Extraction of Robust Voids and Pockets in Proteins

Raghavendra Sridharamurthy, Talha Bin Masood, Harish Doraiswamy, Siddharth Patel, Raghavan Varadarajan, and Vijay Natarajan

Abstract Voids and pockets in a protein, collectively called as cavities, refer to empty spaces that are enclosed by the protein molecule. Existing methods to compute, measure, and visualize the cavities in a protein molecule are sensitive to inaccuracies in the empirically determined atomic radii. This paper presents a topological framework that enables robust computation and visualization of these structures. Given a fixed set of atoms, cavities are represented as subsets of the weighted Delaunay triangulation of atom centres. A novel notion of (ε,π)-stable cavities helps identify cavities that are stable even after perturbing the atom radii by a small value. An efficient method is described to compute these stable cavities for a given input pair of values (ε,π). This approach is used to identify potential pockets and channels in protein structures.

1 Introduction

A cavity in a protein molecule refers to both voids (without openings) and pockets (with openings). These cavities play a key role in determining the stability and

Raghavendra Sridharamurthy · Talha Bin Masood
Department of Computer Science and Automation, Indian Institute of Science, Bangalore, e-mail: g.s.raghavendra@gmail.com, tbmasood@csa.iisc.ernet.in

Harish Doraiswamy
Department of Computer Science and Engineering, NYU Polytechnic School of Engineering, USA, e-mail: harishd@nyu.edu

Siddharth Patel · Raghavan Varadarajan
Molecular Biophysics Unit, Indian Institute of Science, Bangalore, e-mail: {spatel, varadar}@mbu.iisc.ernet.in

Vijay Natarajan
Department of Computer Science and Automation, Supercomputer Education and Research Centre, Indian Institute of Science, Bangalore, e-mail: vijayn@csa.iisc.ernet.in

function of proteins. From the biologist's point of view, obtaining a stable protein is the starting point of many applications, from in-vitro studies of binding and interactions, to using the protein as an antigen or vaccine. Whereas surface pockets often form part of the active site of enzymes or interacting sites for other proteins, internal voids are often relevant structurally as features that affect the overall thermodynamic stability of the protein. It is established that filling up internal voids improves the packing of the protein thus increasing stability. In this respect, detecting and visualizing structurally robust cavities inside the protein informs the biologist on which mutations to perform to improve internal packing and get a stable protein.

Related work. Several methods have been proposed to locate cavities in protein molecules. In this paper, we focus our attention on geometric methods. Edelsbrunner et al. [11, 13] and Liang et al. [21, 22] proposed a definition that is based on the theory of alpha shapes and discrete flows in Delaunay triangulations. Kim et al. [16, 18] proposes a definition of cavities based on an alternate representation of a set of atoms called beta shapes that faithfully captures proximity. Tools based on the above approach are available and widely used [6, 17, 19]. Till and Ullmann [32] employed a graph theoretic algorithm to identify cavities and compute their volume. Parulek et al. [28] used graph based methods on the implicit representation of molecular surfaces to identify pockets and potential binding sites. Varadarajan et al. [3] employed a Monte Carlo procedure to position water molecules together with a Voronoi region-based method to locate empty space. They discussed the importance of accurate identification of cavities for the study of protein structure and stability. Novel Voronoi diagram-based techniques for the extraction and visualization of cavities have also been developed from the viewpoint of studying and interactively exploring access paths to active sites [23, 24, 29, 30]. Krone et al. [20] presented a visualization tool for interactive exploration of protein cavities in dynamic data.

Motivation. The input in the above-mentioned methods are protein structures determined from x-ray crystallography data or other lower resolution data. These cavity detection methods are sensitive to inaccuracies that are inherent in the crystallographic measurements. While the measurements may guarantee high resolution, it is important to note that even small inaccuracies may cause a difference in the reported number of cavities. Inaccuracies may also arise due to fundamental limitations such as the notion of radii of atoms, which is determined empirically. For example, as illustrated in Figure 1, presence of such inaccuracies may result in a cavity detection method to report two distinct but large cavities in place of one, or report very small volume cavities. Figure 2 illustrates the problem as it occurs in a lyzosyme protein.

Contributions. In this work, we aim to develop an interactive method to compute robust cavities in proteins. Our goal is to enable the user to reduce, if not completely eliminate, the inaccuracies mentioned earlier. In order to achieve this, we first provide a novel definition for robustness in the presence of inaccuracies in the measured radii.

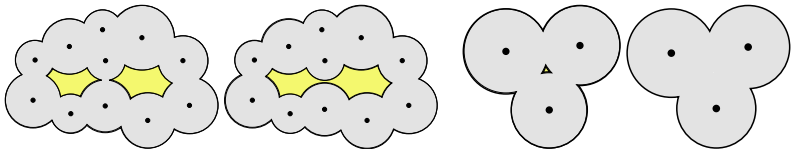


Fig. 1 Left: Two cavities that are apparently very near to each other may be a single cavity. **Right:** A very small cavity may be reported whereas no such cavity may exist.

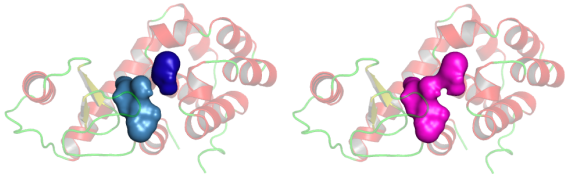


Fig. 2 Left: Two cavities that appear very near to each other in a lyzozyme protein (PDB ID: 200L). The solid surface represents cavities while the protein is shown as cartoon for context. **Right:** The two cavities may be a single cavity.

We then propose a method for computing robust and stable cavities in proteins[1]. This is accomplished through the use of a simple and succinct structure called the alpha complex to represent protein molecules. The alpha complex is a simplicial complex that can be stored as a filtration, a series of simplicial complexes K^i with $K^i \subset K^{i+1}$. In order to identify the set of cavities that are stable with respect to small perturbations in the atom radii, our method symbolically modifies the radii of a select set of atoms by systematically processing and modifying the filtration. We show that this modification results in controlled changes in the number and properties of cavities and does not violate key properties of the filtration. The method is efficient in terms of running time performance and also supports the elimination of very small or insignificant voids as measured by the notion of topological persistence [14].

We develop software to visualize the stable cavities together with the molecule, and to calculate cavity volumes and surface areas. This software provides an interactive framework that a biologist can use to decide which cavities are more relevant and what mutations to perform. The software also supports exporting the detected cavities with the relevant biochemical context to enable their visualization in PyMOL [5].

Finally, we use this software to demonstrate the applicability of the notion of robust voids and pockets and apply it to detect potential channels and pockets in several proteins.

2 Geometry representation of biomolecules

In this section, we briefly introduce the mathematical background required to define and represent the structure of biomolecules [8, 9, 27].

[1] A preliminary version of this work appeared as a short paper in the Proceedings of Eurographics Conference on Visualization [31].

Simplicial complex. A *k-simplex* σ is the convex hull of $k+1$ affinely independent points. A vertex, edge, triangle, and tetrahedron are k-simplices of dimension $0-3$. A simplex τ is a *face* of σ, $\tau \leq \sigma$, if it is the convex hull of a non-empty subset of the $k+1$ points. A *simplicial complex* K is used to represent a topological space and is a finite collection of simplices such that (a) $\sigma \in K$ and $\tau \leq \sigma$ implies $\tau \in K$, and (b) $\sigma_1, \sigma_2 \in K$ implies $\sigma_1 \bigcap \sigma_2$ is either empty or a face of both σ_1 and σ_2. A *subcomplex* of K is a simplicial complex $L \subseteq K$.

Voronoi diagram and Delaunay triangulation. Let $S \subseteq \mathbb{R}^d$ be a finite set of points. The *Voronoi cell* V_p, of a point $p \in S$, is the set of points in $\mathbb{R}^d$ whose Euclidean distance to p is smaller than or equal to any other point in S. The collection of Voronoi cells of all the points in S partitions $\mathbb{R}^d$, and is called the *Voronoi diagram* (Figure 3(a)). The *Delaunay triangulation* D of S is the dual of the Voronoi diagram and partitions the convex hull of S, see Figure 3(b). The *weighted Voronoi diagram* and *weighted Delaunay triangulation* are similarly defined for a set of balls, which is considered as a set of weighted points. The weight is equal to the square of the radius of the ball, and the distance between a weighted point p with weight w_p and a point $x \in \mathbb{R}^d$ is given by the *power distance* $\|x-p\|^2 - w_p$.

Alpha complex. Molecules are often represented using a space-filling model such as a union of balls. The weighted Voronoi diagram helps represent the contribution from each atom to the union of balls. Consider an atom p. Define B_p as an open ball having the radius of the atom p. Let V_p be the weighted Voronoi cell correspond-

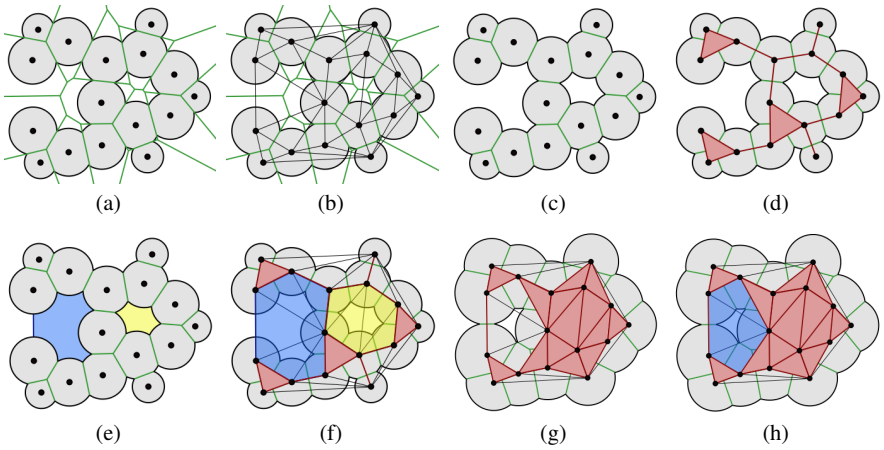


Fig. 3 **(a)** Voronoi diagram of a weighted point set in $\mathbb{R}^2$, Voronoi edges are in green. **(b)** The Delaunay complex is the dual of the Voronoi diagram. **(c)** Intersection of the weighted Voronoi diagram and the union of balls. **(d)** The dual complex is the dual of this partition of the union of balls that captures the incidence relationship. In this particular case, $\alpha = 0$. **(e)** Void and pocket in a collection of 2D balls. Void is shown in yellow and pocket is shown in blue. **(f)** Void and Pocket shown as the connected components of the complement of alpha complex *i.e.*, $D-K$. **(g)** The dual complex shown for some $\alpha > 0$ where the void has been filled up and original pocket has become a void. **(h)** The new void is highlighted using blue, Delaunay edges (in black) and alpha complex (in red) are also shown to provide context.

ing to p. The contribution from each atom p is equal to $B_p \cap V_p$, the intersection between the ball corresponding to the atom and the weighted Voronoi cell of p, see Figure 3(c). The corresponding dual structure is a subcomplex of the weighted Delaunay triangulation and called the *dual complex*, see Figure 3(d).

Edelsbrunner et al. [7, 12, 15] consider a growth model, where the ball weights grow, and track the changes in the dual complex. The growth parameter, α, corresponds to a radius $\sqrt{r_p^2 \pm \alpha^2}$ for a ball centered at p with radius r_p. Positive values of α correspond to growing the balls and negative values correspond to shrinking the balls. The weight of the point $w(p)$ increases or decreases by α^2 and hence ranges between $-\infty$ and ∞. A negative weight corresponds to imaginary radius. Note that $\alpha = 0$ corresponds to no growth. The dual complex corresponding to a set of balls after they are grown by α is called the *alpha complex*.

Given a simplicial complex K, a finite sequence $\emptyset = K^0, K^1, \ldots, K^m = K$ of subcomplexes of K is a *filtration* if $K^0 \subset K^1 \subset \cdots \subset K^m$. Figures 3(d) and 3(g) show two subcomplexes (in red) which are part of a filtration. The *rank* of a subcomplex refers to its position in the filtration. The set of alpha complexes obtained by varying α from $-\infty$ to ∞ is a filtration of the Delaunay triangulation. In particular, we consider the filtration that is generated by inserting the simplices one at a time and if more than one simplex appear at the same value of α, we order them based on their dimension $(0 < 1 < 2 < 3)$. A vertex is inserted into the filtration when the weight of the ball becomes positive.

Voids and pockets. Let the alpha complex K represent a molecule at a given value α and D be the Delaunay complex of the weighted point set. A *cavity* is a maximally connected component of the complement $D - K$. *Voids* and *pockets* are cavities that are, respectively, bounded and not bounded by the union of balls [10]. Figures 3(e) and 3(f) illustrate a void and a pocket in 2D. Figures 3(g) and 3(h) show how they are affected by the growth model.

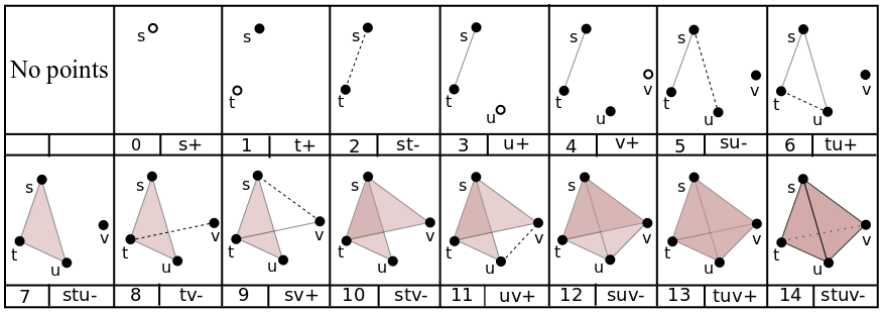


Fig. 4 A filtration generated by inserting simplices in a particular order. A k-simplex corresponds to an overlap of $k+1$ balls. For each simplex, the box in the bottom left shows the rank/arrival time of the simplex, the box in the bottom right shows the simplex along with it's behaviour. The '+' implies that it is a positive simplex (creator) and the '-' implies that it is a negative simplex (destroyer). For example, the triangle '$tuv+$' creates a void and tetrahedron '$stuv-$' destroys it. The persistence of this void is therefore 1.

Topological persistence. The *boundary* of a triangle consists of its edge faces. The boundary of a collection of triangles is the formal sum of boundary edges of the individual triangles, where addition is performed modulo 2. A 2-*cycle* (two-dimensional cycle) is a collection of triangles whose boundary is empty. Cycles of other dimensions are defined similarly. A void is represented by a 2-cycle. The alpha complex K helps represent and track voids via the growth process. A void is said to be created when the last triangle in the 2-cycle is inserted into the filtration and it is destroyed when the volume that it occupies is filled by the last tetrahedron. *Topological persistence* of a 2-cycle measures its lifetime ($k \geq 0$) in a filtration [14]. It is equal to the difference between the α-values when the cycle is created and destroyed. Given a filtration, the persistence of cycles can be computed efficiently [14]. The insertion of every simplex either creates a cycle or destroys a lower dimensional cycle. The persistence value associated with a simplex is equal to the topological persistence of the corresponding cycle. Figure 4 illustrates creation and destruction of cycles in a filtration of a small simplicial complex.

A void is represented by a 2-cycle and hence it has a well-defined creator and destroyer. However, this is not the case for a pocket, which may not necessarily be created as a result of a simplex insertion. This is because, initially $K = \emptyset$ and hence $D - K = D$ is a single connected component. This component corresponds to a pocket and has no creator. Hence, we cannot directly apply the notion of persistence to measure pockets. The notion of persistence of a void intuitively captures the volume of the void in terms of the range of α-values. To be consistent, we use a similar notion for pockets as well. We fix a value of $\alpha = \alpha_0$ and define the birth time of a pocket that has no creator to be equal to α_0. The α-value when the pocket interior is filled corresponds to its destruction time. Thus, similar to voids, the persistence of a pocket is equal to its lifetime and approximates the volume of the pocket in terms of the range of α-values.

3 Robust cavities and their computation

We introduce a notion of robust cavities based on two parameters, one local and another global. The local parameter is referred to as stability and the global parameter is specified by topological persistence. In order to simplify the description, we assume that the cavities are computed for the α-complex corresponding to $\alpha = 0$. However, the proposed definitions, methods, and subsequent analysis are valid for all values of α.

3.1 ε-stable and π-persistent cavities

Consider the interval $[-\varepsilon, \varepsilon]$ of α values, where $\varepsilon \geq 0$. A cavity is called an ε-*stable* cavity if it remains a single connected cavity within all α-complexes for α values in the range $[-\varepsilon, \varepsilon]$. In other words, using the lifetime terminology, the cavity is

born, possibly split into multiple components, and destroyed at α-values that lie strictly outside of this interval. A cavity is π-*persistent* if its topological persistence is greater than π *i.e.*, the cavity size measured in terms of its lifetime is greater than π. The persistence of pockets is defined in this case by setting $\alpha_0 = -\varepsilon$. Combining the two notions of robustness, we call a cavity to be (ε,π)-*stable* if it is both ε-stable and π-persistent.

The above definitions help measure the stability of the cavities when the atom radii are perturbed by a small value. The local parameter considers perturbation within a small interval centered at the α-value of interest whereas the global parameter measures the size of the cavity in terms of its lifetime in the filtration. Cavities of interest may often not be stable with respect to both notions. For example, a large sized cavity (π-persistent for some large π) may be born within the interval $[-\varepsilon,\varepsilon]$. However, note that a small perturbation in the radii of atoms that line the surface of the cavity could result in an earlier birth time, hence making the cavity to be ε-stable. We aim to extract all cavities that are either stable as is or can be made stable via a small perturbation.

3.2 Computing (ε,π)-cavities

The location of the atoms that constitute a protein molecule together with their van der Waals radii is obtained from the protein data bank in pdb format. Given ε and π, we compute the set of (ε,π)-stable cavities as follows.

1. Compute the weighted Delaunay triangulation of the input [19]. The atom centres form the set of points that are weighted using their van der Waals radii.
2. Build the alpha complex [7], which is a filtration of the weighted Delaunay triangulation.
3. **Modify the filtration based on the value of ε.**
4. Compute the set of (ε,π)-stable cavities by identifying all cavities [10, 21] of the modified filtration at $\alpha = 0$, and retaining only those cavities that have persistence greater than π.

The key idea in our proposed method is a modification of the filtration (Step 3) in order to compute the set of stable cavities. The filtration of the weighted Delaunay triangulation as defined by the α-value provides an explicit representation of the birth / death times of each cavity and its evolution. We propose to alter the birth / death times of the cavities by modifying the filtration instead of directly modifying the radii of atoms that line the surface of the cavity. While the latter approach follows directly from the definition, it is cumbersome and computationally inefficient. For example, varying the radii without explicit control may lead to changes in the triangulation and the alpha complex. These changes need to be explicitly tracked, else they may lead to inconsistencies between the alpha complex that represents the molecule and the space-fill model. Resolving such inconsistencies would

necessitate the re-computation of all representations. On the other hand, the former approach is simpler and computationally efficient.

Modifying the filtration. We now describe this step in detail. One or more simplices are inserted to obtain a rank $i+1$ simplicial complex from a rank i simplicial complex in the filtration. Higher ranks correspond to higher values of α. The topology of voids and pockets may change when the simplices are inserted. In particular, consider a triangle whose insertion changes the topology of a cavity. When this cavity is a void, the triangle splits the void into two voids (C1). In case of pockets, the insertion of the triangle could cause one of the following:

(C2) split the pocket into two pockets,
(C3) close one mouth of the pocket (for pockets with more than one mouth),
(C4) split the pocket into a pocket and a void, or
(C5) destroy the pocket and create a new void.

On the other hand, the insertion of a tetrahedron always destroys a void. These topology changes may be avoided by delaying the insertion of the simplices that cause a change in the topology of the cavities.

Let K^j and K^l be the alpha complexes corresponding to $\alpha = -\varepsilon$ and $\alpha = \varepsilon$ respectively. Consider the set of simplices, Σ, inserted into the filtration for values of α in the range $[-\varepsilon, \varepsilon]$. Let $\Sigma_t \subset \Sigma$ be a subset of the set of triangles that modifies the topology of a cavity and $\Sigma_T \subset \Sigma$ be the set of all tetrahedra in Σ. As mentioned earlier, our goal is to selectively alter the radii of atoms by altering the birth / death times of a cavity. In order to accomplish this, we delay the insertion of a select few simplices $\sigma_i \in \Sigma_t$ and all simplices in Σ_T such that $\sigma_i \notin K^j$ but $\sigma_i \in K^l$, where $K^j \subset K^l \subset D$. This delay corresponds to change in radii of the corresponding atoms enclosing the cavity.

Identifying the set Σ_t. The set Σ_t consists of all triangles that satisfies conditions C1, C2 and C3, while triangles that satisfy conditions C4 or C5 are optionally inserted into Σ_t. This is because a triangle that satisfies condition C4 or C5 creates a new void destroying the existing pocket. Depending on whether the perturbation decreases or increases the radii of the corresponding atoms, both the original pocket and the new void can be considered to be stable respectively.

Delayed simplex insertion. Simplicial complexes in the filtration of the weighted Delaunay triangulation and the order of simplices that are inserted to generate the filtration satisfy several containment and incidence properties. These properties should be satisfied for the modified filtration as well. Towards this, we propose a conservative but computationally efficient approach to modify the filtration:

1. Move all tetrahedra in Σ_T to the end of the filtration. All such tetrahedra are present in D but not in any $K^i \subset D$.
2. For each triangle in Σ_t, find its incident tetrahedra τ_1, τ_2.
3. Delay the insertion of the triangle and the two tetrahedra, τ_1 and τ_2, to the end of the filtration.

The above modification is illustrated using a 2D analogue in Figure 5. Consider a void split into two as shown in Figure 5(a). Assume that the highlighted edge

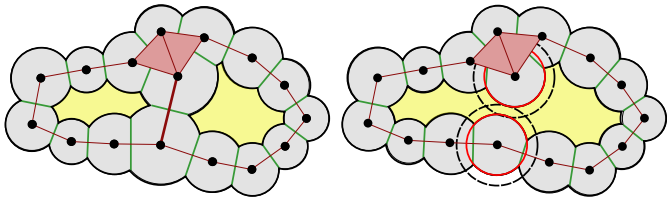


Fig. 5 2D illustration of simplex insertion causing a void to split. **(a)** Voids occur near to each other and the edge that splits the single void into two. **(b)** The two voids merge into one if the simplex insertion is delayed. The modified radii corresponding to the delay is highlighted in red.

(triangle in 3D) is inserted into the alpha complex for α lying in the interval $[-\varepsilon, \varepsilon]$. Further, it also satisfies the criterion that it bounds two different voids. So it becomes a candidate for delayed insertion. We move the edge to the end of the filtration, which means that it does not belong to the alpha complex as shown in Figure 5(b). Also the radii of atoms centered at the end points of the edge (triangle in 3D) are decreased accordingly. Selective modification of the radii of a specific set of atoms is hence achieved in a controlled manner.

After the filtration is modified, we recompute the set of cavities at $\alpha = 0$, which correspond to the set of ε-stable cavities. From this set, we retain cavities having persistence greater than π to obtain the set of (ε, π)-stable cavities. Note that the persistence is computed with respect to the original filtration.

Discussion. While computing stable cavities, our technique creates a single large stable cavity from two nearby smaller cavities. Larger cavities are more relevant for the study of stability of the molecules than smaller cavities. Biologists are therefore interested in identifying such structures in order to fill them. Therefore, in the presence of uncertainty, we choose to create a single large cavity instead of retaining the two smaller cavities.

Time complexity. Let m be the number of simplices in the Delaunay triangulation of the input protein having n atoms, $m = O(n^2)$. Computing the set of cavities takes $O(m\alpha(m))$ time using the union-find data structure. Here, α is the inverse Ackermann function. Given ε, Σ_t and Σ_T are computed in $O(m)$ time using a sequential search over the filtration. Identifying the set of tetrahedra incident on triangles in Σ_t, and moving all the simplices to the end of the filtration takes $O(m)$ time. Thus the time required to modify the filtration is $O(m\alpha(m))$.

3.3 Implementation notes

The filtration obtained from the alpha complex (Step 2) is stored as a list. The index of each simplex in this list represents the rank of that simplex. For each triangle, we additionally store the indices of the incident tetrahedra. For a given ε, we first compute the ranks k_1 and k_2 of the alpha complex at $\alpha = -\varepsilon$ and $\alpha = \varepsilon$, respectively,

and then we compute the set of pockets at rank k_2 [10]. Each pocket is stored as a set of tetrahedra and each tetrahedron has an entry that stores its parent pocket index.

Next, the set Σ_t and Σ_T are computed. The set Σ_T is simply the set of all tetrahedra σ_k having rank $k_1 \leq k \leq k_2$. In order to identify triangles that splits a void into two, it is sufficient to track the connected components of the complement of the α-complex. This is however not true for triangles that modify the topology of a pocket. The addition of such a triangle may not change the number of components present in the complement of the α-complex. In order to identify such triangles, starting from the set of pockets computed at rank k_2, we traverse the filtration in reverse from rank k_2 to rank k_1, and explicitly track the change in topology of the set of pockets. We also create and store the set $\Sigma_T^{'}$, which consists of the incident tetrahedra of all triangles present in Σ_t.

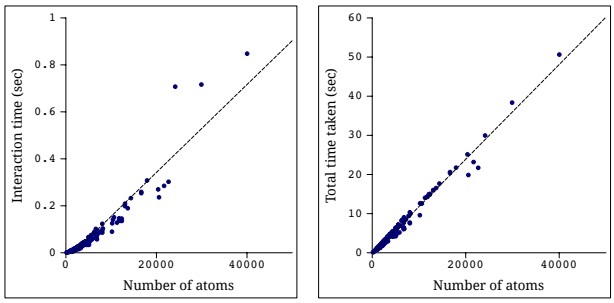
Let $\Sigma_S = \Sigma_t \bigcup \Sigma_T \bigcup \Sigma_T^{'}$. The simplices in Σ_S are sorted in the increasing order of their ranks. Instead of explicitly moving these simplices to the end of the filtration, we perform an implicit move. These simplicies are marked as invalid within the list that represents the original filtration. The new filtration is obtained by traversing the original filtration, ignoring the invalid simplicies, followed by traversing the simplices in Σ_S. An advantage of using this approach is that, when the value of ε is changed, it is easy to revert to the original filtration and recompute the new filtration.

4 Experimental results

We have developed a software tool RobustCavities that interactively computes the set of stable cavities. The values of ε and π can be specified interactively by the user using a slider widget present in the tool. Following is a brief list of features supported in RobustCavities :

- Computation of stable cavities in a protein for specified values of ε, π and α.
- Computation of volume and surface area of cavities.
- Visualization and interactive exploration of cavities with support for multiple rendering modes and colormaps.
- Export cavities with the relevant biochemical context in order to be used in PyMOL [5]. In particular, we support skin mesh [4], union of balls, and tetrahedral representation of the detected cavities. We also provide Python scripts which allow users to load these representations using different colormaps in PyMOL.

We first report experimental results that demonstrate the efficiency of our technique. We then present various examples of stable cavities present in different protein molecules. Finally, we demonstrate the utility of our technique in identifying potential channels and pockets in protein molecules. All experiments were performed on a workstation with a 8-core 2GHz Intel Xeon processor, 16 GB RAM, and an NVidia GTX 600Ti graphics card.



(a) (b)

Fig. 6 (a) Graph showing variations in the interaction time with respect to varying number of atoms. **(b)** Graph showing variations in the total time with respect to varying number of atoms.

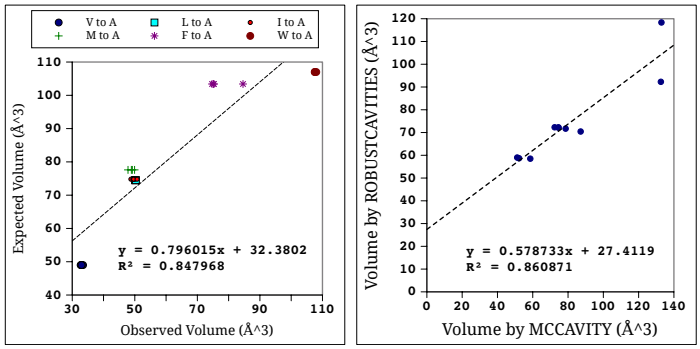
4.1 Performance and validation

Efficiency. In order to test the efficiency of our technique, we computed the set of robust cavities of over 200 proteins having number of atoms ranging from 184 to 40,026. The value of $\alpha = 0$, $\varepsilon = 1$ and $\pi = 0.01$ was used in these experiments.

We first measure the *interaction time*, which is equal to the time taken to modify the filtration and to identify the set of cavities from this modified filtration.This is essentially the time taken to update the set of cavities once the user changes the parameter values. Figure 6(a) plots the interaction time against the number of atoms in the protein. Note that even for very large proteins having 40,000 atoms, the set of robust cavities are computed within a second. Also note the near-linear behaviour of the interaction time.

Next, we measure the total time taken to compute the set of robust cavities. This includes time to compute the original filtration in addition to the interaction time. Figure 6(b) plots the variation of total time taken against the number of atoms in the protein. Even though the total time is significantly greater than the interaction time, it still requires only around 50 seconds for a protein with 40,000 atoms. Also, this performance is acceptable since the computation of the original filtration is a one-time operation done when loading the protein. Among the pre-processing steps, computing alpha complex is the most time consuming step, which can be improved further by employing the method proposed by Mach and Koehl [25].

Validation of computed volumes. As mentioned earlier, RobustCavities also reports the volume and surface area of the cavities identified. Proteins adopt a variety of structures and it is known that the extent of packing of protein chain differs within the same structure as well as between different structures. Because of this reason the volume of the cavity which is created by converting a large residue to a small residue may not always be the same in different protein structures. To mitigate this problem we only examine completely buried cavities where it is known that proteins take up a close packed structure.



(a) (b)

Fig. 7 **(a)** Plot of the computed and actual volumes of the artificial voids that were generated using mutant models. **(b)** Comparison of normalized volumes computed using ROBUSTCAVITIES and MCCAVITY.

Different cavity volume computation methods may employ different molecular models resulting in a variation in the volumes that they report. We perform an additional normalization of the computed volumes using model mutants [3] to eliminate such variations. We use 28 different model mutants to create a set of artificial voids. We use the resulting volumes to compute a linear normalization function, as shown in Figure 7(a). The volumes computed using our computation are normalized as follows:

$$Volume = 0.8 \times ComputedVolume + 32.4.$$

The expected volume against which we compare our observed volume is the Voronoi volume of the cavity, which is created on a large to small residue substitution, averaged over many examples. Since we compare a specific large to small substitution in a protein with an averaged ideal value, there is a difference in the volumes, which is reflected as the constant in the above linear transformation.

In order to verify the correctness of the volumes computed by our software, we compare volume for some of these mutants to the volumes computed using MCCAVITY [3], see Figure 7(b). The graph shows normalized volumes and we observe that there is high correlation between the two sets of volumes. In the absence of an ideal normalizing function, the correlation coefficient helps determine if the volumes computed are consistent with data available from other methods.

4.2 Stable cavities and their properties

We now illustrate different examples of robust cavities identified using our software. Unless otherwise specified, we use values $\alpha = 0$, $\varepsilon = 1$ and $\pi = 0.01$ in following examples. Note that a value of $\varepsilon = 1$ is equivalent to a change of the radius of an atom by at most 0.2Å, which is within the resolution at which the input data is

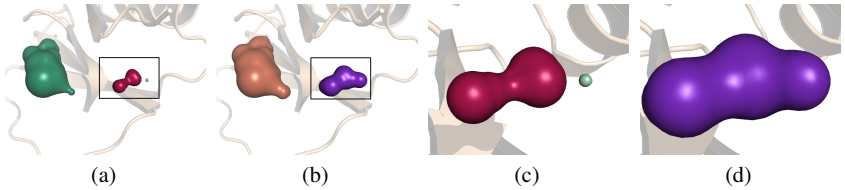


Fig. 8 Visualization of cavities in the protein 2CI2. **(a)** Cavities in the protein. **(b)** The set of $(1.0, 0.01)$-stable cavities. **(c)** Two of the nearby voids in the protein. **(d)** These cavities merge together resulting in a single stable void. Figures (c) and (d) are zoomed in for better context.

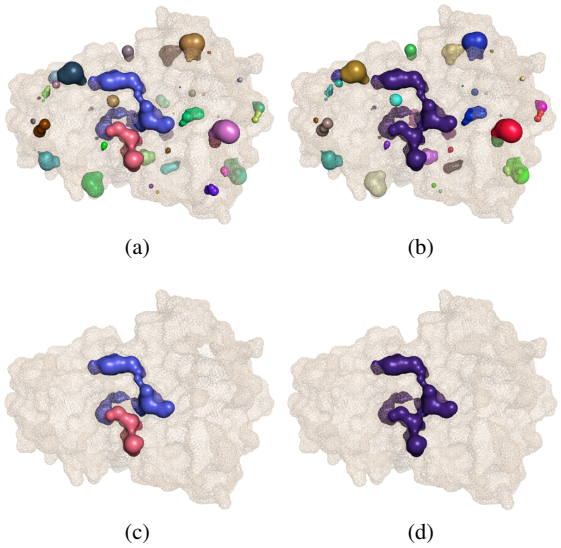


Fig. 9 Visualization of cavities in the protein 4B87. The molecular surface is shown for context. **(a)** The set of cavities in the protein. Number of cavities = 72. **(b)** The set of $(1.0, 0.01)$-stable cavities. Number of stable cavities = 70. **(c)** Two of the nearby pockets in the protein. **(d)** These pockets merge together resulting in a single stable pocket.

available and hence within the tolerance threshold. In the following, we refer to a protein by specifying its PDB ID from the Protein Data Bank [2].

Figure 8 shows protein 2CI2, which has three cavities. Two of the cavities are voids, and are quite close as can be seen in the Figure 8(a). After modification, these two voids are detected as a single void as shown in Figure 8(b). Figures 8(c) and 8(d) show a close-up view of the two merging voids. Similarly, Figures 9 and 10 show stable cavities for proteins 4B87 and 1DKF, respectively. In both these proteins, we observe two significant pockets merging into a single stable pocket after filtration modification.

Properties of stable cavities. Figures 11 and 12 plot the number and volume of (ε, π)-stable cavities for various values of ε. Note that increasing the value of ε implies that cavities from a wider range of α-values are considered. This could potentially increase the number of ε-stable cavities. However, such cavities usually

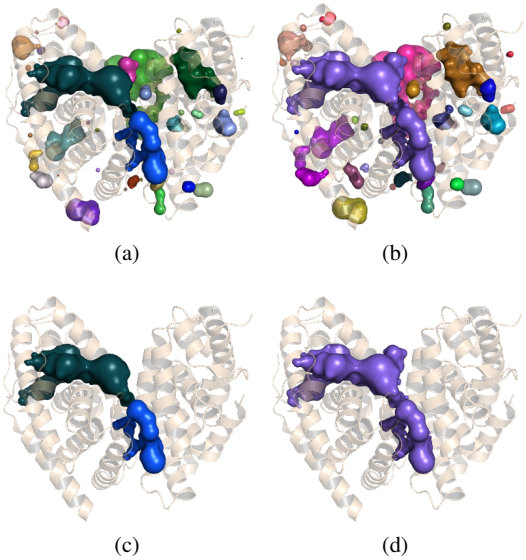
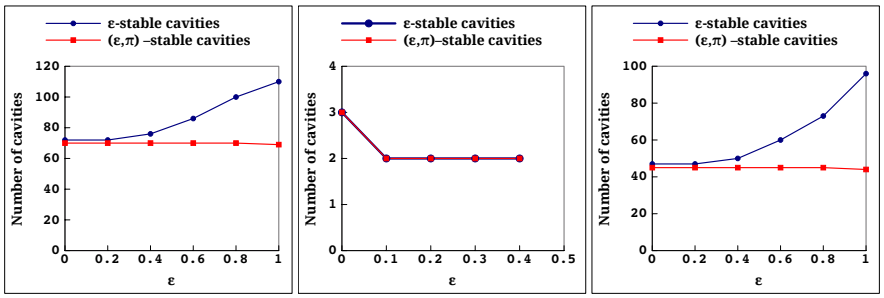


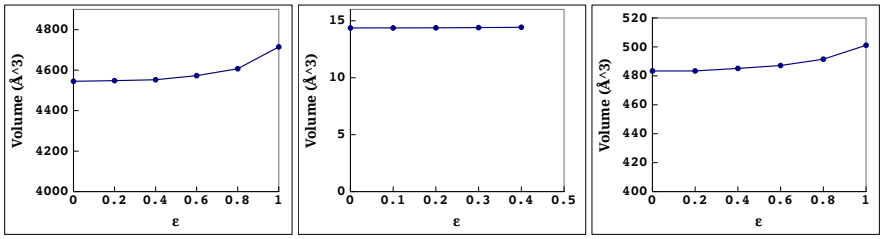
Fig. 10 Visualization of cavities in the protein 1DKF. A cartoon representation of the secondary structure is shown for context. **(a)** All cavities in the protein. Number of cavities = 56. **(b)** The set of $(1.0, 0.01)$-stable cavities. Number of stable cavities = 39. **(c)** Two of the nearby pockets in the protein. **(d)** These pockets merge together resulting in a single stable pocket.

have low persistence and are therefore not (ε, π)-stable. The total volume of all stable cavities increases marginally ($< 1\%$) with increasing ε. The merging of two nearby cavities into a single stable cavity does not effect the total volume. However, volumes of individual cavities could change drastically. We have observed that the volume of a stable void is approximately equal to the sum of the volumes of the original voids.



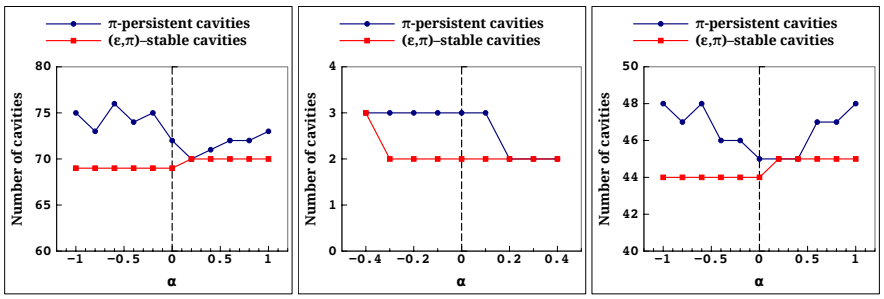
(a) Protein 4HHB (b) Protein 2CI2 (c) Protein 4B87

Fig. 11 Graphs showing the variation of the number of cavities with varying ε. Note that there is an increase in the number of ε-stable cavities as we consider a larger interval. But, the number of (ε, π) cavities is less than or equal to the original number of cavities.



(a) Protein 4HHB (b) Protein 2CI2 (c) Protein 4B87

Fig. 12 Graphs showing the variation of the total volume of cavities with varying ε. The increase in total volume is insignificant.



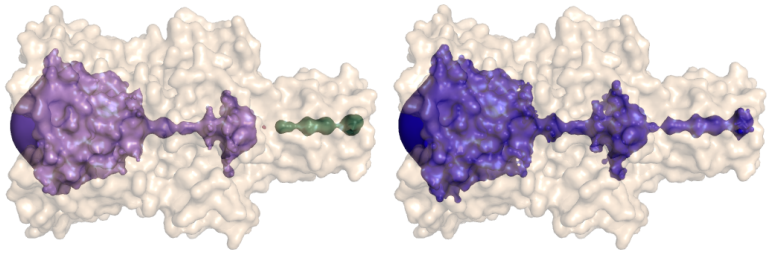
(a) Protein 4HHB (b) Protein 2CI2 (c) Protein 4B87

Fig. 13 Graphs showing the variation of the number of cavities for constant ε and varying α. The number of (ε,π)-stable cavities does not vary much but there is significant variation in the number of π-persistent cavities.

Robustness of (ε,π)-stable cavities. Figure 13 plots the number of (ε,π)-stable cavities and π-persistent cavities for various values of α for $\varepsilon = 1.0$ in case of 4HHB and 4B87 and $\varepsilon = 0.3$ in case of 2CI2. Note that the number of (ε,π)-stable cavities is mostly constant, while there is a significant variation in the the number of π-persistent cavities. This is because, when using only persistence, even though small (noisy) voids are removed, a small change in the radius could change the number of voids. On the other hand, since our method adds in the additional constraint of stability, only the robust voids are retained.

4.3 Detecting potential channels and pockets in proteins

Due to experimental errors or changed protein conformation, a true pocket could be labelled as a void, or a pore (also referred to as through-channels) may be labelled as a disconnected pocket by a cavity detection algorithm. Existing software used



(a) 2OAR - Before (b) 2OAR - After

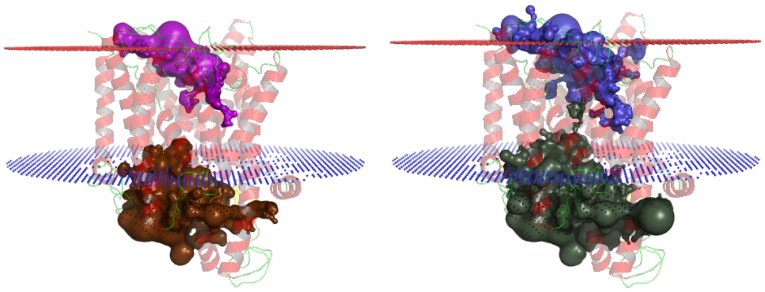
Fig. 14 Detection of potential channel in trans-membrane protein 2OAR. **(a)** At $\alpha = 0$, two pockets and a small void is detected. **(b)** After modification, these cavities merge together revealing the pore present in this trans-membrane protein.

for finding cavities in proteins fail to identify such cavities. However, since our technique is robust to errors (specified using ε), it is possible to detect such potential pockets and pores.

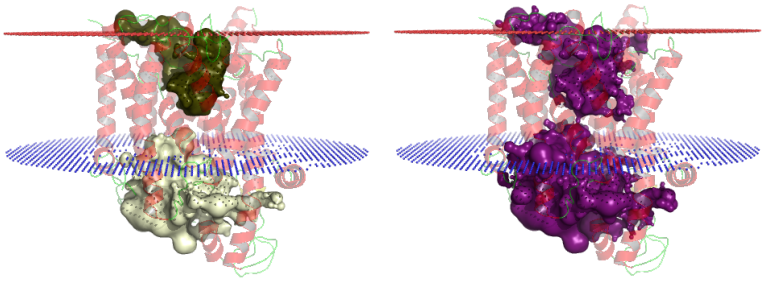
While the case of identifying potential pores (or potential channels) is taken care of by conditions C2 and C3 (refer Section 3.2), in order to identify pockets that appear as voids, we additionally delay insertion of triangles satisfying conditions C4 and C5. Thus stability of pockets is given preference over stability of voids, since pockets usually correspond to functionally important regions of proteins as they are accessible from the outside environment.

Figure 14 illustrates an example where we detect a potential channel in protein 2OAR. This is a trans-membrane protein with a known ion-channel going though it. By default, two pockets and a small void are detected instead of the ion-channel. Using RobustCavities with $\varepsilon = 1.4$ (which corresponds to maximum change of $0.41\mathring{A}$ in atomic radius), the void merges together with the two pockets to correctly identify this channel.

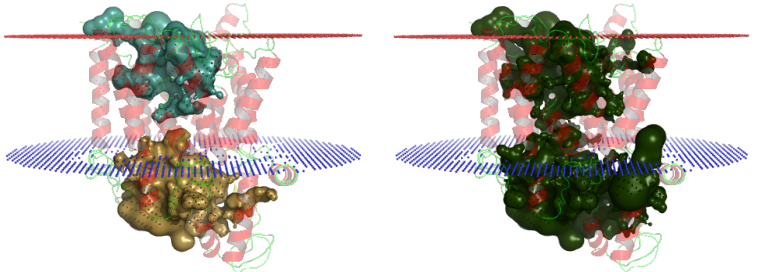
We now demostrate the utility of our potential channel detection technique using the example of translocase SecY protein [1]. We consider three of its structures – 1RHZ, 2YXQ and 2YXR. This unique transporter protein has its transmembrane channel plugged in its wild type conformation (1RHZ). This plug only opens when specific molecules need to be transported. We tried detecting a channel through the wild type 1RHZ structure. Even after modifying the filtration, all cavities remained stably disconnected as shown in Figure 15(b). To probe the mechanics and regulation of this transporter, researchers created a half and full plug deletion mutants of the same protein, labelled as 2YXQ and 2YXR respectively [26]. In 2YXQ, half of the plug region was deleted while in 2YXR the plug was deleted completely. Even after plug deletions, the protein compensates for the deletion and attains a tightly packed structure due to its dynamic nature; other methods still fail to detect a channel in this plug deletion mutant. However, after using RobustCavities with $\varepsilon = 1.5$ (which corresponds to maximum change of $0.48\mathring{A}$ in atomic radius) to modify filtration, we are able to identify the potential channels in these mutants as is shown in Figures 15(d) and 15(f). In summary, we find that the trans-membrane pore is not present in the wild type structure but becomes progressively larger in



(a) 1RHZ - Before (b) 1RHZ - After

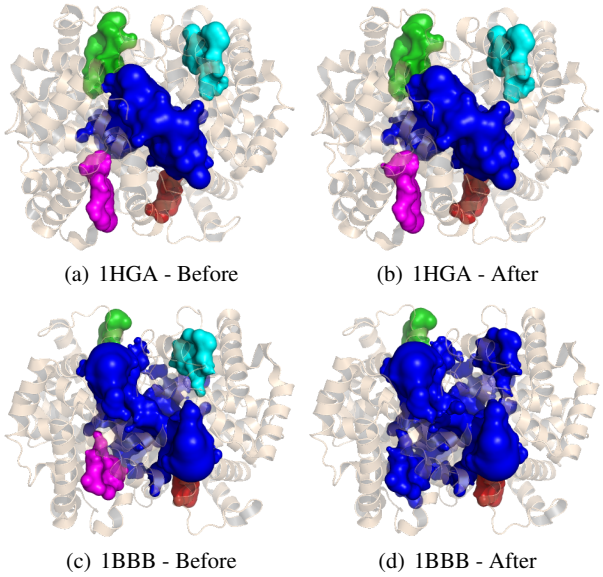


(c) 2YXQ - Before (d) 2YXQ - After



(e) 2YXR - Before (f) 2YXR - After

Fig. 15 The case study of protein translocase SecY. We show results for three structures of this protein viz. 1RHZ, 2YXQ and 2YXR. 1RHZ is the closed structure of this protein, while 2YXQ and 2YXR are mutants with half and full plug deletions respectively. In all the figures the membrane is shown as blue and red planes, where blue plane corresponds to intra-cellular region while red plane denotes extra-cellular region. **(a)** The two pockets in 1RHZ at $\alpha = 0$. **(b)** These two pockets remain disconnected even after modifying filtration. **(c)** The two pockets of interest in 2YXQ at $\alpha = 0$. Please note that these pockets are slightly larger than the pockets in closed structure 1RHZ. **(d)** The pockets merge to reveal transmembrane pore in this mutant. **(e)** The two relevant pockets of in 2YXR at $\alpha = 0$. It can be observed that these pockets are larger than the pockets detected in both 1RHZ and 2YXQ. **(f)** In this mutant too, the two pockets merge to reveal transmembrane connection.



(a) 1HGA - Before (b) 1HGA - After

(c) 1BBB - Before (d) 1BBB - After

Fig. 16 Cavity structures in two states of Hemoglobin. In all the images, central cavity is shown in blue color, while red, cyan, green and magenta colors are used for heme sites in chains A, B, C and D, respectively. The first row in the figure is for 1HGA which is low affinity T state, while second row corresponds to high affinity R state (1BBB) of Hemoglobin. **(a)** The central cavity and four heme cavities in low affinity state of Hemoglobin. **(b)** Even after applying modification, the heme sites don't merge with central cavity. **(c)** The central cavity and four heme cavities in high affinity state of Hemoglobin. **(d)** The heme sites in chains B and D merge with the central cavity after application of RobustCavities .

the half plug and full plug deletions mutants. This is consistent with experimental data which show that plug deletion leads to increased translocation of proteins with defective signal sequences as well as small molecules and increase the propensity for the channel to adopt an open state.

In our final example shown in Figure 16, we study cavity structures in low and high affinity states of the protein Hemoglobin. As shown in Figures 16(a) and 16(c), both low and high affinity structures consist of four heme sites surrounding a central cavity. Also, all these cavities are disconnected at $\alpha = 0$ in both the structures. However, after modifying filtration, while the topology of cavities in low affinity structure remains unchanged (Figure 16(b)), two heme sites in chains B and D of high affinity structure merge with central cavity (Figure 16(d)). It is known that Oxygen binding to heme in hemoglobin causes a conformational change in the rest of the structure which leads to an increase in oxygen binding affinity. The binding results in the conformation transition from tense form (low affinity T state) to relaxed form (high affinity R state). This important conformational change is being correctly captured by the change in topology of the cavities of the R state (Figure 16(d)).

5 Conclusions

We have defined a novel notion of robust cavities that is insensitive to the perturbation of the atomic radii. Robust cavities are computed via a controlled modification of the filtration that represents the molecule and its cavities. Identifying robust cavities is important so that the biologist only targets these cavities in tedious mutation-based experiments. The method addresses the inaccuracies in the measurements of the radii by selectively varying the radii for a specific set of atoms. However the positional uncertainties which arise due to the motion of the molecules is not addressed.

We show several examples which demonstrates using visual evidence that small perturbations in the radii results in a larger and robust cavities. The value of ε used in these experiments is lower than the typical experimental error in crystallographic measurements. We also show the efficiency of our method which allows for interactive exploration of robust cavities with varying ε. Finally, we use our technique to identify robust pockets and pores in different trans-membrane proteins.

In future, we plan to further investigate the relationship between the perturbation in the atom radii corresponding to the delayed simplex insertion and the structural and functional properties of the protein. Future work also includes generalizing the framework to use empirically determined intervals of radii for each atom type and addressing the issue of biological implications of the method.

Acknowledgements. Talha Bin Masood was supported by Microsoft Corporation and Microsoft Research India under the Microsoft Research India PhD Fellowship Award. This work was supported in part by the Department of Science and Technology, India, under Grant SR/S3/EECE/0086/2012, the DST Center for Mathematical Biology, IISc, under Grant SR/S4/MS:799/12, the NYU School of Engineering, and NSF award CNS-1229185. We would like to thank Patrice Koehl for his suggestions and for sharing the source code of Proshape.

References

1. van den Berg, B., Clemons, W.M., Collinson, I., Modis, Y., Hartmann, E., Harrison, S.C., Rapoport, T.A.: X-ray structure of a protein-conducting channel. Nature **427**(6969), 36–44 (2003)
2. Berman, H., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T., Weissig, H., Shindyalov, I., Bourne, P.: The Protein Data Bank. Nucleic acids research **28**(1), 235–242 (2000)
3. Chakravarty, S., Bhinge, A., Varadarajan, R.: A procedure for detection and quantitation of cavity volumes in proteins. Journal of Biological Chemistry **277**(35), 31,345–31,353 (2002)
4. Cheng, H., Shi, X.: Quality mesh generation for molecular skin surfaces using restricted union of balls. Computational Geometry **42**(3), 196–206 (2009)
5. DeLano, W.: The pymol molecular graphics system. `http://www.pymol.org` (2002)
6. Dundas, J., Ouyang, Z., Tseng, J., Binkowski, A., Turpaz, Y., Liang, J.: CASTp: computed atlas of surface topography of proteins with structural and topographical mapping of functionally annotated residues. Nucleic acids research **34**(2), W116–W118 (2006)
7. Edelsbrunner, H.: Weighted alpha shapes. University of Illinois at Urbana-Champaign, Department of Computer Science (1992)

8. Edelsbrunner, H.: Biological applications of computational topology. In: J.E. Goodman, J. O'Rourke (eds.) Handbook of Discrete and Computational Geometry, pp. 1395–1412. CRC Press (2004)
9. Edelsbrunner, H.: Computational Topology. An Introduction. Amer. Math. Soc. (2010)
10. Edelsbrunner, H., Facello, M., Liang, J.: On the definition and the construction of pockets in macromolecules. Discrete Applied Mathematics **88**(1), 83–102 (1998)
11. Edelsbrunner, H., Fu, P.: Measuring space filling diagrams and voids. Tech. rep., UIUC-BI-MB-94-01, Beckman Inst., Univ. Illinois, Urbana, Illinois (1994)
12. Edelsbrunner, H., Kirkpatrick, D., Seidel, R.: On the shape of a set of points in the plane. IEEE Transactions on Information Theory **29**(4), 551–559 (1983)
13. Edelsbrunner, H., Koehl, P.: The geometry of biomolecular solvation. Combinatorial & Computational Geometry **52**, 243–275 (2005)
14. Edelsbrunner, H., Letscher, D., Zomorodian, A.: Topological persistence and simplification. Discrete & Computational Geometry **28**(4), 511–533 (2002)
15. Edelsbrunner, H., Mücke, E.: Three-dimensional alpha shapes. ACM Transactions on Graphics (TOG) **13**(1), 43–72 (1994)
16. Kim, D.S., Cho, Y., Sugihara, K., Ryu, J., Kim, D.: Three-dimensional beta-shapes and beta-complexes via quasi-triangulation. Computer-Aided Design **42**(10), 911–929 (2010)
17. Kim, D.S., Ryu, J., Shin, H., Cho, Y.: Beta-decomposition for the volume and area of the union of three-dimensional balls and their offsets. Journal of Computational Chemistry (2012)
18. Kim, D.S., Sugihara, K.: Tunnels and voids in molecules via voronoi diagram. In: Proc. Symp. Voronoi Diagrams in Science and Engineering (ISVD), pp. 138–143 (2012)
19. Koehl, P., Levitt, M., Edelsbrunner, H.: Proshape: understanding the shape of protein structures. Software at biogeometry.duke.edu/software/proshape (2004)
20. Krone, M., Falk, M., Rehm, S., Pleiss, J., Ertl, T.: Interactive exploration of protein cavities. In: Computer Graphics Forum, vol. 30, pp. 673–682 (2011)
21. Liang, J., Edelsbrunner, H., Fu, P., Sudhakar, P., Subramaniam, S.: Analytical shape computation of macromolecules: II. inaccessible cavities in proteins. Proteins Structure Function and Genetics **33**(1), 18–29 (1998)
22. Liang, J., Edelsbrunner, H., Woodward, C.: Anatomy of protein pockets and cavities. Protein Science **7**(9), 1884–1897 (1998)
23. Lindow, N., Baum, D., Bondar, A., Hege, H.: Dynamic channels in biomolecular systems: Path analysis and visualization. In: Proc. IEEE Symposium on Biological Data Visualization (BioVis), pp. 99–106 (2012)
24. Lindow, N., Baum, D., Hege, H.: Voronoi-based extraction and visualization of molecular paths. Visualization and Computer Graphics, IEEE Transactions on **17**(12), 2025–2034 (2011)
25. Mach, P., Koehl, P.: Geometric measures of large biomolecules: Surface, volume, and pockets. Journal of Computational Chemistry **32**(14), 3023–3038 (2011)
26. Minor, D.L.: Puzzle plugged by protein pore plasticity. Molecular cell **26**(4), 459–460 (2007)
27. Munkres, J.: Elements of Algebraic Topology, vol. 2. Addison-Wesley Menlo Park, CA (1984)
28. Parulek, J., Turkay, C., Reuter, N., Viola, I.: Implicit surfaces for interactive graph based cavity analysis of molecular simulations. In: Biological Data Visualization (BioVis), 2012 IEEE Symposium on, pp. 115–122 (2012)
29. Petřek, M., Košinová, P., Koča, J., Otyepka, M.: MOLE: A Voronoi diagram-based explorer of molecular channels, pores, and tunnels. Structure **15**(11), 1357–1363 (2007)
30. Petřek, M., Otyepka, M., Banáš, P., Košinová, P., Koča, J., Damborskỳ, J.: Caver: a new tool to explore routes from protein clefts, pockets and cavities. BMC bioinformatics **7**(1), 316 (2006)
31. Sridharamurthy, R., Doraiswamy, H., Patel, S., Varadarajan, R., Natarajan, V.: Extraction of robust voids and pockets in proteins. In: EuroVis-Short Papers, pp. 67–71. The Eurographics Association (2013)
32. Till, M.S., Ullmann, G.M.: Mcvol-a program for calculating protein volumes and identifying cavities by a monte carlo algorithm. Journal of molecular modeling **16**(3), 419–429 (2010)