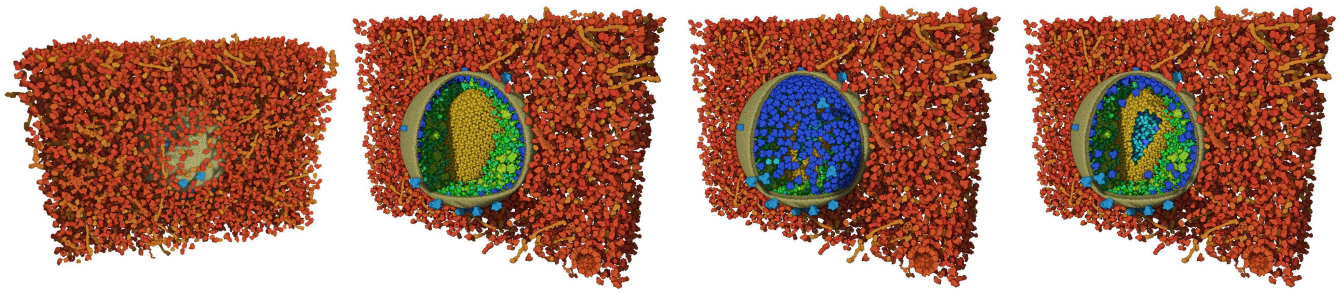
Eurographics Conference on Visualization (EuroVis) 2016
K.-L. Ma, G. Santucci, and J. van Wijk
(Guest Editors)

Volume 35 (2016), Number 3

Visibility Equalizer
Cutaway Visualization of Mesoscopic Biological Models

M. Le Muzic[†1], P. Mindek[1], J. Sorger[1,2], L. Autin[3], D. S. Goodsell[3], and I. Viola[1]

[1]TU Wien, Austria [2]VRVis Research Center, Vienna, Austria [3]The Scripps Research Institute, La Jolla, California, USA



(a) (b) (c) (d)

Figure 1: The workflow of our method for a model of HIV surrounded with blood plasma proteins. (a) The entire dataset is shown. The blood serum (shown in red) is occluding the virus. (b) Clipping objects are added to selectively clip molecules to reveal the HIV capsid. (c) The illustrator decides to show more of the matrix proteins (shown in blue), so their clipping is disabled. However, they are now occluding the view of the capsid. (d) The probabilistic clipping has been used to selectively remove those matrix proteins occluding the capsid, but some of them are left in the scene to indicate the presence of this type of protein on the virus membrane. The capsid has been clipped with view space clipping to reveal its internal structure.

Abstract

In scientific illustrations and visualization, cutaway views are often employed as an effective technique for occlusion management in densely packed scenes. We propose a novel method for authoring cutaway illustrations of mesoscopic biological models. In contrast to the existing cutaway algorithms, we take advantage of the specific nature of the biological models. These models consist of thousands of instances with a comparably smaller number of different types. Our method constitutes a two stage process. In the first step, clipping objects are placed in the scene, creating a cutaway visualization of the model. During this process, a hierarchical list of stacked bars inform the user about the instance visibility distribution of each individual molecular type in the scene. In the second step, the visibility of each molecular type is fine-tuned through these bars, which at this point act as interactive visibility equalizers. An evaluation of our technique with domain experts confirmed that our equalizer-based approach for visibility specification is valuable and effective for both, scientific and educational purposes.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Viewing algorithms

[†] The first two authors contributed equally.

Contact: {mathieu | mindek}@cg.tuwien.ac.at

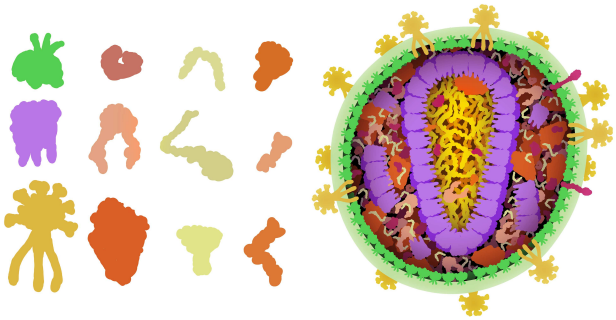
1. Introduction

Molecular biology is an emerging field that is characterized by rapid advances of the current state of knowledge. New discoveries have to be communicated frequently to a large variety of audiences. However, due to their structural and phenomenal complexity, it is challenging to convey the discoveries of molecular phenomena. On a mesoscale level, where thousands or millions of macromolecules form a given structure, this challenge is amplified by the presence of multiple spatio-temporal scales. Currently, illustrations are the most widely-used form of communicating mesoscale structures. To reveal the internal composition of mesoscale structures, such as viruses or bacteria, illustrators often employ clipping, section views or cutaways in their work.

Considering the rapid evolution of knowledge in the field of biology, it is necessary to adapt the traditional illustration pipeline so that new knowledge can be easily added into illustrations, instead of tediously redrawing the entire structure. Virtual 3D models of cells and other mesoscale molecular structures can be utilized for these purposes. Biologists have designed tools, such as *cellPACK* [JAAA*15], to procedurally generate 3D models that represent the structure of micro-organisms such as viruses, or entire cells at atomic resolution. Based on a set of input parameters, individual molecules are algorithmically assembled into these complex organic static structures. The parameter set consists of a specification of molecular types, concentrations and spatial distribution that define where the instances are distributed in a given compartment. The resulting 3D models, in the most complex cases, may consist of thousands of various molecular types, which in turn, may result in millions of molecules and billions of atoms. The instances are densely packed within the predefined compartments, to replicate the actual molecular crowding phenomena prevailing in living organisms. Due to the high density of these scenes, inner structures that are essential for conveying the function of the organism remain hidden. It is therefore important to develop visualization techniques that would procedurally reproduce the occlusion management methods used in traditional illustration. Currently, this is achieved by placing clipping objects in the scene, which remove specified parts of the displayed model. However, illustrators have to make sure that the essential information, e.g., the ratio of multiple molecular ingredients, is represented and not either hidden in the volume or clipped away (Fig. 2a). To do this, they would need to visually inspect the presence of each single ingredient in the resulting visualization (Fig. 2b).

To alleviate this process, we present our first contribution: a method that quantifies the overall visibility of the model contents. We display a stacked bar for each molecular type that encodes the ratio of visible, clipped, and occluded instances of the respective type for the current viewpoint and clipping setting. During the process of placing clipping objects in the scene, these bars continuously reveal molecular types that are underrepresented or overrepresented. This enables the illustrator to modify the placement of the clipping objects in such a way that every molecular type is adequately represented in the scene. We call this the *coarse level* of the visibility specification process.

To preserve important structures that would be removed by clipping objects such as cutting planes, traditional illustrations also often reintroduce parts of the removed structures in front of the revealed cross sections. In Figure 2b, for instance, the glycoproteins (yellow molecules) of the HIV particle that are not occluding the object of interest, in this case the capsid containing the RNA, are left in the illustration to communicate their presence on the surface of the virus (Fig. 2a). In this way, the main components of the virus particle can be illustrated in a single image. The process of fine-tuning the visibility is extremely time-consuming, as the illustrator has to manually pick individual molecular instances to be reintroduced or removed from the scene.



(a) (b)

Figure 2: (a) Various types of protein molecules and (b) their embedding into a mesoscale model. It is a demanding task to determine which molecular types are visible and how many of their instances are shown.

To speed up this visibility fine-tuning process, we propose our second contribution: a novel method for managing the visibility of instances. Instead of binary enabling or disabling the presence of all instances for a given molecular type, we offer the possibility to show a desired number of them. The main purpose of this approach is to increase the visibility of hidden structures by removing redundant parts of occluding instances while preserving some of them. An analogy of this approach can be drawn with "Screen-Door Transparency", a trivial way to obtain transparency in computer graphics by placing small holes in a polygon to reveal what is present behind. Additionally, we propose the novel metaphor of the *visibility equalizer* for controlling this effect efficiently. To explain its role, we use the metaphor of hi-fi sound reproduction where the *volume control* is used for adjusting the output sound *uniformly* on all frequencies. Such a mechanism corresponds to our coarse level visibility specification of the clipping objects, where all molecular types are uniformly removed from the clipped regions. However, hi-fi sound systems also allow users to fine-tune the volume levels of individual frequency bands through an *equalizer*. Following this metaphor, the visibility levels of molecular types are represented as stacked bars that form our visibility equalizer. All visibility bars are interactive, thus allowing the user to adjust desired levels just as with a sound equalizer. After drawing relation to previously proposed visibility management techniques in the next section, we will discuss the technique of visibility equalizers in further detail in the remaining sections of the paper.

2. Related Work

Related work can be categorized into occlusion management techniques and molecular visualization. We will concentrate on the former, according to the focus of this paper.

2.1. Occlusion Management

Related occlusion management techniques can be categorized into object centric approaches and transfer function based approaches. In object centric approaches, the geometry or parts of the volume that are obstructing one or more particular objects of interest are (partially) removed. In transfer function based approaches, the user assigns importances to intervals of the volume data values.

Object Centered Approaches. Cutaway and ghosting techniques were first introduced by Feiner & Seligmann [FS92] in 1992. Their work inspired several follow-up approaches [DWE02, DWE03, WEE03, VKG04, VKG05, KTH*05] that were later summarized in the survey by Viola & Gröller [VG05] under the collective term of *smart visibility* techniques. They coined this term to describe expressive visualization techniques that smartly uncover the most important features of the displayed data, i.e., cutaway views, ghosted views, and exploded views.

Krüger et al. [KSW06] developed a system that applies transparency and shading to enable focus&context visualization in volume data sets with a simple point&click interface. Li et al. [LRA*07] propose a cutaway design based on the geometry of the occluder in contrast to previous approaches that were based on the occludee. Burns & Finkelstein [BF08] applied the concept of importance-driven cutaways for volume data to polygonal models. Lawonn et al. [LGV*16] extend this approach to present a composite technique that combines the visualization of blood flow with the surrounding vessel structures. Baer et al. [BGCP11] published a perceptual evaluation of smart visibility techniques for two ghosted view approaches in comparison to semi-transparent approaches. The results clearly favored the ghosted view techniques. Sigg et al. [SFCP12] propose an approach for automatic cutaway box placement with optimized visibility for target features that are specified as degree-of-interest functions during interactive visual analysis of the volume data. Lidal et al. [LHV12] defined five design principles for cutaway visualization of geological models. The approach by Diaz et al. [DMNV12] preserves the relevant context information in volume clipping by allowing the user to extrude segmented surfaces such as bone structures from the clipping plane.

Since these approaches are object centered, they deal with (partial) occlusion of individual objects. For our data, partial or even complete occlusion of individual molecules is not an issue. The data is not composed of large singular entities such as polygonal or segmented volumetric objects where each single one has a semantic meaning. Instead, there are thousands or hundreds of thousands of instances that stem from only a couple of dozen molecule types. Therefore it does not matter if individual instances are occluded, as long as the structures that they form are preserved. Our approach is therefore fundamentally different from existing occlusion management approaches as it combines principles from object centered and transfer function based approaches.

Transfer Function Based Approaches. Since our molecule data is composed of a dense point cloud that resembles volumetric data on a nonregular grid, our approach is also related to transfer function based approaches. However, instead of a wide range of attribute values distributed over voxels, our data features a comparably smaller number of molecule types. This characteristic enables our visibility equalizer approach. The context-preserving volume rendering model [BGKG05] uses a function of shading intensity, gradient magnitude, distance to the eye point, and previously accumulated opacity to selectively reduce the opacity in less important data regions. Contours of surfaces that would be removed due to opacity remain visible, as the amount of illumination received is taken as a measure whether a point should be visible or not. Burns et al. [BHW*07] propose a multimodal approach that combines CT scan data and real-time ultrasound data. Importance driven shading is used to emphasize features of higher importance that have been revealed through the ghosting.

In his PhD thesis [Vio05], Viola presents an optimization strategy for automatically assigning visual mapping to voxels so that segmented objects in the volume are visible as specified by the user. Correa et al. [CM11] used a similar approach for applying visibility directly to voxels, without the notion of segmented objects. In our approach, we control visibility by interacting with the stacked bars of the visibility equalizer to modify the clipping object properties for each icndividual molecule type. Ruiz et al. [RBB*11] propose an approach for automatic transfer function optimization by minimizing the informational divergence between a user specified target distribution and the visibility distribution captured from certain viewpoints.

Transfer function based approaches are well suited for volumetric data that contains segmentable structures, such as the organs or bones in a medical scan. For molecular data this only holds partially true, as some types of molecules do indeed form continuous structures that could be made visible with a transfer function (e.g., membranes, nucleus). On the other hand, within these structures there is a more noise-like distribution of these molecules that cannot be segmented into solid structures.

2.2. Multi-Scale Visualization of Molecular Structures

Lindow et al. [LBH12] were the first to introduce a fast method for the real-time rendering of large-scale atomic data on consumer level hardware. They utilize instancing on the GPU to repeat these structures in the scene. For each molecule type, a 3D grid of the atoms is created and stored on the GPU. Falk et al. [FKE13] further refined the method with improved depth culling and hierarchical ray casting to achieve faster rendering performance for even larger scenes.

A novel and more efficient approach for rendering large molecular datasets was introduced by Le Muzic et al. [LMPSV14], which is based on brute-force rasterization rather than ray-tracing. To reduce the number of drawn geometries they rely on the tessellation shader to perform dynamic level-of-detail rendering. Their approach allows switching between different degrees of coarseness for the molecular structures on the fly, thus allowing the entire scene to be rendered in a single draw call efficiently, approaching

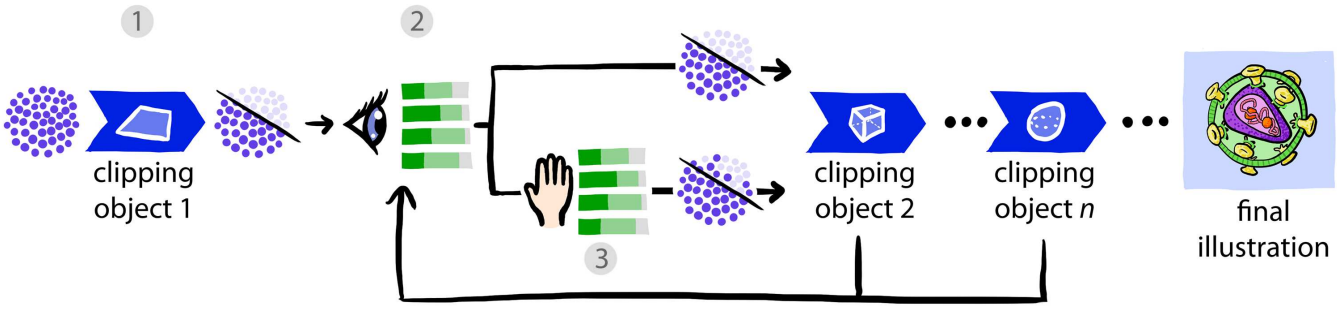


Figure 3: An illustration of the workflow with the visibility equalizer. (1) Clipping objects filter out elements in the data based on their type and location. (2) The clipping is applied in serial, i.e., the output of a clipping object constitute the input of the next one. The visibility information of the entire scene is routinely collected and updated in the visibility equalizer to keep the viewer informed about the current state of the data. (3) The clipping parameters of a given clipping object can later on be refined by interacting with the bar charts of the visibility equalizer to offer more control on the clipping, such as fuzziness.

zero driver overhead. As a follow up, they developed and released cellVIEW [LAPV15], a tool designed for rendering large-scale molecular scenes, which was implemented with Unity3D, a popular 3D engine. cellVIEW was primarily developed to showcase large molecular structures generated with cellPACK [JAAA*15], a tool developed to procedurally generate accurate and multi-scale models of entire viruses and cells.

Our visibility equalizer technique is built upon cellVIEW to improve the navigation and exploration of large molecular scenes generated with cellPACK. cellVIEW leverages GPU computing and parallel programming to enable real-time rendering and therefore, to provide a smooth and responsive user-experience, the visibility equalizer was developed with the same programming paradigm.

3. Overview

The two main components of our method are the *clipping objects* and the *visibility equalizer*. The workflow of our method is depicted in Figure 3. We distinguish between object space object clipping and view space object clipping. Object space clipping discards elements according to their distance to a geometric shape. View-space clipping discards elements according to whether or not they occlude certain objects in the current viewport. The role of the visibility equalizer is two fold: to provide important information about the visibility of molecular species, and to let users override the behavior of the clipping objects by directly manipulating the equalizer values. Each set of stacked bars show three types of quantitative information for a given type of ingredient: a - the ratio of visible instances to the total number of instances; b - the ratio of occluded instances to the total number of instances; c - the ratio of dicarded instances to the total number of instances. The visual encoding of the visibility equalizer is illustrated in Figure 4. By dragging the light green bar, the proportion of instances affected by the object-space clipping is modified, while dragging the dark green bar modifies the proportion of instances affected by the view-space clipping.

4. Object Space Clipping

Clipping objects define how instances shall be discarded depending on their location or type. They can operate either in object space or in view space. In this Section, we will explain in detail how clipping objects operate in object space.

4.1. Clipping Object Distance

Clipping objects are associated with geometric shapes to specify a region of the domain that is influenced by the clipping. Our system currently supports the following set of primitive shapes: plane, cube, sphere, cylinder and cone. We compute the distance between the instance centroids and the clipping region to identify instances that lie inside that region. To accelerate the computation, we solve the problem analytically using a mathematical description of the clipping region as a 3D signed distance field (SDF).

Due to the architecture of the rendering technique which is employed, the instances information (position, rotation, type, radius, etc.) is already stored large buffers stored on the GPU memory. To speed up the clipping operation and avoid data transfer costs, the distance is computed in parallel in a compute shader program, prior to the rendering, with one thread allocated per instance. The position, rotation, scale and geometry type of every clipping object must be additionally uploaded to the GPU in order to define the correct clipping region SDF. In a single thread and for each clipping object, the required information is fetched from the video memory and is used to compute the signed distance between an instance and the clipping object. When an object needs to be clipped, a dedicated flag in a GPU buffer is updated. The flag is then accessed during the rendering to discard the clipped instance.

4.2. Clipping Filtering

A clipping object also comprises a set of parameters that override the clipping state of instances based on their type. This filtering operation is performed if an instance is located inside the clipping region, in the same compute shader program described in Section 4.1. The first parameter controls the percentage of discarded instances

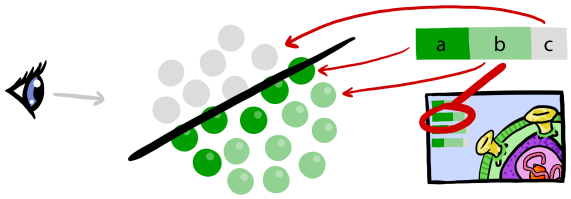


Figure 4: Illustration of the visibility equalizers. Each molecular ingredient has its own stacked bar showing (a) instances visible from the current viewpoint, (b) occluded instances, (c) instances clipped away by the clipping objects.

in the clipping region for a given type. We refer to this value as *object space clipping probability*. This value can be increased or decreased by dragging the light green bar in the visibility equalizer. It is important to mention that with multiple clipping objects, interaction with the light green bar in the visibility equalizer will only affect the clipping probability of the selected clipping object.

Upon start-up of the system, each instance is given a uniformly distributed random number between 0 and 1, and which will remain unchanged. Then, for each instance, we compare this value with the clipping probability in the computer shader program. If the constant random number is higher than the clipping probability, the instance is marked as discarded, and will not be rendered. For example, if the clipping probability value is equal to zero, all instances in the clipping region will be clipped, whereas if the value is equal to one, no instances will be clipped. A value between zero and one thus controls the degree of fuzziness of a clipping, as explained in Figure 5.

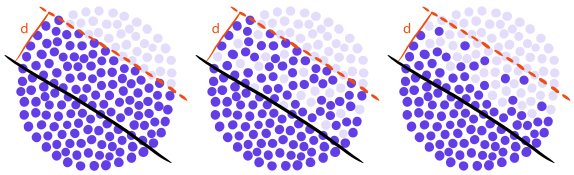
The other parameters allow users to control the clipping based on properties such as the size of the molecules (weight) and their overall number (concentration). Via the user interface and for a given clipping object, the user defines ranges that correspond to the desired concentration and molecular weight. Instances whose properties lie outside these ranges are simply discarded and will not be rendered.

4.3. Falloff Function

To increase control over the object space clipping, we use a falloff function. The falloff function gradually modulates the effect of the clipping with respect to the distance from the clipping shape as illustrated in Figure 5. The farther away from the clipping surface an instance is, the higher its clipping probability will be. The object space clipping probability of a molecule on the 3D position p is multiplied by the falloff function $f(p)$. The falloff function is defined as follows:

$$f(\vec{p}) = 1 - min(1, (d(\vec{p})/m)^c) \quad (1)$$

where $d(p)$ is the distance to the clipping surface from the point p. The function is parametrized by m and c, where m is the maximum distance up to which the object-space clipping probability takes effect, and c specifies the exponent of the falloff function. It is important to mention that the falloff function will not preserve the user-defined clip-ratio displayed in the visibility equalizer.



(a) (b) (c)

Figure 5: Illustration of the falloff function mechanism: (a) elements located further than distance d from the clipping object are clipped, (b) elements between the clipping object and d are uniformly clipped, (c) elements are removed gradually based on their distance to the clipping object to further customize its behaviour.

5. View Space Clipping

While object space clipping with primitive shapes allows for a high degree of flexibility, it may also require cumbersome manual operations for more complex set-ups. We therefore provide additional functionalities to selectively remove occluding instances in front of a set of ingredients set focus, to ensure them a maximum degree of visibility. The focus state can be manually specified via the visibility equalizer by ticking a dedicated checkbox in front of the stacked bars.

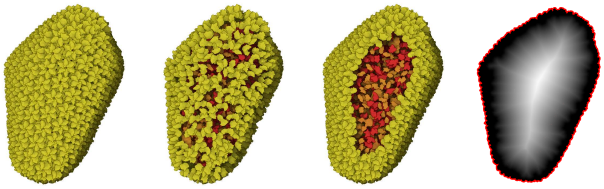
5.1. Occlusion Queries

When an ingredient type is set as focus, occluding instances of a different type may be selectively removed to reveal the occludees. To identify occluding instances, we perform occlusion queries. Nowadays, modern graphics hardware is able to perform occlusion queries easily using the fixed-functionality. This method requires one draw call per query, which may induce driver overhead with several thousands of queries. We compute the queries manually instead, using a custom shader program, because it allows the queries to be computed in a single GPU draw call, thus approaching zero driver overhead. In-depth technical details about this approach are described by Kubisch & Tavenrath [KT14].

We first render all the focused ingredients to an off-screen texture. This texture will then serve as a depth-stencil mask for the occlusion queries. There can be several ingredient types constituting the object of focus for a given clipping object. Thereafter, we render the bounding spheres of the potential occluders in a single draw call using instancing, on top of the depth-stencil mask. Thanks to early depth-stencil functionality, available on modern graphics hardware, fragments that will pass the test and be executed are guaranteed to belong to an occluder. We then update the clipping state of the occluding instance by updating its corresponding occlusion flag stored in the main video memory directly from the fragment program.

5.2. Clipping Filtering

Similarly to the object space clipping, we provide an additional parameter to control the degree of fuzziness of the view-dependent



(a) (b) (c) (d)

Figure 6: View-space clipping, (a) shows the full HIV Capsid, (b) shows the uniformly distributed clipping, (c) demonstrate the aperture effect and (d) shows the results of the 2D distance transform of the clipping mask.

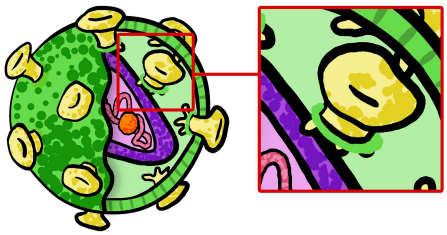


Figure 7: Illustration of contextual anchoring with an HIV particle. Despite the cutaway, some of the glyco-proteins (in yellow) are displayed and their surrounding lipid molecules (green) is preserved as contextual information.

clipping, which we refer to as *view-space clipping probability*. This value is set by the user for each ingredient type, and is modified by dragging the dark green bar in the visibility equalizer. The view-space clipping probability is evaluated after an instance is flagged as occluder in the same shader program mentioned in Section 5.1. We compare the clipping probability with a random number, initially defined and described in Section 4.2. If the random number is higher than the clipping probability, the instance will remain as clipped, otherwise it will be displayed. This will however result in a uniformly distributed number of visible occluders over the object of focus. Such a distribution might not always be the best design choice, because it fragments heavily the overall structure of the occluders and makes it difficult to see the occludees, as shown in Figure 6b.

We propose an alternative technique for fuzzy removal of occluding instances, which we dub the *aperture effect*. We define an additional parameter, the aperture coefficient, which controls the 2D distance from the instance to the edges of the occludees mask below which occluding instances shall be clipped. A example of the aperture effect is shown in Figure 6c. To enable this effect we compute the 2D distance transform of the occludees mask which we store in a separate off-line texture. We use the GPU Jump Flooding Algorithm by Rong & Tan [RT06] to interactively recompute the 2D distance field every frame. After the computation of the distance transform, the texture stores the distance to the contours of the shape for each pixel. Then, while computing occlusion queries in the fragment shader, we simply look-up the distance for the current fragment, and discard instances according to the user-defined aperture coefficient.

5.3. Contextual Anchoring

When observing still images of cut-away scenes, it might be challenging to perceive the depth of objects correctly, despite using lighting-based depth cues. We propose an additional method for depth guidance, which we call *contextual anchoring*. The concept is to override the results of clipping, to preserve elements located in proximity to non-clipped elements that would normally be clipped. This principle in shown in Figure 7, where we can observe parts of the green membrane anchored around channel molecules, and which indicate that they are located on the surface of the object. We were able to procedurally reproduce this effect by applying a depth bias to the occlusion queries computation for selected focused molecules. This bias will ensure that contextual elements will no longer overlap the focus and will therefore be preserve as illustrated in Figure 8.

6. Equalizing Visibility

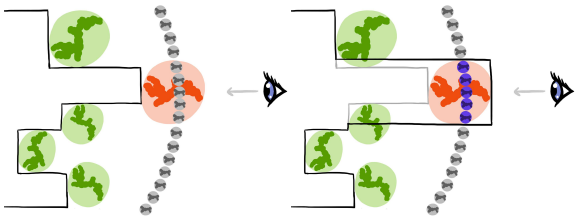
The visibility equalizer comprises a series of stacked bars that convey important visibility information for each ingredient type. The three colors correspond respectively to the number of visible, occluded and clipped instances, as explained in Figure 4. In order to fill the stacked bars with correct values, we must count the number of clipped and visible instances, and this operation must be repeated on every update.

6.1. Counting Clipped Instances

We perform the counting of the clipped instances on the GPU, in a dedicated compute shader program, since all the data already reside in the video memory. We previously declare a dedicated buffer on the GPU to hold the number of clipped instances for each ingredient type, and which shall be cleared before each counting operation. Counting the clipped instances is a rather straightforward task since the clipping state of each instance is routinely computed and stored in a dedicated GPU buffer. Once the clipping routine is performed, we simply browse through all the instances, and if an instance is flagged as clipped, we increase the counter in the GPU buffer that corresponds to the number of clipped instances for the given type. It is important to use an atomic increment operation for the counting to avoid concurrent accesses to the same counter value from different threads.

6.2. Counting Visible Instances

In order to count the number of visible instances for a given viewpoint, we first need to generate an instance buffer, which is a texture that contains, for each pixel, the unique instance id of the rendered molecule. We first start to flag visible instances in a post-processing shader, by browsing all the pixels of the instance buffer. In case an instance is present in the instance buffer, it is guaranteed to have at least one pixel visible on the screen, and it is therefore flagged as



(a) (b)

Figure 8: The principle of the depth-bias used for contextual anchoring. The dark bars represents the depth values of the mask from the side, in one dimension. Elements in grey correspond to potential occluders, while elements in red and green correspond to occludees. The red type is subject to contextual anchoring. (a) Without contextual anchoring, the depth of occluders (grey) is overlapping the depth of the mask and will therefore be discarded. (b) With contextual anchoring, the depth of the occludees (red) is shifted so that context elements (purple) no longer overlap the focus and remain unclipped.

visible in a dedicated GPU buffer. To store the number of visible instances per type, we also need to declare an additional GPU buffer, which must be previously cleared each time visible instances are counted. In a second stage, similarly to the counting of the clipped instances, we browse through all the instances in a dedicated compute shader, while fetching the visibility information which was previously computed. Should an instance be flagged as visible, the counter that corresponds to the number of visible instances for the given type will be increased using an atomic increment operation. Once the information about the number of visible and clipped instances is obtained, the data is then transferred to the CPU and used to update the visibility equalizer.

7. Results and Performance Analysis

To showcase the capabilities of our method, we applied it to three different mesoscale molecular scenes. For the rendering, we used cellVIEW [LAPV15], a tool designed to efficiently render large molecular scenes on the GPU and implemented with Unity3D. The different datasets have been generated by the domain experts with cellPACK [JAAA*15], a modeling tool for procedural generation of large biomolecular structures. cellPACK summarizes and incorporates the most recent knowledge obtained from structural biology and system biology to generate comprehensive mesoscale models. Based on experimentally obtained data (such as proteins structure, concentration and spatial distribution), the tool is able to generate entire models of viruses and cells via a packing method based on collision constraints.

The first dataset is a model of an HIV particle in blood serum that contains 20502 instances of 45 different protein types and 215559 instances of lipid molecules. In Figure 9a, we show an example of a single clipping plane used to reduce the concentration of the blood serum molecules, so that the HIV proteins are visible. However, to avoid misleading the viewer about the actual concentration of the blood molecules, we render clipped proteins with a ghosting effect. This communicate true information about the concentration, while reducing visual clutter caused by the dense arrangement of blood serum proteins. Figure 1 shows sequential step for production a comprehensible cut-away illustration with the HIV dataset.

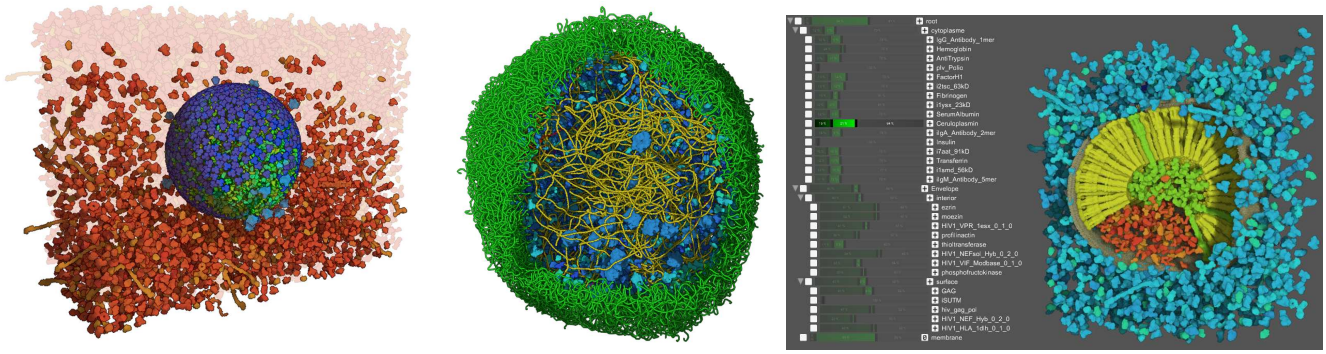
The second dataset is a model of *Mycoplasma mycoides* that contains 5380 proteins of 22 different types. Figure 9b shows how fuzzy-clipping is used to reduce visual clutter to illustrate the positions of the ribosomes (shown in blue) within the cell.

The third dataset, shown in Figure 9c is a model of an immature HIV which contains 1081 instances of 13 different protein types. We applied several clipping objects to reveal the internal structure of the virus. The blood serum (blue) has been preserved around the particle using the fuzzy clipping to illustrate how it encloses the HIV particle. The visibility equalizer is displayed as well, showing the ratios of visible and clipped instances of the individual molecular ingredients. The white boxes to the left of each stacked bar are used to mark the given ingredient or compartment as focus.

Figure 10 shows the mature HIV dataset clipped with a single plane. The contextual anchoring is applied to reintroduce parts of the clipped membrane (grey) around the envelope proteins (blue).

The visibility equalizer is designed to limit the computational overhead in order to offer a fast and responsive user experience. To demonstrate the responsiveness of our method, we measured the computation time for the object-space clipping, view-space clipping and 2D distance transform, respectively. The application was running on a machine equipped with an Intel Core i7-3930 CPU 3.20 GHz machine coupled with a GeForce GTX Titan X graphics card with 12GB of video RAM. The computation of the object-space clipping, compared to the rendering task performed by cellVIEW, is very lightweight and does not impact the overall performance too much. It took **0.3 milliseconds** to evaluate the 236061 instances of the HIV + blood dataset without clipping any of them. It took **0.5 milliseconds** in total to slice the dataset in half and **0.6 milliseconds** to clip it entirely. The increasing cost corresponds to the writing operations to the video memory, which are performed when an instance is clipped. It is important to mention that neither the shape of the clipping object nor the number of clipping objects have a meaningful influence on the performance.

The view-space clipping, however, requires more computational work that could impact the responsiveness. Indeed, for computing occlusion queries, occluders and occludees must be additionally rendered, which adds extra work to the rendering pipeline. For this reason, only the bounding spheres of the molecules are rendered instead of their entire structures, which may consist of hundreds or thousands of spheres, in order to guarantee a minimal computational overhead. We measured **0.07 milliseconds** for rendering the depth-stencil mask with 12142 instances (HIV proteins), and **0.57 milliseconds** for the computation of the 223919 occlusion queries corresponding to the remaining objects of the scene (blood proteins + lipid residues). Additionally, the 2D distance transform that is needed for the aperture effect also requires additional computation. It took **0.15 milliseconds** for computing the distance transform of the previous depth-stencil mask at a resolution of 512 by 512 pixels. Unlike object-space clipping, the view-space clipping computation cost will keep increasing with additional operations. Therefore, it



(a) (b) (c)

Figure 9: Advanced clipping options in real test systems. (a) A falloff function is used to gradually clip serum molecules (red) from bottom to top to reveal the HIV capsid, with ghosting to give cues about the overall concentration. (b) Selective clipping is used to reveal the location of ribosome (blue) in a model of Mycoplasma mycoides. (c) Internal structures of a immature HIV model are shown by several clipping objects. On the left, the visibility equalizer is shown.

is a good strategy to keep a low number of view space clipping objects, especially with very large scenes.

8. User Feedback

We evaluated the usefulness of our tool by collecting informal user feedback from domain experts in biological illustration and molecular biology. In both cases, we did a remote walk-through introduction of our software, while collecting first impressions. Additionally, we gave them an executable version of the software and asked them to write a short summary of their experience after trying the tool by themselves. We first sent an early version of our tool to a biomedical illustrator with a strong academic background in chemistry. His overall feedback was very positive, he enjoyed the responsiveness of the tool, and the novel concept of fuzziness and gradient clipping. Here is a quote from his written feedback:

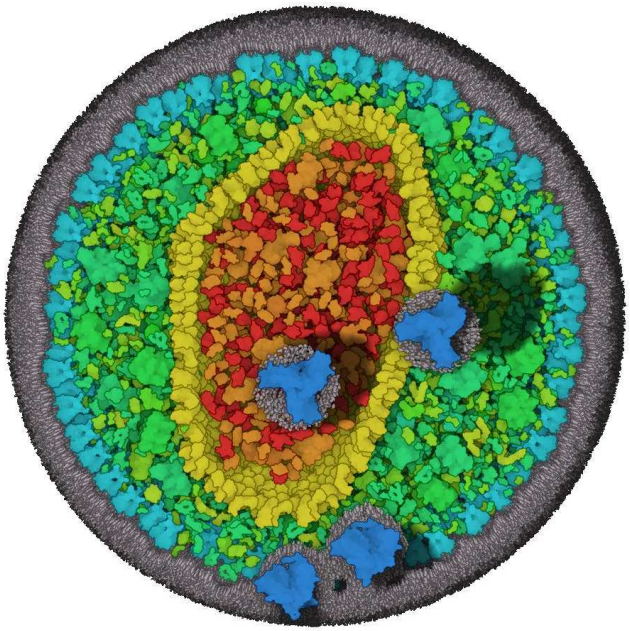


Figure 10: HIV clipped with a plane. Contextual anchoring is used to indicate the proximity of envelope proteins (dark blue) with the lipid membrane (grey) The dark spots represent shadows projected into interior proteins.

"...in my opinion it can be a very useful toolkit for an illustrator in the biomedical field...It also seems very promising for interactive teaching and also for animation purposes... One very useful feature of the software is the possibility to "cut" planes of interest of a particular space, and keeping the information of all "layers" by creating a "gradient" of concentration of the ingredients of the displayed molecular recipe. A visualization that resembles an "exploded model" but for biological assembly and it can be achieved without manually selecting every instance you would like to hide."

Secondly, we interviewed an expert in the domain of molecular biology and visualization. For this second interview, the overall feedback was also quite positive. He greatly enjoyed how easy and fast it was to perform clipping, and also enjoyed the user interface for manipulating the cut object parameters. He also wished for several additional features to improve the usability of the tool, such as filtering based on biomolecular properties and rendering the ghosts of the clipped instances. These features have since been implemented in the current version of the software, as seen in Figure 9a. Here is a quote from the written feedback we collected:

"...The aperture cutting feature is especially useful for exploring a feature or object in the context of a crowded molecular environment. The ability to retain a subset of the clipped objects ("fuzzy clipping") is an interesting feature that could be very useful under certain circumstances. The feature is useful if one wants to get an impression of reducing the concentration of some of the molecular ingredients, or of what a gradient of certain molecular ingredients would look like."

9. Conclusion and Future Work

In this paper, we present a novel method for authoring cutaway illustrations of mesoscopic molecular models. Our system uses clipping objects to selectively remove instances based on their type and location. To monitor and fine-tune the process, we introduce the visibility equalizer. It keeps the user informed about the number of molecular instances removed by the clipping objects, or occluded in the current viewport. Moreover, the visibility equalizer allows the users to directly override the behaviour of the clipping objects in order to fine-tune the visibility of molecular ingredients within the scene.

The visibility equalizer concept demonstrates a scenario where a visualization metaphor, such as the stacked bar chart, can serve as a user interface for performing a specific task, in our case to manipulate 3D data to authorize cutaways. The method allows users to create comprehensive illustrations of static biological models in realtime. This was confirmed by gathering feedback from domain experts. While the concept was applied to a specific domain, we also wish to develop other examples where the (information) visualization would act simultaneously as an interface to steer the view.

There are also several follow-up ideas which we would like to focus on in the future, to strengthen data exploration and showcasing with cellVIEW. Firstly, we would like to explore automatic clipping mechanisms to assist the user with the placement of clipping objects based on the nature of the scene and shape analysis. Secondly, we would also like to try our visibility equalizer concept with time-dependent datasets and enhance it to provide the means for authoring illustrations of dynamic datasets.

Our Visibility Equalizer method is built on top of cellVIEW and Unity3D, which are both free to use for non-commercial use, the source code is publicly available, as well as the showcased scenes modelled with cellPACK (https://github.com/illvisation/cellVIEW).

Acknowledgement

This project has been funded by the Vienna Science and Technology Fund (WWTF) through project VRG11-010 and also supported by EC Marie Curie Career Integration Grant through project PCIG13-GA-2013-618680. Johannes Sorger has been partially supported in the scope of the FWF-funded project P24597-N23 (VISAR) and the COMET K1 program of the Austrian Funding Agency (FFG). Ludovic Autin received support from the National Institutes of Health under award number P41GM103426. We would like to thank Aysylu Gabdulkhakova and Manuela Waldner for insightful comments.

References

[BF08] BURNS M., FINKELSTEIN A.: Adaptive cutaways for comprehensible rendering of polygonal scenes. In *SIGGRAPH Asia* (Singapore, 2008), ACM, pp. 154:1–154:7. 3

[BGCP11] BAER A., GASTEIGER R., CUNNINGHAM D., PREIM B.: Perceptual evaluation of ghosted view techniques for the exploration of vascular structures and embedded flow. *Computer Graphics Forum 30*, 3 (2011), 811–820. 3

[BGKG05] BRUCKNER S., GRIMM S., KANITSAR A., GRÖLLER M. E.: Illustrative context-preserving volume rendering. In *Proceedings of the Seventh Joint Eurographics / IEEE VGTC Conference on Visualization* (Leeds, United Kingdom, 2005), EUROVIS'05, pp. 69–76. 3

[BHW*07] BURNS M., HAIDACHER M., WEIN W., VIOLA I., GRÖLLER M. E.: Feature emphasis and contextual cutaways for multimodal medical visualization. In *Proceedings of the 9th Joint Eurographics / IEEE VGTC Conference on Visualization* (Sweden, 2007), EUROVIS'07, pp. 275–282. 3

[CM11] CORREA C., MA K.-L.: Visibility histograms and visibility-driven transfer functions. *Visualization and Computer Graphics, IEEE Transactions on 17*, 2 (Feb 2011), 192–204. 3

[DMNV12] DÍAZ J., MONCLÚS E., NAVAZO I., VÁZQUEZ P.: Adaptive cross-sections of anatomical models. *Computer Graphics Forum 31*, 7 (2012), 2155–2164. 3

[DWE02] DIEPSTRATEN J., WEISKOPF D., ERTL T.: Transparency in interactive technical illustrations. *Computer Graphics Forum 21*, 3 (2002), 317–325. 3

[DWE03] DIEPSTRATEN J., WEISKOPF D., ERTL T.: Interactive cutaway illustrations. *Computer Graphics Forum 22*, 3 (2003), 523–532. 3

[FKE13] FALK M., KRONE M., ERTL T.: Atomistic visualization of mesoscopic whole-cell simulations using ray-casted instancing. *Computer Graphics Forum 32*, 8 (2013), 195–206. 3

[FS92] FEINER S., SELIGMANN D.: Cutaways and ghosting: satisfying visibility constraints in dynamic 3d illustrations. *The Visual Computer 8*, 5-6 (1992), 292–302. 3

[JAAA*15] JOHNSON G. T., AUTIN L., AL-ALUSI M., GOODSELL D. S., SANNER M. F., OLSON A. J.: cellPACK: a virtual mesoscope to model and visualize structural systems biology. *Nature methods 12*, 1 (Jan. 2015), 85–91. 2, 4, 7

[KSW06] KRÜGER J., SCHNEIDER J., WESTERMANN R.: Clearview: An interactive context preserving hotspot visualization technique. *Visualization and Computer Graphics, IEEE Transactions on 12*, 5 (Sept 2006), 941–948. 3

[KT14] KUBISCH C., TAVENRATH M.: Opengl 4.4 scene rendering techniques. *NVIDIA Corporation* (2014). 5

[KTH*05] KRÜGER A., TIETJEN C., HINTZE J., PREIM B., HERTEL I., STRAUSSG.: Interactive visualization for neck-dissection planning. In *Proceedings of the Seventh Joint Eurographics / IEEE VGTC Conference on Visualization* (Leeds, United Kingdom, 2005), EUROVIS'05, pp. 295–302. 3

[LAPV15] LE MUZIC M., AUTIN L., PARULEK J., VIOLA I.: cellVIEW: a tool for illustrative and multi-scale rendering of large biomolecular datasets. In *Eurographics Workshop on Visual Computing for Biology and Medicine* (Sept. 2015), pp. 61–70. 4, 7

[LBH12] LINDOW N., BAUM D., HEGE H.-C.: Interactive rendering of materials and biological structures on atomic and nanoscopic scale. *Computer Graphics Forum 31*, 3 (2012), 1325–1334. 3

[LGV*16] LAWONN K., GLASSER S., VILANOVA A., PREIM B., ISENBERG T.: Occlusion-free blood flow animation with wall thickness visualization. *Visualization and Computer Graphics, IEEE Transactions on 22*, 1 (Jan 2016), 728–737. 3

[LHV12] LIDAL E. M., HAUSER H., VIOLA I.: Design principles for cutaway visualization of geological models. In *Proceedings of Spring Conference on Computer Graphics (SCCG 2012)* (May 2012), pp. 53–60. 3

[LMPSV14] LE MUZIC M., PARULEK J., STAVRUM A.-K., VIOLA I.: Illustrative visualization of molecular reactions using omniscient intelligence and passive agents. *Computer Graphics Forum 33*, 3 (2014), 141–150. 3

[LRA*07] LI W., RITTER L., AGRAWALA M., CURLESS B., SALESIN D.: Interactive cutaway illustrations of complex 3D models. In *SIGGRAPH '07* (San Diego, California, 2007), ACM. 3

[RBB*11] RUIZ M., BARDERA A., BOADA I., VIOLA I., FEIXAS M., SBERT M.: Automatic transfer functions based on informational divergence. *Visualization and Computer Graphics, IEEE Transactions on 17*, 12 (Dec 2011), 1932–1941. 3

[RT06] RONG G., TAN T.-S.: Jump flooding in gpu with applications to voronoi diagram and distance transform. In *Proceedings of the 2006 Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2006), I3D '06, ACM, pp. 109–116. 6

[SFCP12] SIGG S., FUCHS R., CARNECKY R., PEIKERT R.: Intelligent cutaway illustrations. In *Visualization Symposium (PacificVis), 2012 IEEE Pacific* (Feb 2012), pp. 185–192. 3

[VG05] VIOLA I., GRÖLLER E.: Smart visibility in visualization. In *Computational Aesthetics in Graphics, Visualization and Imaging* (2005), The Eurographics Association. 3

[Vio05] VIOLA I.: *Importance-Driven Expressive Visualization*. PhD thesis, Institute of Computer Graphics and Algorithms, Vienna University of Technology, Favoritenstrasse 9-11/186, A-1040 Vienna, Austria, June 2005. 3

[VKG04] VIOLA I., KANITSAR A., GRÖLLER M. E.: Importance-driven volume rendering. In *Proceedings of the Conference on Visualization '04* (Washington, DC, USA, 2004), VIS '04, IEEE Computer Society, pp. 139–146. 3

[VKG05] VIOLA I., KANITSAR A., GRÖLLER M. E.: Importance-driven feature enhancement in volume visualization. *Visualization and Computer Graphics, IEEE Transactions on 11*, 4 (July 2005), 408–418. 3

[WEE03] WEISKOPF D., ENGEL K., ERTL T.: Interactive clipping techniques for texture-based volume visualization and volume shading. *Visualization and Computer Graphics, IEEE Transactions on 9*, 3 (July 2003), 298–312. 3