spheres are used to initialize the distance field and to make later computations more efficient.

4.2.1 Ligand Bounding Sphere

We define the ligand bounding sphere as the smallest sphere completely containing all ligand atom spheres. If $p_b \in \mathbb{R}^3$ is the position and $r_b \in \mathbb{R}$ is the radius of the

ligand bounding sphere, then $\left\|p_b - p_i^l\right\| + r_i^l \leq r_b$, for all $i = 1...m$, and no smaller sphere exists for which the inequality also holds. The minimal bounding sphere of a set of spheres can be tangent to more than four input spheres, but it is already defined uniquely by at most four spheres. An illustration of the ligand bounding sphere is depicted in Fig. 3(b) as blue circle. We compute the minimal bounding sphere using a simple iterative algorithm. More efficient algorithms have been proposed [10], but since the number of atoms in a ligand is generally small and the computation has to be performed only once per conformation, the simple algorithm is sufficient. We start by computing the minimal bounding sphere of four randomly selected input spheres [11]. Then, the bounding condition of this bounding sphere is checked for all remaining input spheres. If a sphere is not enclosed by the current bounding sphere, the sphere is interchanged with one of the previous four selected spheres such that the radius of the new minimal bounding sphere becomes maximal. This procedure is repeated until all input spheres are enclosed by the current bounding sphere.

We compute the ligand bounding sphere for each ligand conformation separately. Then, the overall ligand bounding radius $r_{max} \in \mathbb{R}$ is the maximum of the radii of all ligand bounding radii r_b computed for all conformations. The sphere with radius r_{max} can be used to determine positions on the grid, where all ligand conformations, no matter what orientations they have, can be placed without intersecting any receptor atom sphere. This is illustrated in Fig. 3(b).

4.2.2 Ligand Inscribed Sphere

The ligand inscribed sphere is also computed for each ligand conformation separately. To determine the ligand inscribed sphere for a single conformation, we place the sphere at the center of the ligand bounding sphere. We then determine the maximal radius such that the sphere is completely enclosed by the atoms of the ligand. The ligand inscribed sphere is illustrated as red circle in Fig. 3(b). Note that the ligand inscribed sphere has a negative radius, if the center of the ligand bounding sphere lies completely outside the ligand conformation.

From the ligand inscribed spheres of all ligand conformations we compute the radius $r_{min} \in \mathbb{R}$ as the minimum radius of the ligand inscribed spheres over all conformations. The sphere with radius r_{min} will be used to exclude grid positions from being tested with either the ligand bounding sphere or the ligand geometries. The reason for this is that if a sphere with radius r_{min} intersects the protein atoms, all ligand conformations placed at the same position, no matter what orientations they have, will also intersect the protein atoms (see Fig. 3(a)).

The two spheres with radii r_{max} and r_{min} allow us to speed up the surface computation as will be described next.

4.2.3 Distance Field Initialization

In phase I, we uniformly sample the ligand positions and initialize the discrete distance function. Note that we use the same samples for the ligand positions and the distance function. In addition, we detect all positions for which we have to investigate all ligand conformations and orientations.

To compute the discretization of the sample positions and the distance field, we first compute the minimal axis-aligned bounding box of the atom spheres of the receptor molecule. This box is then extended in all directions by r_{max}. Afterwards, the box is uniformly sampled in all directions with a user-defined grid spacing g.

The distance field is initialized with the negative value of the maximal atomic radius r_r of the receptor. Then we iterate over all sample positions T and perform at each position an intersection test of the receptor atom spheres with the minimal ligand inscribed sphere with radius r_{min}. If an atom i intersects the sphere, which means that $\|p_i^r - T\| < r_i^r + r_{min}$, all ligand conformations and orientations also intersect this atom. Hence, the position can be ignored for further investigations. In case that the minimal ligand sphere does not intersect the receptor, we perform a second intersection test with the maximal ligand bounding sphere having radius r_{max}. If this sphere does not intersect the receptor atom spheres, all ligand conformations and orientations at this position are valid, that is, they do not intersect the receptor. In this case, we set the distance value at this position to r_{max}. For all remaining positions $\tilde{T}$, where the maximal ligand sphere intersects the receptor and the minimal ligand sphere does not, we have to investigate all ligand orientations and conformations. These positions are marked by yellow points in Fig. 3(a).

Finally, we iterate over all sample positions T at which the distance field has a value equal to r_{max}. If one of the 26 neighboring grid points has a distance that is smaller than r_{max}, we update the distance function in the neighborhood of the maximal ligand bounding sphere using the distance function that is depicted in Fig. 4(a). This function has a value of r_{max} in the center, 0 at the border of the sphere, and outside the sphere, the values take on the negative distance to the sphere border. The new distance at a neighboring position p is set to the maximum of the old distance and $r_{max} - \|p - T\|$. After this phase, the implicit function defined by the current distance function represents the discrete SES with probe radius r_{max} (see Fig. 3(b)).

4.3 Phase II

In the second phase of the algorithm, the discrete distance function is refined in the vicinity of the LES. For this, we consider all the sample points (yellow points in Fig. 3(a)) that might contribute to this refinement. At all these points, all conformations of the ligand are placed one after another and the distance field is updated accordingly. Of course, considering only a single orientation per ligand conformation will lead to large errors in approximating the LES. Therefore, we use a user-defined number o of orientations for each ligand conformation. These orientations should be sampled such that the space of all orientations is well represented. In the following, we describe this sampling before we describe the final refinement step.

4.3.1 Precomputation of Ligand Orientations

To obtain the most different ligand orientations, the computation of the orientations is done for each conformation separately. Consider a single conformation with m atoms whose positions are $p_i^l \in \mathbb{R}^3$ and atomic radii are $r_i^l \in \mathbb{R}$, with $i = 1...m$. For the purpose of sampling the ligand orientations, we transform the ligand such that the center of its ligand bounding sphere lies in the origin. The transformed atom positions $\tilde{p}_i^l \in \mathbb{R}^3$ are given by $\tilde{p}_i^l = p_i^l - p_b$, for all $i = 1...m$. Due to transforming the ligand such that its bounding sphere center lies in the origin, for each rotation of the transformed ligand, all atoms are always inside the ligand bounding sphere with radius r_b. For the computation of the o orientations, we first uniformly sample points on the surface of the unit sphere. The vectors from the center of the unit sphere to the sampled points represent the axes for the rotations of the orientations. Furthermore, the angles for the rotations around each axis are uniformly sampled. We initially compute $\tilde{o} = 10 \cdot o$ orientations.

Afterwards, the differences between orientations are computed and the most significant o orientations are selected. Thus we aim at maximizing the minimal distance between the ligand orientations. This allows us to filter unnecessary orientations due to, for example, symmetry properties of the ligand. Let R_j be the initial sampled rotations with $j = 1...\tilde{o}, \tilde{o} \gg o$. We approximate the difference between the orientations by the root mean square metric (RMSM) of the atom positions. Let $D \in \mathbb{R}^{\tilde{o},\tilde{o}}$ be the symmetric matrix that stores the distances between the orientations, then

$$d_{j,k} = \sqrt{\sum_{i=1}^{m} (R_j \tilde{p}_i^l - R_k \tilde{p}_i^l)^2}.$$

with $d_{j,k}$ being the matrix element in row j and column k. The goal is to find the o most different orientations of the $\tilde{o}$ orientations. This can be mathematically formulated by finding the set of o orientations where the minimal distance becomes maximal. We approximate the solution to this optimization problem using k-means clustering, which is much faster than computing the optimal solution. The k-means clustering algorithm starts by randomly selecting o orientations as cluster centers. Then, each orientation R_i is assigned to the cluster center R_j with the minimal distance $d_{i,j}$. This creates o clusters of orientations. In each cluster, the orientation whose maximal distance within the cluster becomes minimal is computed. If this orientation is different to the previous selected center, the orientations are interchanged. Then the assignments and the new cluster centers are recomputed. This is repeated until no cluster center changes anymore. To achieve an even better solution, the algorithm can be run several times. The best solution is then the one with the maximal minimal distance.

4.3.2 Distance Field Refinement

Now that we have computed a discretization of the grid as well as the orientations of the ligand conformations, we can describe the final part of the algorithm. During this last step, we need to perform intersection tests between the ligand and the receptor molecule for all orientations R and all conformations k at the precomputed positions $\tilde{T}$. If the ligand does not intersect the protein for a certain position, orientation, and conformation, the distance function is updated in its local neighborhood using a distance function similar to those depicted in Fig. 4(b) and (c). In detail we perform the following steps.

1: **for** $k = 1...c$ **do**
2: **for all** orientations R **do**
3: **for all** ligand positions $\tilde{T}$ **do**
4: **if** $I_{r,l}(k, \tilde{T}, R) = 1$ **then**
5: **for all** distance samples $p \in \mathbb{R}^3$ **do**
6: $d_{r,l}(p) = \max\left(d_{r,l}(p), \max_{j=1...m} r_j^l - \left\| p - (Rp_{jk}^l + \tilde{T}) \right\|\right)$
7: **end for**
8: **end if**
9: **end for**
10: **end for**
11: **end for**

Note that the values of the distance function are only updated if they become larger. During the distance updates, points reachable by a ligand become positive and unreachable points stay negative, but might still increase. Since the ligand excluded surface is

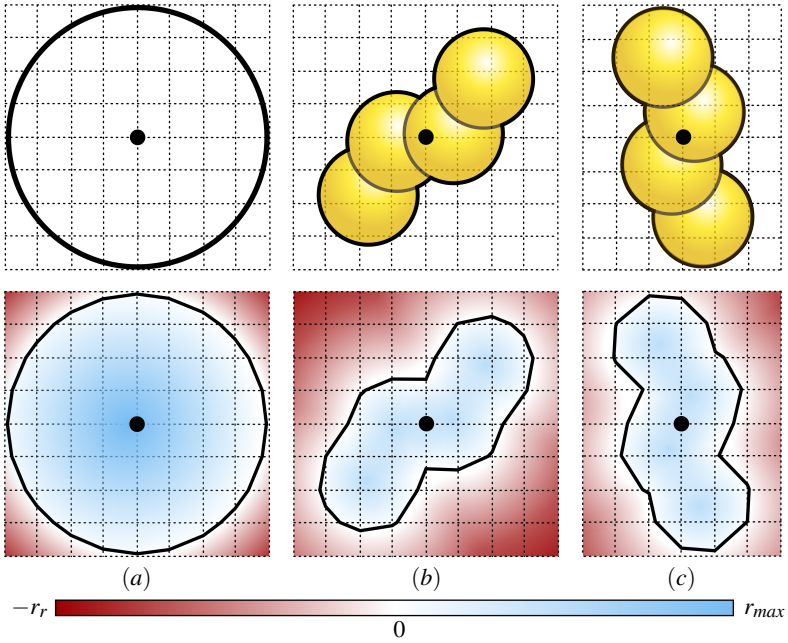

Figure 4: Illustration of local discrete distance fields. (a) The bottom image shows the local distance field of the ligand bounding sphere (top), which is used in phase I of the algorithm. (b) and (c) Local distance fields (bottom) for two ligand orientations (top). These distance fields are used in phase II.

the surface which separates positive and negative values, it is defined by the implicit description $d_{r,l}(p) = 0$.

4.4 Cavity Structure

Apart from computing the LES, our sampling approach can also be used to extract the cavity structure of the receptor. Since we have already computed all valid and invalid ligand states, we simply need to store this information. To do so, we maintain a bit array of length $c \cdot o$ at each sample point. In this bit array, for each ligand conformation and orientation, we store whether it is valid or not. If the conformation and orientation is valid, we set the bit to 1. If it is not valid, it is set to 0. Note that for sample points where we can place the maximal ligand bounding sphere, all bits of the array are set to 1. On the other hand, all bits of the array are set to 0 if the minimal ligand inscribed sphere intersects the receptor molecule. Thus we only have to store this bit array for all sample points $\tilde{T}$ (see yellow points in Fig. 3(a)).

All points with at least one bit set to 1 define valid sample points of a cavity. However, usually we are only interested in positions that represent real cavities. Therefore, we remove sample points that are outside the molecule by utilizing ambient occlusion as it was done before [29]. For each sample position with at least one valid ligand state, we cast a set of uniformly distributed rays from this position. The ambient occlusion value for this sample is the quotient of the number of rays that hit any receptor atom sphere and the overall number of rays. Thus, the higher the ambient occlusion value the less ambient light is received at this position and the deeper the sample lies inside the receptor molecule. All sample points with a value less than a user-defined threshold are removed from the following considerations.

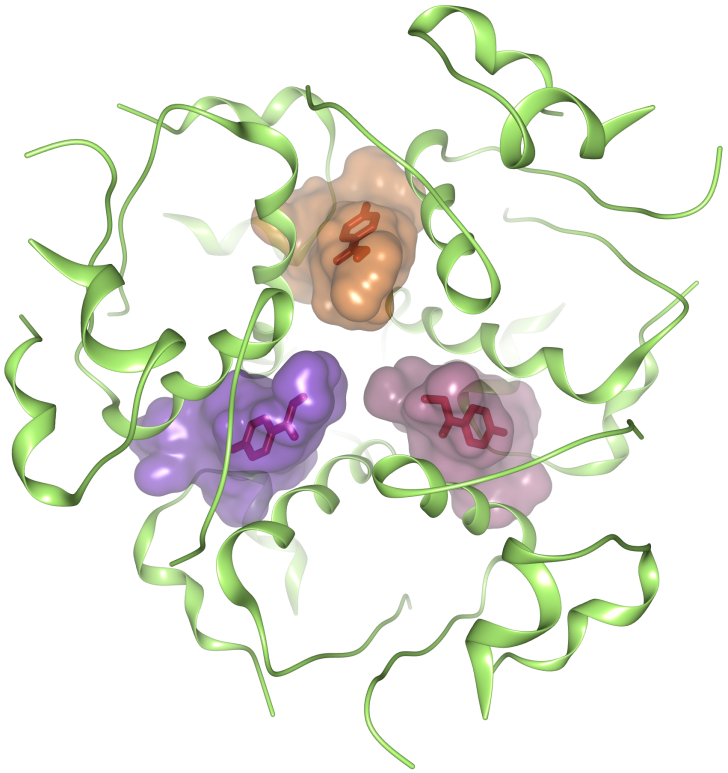

Figure 5: Cavities of hexameric insulin (PDB-Id: 3MTH) for methylparaben, computed with the LES algorithm (with 200 orientations). The three main cavities are depicted by their boundaries, together with methylparaben. The insulin molecule is depicted by its secondary structure.

The remaining sample positions are then clustered to get the cores of the cavities. Two neighboring sample positions belong to the same cluster if at least one entry is 1 at the same position in both bit arrays. This means, for all samples of the same cluster, the ligand can move from one sample position to a neighboring one without changing the orientation or conformation. At a sample position, the ligand can change its orientation or conformation to another valid state. Each cluster defines a core of a cavity. Additionally, small clusters can be filtered out according to a user-defined minimal cluster size.

From the cores of a cavity, its surface can be easily generated using again a discrete distance field. This is also initialized with a negative value, for example $-r_r$. Then the distance field is updated according to the maximum of all distance functions of valid ligand states of the cavity. The surface of the cavity is then defined as the implicit surface given by this distance field. Due to the fact that we store for each sample point its valid states, we can easily compute trajectories of the ligand from inside the cavity to the outside or vice versa.

5 Implementation

In this section, we describe implementational details for the two most expensive steps of the algorithm.

5.1 Intersection Tests

During the execution of the algorithm, many intersection tests between spheres have to be carried out. In phase I, we need intersection tests between the minimal ligand inscribed sphere and the maximal ligand bounding sphere with the atom spheres of the receptor. In phase II, intersection tests between the ligand atoms and the receptor atoms need to be computed. Without using a data structure to reduce the number of intersection tests, $n \cdot m$ sphere-sphere intersections tests need to be carried out for a single receptor ligand intersection test in the worst case, where n is the number of receptor atoms and m the number of ligand atoms. We can stop if the first intersection has been found, but if the molecules do not intersect, all intersection tests are needed. Hence, a data structure is required to reduce the number of intersection tests.

Since the receptor molecule is static, we can store the atoms in a 3-dimensional grid. In each cell of this grid, we record all receptor atoms that intersect with the cell. For molecular data, typically a uniform grid is used with a grid size of approximately 1 Å. Then, in order to compute the intersection of a sphere with the atoms of the receptor, all grid cells are accumulated that intersect with the sphere. Only atoms stored in all accumulated grid cells need to be tested for intersection, which enormously reduces the number of intersection tests. Since the atom spheres of a molecule cannot be arbitrarily dense, because they have a fixed minimal distance to each other, the number of atoms that need be tested is much smaller than n. With this, the complexity for an intersection test of a single sphere with the receptor molecule reduces from linear to constant [46].

In addition to the usage of a 3-dimensional grid as accelerator, we can further speed-up the intersection tests by computing them in parallel. This is possible since intersection tests of spheres with the receptor molecule are independent of each other. On the CPU, we can use OpenMP. Faster computations can be achieved by running the intersection tests on the GPU. In order to be platform independent, we used OpenCL [1]. In our implementation, each thread computes the intersection of one sphere with the receptor molecule. Here, the grid is stored using two arrays, one containing an integer value for each cell that represents an index to the second array with floating point values. The second array stores all spheres per cell, where each sphere consists of four values, three for the position and one for the radius. The integer array maps to the first sphere of the corresponding cell. All remaining spheres of the cell are stored in the consecutive entries. The end of the list is indicated by a sphere with negative radius.

5.2 Distance Field Updates

The distance field updates are the most expensive part of the algorithm. Recall that distance field updates are needed for two types of *instances*, that is, for the *maximal ligand bounding sphere* and for the *ligand conformations with different orientations*. If we place any of these instances at a particular sample position, updating the distance field means that for each point in the distance field, we have to compute the distance to the boundary of this instance. For the maximal ligand bounding sphere, this boundary is the sphere surface. For a ligand conformation, this boundary is the van der Waals surface of the ligand atoms, which is the surface enclosing all ligand atom spheres.

Since the distance field defining the LES is negative inside the LES and positive outside, the signs of the distances we compute to the instance boundaries need to have inversed signs. That is, we need to have positive distances inside the instance and negative distances outside. Recall that we have initialized the distance field with the negative maximal atom radius of the receptor. If we wanted to compute the correct

distance field, we would have to initialize the distances with $-\infty$. However, since we are not interested in the distance field itself but only in the implicit surface defined by $d_{r,l}(p) = 0$, only the grid points close to the LES need to have the correct values. For all other grid points, it suffices to have the correct distance sign. Once we have computed the distance of a particular grid point to the instance boundary, the distance is updated only if the new distance value is larger than the current one. That is, the values of the distance field only increase.

Now, if s is the number of grid points and we update the distance field in the naïve way, that is, all grid points for each instance at each sample position, the time complexity is $O(s^2)$. Even for medium-sized grids, this would result in very long computation times. Hence, in the following, we describe two ways to reduce this run time.

5.2.1 Local Distance Fields

The first observation is that we do not have to update the whole distance field, but only a part that is in the local neighborhood of the sample point. Here, the size of the neighborhood is defined by the particular instance. In fact, it is sufficient to consider all those grid points p that are inside the axis-aligned bounding box of the instance plus those points within a distance $\|p - p_{bb}\|_\infty \leq g$ to the bounding box (see Fig. 4). Thus, the instance will be completely surrounded by grid points with negative distance to the instance.

Since we use the same points for the distance field and the sample points of the instances, we can precompute the distances for the local neighborhood of each instance. To do so, we compute local distance fields placed in the origin with the same grid spacing g. We do this for each instance, that is, for the maximal ligand bounding sphere and all ligand conformations and orientations (see Fig. 4). Then, during an update step, the local distance field only needs to be moved to the current position and we can compare the values of the global distance field with those of the local distance field directly. The maximum of these two values determines the new value.

Since the average size $\tilde{s}$ of the local distance field is usually much smaller than s, the run time reduces from s^2 to $\tilde{s} \cdot s$. However, since $\tilde{s}$ is a fixed fraction of s, that is, $\tilde{s} = \alpha s$, where α is constant, the complexity of the algorithm remains $O(s^2)$.

5.2.2 Use of KD-Tree

A second observation can be used to further reduce the number of grid points that need to be updated: The only grid points defining the implicit surface $d_{r,l}(p) = 0$ are the ones in whose neighborhood the sign of the distance values changes. Furthermore, recall that the distance values only increase over the duration of the algorithm. Hence, if a grid point p has a positive value and all of its 26 neighbors also have positive distance values, p will not contribute to the definition of the implicit surface. Hence, we do not need to consider p any further.

For this purpose, we maintain a KD-tree that only contains those sample points that possibly need further updates. Using the KD-tree, these points can be quickly queried. Furthermore, we dynamically update the KD-tree by removing all samples that become positive and also have only positive neighbors. Thus, if we have placed an instance at a particular sample point T, we use the KD-tree to quickly identify the grid points in the local neighborhood of T with respect to the instance size and only update those points, if needed.

5.2.3 Parallel Distance Updates on the GPU

While we used the KD-tree to accelerate the CPU implementation, for the GPU we implemented a parallel distance field update. For this, the OpenCL kernel gets three inputs: the overall distance field, the local distance field, and a set of valid ligand positions for this field. Each thread corresponds to one sample point of the ligand distance field. The thread iterates over the ligand positions and detects in each step the corresponding sample point in the distance field. If the value in the distance field is smaller than the value in the ligand distance field, it is replaced by the new value. Note that this can lead to concurrent accesses of different threads on the same sample point in the distance field. However, as mentioned before, the values in the distance function can only increase. Thus, if we run the OpenCL kernel several times, the distance values converge to the correct result. To ensure the correct result, we use a single boolean variable, which is initialized with 'false' before each call of the kernel. If during the call a distance value has changed, the variable is set to true. The kernel is called again as long as the variable is 'true'. Thus we call the kernel at least two times and in practice on average three to four times.

6 Visualization

The results of the above algorithms are a discrete scalar field describing the distance function in the local vicinity of the LES, and the cavities given by all their valid ligand states. In this section, we briefly describe how we render the LES and the cavities.

The LES can be visualized either by extracting a triangular mesh with the marching cubes algorithm [31] or by direct isosurface ray casting [13]. With todays GPU implementations of marching cubes, the surface can be generated nearly as fast as with direct ray casting. With some additional effort, the triangular surface can be colored according to arbitrary molecular properties. It can also be used to measure the area of the surface and its enclosed volume. Since the discrete distance function describing the LES is a signed distance function which is positive outside the LES and negative inside, ray casting the LES can be further accelerated. Outside the LES, the distance in the distance field is always smaller or equal to the minimal Euclidean distance to the LES. Hence, instead of a constant step size for the direct isosurface ray casting, we can directly use the distance in the discrete field. This is similar to the classical sphere tracing by Hart [15], which is usually faster than a ray casting with constant step size.

The cavities can be visualized both by its cores and their surfaces. To render their surfaces, we construct for each cavity a distance field similar to the LES distance field and render it in the same way (Fig. 5). The cavity cores are visualized by placing at each core sample position a small sphere (Fig. 6). For the sphere rendering, we use a modern GPU-based ray casting [42]. The depth perception of all representations can be improved by ambient occlusion or surface cuts.

7 Results

All results have been produced on an Intel Xeon X5650 E5540 2.66 GHz with 6 cores and an Nvidia GeForce GTX 680. For measuring the quality and performance, we used several molecules from the PDB [36] as well as two other data sets.

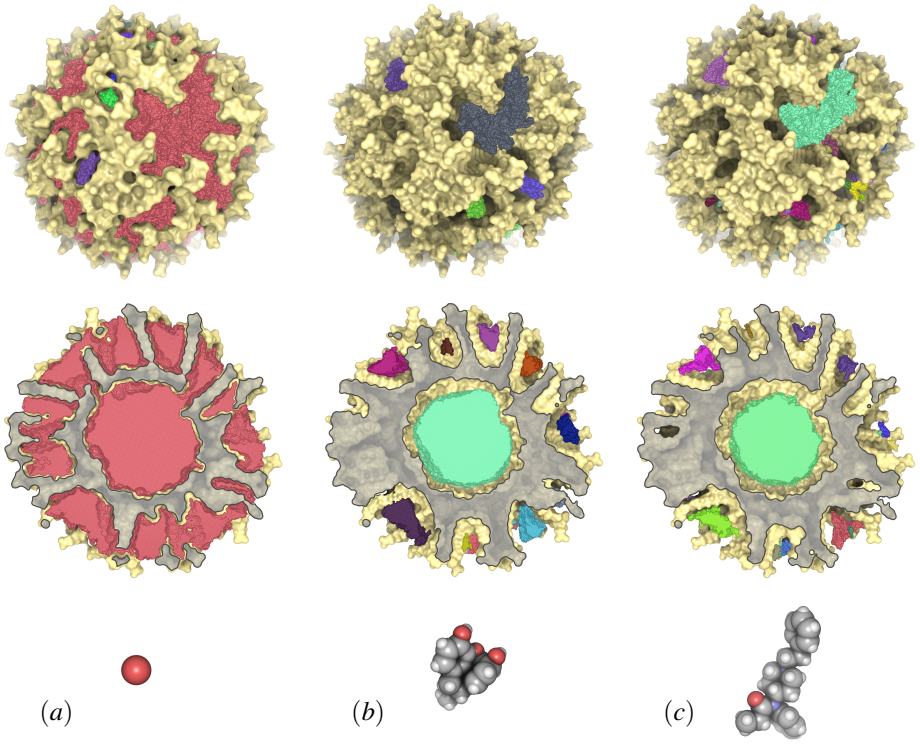

Figure 6: SES and LES with cavity cores of a dendritic core multi-shell nanotransporter: (a) SES, (b) LES of morphine, (c) LES of fentanyl. In the image, the probe sphere and the ligands have been scaled by a factor of 3. The surface of the nanotransporter is depicted in yellow, the cut of the surface is gray, and the cavity cores are colored according to their clustering.

7.1 Parameters

Although the surface definition is independent of parameters, the algorithm requires parameters due to the discretization. The two parameters used for the LES computation are the grid spacing g and the number of orientations o. In Fig. 7, we have plotted how the enclosed volume and the surface area change depending on the number of orientations. For this we used ketosteroid isomerase with ligand equilenine (PDB-Id: 1OGZ, Fig. 1). We computed the LES with 10 to 200 orientations by increasing the number of orientations by 5 in each step. From 200 to $1,000$ orientations, we increased the number of orientations by 100 each, and from $1,000$ to $10,000$ orientations, a step size of $1,000$ orientations was used. For each number of orientations, we generated a triangular mesh of the LES and computed its surface area and the enclosed volume. It can be observed that the main changes in surface area and volume occur between 10 and $1,000$ orientations. Furthermore, it can be observed that for a smaller grid spacing the changes are more rapid than for larger grid spacings.

In our experiments, we generally used 200 or fewer orientations. Only in the case of the HIV-protease with inhibitor amprenavir we were not able to find the binding site with this number of orientations. With $1,000$ orientations, however, we found the binding site even with a grid spacing of 0.75 Å (Fig. 9).

The cavity analysis only needs one parameter, which determines when a sample

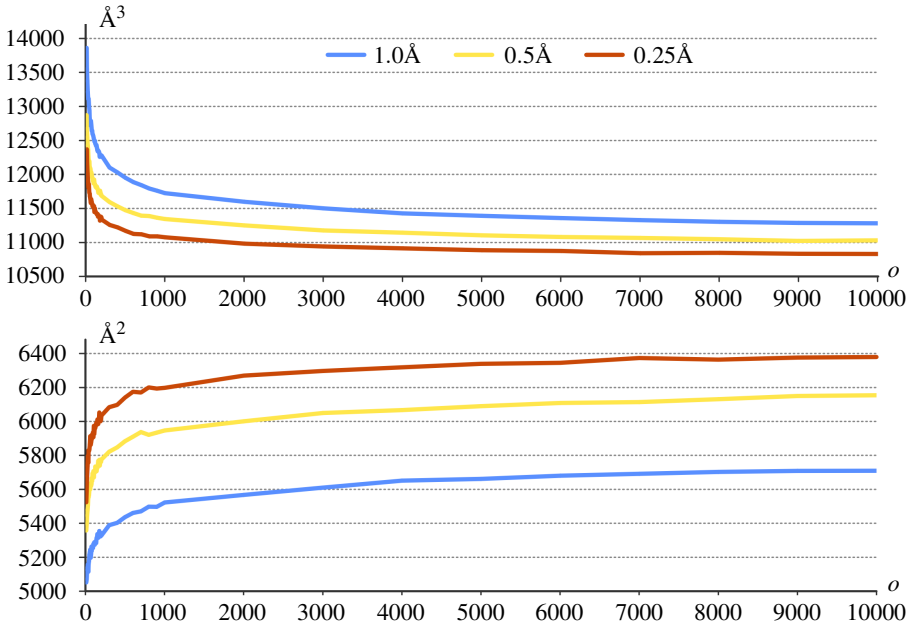

Figure 7: For the graphs shown above, the LES of ketosteroid isomerase (PDB-Id: 1OGZ; also see Fig. 1) with respect to the ligand equilenine was computed for different grid spacings and an increasing number of orientations. Note, however, that the used orientations were independent of each other. The graphs in the two images plot the enclosed volume (top) of the LES and the surface area (bottom) of the LES against the number of ligand rotations used for computing the LES.

point is considered to lie outside the molecule. To determine the degree of burriedness we used ambient occlusion by casting a fixed number of rays. We always used 100 rays. The user can then set a threshold at what ambient occlusion value a sample point is considered to lie outside the molecule.

7.2 Performance

Let $s \in \mathbb{N}$ be the number of sample positions which is defined by the grid spacing g and the size of the receptor. Note that s grows cubically when linearly decreasing g. Because of the grid data structure, the cost of each intersection test with the receptor molecule is constant, Hence, the intersection tests have a complexity of $O(c \cdot o \cdot s)$. An upper bound for the number of values in the local distance field of the ligand is $\alpha \cdot s$, where α is the ratio of the volume of the local field and the overall distance field. Hence the updates of the distance field have a complexity of $O(c \cdot o \cdot \alpha \cdot s^2)$. The k-means clustering is the most expensive part for the computation of the orientations. It grows polynomially with o. However, compared to the rest of the algorithm, in practice the computation of the orientations is negligible.

The maximal memory requirements for the algorithm are given by the following data structures. The global distance field with a space complexity of $O(s)$, the local distance field with $O(\alpha \cdot s)$, the grid data structure of the receptor molecule with $O(n)$, the KD-tree with a worst case complexity of $O(s)$, and the ligand orientations at all sample points with a complexity of $O(s \cdot o \cdot c)$. For the intersection tests on the GPU, the grid of the receptor molecule and a set of spheres are required. For the distance

updates on the GPU, the global and local distance fields and a set of update positions given by sample indices are required. Based on performance tests, for each call of an OpenCL kernel, we set the number of spheres for the intersection tests to $65,536$ and the number of positions for the distance updates to $1,024$.

In phase I we detect all sample positions that have to be investigated for each ligand conformation and orientation. This preprocessing accelerates the algorithm because many of the sample positions can be ignored in phase II. The amount of acceleration depends mainly on the size of the ligand and the ratio between this size and the size of the receptor, but also on the geometrical complexity of the receptor. During our tests, we observed that for water, which has a bounding radius of 1.52 Å, approximately 90% of the sample positions can be ignored in phase II. For morphine, with a bounding radius of 5.44 Å, still around 75% of the sample positions can be ignored and for fentanyl, whose bounding radius is 8.47 Å, approximately 68% can be ignored. For all tested ligands, on average the number of sample points was higher than 60%. Note that the percentages for each ligand were averaged over a couple of receptors of different size, ranging from 272 atoms to $58,870$ atoms.

Detailed timings for all parts of the algorithm are given in Table 1. In all our examples, we observed that the computations for common receptor-ligand systems were 6 to 10 times faster on the GPU compared to the CPU with KD-tree. On the CPU, the use of the KD-tree accelerated the computation by a factor of at least two. Furthermore, we observed that the intersection tests can be slower than the distance updates, although the complexity of the distance updates is worse (see, for example, 1HPV or the nanotransporter with morphine).

Most of the memory is required to store the valid ligand orientations at each sample position. For the nanotransporter and morphine in Table 1, this structure requires approximately 50 MB. For 1HPV and amprenavir with a grid spacing of 0.25 Å, approximately 3 GB are required. However, this memory is only temporarily needed if cavities are computed. The largest memory consumption on the GPU is due to the global distance field. For the nanotransporter and morphine, this field requires approximately 25 MB. For 1HPV and amprenavir with a grid spacing of 0.25 Å, approximately 66 MB are required. Detailed memory requirements for the data sets in Table 1 are given in table 2.

While the SES can be computed interactively, the computation of the LES usually takes several minutes up to hours depending on the grid spacing and the number of orientations (see Table 1). On the other hand, the LES better reflects the actual accessibility of a certain ligand, which can be seen in Figs 1, 2, and 6.

Table 1: Run times (given in seconds) for LES and cavity computation for different receptors, ligands, grid spacings (g) and number of orientations (o) both on CPU (*without KD-tree) and GPU. Phases I and II are split into the two most time-consuming steps: intersection tests (IT) and distance updates (DU). Note that for 1HPV, four conformations of amprenavir were used with 1,000 orientations each.

system	**receptor (#atoms)**	**ligand (#atoms)**	***g*** **(in Å)**	***o***	**phase I IT**	**phase I DU**	**phase II IT**	**phase II DU**	**LES**	**boundary detection**	**core computation**	**cavity surface**	**overall**
CPU	1GRM (272)	water (3)	0.25	200	0.2	0.7	32.8	114.7	**148.4**	11.5	0.1	14.1	**174**
GPU	1GRM (272)	water (3)	0.25	200	0.1	0.2	4.6	16.2	**21.1**	0.6	0.1	1.5	**23**
CPU*	1OGZ (944)	equilenine (20)	0.5	200	0.4	5.6	265.4	1057.1	**1328.5**	16.3	0.1	3.5	**1348**
CPU	1OGZ (944)	equilenine (20)	0.5	200	0.4	5.6	255.4	383.1	**644.4**	15.7	0.1	3.6	**664**
GPU	1OGZ (944)	equilenine (20)	0.5	200	0.1	0.4	29.5	50.9	**80.9**	0.7	0.1	0.5	**82**
CPU	nanotrans. (16487)	water (3)	0.5	200	0.4	1.6	87.7	297.7	**387.3**	125.2	1.6	128.0	**642**
CPU	nanotrans. (16487)	sulfate (5)	0.5	200	0.5	3.3	159.2	557.1	**720.1**	122.7	0.9	236.1	**1080**
CPU	nanotrans. (16487)	morphine (40)	0.5	200	2.8	13.1	2175.6	762.4	**2955.1**	131.7	0.4	533.0	**3620**
GPU	nanotrans. (16487)	water (3)	0.5	200	0.4	1.2	15.8	51.9	**69.3**	4.5	1.6	11.7	**87**
GPU	nanotrans. (16487)	sulfate (5)	0.5	200	0.4	1.2	18.2	58.5	**78.3**	3.9	0.9	20.4	**104**
GPU	nanotrans. (16487)	morphine (40)	0.5	200	0.8	0.9	367.1	139.2	**508.1**	4.2	0.4	82.4	**595**
GPU	1HPV (1516)	amprenavir (35)	1.0	4000	0.1	0.1	345.2	152.7	**498.1**	0.3	0.1	0.1	**499**
GPU	1HPV (1516)	amprenavir (35)	0.5	4000	0.3	0.9	2436.7	1970.2	**4408.1**	1.7	0.1	0.2	**4410**
GPU	1HPV (1516)	amprenavir (35)	0.25	4000	1.2	21.9	18540.9	66939.2	**85503.2**	11.8	0.5	6.3	**85522**

System: Intel Xeon X5650 E5540 2.66 GHz with an Nvidia GeForce GTX 680.

Table 2: Memory requirements for the computation of the LES for different receptor and ligand molecules. The grid spacing is denoted g and the number of ligand orientations o. For 1HPV with the ligand amprenavir, 4 conformations with 1,000 orientations per confirmation were used. Storage of the valid ligand states is not necessary for the LES computation, but temporarily it is used to extract the cavities. The LES surface mesh was extracted from the global distance field by using the marching cubes algorithm.

receptor (#atoms)	**ligand (#atoms)**	***g* (in Å)**	***o***	**global distance field (in MB)**	**local distance field (in KB)**	**grid (in KB)**	**kd-tree (in MB)**	**valid ligand states (in MB)**	**LES surface mesh (in MB)**
1GRM (272)	water (3)	0.25	200	4.5	13.2	50.8	2.1	5.0	**2.1**
1OGZ (944)	equilenine (20)	0.5	200	5.3	76.9	127.9	1.4	9.0	**1.1**
nanotrans. (16482)	water (3)	0.5	200	19.3	2.8	2381.3	8.7	15.7	**13.1**
nanotrans. (16482)	sulfate (5)	0.5	200	20.6	5.2	2381.3	10.3	13.3	**11.8**
nanotrans. (16482)	morphine (40)	0.5	200	24.9	47.5	2381.3	12.5	49.7	**8.2**
1HPV (1516)	amprenavir (35)	1.0	4000	1.1	26.8	220.1	0.3	46.9	**0.4**
1HPV (1516)	amprenavir (35)	0.5	4000	8.5	197.9	220.1	2.2	383.9	**1.5**
1HPV (1516)	amprenavir (35)	0.25	4000	66.2	1519.6	220.1	16.3	3106.9	**6.3**

8 Feedback by Domain Scientists

Apart from an evaluation in terms of computational complexity and performance, we carried out a small survey about the usability of the proposed new molecular surface by performing a *structured interview* about potential advantages and disadvantages of the LES. We interviewed four senior experts from different labs, working in molecular dynamics simulation and dealing with a wide range of applications. In the following we summarize the most important statements.

Advantages: All experts agreed that the LES provides valuable ligand-specific information. In particular, the LES can help identify binding sites that have so far been unknown from experimental data. Furthermore, the LES allows one to discard cavities that are poor candidates as hosts for a ligand. Thus, more expensive methods computing the binding free energies or molecular dynamics simulations using force fields can be applied more effectively. If a protein can bind different ligands, the LES might also enable the identification of different binding sites for different ligands. The pure visualization of the LES is of interest, because in many applications of molecular modeling, the chemical intuition of the observer is an important addition to fully automatic methods. This intuition is even further supported by the identified cavities.

Disadvantages: If both the protein and the ligand are highly flexible, the computation of the LES is expensive. Then it might be favorable to use the SES to identify binding sites and cavities and compute the LES only for a few time steps or locally near the binding site.

9 Discussion

The most crucial parameter of our algorithm is the number of orientations. We tried to investigate the quality of the results depending on the number of orientations by plotting the surface area and the enclosed volume versus the number of orientations (Fig. 7). It is clear that both curves will converge, but the plots gives us a good indication, how fast this convergence will be. What can be clearly seen is that the largest changes occur between 10 and $1,000$ orientations. This suggests that with less than $1,000$ orientations the LES can be very well approximated. In our experiments, we typically used 200 or even less orientations and were able to reproduce most known cavities. In rare cases, however, it might be necessary to use more than this number or even more than $1,000$ orientations. Hence, if the run time is of minor importance, we suggest using between 500 and $1,000$ orientations.

More important in terms of run time is the choice of the grid spacing. The run time grows quadratically with the number of grid points, which again grows inversely cubically with the grid spacing. For grid-based cavity analysis, commonly a minimal grid spacing of 0.5 Å is used. We either used a grid spacing of 0.5 Å or 0.25 Å. While the latter is clearly favorable in terms of precision, in our experiments, we measured an approximate 28-fold run time. Furthermore, in all our experiments, a grid spacing of 0.5 Å was sufficient to find the known binding sites. Hence, we suggest using a grid spacing of 0.5 Å.

If the ligand binding site is very tight and the flexibility of the ligand is high, it might be necessary to use a large number of conformations. Since the run time depends linearly on the number of conformations, this is expensive. In this case, it might be possible to reduce the complexity by taking into account the redundancy in the conformations.

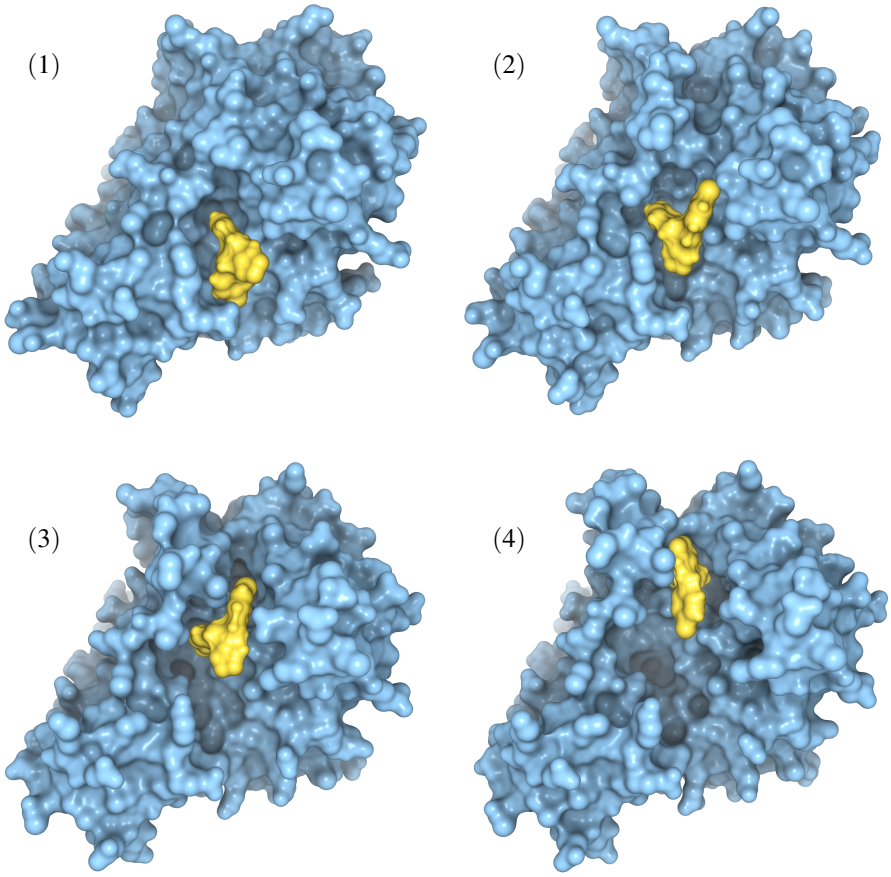

Figure 8: LES (blue) depicted for four selected time steps of a simulation trajectory of β-lactamase (data courtesy of Gregory L. Bowman and Philipp L. Geissler [4]). During the simulation, the active site of the β-lactamase opens. The sequence of images shows the computed positions of the CBT ligand (yellow) closest to the active site.

10 Conclusion and Future Work

We have defined a new type of molecular surface, the ligand excluded surface (LES), which naturally extends the definition of the solvent excluded surface (SES). Instead of approximating the ligand by a probe, the LES considers the complex geometry of a ligand described by its van der Waals atoms. Thus, the LES represents a more accurate approximation of the regions accessible to the ligand than the SES. We have presented a grid-based algorithm to compute an approximation of the LES. This algorithm has a complexity of $O(s^2)$, where s is the number of grid points. Despite this complexity, we show that it is feasible to calculate the LES for typical receptors and ligands in a reasonable time. This is possible because we identified the most time-consuming parts of the algorithm and optimized these parts algorithmically but also by parallelizing them on the CPU and the GPU.

We have also shown that with a minor extension of the algorithm, the cavity structure of the molecule with respect to the ligand can be computed with almost no runtime overhead. In addition to the positions, we also obtain the possible orientations and

conformations of the ligand. This information might be exploited to steer molecular docking simulations, where the ligand binding path into the active site is detected (see Fig. 8).

It is obvious that with the calculation of the LES, a new computational challenge is associated. Below we outline some ideas on how the computation could be improved and extended.

First, the fixed user-defined number of orientations might be eliminated by measuring the error rate between increasing numbers of orientations. For this, we might start with a few orientations and subsequently add more orientations while at the same time measuring the rate of change of the volume. As long as the change is larger than a predefined fraction, we would continue adding more orientations. This strategy, however, requires a suitable way to successively pick new orientations from a large set of orientations such that the distances between neighbored orientations remains similar. But it is not obvious, how the selection can be done most efficiently.

Currently, each sample position with at least one valid ligand state belongs to exactly one cavity core. However, it is possible that the ligand cannot continuously change from one valid state to another valid state at the same sample position without intersecting the receptor molecule. Hence, the cavity cores might have to be split at such positions which also means that a sampling position can belong to different cavities and, thus, that the cores can overlap. In the future, we want to handle such situations more correctly. For this, it is necessary to define valid changes of orientations and conformations within a sample position.

Despite our algorithmic optimizations, the run time of the algorithm can still be very long, particularly for large molecules and a small grid spacing. In terms of cavity analysis, one is often only interested in cavities existing in certain regions. Thus, it might be sensible to restrict the cavity computation to user-defined regions. Furthermore, a combination of our approach with one that approximates the ligand by a sphere, such as our molecular path computation [29], might significantly speed up the computation.

We would also like to investigate whether the use of tetrahedral meshes with the same number of grid points as for hexahedral meshes improves the surface quality and leads to a faster convergence.

The focus of this work was the definition and computation of a purely geometry-based representation of the accessibility of a receptor with respect to a ligand molecule. The definition of the LES could be extended further by considering physico-chemical properties, such as the binding affinity of the ligand. By considering only those positions and orientations for which the binding affinity is high, the LES might even better reflect the true accessibility of the ligand.

11 Acknowledgments

The authors thank Greg Bowman from the Miller Institute for his feedback and for providing the dynamic data set (see Fig. 8), Marcus Weber from Zuse Institute Berlin for his feedback and Marcus Weber and Amir Sedighi for providing the CMS nanotransporter (see Fig. 6), and finally Ana-Nicoleta Bondar from Freie Universität Berlin and Frank Cordes from GETLIG & TAR for their helpful comments and feedback.

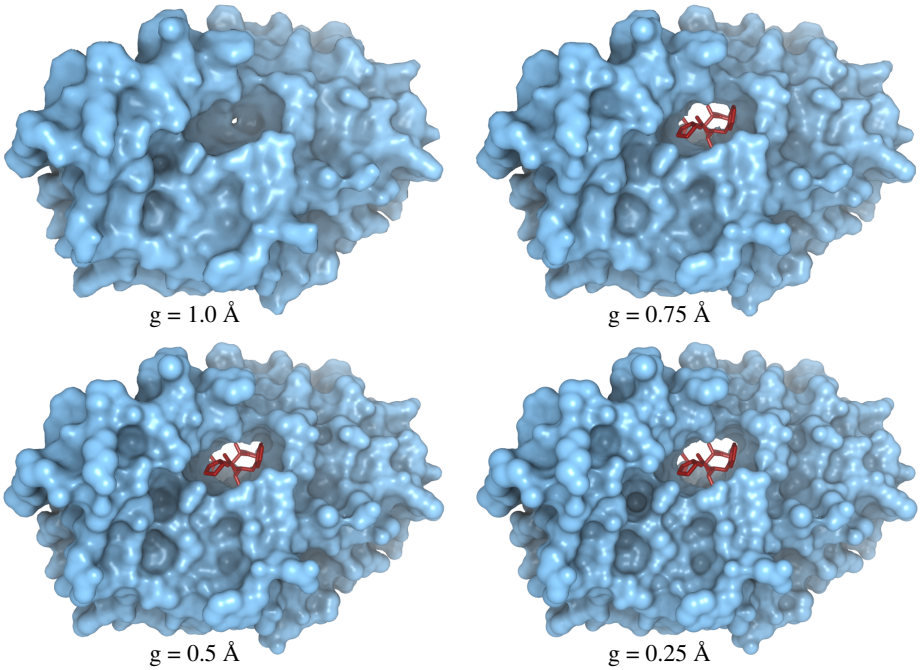

Figure 9: LES of HIV protease (PDB-Id: 1HPV) for ligand amprenavir (red) with different grid resolutions. For the computation of the LES, four ligand conformations with 1,000 orientations were considered. For a grid spacing of 1.0 Å, the ligand binding site was not found.

References

[1] OpenCL Website, 2014. http://www.khronos.org/opencl/.

[2] C. Bajaj and V. Siddavanahalli. Adaptive grid based methods for computing molecular surfaces and properties. Technical report, The University of Texas at Austin, 2006. ICES Report 06-56, Institute for Computational and Engineering Sciences.

[3] T. A. Binkowski, S. Naghibzadeh, and J. Liang. CASTp: Computed atlas of surface topography of proteins. *Nucleic Acids Research*, 31(13):3352–3355, 2003.

[4] G. R. Bowman and P. L. Geissler. Equilibrium fluctuations of a single folded protein reveal a multitude of potential cryptic allosteric sites. *Proc Natl Acad Sci U S A*, 109(29):11681–6, Jul 2012.

[5] G. P. Brady Jr. and P. F. W. Stouten. Fast prediction and visualization of protein binding pockets with pass. *Journal of Computer-Aided Molecular Design*, 14(4):383–401, 2000.

[6] M. L. Connolly. Molecular surface triangulation. *Journal of Applied Crystallography*, 18:499–505, 1985.

[7] J. Cortés, S. Barbe, M. Erard, and T. Siméon. Encoding molecular motions in voxel maps. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 8:557–563, March 2011.

[8] J. Cortés, T. Siméon, V. Ruiz de Angulo, D. Guieysse, M. Remaud-Simon, and V. Tran. A path planning approach for computing large-amplitude motions of flexible molecules. *Bioinformatics*, 21(suppl 1):i116–i125, 2005.

[9] H. Edelsbrunner, M. A. Facello, P. Fu, and J. Liang. Measuring proteins and voids in proteins. In *System Sciences, 1995. Proceedings of the Twenty-Eighth Hawaii International Conference on*, volume 5, pages 256–264. IEEE, 1995.

[10] K. Fischer. *Smallest enclosing balls of balls: Combinatorial structure & algorithms*. PhD thesis, ETH Zürich, 2005.

[11] M. L. Gavrilova and J. Rokne. Updating the topology of the dynamic Voronoi diagram for spheres in Euclidean d-dimensional space. *Comput. Aided Geom. Des.*, 20:231–242, 2003.

[12] J. Greer and B. L. Bush. Macromolecular shape and surface maps by solvent exclusion. *Proceedings of the National Academy of Sciences*, 75(1):303–307, January 1978.

[13] M. Hadwiger, C. Sigg, H. Scharsach, K. Bühler, and M. H. Gross. Real-time ray-casting and advanced shading of discrete isosurfaces. *Computer Graphics Forum*, 24(3):303–312, 2005.

[14] M. Haranczyk and J. A. Sethian. Navigating molecular worms inside chemical labyrinths. *Proceedings of the National Academy of Sciences*, 106(51):21472–21477, 2009.

[15] J. C. Hart. Sphere tracing: a geometric method for the antialiased ray tracing of implicit surfaces. *The Visual Computer*, 12(10):527–545, 1996.

[16] M. Hendlich, F. Rippmann, and G. Barnickel. Ligsite: automatic and efficient detection of potential small molecule-binding sites in proteins. *J Mol Graph Model*, 15(6):359–63, 389, Dec 1997.

[17] J. Higo and N. Go. Algorithm for rapid calculation of excluded volume of large molecules. *J. Comp. Chem.*, 10:376–379, 1989.

[18] B. Huang. Metapocket: A meta approach to improve protein ligand binding site prediction. *OMICS*, 13(4):325–330, 2009.

[19] B. Huang and M. Schroeder. LIGSITEcsc: predicting ligand binding sites using the Connolly surface and degree of conservation. *BMC Struct Biol*, 6:19, 2006.

[20] M. Krone, K. Bidmon, and T. Ertl. Interactive visualization of molecular surface dynamics. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1391–1398, 2009.

[21] M. Krone, J. E.Stone, T. Ertl, and K. Schulten. Fast Visualization of Gaussian Density Surfaces for Molecular Dynamics and Particle System Trajectories. Vienna, Austria, 2012. Eurographics Association.

[22] M. Krone, M. Falk, S. Rehm, J. Pleiss, and T. Ertl. Interactive exploration of protein cavities. *Computer Graphics Forum*, 30(3):673–682, 2011.

[23] M. Krone, S. Grottel, and T. Ertl. Parallel contour-buildup algorithm for the molecular surface. In *Proceedings of IEEE Symposium on Biological Data Visualization (BioVis '11)*, pages 17–22, 2011.

[24] R. A. Laskowski. Surfnet: A program for visualizing molecular surfaces, cavities, and intermolecular interactions. *Journal of Molecular Graphics*, 13:323–330, 1995.

[25] A. T. R. Laurie and R. M. Jackson. Q-sitefinder: an energy-based method for the prediction of protein-ligand binding sites. *Bioinformatics*, 21(9):1908–1916, 2005.

[26] B. Lee and F. M. Richards. The interpretation of protein structures: Estimation of static accessibility. *Journal of Molecular Biology*, 55:379–400, 1971.

[27] D. G. Levitt and L. J. Banaszak. Pocket: A computer graphics method for identifying and displaying protein cavities and their surrounding amino acids. *J. Mol. Graph.*, 10(4):229–234, Dec. 1992.

[28] N. Lindow, D. Baum, A.-N. Bondar, and H.-C. Hege. Exploration of cavity dynamics in biomolecular systems. *BMC Bioinformatics*, 14, 2013.

[29] N. Lindow, D. Baum, and H.-C. Hege. Voronoi-based extraction and visualization of molecular paths. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2025 – 2034, 2011.

[30] N. Lindow, D. Baum, S. Prohaska, and H.-C. Hege. Accelerated visualization of dynamic molecular surfaces. *Comput. Graph. Forum*, 29:943 – 952, 2010.

[31] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *ACM Siggraph Computer Graphics (Proc. of SIGGRAPH 87)*, volume 21, pages 163–169. ACM, July 1987.

[32] P. Medek, P. Beneš, and J. Sochor. Computation of tunnels in protein molecules using Delaunay triangulation. *Journal of WSCG, University of West Bohemia, Pilsen*, 15(1-3):107–114, 2007.

[33] J. Parulek and A. Brambilla. Fast blending scheme for molecular surface representation. *IEEE Trans. Vis. Comput. Graph.*, 19(12):2653–2662, 2013.

[34] J. Parulek and I. Viola. Implicit representation of molecular surfaces. In H. Hauser, S. G. Kobourov, and H. Qu, editors, *PacificVis*, pages 217–224. IEEE, 2012.

[35] M. Pavlov and B. Fedorov. Improved technique for calculating X-ray scattering intensity of biopolymers in solution: Evaluation of the form, volume, and surface of a particle. *Biopolymers*, 22:1507–1522, 1983.

[36] Protein Data Bank. http://www.pdb.org.

[37] L. H. Pearl and A. Honegger. Generation of molecular surfaces for graphic display. *Journal of Molecular Graphics*, 1(1):9–12, 1983.

[38] M. Petřek, P. Košinová, J. Koča, and M. Otyepka. MOLE: a Voronoi diagram-based explorer of molecular channels, pores, and tunnels. *Structure*, 15(11):1357–1363, Nov 2007.

[39] M. Petřek, M. Otyepka, P. Banas, P. Kosinová, J. Koca, and J. Damborsky. CAVER: a new tool to explore routes from protein clefts, pockets and cavities. *BMC Bioinformatics*, 7(1):316+, June 2006.

[40] F. M. Richards. Areas, volumes, packing, and protein structure. *Annual Review of Biophysics and Bioengineering*, 6:151–176, 1977.

[41] M. F. Sanner, A. J. Olson, and J.-C. Spehner. Fast and robust computation of molecular surfaces. In *Symposium on Computational Geometry*, pages C6–C7, 1995.

[42] C. Sigg, T. Weyrich, M. Botsch, and M. Gross. GPU-based ray-casting of quadratic surfaces. In *Proceedings of the Eurographics Symposium on Point-Based Graphics*, pages 59–65, 2006.

[43] O. S. Smart, J. G. Neduvelil, X. Wang, B. A. Wallace, and M. S. Sansom. Hole: a program for the analysis of the pore dimensions of ion channel structural models. *J Mol Graph*, 14(6):354–60, 376, Dec 1996.

[44] R. Sridharamurthy, H. Doraiswamy, S. Patel, R. Varadarajan, and V. Natarajan. Extraction of Robust Voids and Pockets in Proteins. In M. Hlawitschka and T. Weinkauf, editors, *EuroVis - Short Papers*, pages 67–71, Leipzig, Germany, 2013. Eurographics Association.

[45] M. Totrov and R. Abagyan. The contour-buildup algorithm to calculate the analytical molecular surface. *Journal of Structural Biology*, 116:138–143, 1996.

[46] A. Varshney, F. P. B. Jr., J. William, and W. V. Wright. Linearly scalable computation of smooth molecular surfaces. In *IEEE Computer Graphics and Applications*, volume 14, pages 19–25, 1994.

[47] M. Weisel, E. Proschak, and G. Schneider. PocketPicker: analysis of ligand binding-sites with shape descriptors. *Chemistry Central Journal*, 1(7):1–17, 2007.

[48] E. Yaffe, D. Fishelovitch, H. J. Wolfson, D. Halperin, and R. Nussinov. MolAxis: Efficient and accurate identification of channels in macromolecules. *Proteins: Structure, Function, and Bioinformatics*, 73(1):72–86, 2008.