RESEARCH

Interactive Exploration of Ligand Transportation through Protein Tunnels

Katarína Furmanová[1*], Miroslava Jarešová[1], Jan Byška[1,2], Adam Jurčík[1], Július Parulek[2], Helwig Hauser[2] and Barbora Kozlíková[1]

Abstract

Background: Protein structures and their interaction with ligands have been in the focus of biochemistry and structural biology research for decades. The transportation of ligand into the protein active site is often complex process, driven by geometric and physico-chemical properties, which renders the ligand path full of jitter and impasses. This prevents understanding of the ligand transportation and reasoning behind its behavior along the path.

Results: To address the domain expert needs we design an explorative visualization solution based on a multi-scale simplification model. It helps to navigate the user to the most interesting parts of the ligand trajectory by exploring different parameters of the ligand and its movement, such as its distance to the active site, changes of amino acids lining the ligand, or ligand "stuckness". The process is supported by three linked views – 3D representation of the simplified trajectory, scatterplot matrix, and bar charts with line representation of ligand-lining amino acids.

Conclusions: The usage of our tool is demonstrated on simulation of molecular dynamics simulation generated by the domain experts. The tool was tested by the domain experts from protein engineering and the results confirm that it helps to navigate the user to the most interesting parts of the ligand trajectory and to understand the ligand behavior.

Keywords: Molecular Visualization; Bioinformatics Visualization; Computational Proteomics

Background

The study of reaction processes between different types of molecules has been an important research problem already for decades. A proper understanding of the processes occurring when two or more molecules react helps in the design of new chemical matters, e.g., in drug design or protein engineering. Here, the researchers aim to combine a protein with a given ligand in order to design a new drug or to change protein properties and their function. In these particular cases the ligand has to be transported from the outer solvent to the protein active site where the chemical reaction between the ligand and the amino acids surrounding the active site takes place. The consecutive reaction then changes the composition and properties of both molecules. In protein engineering, for example, the goal is to alter the protein properties so that the new protein is, e.g., more stable and resistant to organic cosolvents [1].

The design complexity of such reactions lies namely in the transportation of the ligand to the protein active site. As the active site is usually buried deeply in the protein structure and thus inaccessible directly from its surface, the ligand has to find a suitable transportation path through the protein structure. This process, called molecular docking, is very complex, lengthy, and its analysis is heavy on computational resources. Therefore, researchers aim at solutions that simplify and ease the analysis for proper ligand binding. Currently available solutions often focus on detection of possible ligand transportation paths through the protein, called tunnels. These solutions are mostly based on the geometric analysis, e.g., CAVER [2], MOLE [3], or MolAxis [4]. Figure 1 shows an example of small ligand passing through a tunnel computed by CAVER algorithm and visualized using CAVER Analyst [5]. The tunnel is colored with respect to the hydrophobicity of the surrounding amino acids and the geometric bottleneck of the tunnel is clearly visible.

Other approaches, such as MoMA-LigPath [6], aim at simulating the ligand transportation itself. Nevertheless, simulating the ligand docking using current computational approaches is still a challenging problem. There are several available variants of molecular

[*]Correspondence: furmanova@mail.muni.cz
[1]Masaryk University, Brno, Czech Republic
Full list of author information is available at the end of the article

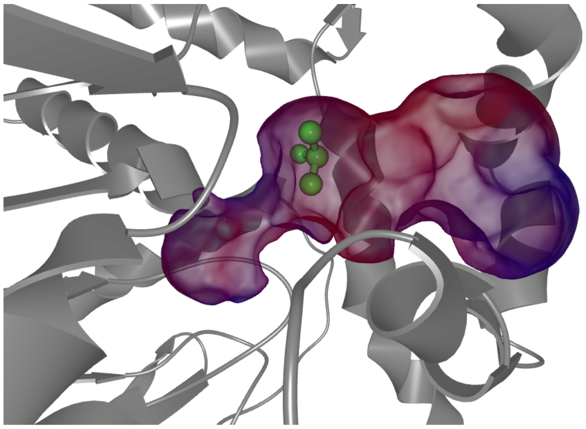


Figure 1 Small ligand (green) passing through a tunnel represented by the transparent surface. The surface encloses the available empty space along the tunnel centerline and is colored with respect to the hydrophobicity of the surrounding amino acids (hydrophobic – red, hydrophylic – blue, neutral – violet). The active site is located in the left ending of the tunnel.

simulation methods devised specifically to this problem. Among these, Steered Molecular Dynamics [7] and Random Acceleration Molecular Dynamics [8] are able to simulate the ligand binding. Such simulations produce a large amount of data containing the ligand movements that are to be explored by the domain expert. As the length of these simulations reaches often hundreds of thousands of time steps, it becomes impossible for the domain experts to visualize and observe the ligand movement in a frame-by-frame manner. Moreover, the simulation often contains movements that are irrelevant to the ligand binding. For example, a significant portion of the simulation the ligand usually spends outside the protein searching for a proper tunnel gorge to enter the molecule. It also often happens that the ligand enters the molecule via a wrong tunnel and is then evicted from the molecule and searches for the entrance repeatedly. Therefore, the "true" active site entering can be in fact captured in a smaller subset of the original sequence.

In this paper, we propose a new visual analysis system that addresses the aforementioned challenges, i.e., the transportation of a ligand to the active site. We aim to provide the domain experts with a tool for intuitive and interactive exploration of already captured molecular dynamics simulation containing the process of ligand binding. Using our proposed solution the users are able to distinguish between the parts of the simulation where the ligand searches for the proper path to the active site (searches the tunnel gorge or enters the wrong tunnel) and the part where the ligand finally leads to and reaches the active site. Furthermore, our solution introduces a method for simplification of the original scattered ligand trajectory. We suggest the automatic simplification of the trajectory which can be further manually adjusted, i.e., selected parts of the trajectory can be further simplified. So the trajectory can be simplified in a hierarchical manner and further interactively explored using the other proposed views. One of these views is the 3D visualization where the simplified trajectory can be displayed in the spatial context of the whole molecule. The trajectory can be color-coded according to different criteria and selections performed in the other views are highlighted by a thick tube. Another available view is the scatterplot matrix which helps to explore different properties of the ligand along its trajectory, e.g., the direction or distance to the active site. Selections performed in these scatterplots are highlighted not only in the 3D view but also in another proposed view, the bar chart representing the temporal changes of selected ligand properties. This view enables to aggregate the time steps to better reveal the trends in ligand behavior. The bar chart is equipped with the line representation of amino acids lining the ligand along its trajectory. This helps the user to reveal the closest amino acids directly influencing the ligand behavior in different time steps.

Our proposed solution was thoroughly tested and evaluated by the domain experts from protein engineering area. The evaluation was performed on several simulations captured and analyzed before using the traditional approach (manual exploration of ligand in individual time steps). Therefore, we also describe the process of using our solution when exploring these molecular dynamics simulations and summarize the feedback from the domain experts.

Related Work

The interactive exploration of ligand paths leading through protein structures is a complex problem that requires an elaborate analysis of existing and convenient approaches. As our system addresses different areas, this section will be divided with respect to these areas as well. Studying the ligand transportation path is tightly related to the analysis of its trajectory. Thus, the first part will cover the topic related to the trajectory analysis, simplification, and visualization. Then the existing representations of tunnels and their surrounding amino acids will be addressed.

Trajectory Analysis, Simplification, and Visualization

When searching for the most appropriate trajectory simplification method, the taxonomy of different movement patterns introduced by Dodge et al. [9] can help to categorize the type of motion and to propose an appropriate solution for the analysis and simplification.

Then, Andrienko et al. [10] describe a legacy of tools and approaches to analyze trajectory data. With respect to the trajectory simplification, they introduce two approaches to data abstraction representing a necessary step to achieve the reduction of the data in order to achieve informative visualizations. The first approach is characterized by omitting unnecessary positions and segments, while the second exploits data subsampling. On the other hand, they state that there is no general method to sample trajectories. Thus the suitability of particular sampling method is deduced from what information is considered important. A recent work of Vrotsou et al. [11] introduce a systematic stepwise methodology for trajectory simplification with emphasis on visual analysis. Even though they primarily developed the tool to analyze three dimensional trajectories, it is not limited to them only.

In the field of molecular biology as well as systems biology, there are several examples of methods focusing on trajectory analysis. Bidmon et al. [12] present an abstract way of identification and visualization of solvent molecules' pathways within molecular dynamics. In comparison with the previous solutions, their approach preserves valuable information on the directions and velocities of water molecules routing along these paths. Another approach that describes a guidance through a complex simulation trajectories in systems biology is presented by Luboschik et al. [13]. The method addresses biochemical reaction networks and aims to provide the users with a tool for investigating the overall behavior of a modeled system and detailed behavior at the same time.

Visualization of Cavity and Tunnel Features

Phillips et al. [14] propose a method to quantitatively estimate molecular features, e.g., volume and surface areas, via a ray-casting technique. This involves the computation of cavities. Lindow et al. [15] introduce a technique that allows to extract significant paths from the molecules. In their approach the authors utilize a Voronoi diagram of spheres. Their final visualization is achieved by means of placing light sources on the extracted paths to enhance the presence of tunnels. Parulek et al. [16] exploit scatterplots to communicate the evolution of protein voids. In their later study [17] they also suggest to utilize amino acids physico-chemical properties related to cavities to help users to navigate through their occurrences. The visualization is then achieved in the focus and context manner. Lindow et al. [18] present an approach for visualization of temporal evolution of cavities in a temporal graph, which, to a certain extent, resembles our temporal tunnel model. In this work they focus on the interactive exploration of the dynamics of protein cavities that can form the transportation path for a ligand. They calculate and visualize the cavity volume and analyze the time-dependent changes of the cavity structure. The cavity dynamics is captured by rendering the cavities in a single image. The final visualization is achieved through the molecular surface representation colored according to time. Krone et al. [19] present a similar approach where they extract and track tunnels in MD simulations. They exploit temporal graphs to communicate the evolution of surface areas of tunnels. In our technique, we go one step further, where in addition, we incorporate the ligand-protein interaction information and provide users with the means to navigate through details of this interaction. Moreover, we analyze sequences of several thousands of frames, which is not the case in aforementioned approaches. Kozlíková et al. [20] propose a way to seamlessly visualize the geometry and shape of tunnels across MD simulation. Here, they aim at 3D visualization solely, which is not suitable way of exploring and understanding of thousands of simulation time-steps. Byška et al. [21] introduce an approach to interactively explore the tunnel objects in MD simulations. Similarly to our proposed approach, they visualize the time-varying tunnel as a profile graph that includes information on surrounding amino acids. Nevertheless, they focus on a single tunnel instance and do not provide the means to explore the ligand-protein interaction.

Problem Description and Input Data

When studying long molecular dynamics simulations, the researchers have to face namely the following high-level tasks:

- Detect the part of the simulation where the ligand enters the protein and finally reaches the active site.
- Explore this route in detail and detect its bottlenecks.

In consequence, these steps are performed in order to reveal the parts of the trajectory where the ligand gets stuck. It means that in such parts there are some obstacles made by the surrounding amino acids. These obstacles can be geometric (the empty space between these amino acids is too narrow) or physico-chemical (the properties of the amino acids are incompatible with the ligand properties), or their combination. The geometric obstacles can be detected by using an algorithm for tunnel computation in molecular dynamics (e.g., CAVER [2] or MOLE [3]). These tools are able to produce the information about time evolution of individual tunnels and for each tunnel detect its bottleneck – the narrowest part limiting the size of the ligand aiming to pass through this tunnel.

However, this bottleneck can be to some extent influenced by the passing ligand and this influence is also determined by the physico-chemical properties of the ligand and the bottleneck-lining amino acids (e.g., hydrophobicity or partial charges of individual atoms). To reveal such dependencies, it requires the involvement and experience of the domain expert who has to study the ligand trajectory captured in the molecular dynamics simulation.

In protein engineering, all these efforts can finally lead to the detection of amino acids along the ligand trajectory which caused some problems, i.e., were the key players in situations when the ligand got stuck. Such amino acids are then the best candidates for subsequent mutation of the protein chain when these amino acids are replaced by more suitable ones wrt. their size and properties. On the other hand, in drug design the aim is to propose modifications of the ligand to increase the binding likelihood in order to design a better drug produced from this ligand.

The input data is obtained from the simulations of molecular dynamics which contains the movements of the protein and one or more ligands. Ligands can follow different routes – they can be transported from the outside environment to the protein active site or vice versa (after the desired reaction the product leaves the protein).

The length of the simulations may vary from few hundreds to hundreds of thousands of time steps. This depends namely on the ligand velocity and its ability to find the proper path to the active site.

Long simulations can often contain many time steps when the ligand followed an impasse – it tried to find the proper entrance point to the protein (tunnel gorge) or entered a wrong tunnel and had to return. Such parts of the simulation could be of less interest so that the user should be provided with a possibility to filter them out and focus only on the interesting parts. Figure 2 shows such example when the ligand trajectory consists of 50.000 time steps. It is visible that in most of the time the ligand travels around the protein chain (depicted in blue). Therefore, only a fraction of the trajectory captures its transportation to the active site.

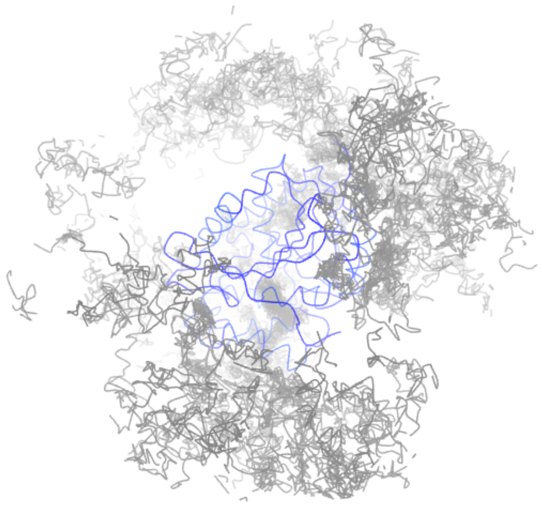


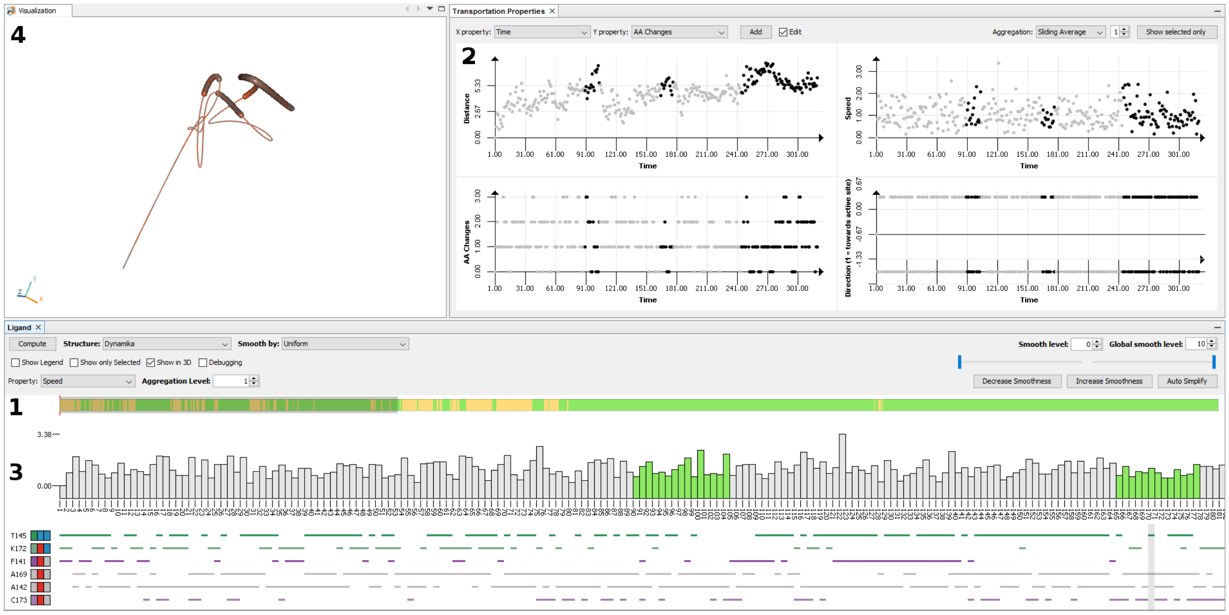
Figure 2 Ligand trajectory (gray) for a simulation containing 50.000 time steps. The protein chain is depicted in blue.

System Overview

Our proposed solution consists of several linked views covering individual steps of the exploration workflow (see Figure 3). In the first part, the user is provided with the overview representation of the whole loaded trajectory (Figure 3 1). This simple 1D representation encodes different routes of the ligand – green color highlights the parts of the trajectory where the ligand is inside the protein, white and yellow colors encode the situation when the ligand is outside the protein or on it surface respectively. Blue color shows the time portions when the ligand was close to the active site. By selecting a part of the trajectory using this representation, the user is automatically navigated to this selected part and all following representations show only this part.

Among these views belong the scatterplot matrix (Figure 3 2) enabling to explore different attributes of the ligand and its trajectory. These attributes are described in detail in section Derivation of Attributes. By interactive manipulation with these matrices the user can select a subset of time steps in which the ligand behaved in a desired way. The selection is highly dependent on the current tasks the user aims to perform. For example, when the user searches for the parts of the simulation where the ligand got stuck for a certain amount of time steps and at the same time was close to the active site, he or she plots and interacts with the matrix showing the relationship between the ligand stuckness and its distance from the active site.

The selected time steps can be further explored using the combination of bar chart and line representations (Figure 3 3). The bar chart enables to follow a selected attribute over time. Furthermore, individual bars can represent an aggregated information for selected portions of time steps. The aggregation is performed on uniformly divided time intervals. Then they show the average values of selected attributes. The line representation informs about the amino acids closest to the ligand along its path. This information helps to determine those amino acids which play a substantial role in the ligand transportation.



Visualization

Transportation Properties

X property: Time   Y property: AA Changes   Add   ☑ Edit   Aggregation: Sliding Average   1   Show selected only

Ligand

Compute   Structure: Dynamika   Smooth by: Uniform

□ Show Legend   □ Show only Selected   ☑ Show in 3D   □ Debugging

Property: Speed   Aggregation Level: 1

Smooth level: 0   Global smooth level: 10

Decrease Smoothness   Increase Smoothness   Auto Simplify

Figure 3 Overview of the proposed system. 1 – overview of the whole loaded trajectory, 2 – scatterplot matrix plotting different attributes of the ligand trajectory, 3 – bar charts and line representation showing ligand-lining amino acids, 4 – spatial representation of the trajectory.

These views are linked with the three-dimensional view (Figure 3 4) where the user can observe the ligand trajectory in the context of the protein structure. As the original trajectory is highly scattered, we propose a simplification algorithm which aims to reveal the trends in the ligand movement and suppress small insignificant movements. The algorithm is described in detail in the Trajectory Simplification section.

Trajectory Simplification

When visualizing the ligand trajectory as is, e.g., using line strip of consecutive ligand positions, the visualization becomes crowded even when analyzing only hundreds of snapshots (see Figure 4). Therefore, we decided to simplify the original trajectory data and to visualize the simplified trajectory. In this manner, we enable the user to deduce information about significant ligand movement directly from its 3D visualization.

We propose two types of ligand trajectory simplification: i) automatic and ii) manual. The automatic simplification is applied to the whole original trajectory, while the manual one enables fine user regulated control over the level of simplification of individual parts of the trajectory. In fact, the automatic simplification can be viewed as an iterative application of the manual simplification.

Manual Simplification

First, we will describe the algorithm of the manual trajectory simplification (see Algorithm 1). The input of

Algorithm 1 Trajectory simplification

```
Input: $T_{in}$ — trajectory, $I$ — simplification interval
Output: $T_{out}$ — simplified trajectory
procedure SIMPLIFY($T_{in}, I$)
    $(T', S') \leftarrow$ CACHELOAD()    ▷ $S'$ — previous simplification
    $S \leftarrow$ UPDATE($S', I$)

    if ISINCREMENTAL($S, S'$) then
        $T_{out} \leftarrow$ SAVITZKYGOLAY($T', I$)
    else
        $\mathcal{L} \leftarrow$ BYLEVELS($S$)    ▷ $\mathcal{L}$ — sets of complex intervals
        $T_{out} \leftarrow T_{in}$
        for all $L \in \mathcal{L}$ do    ▷ in asc. order by $level(L)$
            for all $I_L \in L$ do
                $T_{out} \leftarrow$ SAVITZKYGOLAY($T_{out}, I_L$)
            end for
        end for
    end if

    CACHESAVE($T_{out}, S$)
    return $T_{out}$
end procedure
```

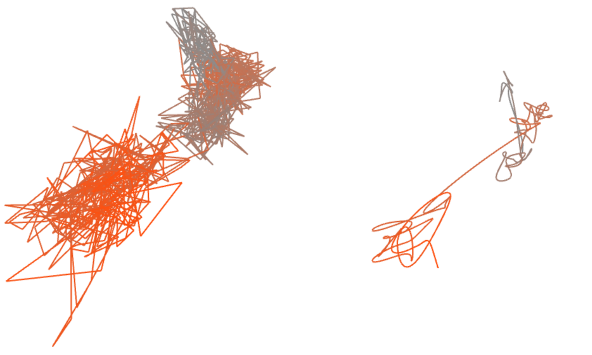


Figure 4 Visualization of 800 snapshots of a ligand trajectory using line strip. Visualization of the original trajectory is crowded (left). On the other hand, visualization of the simplified trajectory clearly reveals its possible important parts (right). The trajectory is colored by time from beginning (gray) towards its end (orange).

the algorithm consist of a trajectory T_{in} and an interval I which denotes a part where the trajectory will be simplified. As a first step, the algorithm retrieves from a cache the current visualized trajectory T' together with its simplification S'. Structure S' is a list of consecutive intervals that span the whole trajectory. Each interval in S' is assigned with a simplification level and as such describes the amount of simplification of a respective part of T_{in}. This representation enables simplification of different parts of T_{in} using different level of detail. In the next step, updated simplification S is obtained by applying I to S'. Then, it is decided whether T' can be incrementally updated to obtain T_{out}. This is true when the level of simplification of T' at the updated interval I is lower than the desired level of simplification. In this case, the current visualized trajectory T' is simplified on I resulting in T_{out}. Otherwise, the visualized trajectory T' cannot be used and the simplified trajectory has to be computed from scratch using T_{in}. This case typically emerges when a user decides to lower the amount of simplification of some part of the trajectory. The computation then proceeds as follows. A list $\mathcal{L}$ is computed from S. For each simplification level in S, we extract from S a set of all intervals on that level, L, and we add L to $\mathcal{L}$. Then, we iterate through $\mathcal{L}$ in ascending order by level of simplification. In each iteration, we have a set of intervals $L \in \mathcal{L}$ and we apply the simplification on all $I_L \in L$ to T_{out}. In both cases, we employ Savitzky-Golay smoothing method [22] to simplify the trajectory. As a last step, we store the simplified trajectory to a cache. The caching is employed to primarily improve performance of automatic simplification. Moreover, the performance of manual simplification is also enhanced, for example, when the user iteratively simplifies the same part of the trajectory until he/she is satisfied with the result.

Algorithm 2 Automatic trajectory simplification

```
Input: $T_{in}$ — trajectory, $\nu$ — complexity neighborhood, $\tau$ — complexity threshold, $\epsilon$ — improvement threshold
Output: $T_{out}$ — simplified trajectory
procedure AutoSimplify($T_{in}, \nu, \tau, \epsilon$)
    $C \leftarrow$ Interval($T_{in}$)                    ▷ $C$ — complex intervals
    $c(x) \leftarrow$ Complexity($T_{in}, \nu$)

    $T_{out} \leftarrow T_{in}$
    repeat
        $P \leftarrow$ FindSimplePoints($c, \tau$)
        $C \leftarrow$ RemoveSimplePoints($C, P$)

        for all $I \in C$ do
            $T_{out} \leftarrow$ Simplify($T_{out}, I$)
        end for

        $c'(x) \leftarrow c(x)$
        $c(x) \leftarrow$ Complexity($T_{out}, \nu$)

        $\Delta c \leftarrow \sum_{x \in T_{out}} max(c(x) - c'(x), 0)$
                                                     ▷ $\Delta c$ — improvement
    until $\Delta c < \epsilon$

    return $T_{out}$
end procedure
```

Automatic Simplification
The automatic algorithm (see Algorithm 2) is iterative and it employs the manual simplification in its iterations. Furthermore, it is based on an idea to simplify only parts of the trajectory that are still complex. The algorithm starts with considering the whole trajectory as complex – the set of complex intervals C is set to the interval spanning T_{in}. Then, the complexity (c) of T_{in} is evaluated in all points of T_{in}. The complexity $c(x)$ in point x is defined as (see Figure 5):

$$c(x) = \sum_{(u,v) \in N(T,x,\nu)} (|u| + |v|)^2 \alpha(u,v), \qquad (1)$$

where $N(T, x, \nu)$ is a set of consecutive tuples of segments of a trajectory T lying in the neighborhood of x and $\alpha(u, v)$ is angle between segments u and v. The neighborhood of x contains all points $y \in T$ such that $d(x, y) < \nu$ where $d(x, y)$ is distance along T. We evaluate the complexity of T in a neighborhood of x in order to take into account local shape of the trajectory in the vicinity of x. Our typical setting for ν is 2 Å an experimentally obtained value.

Further, simplification S is set to empty at the beginning and the resulting trajectory T_{out} is set to T_{in}. The iterative simplification then proceeds as follows. First, a set of simple points P is found by thresholding $c(x)$ by τ. All points $p \in P$ are then removed from C which prevents further simplification of parts of the

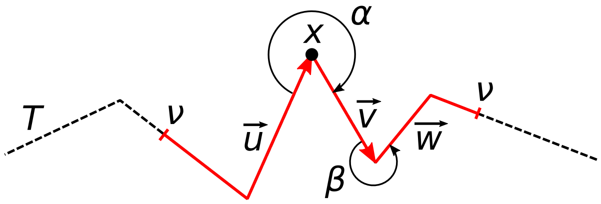


Figure 5 Evaluation of complexity of a trajectory T in point x. The complexity $c(x)$ is contributed to by tuples (u, v) and (v, w), i.e., their angles α and β, as the segments u, v and w lie in the neighborhood of x (red). The neighborhood of x contains all points that are closer (along T) to x then ν.

trajectory that are already simple. Then, T_{out} is simplified in all complex intervals that remained in C. After the simplification, the complexity is evaluated again and the improvement to previous complexity is computed. The iterative simplification ends when the improvement after an iteration (Δc) drops below a user specified threshold ϵ.

Derivation of Attributes

In the previous section we described the simplification of the ligand trajectory. On one hand, such approximation helps to understand the overall ligand movements. On the other hand, it also suppresses vast amount of details that are important for complete understanding of the ligand movements inside protein.

In order to preserve this information and hence allow the biochemist to explore the ligand behavior in detail we evaluate interesting geometric and physico-chemical attributes of the ligand trajectory on multiple levels. These attributes are then communicated to the user in several ways (using the scatterplot matrix, the bar charts, and different coloring of the trajectory in the 3D view).

Based on several discussions with the domain experts, we detected the following attributes as the most significant: ”stuckness” of the ligand in one place, its distance from the active site, the direction of its movement, the amount of surrounding free space, changes of surrounding amino acids and their properties, the hydrophobicity and charge profiles of amino acids and atoms around the ligand, and the speed of the ligand along the trajectory. Figure 6 aims to illustrate the individual attributes.

The remaining part of this section describes these attributes in detail, along with the process of deriving these attributes from the original data, and rationalizes their usefulness.

Stuckness is important attribute which comes from the fact that if the ligand got stuck in some place, it means that there was some obstacle (geometric, physico-chemical, or combined) which prevented the ligand to pass towards the active site. The biochemists are highly interested in these obstacles and want to explore them in more detail in order to reveal the reason of this problem and possibly propose a replacement (mutation) of some amino acids surrounding this problematic spot. We consider the ligand to be stuck if it oscillates around a specific place in the protein. To estimate the stuckness, we measure how much was the movement of the ligand straight, i.e., whether it moved significantly in some direction or its movement was rather random. The stuckness is estimated in a neighborhood of $2n + 1$ consecutive ligand positions (w.r.t. time). The size of the neighborhood n can specified by the user. For a set of ligand positions P we define the stuckness as:

$$s(x) = \frac{\max_{-n \leq i,j \leq n}\{d(P(x+i), P(x+j))\}}{\sum_{-n \leq i < n} d(P(x+i), P(x+i+1))}, \quad (2)$$

where $P(x)$ is position of the ligand in snapshot x and $d(x, y)$ is euclidean distance of $x, y \in \mathbb{R}^3$. To be complete, in the beginning (or in the end) of the trajectory, we shrink the neighborhood from left (or right). Furthermore, we set $n = 4$ by default as this value was obtained experimentally.

Distance of the ligand from the active site provides the biochemists with the information whether some observed behavior occurred in a vicinity of the active site, the surface, or somewhere between. This can be very helpful for instance when we want to mutate some amino acids along the trajectory in order to remove the stuckness of the ligand in that area. Therefore, this attribute is naturally connected with the ligand stuckness. In these cases it is essential to see also where exactly the unwanted behavior of ligand happened. In order to evaluate the current distance of the ligand from the active site, we first extract the position of the active site $A(x)$ in each time step x. Here the active site is defined by a set of surrounding amino acids and we compute the position of the active site as the center of mass of these amino acids, i.e., their atoms. Once $A(x)$ is determined, we compute distance of the ligand position $P(x)$ to the active site as $d_{AS}(x) = d(P(x), A(x))$ where $d(x, y)$ is again euclidean distance of $x, y \in \mathbb{R}^3$.

Direction of the ligand movement with respect to the active site is another essential attribute. During the MD simulation the ligand can enter and exit the protein tunnel repeatedly. Therefore, biochemists want to distinguish between intervals where the ligand was traveling towards the active site or in the opposite direction (i.e., towards the outer solvent). To compute

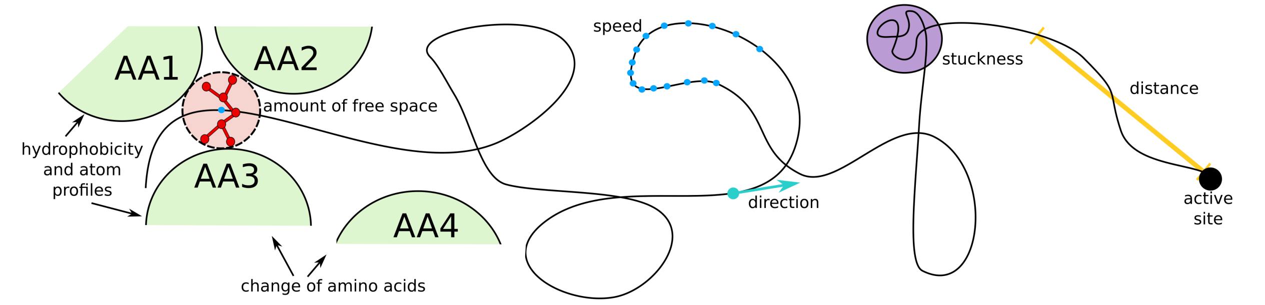


Figure 6 Illustration of the attributes derived for the ligand and its trajectory. AA1-AA4 denote amino acids lining the ligand in a specific position.

the direction we employ the **distance** attribute such that we evaluate whether the distance increases between two subsequent time steps, when moving towards the active site, or decreases, when moving in the opposite direction.

Amount of free space surrounding the ligand can help the biochemists to understand why the ligand in some places got stuck. In such cases there was not enough free space for the passage of the ligand or the ligand was repelled by some amino acid with incompatible physico-chemical properties. the amount of free space around the ligand can be obtained in two ways. When the user has the information about tunnels (e.g., calculated by CAVER tool [2]), we can use it for the assessment of the free space around the ligand. We take the tunnel in the corresponding time step and search for its sphere in which the ligand is positioned. The radius of this sphere is then taken as the descriptor of the free space. However, when the information about the tunnel is not available, we compute the free space in the following way. We denote this approach as the detection of temporal tunnel. The temporal tunnel is defined by a set of spheres where the number of spheres corresponds to the number of time steps in the simulation. Each sphere is taken from one time step and it defines the maximum empty space surrounding the ligand. For each time-step we firstly retrieve ten protein atoms closest to the ligand. Their mean position defines the center of the temporal tunnel sphere while the radius is determined as the minimum distance from this center to the closest protein atom.

Changes of amino acids and their properties help to understand when the ligand was in contact with a specific set of amino acids and when this set changed. In other words, the changes of amino acids can reveal those parts of the trajectory where the ligand significantly changed its position with respect to the protein. The higher number of different amino acids surrounding the ligand between two consecutive time steps corresponds to variations inside the protein tunnel as well. Moreover, whether the ligand will be able to pass through the tunnel and reach the active site is highly dependent not only on the geometric properties of the tunnel but also on the physico-chemical properties of the surrounding amino acids and their atoms (e.g., hydrophobicity or electric charge). This attribute is tightly connected with the ligand stuckness and subsequent proposal of candidates for amino acid mutations.

Hydrophobicity profile is related to the amino acids surrounding the ligand trajectory. It aims to add the user the information about physico-chemical property – hydrophobicity – of these amino acids. This is very useful in cases when the ligand got stuck and this was not caused by any geometric obstacle. This attribute is calculated as the average value from all ligand-lining amino acids when the lining distance is taken as 2 Å.

Charge profile has a very similar purpose as the hydrophobicity profile. It conveys the information about another significant physico-chemical parameter, atom partial charge, calculated for each atom in the distance of 2 Å from the ligand and averaged.

pridal jsem partial

Speed can be easily described and estimated in relative terms. Since the original data are uniformly sampled (i.e., we have the information about the exact ligand position in every two femtoseconds), we can measure the distance between two subsequent time steps and thus obtain the speed of the ligand in that particular area.

Visual Exploration

In this section we describe the individual views supported by our system. All views are interactively linked, i.e., the user can select a specific parts of the ligand trajectory and these parts are highlighted also in the remaining views.

Trajectory Overview

This simple 1D representation helps to navigate the user to the interesting parts of the simulation. It is depicted as a bar colored according to the ligand position with respect to the protein surface and active site. We assort the ligand movement into four categories:

- **Outside** – the ligand is moving outside of protein.
- **Surface** – the ligand is moving alongside the protein surface.
- **Inside** – the ligand is moving within the protein.
- **Active Site** – the ligand is moving in close proximity to the active site.

This representation enables quick filtering of irrelevant data (e.g., parts of the simulation when the ligand is moving outside the protein), as well as navigation to the potentially important parts. The selected time interval is then displayed in the bar chart as well as in the scatterplot matrix.

Bar Chart and Line View

This view provides a closer look at the ligand movement with respect to the properties of the temporal tunnel. It consists of two parts – the bar chart and line representation of all amino acids lining the ligand trajectory.

By default, each bar in the bar chart represents one time step of the simulation, though if needed (e.g., if we select long time sequence) the neighboring bars can be uniformly aggregated. The bars are selectable and the selected time steps are highlighted in the scatterplot matrix as well as in the 3D View. The height of the bar is set to represent one of the geometric or physico-chemical attributes described previously. The color of selected bars corresponds to the coloring of the trajectory overview, indicating the ligand position at a given time step. The user can visualize only the selected bars. When the set of selected bars is not continuous, the user is informed about it by a small wave inserted between the bars.

The information about the surrounding acids is also very crucial when exploring the sites when the ligand got stuck. For each time step within the temporal block we compute the three closest amino acids from the ligand. We provide the visual representation of this list (Figure 7), where each amino acid defines one line in the list. The line is colored with respect to a selected physico-chemical property – our representation supports switching between hydrophobicity, partial charges, and donors and acceptors. The interruption in some of these lines is caused by the fact that the given amino acid is not present around the ligand in the corresponding temporal block.

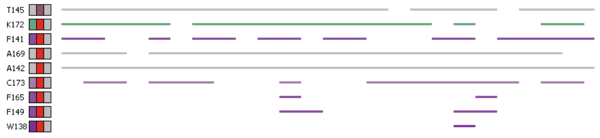


Figure 7 Representation of amino acids surrounding the temporal tunnel and their physico-chemical properties. Coloring corresponds to a selected property, the line interruption means that this amino acid was not detected around the ligand in the corresponding temporal block.

Scatterplot Matrix

For detailed analysis of geometric and physico-chemical attributes of the ligand trajectory at given time steps we propose the scatterplot matrix. The axes of one scatterplot represent a pair of user-selected attributes. Each point in the scatterplot then represents the values of these attributes at one time step. As a result, the scatterplot can easily reveal trends and relationships between attributes. Interactive manipulation with the scatterplot, such as selection, zooming, and change of displayed attributes provides an easy way for manual data filtering. To eliminate the noise in the data caused by ligand jittering, we propose the *sliding window* smoothing function. The sliding window function assigns to each time step the averaged value from its neighborhood. The size of the neighborhood is again regulated by the user.

The plots can further be stacked, forming a matrix and thus showing relationship between multiple attributes at once. The stacked plots are also interactively interconnected and the selection in Scatterplot Matrix is interactively linked with Barchart View and 3D View.

Figure 8 shows the scatterplot matrix for a sequence of 500 time steps.

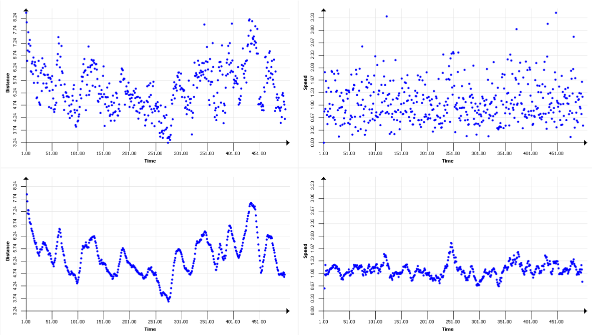


Figure 8 Scatterplot matrix depicting time vs. distance and time vs speed property before (up) and after (down) smoothing by the sliding window function.

3D View

This view provides an overview of the ligand trajectory in 3D. However, as was shown in Figure 2, the ligand trajectory contains a lot of noise, due to the jittery movement of the ligand. Therefore, we introduced the previously described trajectory simplification. Using this function the whole trajectory can be automatically simplified. If the automatic simplification does not lead to satisfactory results, selected parts of the trajectory can be further simplified manually. The trajectory is colored according to one of the geometric and physico-chemical attributes.

Analysis Procedure

In this section we will describe one of case studies which we have performed using our novel exploration tool. In this particular study the biochemists explored molecular simulation of Haloalkane dehalogenase together with the ligand. Since the simulation consists of 50.000 time steps, it would take enormous amount of time to analyze it using only common techniques such as 3D animation. In this section we will show how this can be significantly improved using our exploration tool. After the precomputation stage the user can immediately start to explore individual views described in the previous sections.

We start with the rough analysis of the overall behavior of the ligand using our simulation overview (see Figure 9). Utilizing this view the biochemist can immediately see that the whole simulation contains three main (two long and one relatively short) parts where the ligand was inside the protein (blue and green colors). These parts are divided by two orange and white blocks signalizing that the ligand was moving near to the protein surface (orange) or even entirely left the protein (white).

The mentioned color mapping on the simulation overview suggests that in this particular simulation the ligand started in the active site, then traveled towards the protein surface and finally left the molecule. After spending some time outside it found again its way into a protein tunnel and traveled back to the active side. The whole process is then repeated again while this time ligand spent significantly more time outside the protein or nearby its surface and returned just for last few hunderts of timesteps.

After the first evaluation of the overall ligand trajectory the biochemists stated that they would like to evaluate interactions between the ligand and protein in more details in two particular scenarios (see Figure 9). First, they focused on the part where ligand traveled from the active site to the protein surface (see Section 0.0.1). The second interesting case, according to the biochemist, is when the ligand is trying to find its way back into the protein (see Section 0.0.2). Exploration of these two parts of the simulation allows the biochemist to closely evaluate the influence of the protein tunnel and its properties to the ligand movements and it may help to reveal the essence of the interaction between the observed protein and given ligand.

In order to explore the individual parts of the simulation in more details our tool provides a brushing tool that allows to draw a rectangle over the desired part of the simulation in simulation overview and the corresponding time steps are then selected and used for further exploration.

tohle možná patří spíš do popsiu toho overview jako takového v předchozích sekcích - až to tam teda někdo dopíše :D

0.0.1 *Case 1 - ligand traveling from the active site to the protein surface*

For the first case the biochemist selected the first part of the simulation consisting from circa 17000 time steps (see Figure 9 – Case 1). From this point the biochemist can continue in more detailed exploration using bruising and linking techniques inside other three views – namely using the 3D, scatter plot, and bar chart views (see Figure 10 – top). Each of this view provides slightly different representation of the data and allows the biochemist to perform different operations.

For instance, using the scatter plot set to show distance from active site over time (see Figure **??** – top) immediately revealed that the ligand was actually near the protein surface (represented by high peaks) few additional times during this part of the simulation. The same behavior can be actually seen also in the simulation overview (see Figure 9 – Case 1) as slim orange lines. But it is much more obvious in this scatter plot view.

Another interesting behavior which can be seen with this setup is that ligand is repeatedly traveling back and forward from the active site (lower values) to the surface (higher values) in the first (and again in the last) part of the selected interval. This behavior could suggests that ligand had actually problems to bind itself to the active site.

When evaluating the reactivity between the ligand and protein active site it is important to determine the exact amino acids that interact with the ligand. If the amino acids in the active site of the protein are non-polar then it makes this active site specific to a non-polar substance. On the other hand, if the active site is made up of polar amino acids then the active site is specific to a polar substance. Therefore, the biochemist were interested in seeing whether there are some polar amino acids inside the active site. Note that the polarity of amino acids is closely related to their hydrophobicity – hydrophobic amino acids are non-polar while hydrophilic amino acids are polar.

tohle mám z nějaké stránky - je to třeba přepsat bo copy paste http://ibguides hl

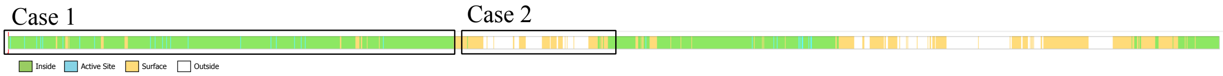


Figure 9 Overview of the simulation consisting of 50.000 time steps. Selected part was explored in detail in our case study.

The desired information can be retrieved using our exploration tool in many ways. Here we show one of the possible solutions used by our domain experts in the real test scenario .

já skončím v pekle

The biochemist first added another scatter plot showing the hydrophobicity profile along the ligand trajectory (see Figure 10). Since both scatter plots are interactively connected in real-time, the biochemist could select the low hydrophobicity values and immediately observed at which distance they are presented with respect to the ligand trajectory.

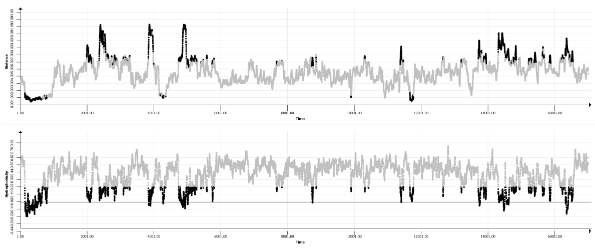


Figure 10 TODO Bottom – Hydrophobicity profile of the trajectory. Top – the distance from the active site over time. As can be seen the most hydrophilic amino acids are surrounding the most distant parts of the trajectory.

The results among others confirmed the behavior common for most of the proteins. Namely the fact that the part of the trajectory in greater distance from the active site (i.e., nearby surface) is surrounded mostly by hydrofilic amino acids while when the ligand reaches the core of the protein it is surrounded by highly hydrophobic amino acid.

However, as can be seen in the bottom part of Figure 10 there are few amino acids that are actually hydrophilic but located in the close vicinity to the active site. In order to easily exactly pinpoint these specific amino acids the biochemist set the scatter plot to visualize the exact relation ship between the distance and hydrophobicity (see Figure 12 – bottom). By creating selection in this view, the biochemist were able to easily pinpoint the part of the simulation closer to the active site while containing the hydrophilic amino acids (see Figure 12 – top).

As the next step, the biochemist were interested in seeing what amino acids are actually responsible for the described behavior. For this reason they utilized the bar chart view containing the amino acid graph (see Figure **??**).

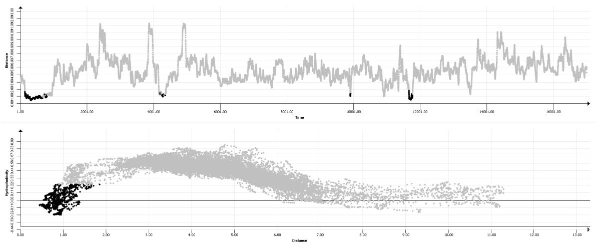


Figure 11 TODO

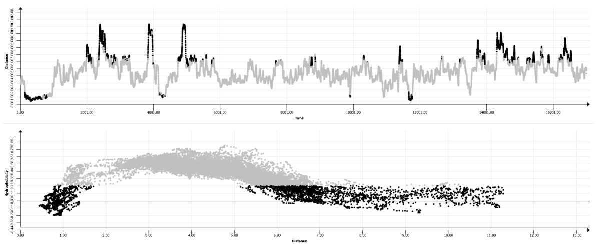


Figure 12 TODO mam ještě aj tenhle ale nevím zda ho použít – to je ta prvni selekce?

In order to easily see required data they confine the view only to selected time steps and set high level of aggregation (each bar contains about 20 time steps). Even though the amino acids view contains list of all amino acids surrounding the ligand in given time steps it was relatively easy to spot the hydrophobic ones due to the ability of the view to color individual amino acids based on their properties. Here the biochemist simply selected the green amino acids (N38 and D103). As can be seen in Figure 14 the selected amino acids exactly corresponds to the estimated active site in the protein.

moc dlouha věta, ale Bára to určitě poladí, viď :)

As the last step, the biochemist went back to the bar chart, lowered the amount of aggregation and selected

tenhle závěr bude chtít taky asi tady polatid

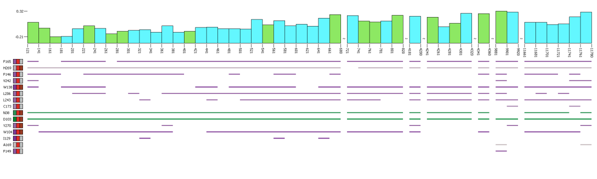


Figure 13 TODO

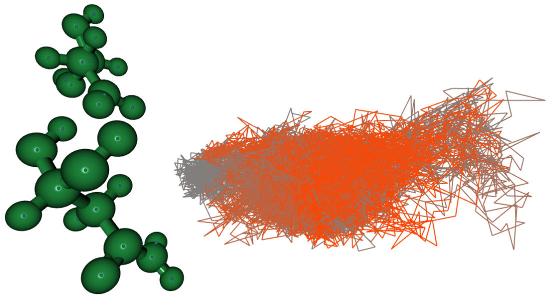


Figure 14 TODO než se zeptaš, SS jsou dodrbané, musím opravit :D

all bar charts containing the two hydrophobic amino acids (see Figure 15). I thought/hoped, that this would clearly show, that these two AA are involved in the "not being able to bind" section (i.e., where the high peaks are) and hence prove my theory stated at the begining, but this clearly have not happen so, what next?.

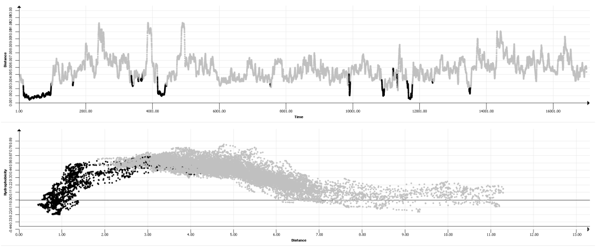


Figure 15 TODO

0.0.2 Case 2 - ligand accessing the protein

na to asi kašlu, není místo co?

Results and Discussion - BK

use case + feedback

Conclusions - All

and future work

Competing interests

The authors declare that they have no competing interests.

Author details

[1]Masaryk University, Brno, Czech Republic. [2]University of Bergen, Norway.

References

1. Koudelakova, T., Chaloupkova, R., Brezovsky, J., Prokop, Z., Sebestova, E., Hesseler, M., Khabiri, M., Plevaka, M., Kulik, D., Kuta Smatanova, I., Rezacova, P., Ettrich, R., Bornscheuer, U.T., Damborsky, J.: Engineering enzyme stability and resistance to an organic cosolvent by modification of residues in the access tunnel. Angewandte Chemie International Edition **52**(7) (2013)
2. Chovancova, E., Pavelka, A., Benes, P., Strnad, O., Brezovsky, J., Kozlikova, B., Gora, A., Sustr, V., Klvana, M., Medek, P., Biedermannova, L., Sochor, J., Damborsky, J.: CAVER 3.0: A tool for the analysis of transport pathways in dynamic protein structures. PLoS Computational Biology **8**(10) (2012)
3. Sehnal, D., Svobodova Varekova, R., Berka, K., Pravda, L., Navratilova, V., Banas, P., Ionescu, C.-M., Otyepka, M., Koca, J.: MOLE 2.0: advanced approach for analysis of biomacromolecular channels. Journal of Cheminformatics **5**(1) (2013)
4. Yaffe, E., Fishelovitch, D., Wolfson, H.J., Halperin, D., Nussinov, R.: MolAxis: Efficient and accurate identification of channels in macromolecules. Proteins: Structure, Function, and Bioinformatics **73**(1) (2008)
5. Kozlikova, B., Sebestova, E., Sustr, V., Brezovsky, J., Strnad, O., Daniel, L., Bednar, D., Pavelka, A., Manak, M., Bezdeka, M., Benes, P., Kotry, M., Gora, A., Damborsky, J., Sochor, J.: CAVER Analyst 1.0: graphic tool for interactive visualization and analysis of tunnels and channels in protein structures. Bioinformatics **30**(18), 2684–2685 (2014)
6. Devaurs, D., Bouard, L., Vaisset, M., Zanon, C., Al-Bluwi, I., Iehl, R., Simeon, T., Cortes, J.: MoMA-LigPath: a web server to simulate protein-ligand unbinding. Nucleic Acids Research **41**(Web Server issue), 297–302 (2013)
7. Isralewitz, B., Gao, M., Schulten, K.: Steered molecular dynamics and mechanical functions of proteins. Current Opinion in Structural Biology **11**(2), 224–230 (2001)
8. Ludemann, S.K., Lounnas, V., Wade, R.C.: How do substrates enter and products exit the buried active site of cytochrome P450cam? 1. Random expulsion molecular dynamics investigation of ligand access channels and mechanisms. Journal of Molecular Biology **303**(5), 797–811 (2000)
9. Dodge, S., Weibel, R., Lautenschütz, A.-K.: Towards a taxonomy of movement patterns. Information Visualization **7**(3), 240–252 (2008)
10. Andrienko, G., Andrienko, N., Bak, P., Keim, D., Wrobel, S.: Visual Analytics of Movement. Springer, Heidelberg (2013)
11. Vrotsou, K., Janetzko, H., Navarra, C., Fuchs, G., Spretke, D., Mansmann, F., Andrienko, N., Andrienko, G.: SimpliFly: A methodology for simplification and thematic enhancement of trajectories. IEEE Transactions on Visualization and Computer Graphics **21**(1), 107–121 (2015)
12. Bidmon, K., Grottel, S., Bös, F., Pleiss, J., Ertl, T.: Visual Abstractions of Solvent Pathlines near Protein Cavities. Computer Graphics Forum (2008)
13. Luboschik, M., Maus, C., Schulz, H.-J., Schumann, H., Uhrmacher, A.: Heterogeneity-based guidance for exploring multiscale data in systems biology. Proceedings of the IEEE Symposium on Biological Data Visualization (BioVis'12) (2012)
14. Phillips, M., Georgiev, I., Dehof, A.K., Nickels, S., Marsalek, L., Lenhof, H.-P., Hildebrandt, A., Slusallek, P.: Measuring properties of molecular surfaces using ray casting. In: Parallel Distributed Processing, Workshops and Phd Forum (IPDPSW), 2010 IEEE International Symposium On, pp. 1–7 (2010)
15. Lindow, N., Baum, D., Hege, H.-C.: Voronoi-based extraction and visualization of molecular paths. IEEE Transactions on Visualization and Computer Graphics **17**(12), 2025–2034 (2011)
16. 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). IEEE
17. Parulek, J., Turkay, C., Reuter, N., Viola, I.: Visual cavity analysis in molecular simulations. BMC Bioinformatics **14**(Suppl 19), 4 (2013)
18. Lindow, N., Baum, D., Bondar, A.-N., Hege, H.-C.: Exploring cavity dynamics in biomolecular systems. BMC Bioinformatics **14**(S-19), 5 (2013)
19. Krone, M., Reina, G., Schulz, C., Kulschewski, T., Pleiss, J., Ertl, T.: Interactive extraction and tracking of biomolecular surface features. Computer Graphics Forum **32**(3pt3), 331–340 (2013)
20. Kozlikova, B., Jurcik, A., Byska, J., Strnad, O., Sochor, J.: Visualizing movements of protein tunnels in molecular dynamics simulations. In: Eurographics Workshop on Visual Computing for Biology and Medicine, VCBM 2014, Vienna, Austria, 2014. Proceedings, pp.

97–106 (2014). http://dx.doi.org/10.2312/vcbm.20141188

21. Byska, J., Le Muzic, M., Groeller, M.E., Viola, I., Kozlikova, B.: AnimoAminoMiner: Exploration of protein tunnels and their properties in molecular dynamics. Visualization and Computer Graphics, IEEE Transactions on **22**(1), 747–756 (2016)
22. Savitzky, A., Golay, M.J.: Smoothing and differentiation of data by simplified least squares procedures. Analytical chemistry **36**(8), 1627–1639 (1964)