Eurographics Conference on Visualization (EuroVis) 2016
K-L. Ma, G. Santucci, and J. 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], and I. Viola[1]

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

Abstract

In scientific illustration and visualization, cutaway views are often employed as an effective technique for occlusion management in densely packed scenes. We propose a novel data-centric 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 that are distributed across a comparably smaller number of different molecular types. Our method constitutes a two stage process. In the first step, culling objects are placed in the scene, creating a cutaway visualization of the model. During this process, histograms 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 histograms, which at this point act as interactive visibility equalizers. The technique has been evaluated by domain experts in scientific illustration.

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

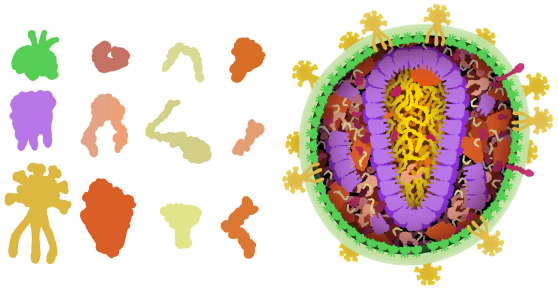
1. Introduction

Molecular biology is an emerging field that is marked by a constant evolution of the current state of knowledge. New discoveries have to be communicated frequently to a large variety of audiences. Since discoveries of complex molecular phenomena at a mesoscopic scale cannot directly be conveyed, illustrations that depict models of these structures are the only way of communicating them.

The traditional pipeline for creating scientific illustrations of molecular structures starts with gathering the knowledge that is required for building models that convey the newly discovered insights. When sufficient knowledge is gathered, illustrators create sketches, in which specific internal and external regions of the illustrated structures are uncovered. To achieve this, occlusion management techniques, such as *cutaways* are applied. Cutaways remove specific parts of the organism model, so that internal structures become visible. When biologists make new findings, the conceptual layout of the original illustration might not be valid anymore. The whole illustration process has to be repeated from the beginning. Such a cycle can take months or even years to complete.

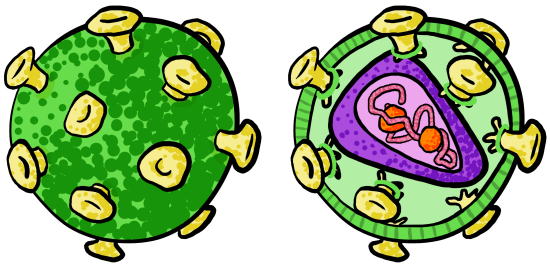
Considering the rapid evolution of knowledge in the field of biology, it is necessary to adapt the traditional illustration pipeline so that new data can be easily plugged in and resulting illustrations can be updated in a very short time. Virtual 3D models of cells and other mesoscale molecular structures can be utilized for these purposes. Biologists use tools, such as *cellPack* [JAAA*15], to procedurally generate these models. Based on a set of input parameters, individual molecules are assembled into large complexes or even entire systems, such as bacterial organisms. Such a parameter set consists of a specification of individual molecular ingredients, as well as spatial compartments that define where the instances of these ingredients are populated. The resulting 3D model can consist of several thousands of instances of these molecular ingredients. The instances are densely packed within the predefined compartments.

† Both first authors contributed equally. Contact: {mathieu | mindek}@cg.tuwien.ac.at



(a) (b)

Figure 1: Which of the proteins shown in (a) are visible in the illustration shown in (b) and which are not?



(a) (b)

Figure 2: (a) Illustration of a HIV virus. Here, outside membrane of the virus particle is visible. (b) Cutaway view of the HIV virus. Despite the cutaway, some of the glycoproteins (yellow molecules) are kept in the view to provide adequate context.

The mesoscale biological models represent the structure of microorganisms, cells, or even viruses at atomic resolution. However, simply displaying such models does not guarantee an adequate view of internal structures. These structures are often the key for the function of the organism, and should be therefore shown in an illustration. The internal structures are occluded because of the high density of the molecular instances present in the models. To solve this problem, visualization techniques need to be developed which reproduce the occlusion management methods used in traditional illustration.

Currently, occlusion management in mesoscale virtual models is carried out by placing clipping objects in the scene, which remove specified parts of the displayed model. During this process, the illustrator does not have a good overview of what instances have been already removed, and which molecular types are still sufficiently represented in the scene. The illustrator has to continuously check the modelled scene against the gathered data and tediously confirm whether all the necessary molecular types are still present.

As a tangible example, let us assume a task where multiple molecular ingredients need to be shown in an illustration. By clipping through the center of a 3D model, many structures can be revealed. However, to find out which of the desired structures have been actually shown by the clipping and which not, one would need to manually check the presence of each single ingredient in the resulting visualization. This situation is illustrated in Figure 1.

To alleviate this process, we present our first contribution. During the process of placing the clipping objects in the the scene, we display *visibility histograms* of the molecular types, which immediately reveal which of them are underrepresented or overrepresented. By looking at the visibility histograms, which are continuously updated, the illustrator is able to modify the placement of the clipping objects in such a way that every molecular type is adequately represented in the scene. This is the coarse-level of the visibility specification process.

In traditional illustration, fine-level visibility specification is often utilized as well. To communicate the biology knowledge well, the illustrations have to sometimes display molecular instances which would be impossible to specify with the simple clipping objects, such as cutting planes. An example is shown in Figure 2. Figure 2a shows an illustration of a HIV virus. In Figure 2b, a cutting plane is used to reveal internal structures of the virus - the capsid containing the RNA. Some of the glycoproteins (yellow molecules) are left in the illustration to communicate their presence on the surface of the virus particle. In particular, those glycoproteins which are not occluding the object of interest, were chosen to be kept in the illustration providing the contextual information. 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 pick individual molecular instances to be reintroduced or removed from the scene. This might be done to control the under and overrepresentation of some of the molecular types, removing instances occluding important aspects of the model, suggesting shapes, etc.

To significantly speed up the fine-level visibility specification in our approach utilizing 3D virtual models, we propose our second contribution - *visibility equalizers*. To explain how the visibility equalizers are used to speed up the process of fine-tuning the visibility in molecular models, we use the metaphor of hi-fi sound reproduction. In the hi-fi sound systems, volume control is the basic tool for adjusting the output sound uniformly on all frequencies. This corresponds with the coarse-level visibility specification through clipping objects in the molecular scenes, where all molecular types are uniformly removed from the clipped regions. However, hi-fi sound system allows users to fine-tune the sound through *equalizers*. With equalizers, the volume of each individual frequency band can be adjusted separately to achieve desired

sound during the reproduction. To achieve similar level of control for the visibility in the molecular models, we make the visibility histograms interactive. Individual bins of the histograms can be dragged to increase or decrease visibility of the individual molecular types within the scene, given the specified clipping objects. The interactive element effectively turns the visibility histograms into visibility equalizers for the molecular models.

Our main contributions are:

- a new workflow for illustrator-authored cutaway illustrations from mesoscale 3D structural models
- a new visual metaphor of visibility equalizers with which allows users to fine tune the cut-away design so that visibility is distributed among the molecular types as desired by the illustrator.

2. Related Work

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 Finer & Seligmann [FS92] in 1992 as an automated approach for generating illustrations that consider the occlusion of user defined objects. In 2002, Diepstraten et al. [DWE02] picked up the technique again and defined a set of rules for computer-based rendering of technical illustrations to achieve a view-dependent transparency model that mimics the ghosting techniques of technical illustrations. They later extended these rules for interactive cutaway illustrations [DWE03].

Analogous to the cutaways for polygonal representations, Weiskopf et al. [WEE03] developed an interactive clipping technique for volume rendering that supports complex clipping geometries. In 2004, Viola et al. [VKG04] developed an automated approach for focus & context visualization for segmented volumetric objects. An assigned object importance determines the visibility priority for the segmented parts of the volume. Contextual information is kept in regions where the context does not occlude the feature of interest. Follow-up work focused on the definition of levels of sparseness and importance compositing for cutaway and ghosting calculations [VKG05]. In 2005, Viola & Gröller [VG05] give an overview of current "smart visibility" techniques. The term describes expressive visualization techniques that smartly uncover the most important features of the displayed data, such as cut-away views, ghosted views, and exploded views. 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. **[which part of your phd thesis should I highlight?]**

A. Krüger et al. [KTH*05] combined visualization and interaction techniques such as cutaway views, silhouettes and color-coded distances to improve the spatial perception of feature arrangement for surgical planning. lymph nodes are emphasized using ghosted views to easily convey their spatial position. J. 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] developed an approach that allows interactive exploration of complex models, e.g., mechanical or anatomical, that requires the user to rig each part of the respective model. Based on the rigging, the system produces cuts that adhere to a set of rules that were inspired by anatomic and mechanical illustrations. The approach by Burns & Finkelstein [BF08] for view dependent cutaways inspired our aperture that is discussed in section XXX. The cutaway shape is determined by the enlarged shape of the focus objects in the depth image. To preserve the information of the cut geometry, they apply shading and contouring/outlining of the cut surfaces, as well as ghosting of the cut geometry contours. 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. The structures visually encode the wall thickness as colored regions in order to preserve important context information. A view dependent peel-away approach for volume data was proposed by Birkeland and Viola [BV09]. 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.

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 design principles for cutaway visualization of geological models. They promote boxes as ideal cutaway shapes for emphasizing the shape and depth of focus features in layered structures, such as geological sediments. Illumination should effectively communicate the shape and spatial ordering inside the cutaway, as well as enhancing relationships between the focus features and the context. They define five design principles that we discuss in section XXX in relation to our approach.

Transfer Function Based Approaches. The context-preserving volume rendering model proposed by Bruckner et al. [BGKG05] is an extension of direct volume rendering. The technique uses a function of shading intensity, gradient magnitude, distance to the eye point, and previously ac-

cumulated 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 culling/ghosting.

Correa et al. [CM11] present visibility histograms for specification of transfer functions for volume rendering. In contrast with this work, the motivation for our method is to design an interface for authoring cutaway illustrations of molecular data. The properties of the data imply that each bin of our visibility equalizer, representing individual molecular ingredients, can be interacted with to change the properties of the clipping objects applied to the 3D scene.

Ruiz et al. [RBB*11] propose an approach for automatic transfer function optimization. The transfer functions are obtained by minimizing the informational divergence or Kullback-Leibler distance 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 solid structures that could be made visible with a TF (membranes, nucleus). On the other side, within these structures there is a more noise like distribution of these molecules that cannot be segmented into solid structures. In regard to object centered approaches, (partial) occlusion of individual molecules is not an issues as the data does not contain 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 belong to a couple of dozen molecule types. our approach is therefore fundamentally different from existing occlusion management approaches as it combines principles from object centered and transfer function approaches.

2.2. Visualization of Molecular Structures

The visualization of the molecular structures in our approach is based on the publicly available cellView [LAPV15]. The tool is capable of rendering structures that are comprised of several billions atoms at interactive frame rates in multiple levels of detail. Lindow et al. [LBH12] were the ïňĄrst to introduce a fast method for the real-time rendering of large-scale atomic data on consumer grade hardware. Similar to cellView, 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.

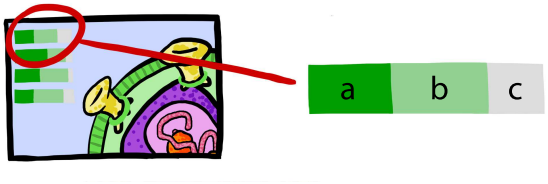


Figure 3: Visual representation of the visibility equalizers.

Other related work is concerned with illustrative molecular visualization. Grottel et al. [GKSE12] and Eichelbaum et al. [ESH13] propose ambient occlusion approaches for large molecular scenes in order to improve the depth perception in these complex structures. Parulek et al. [PJR*14] propose a continuous level of detail scheme for molecular data that offers gradual shape simplification for distant molecules based on a clustering of the atomic spheres.

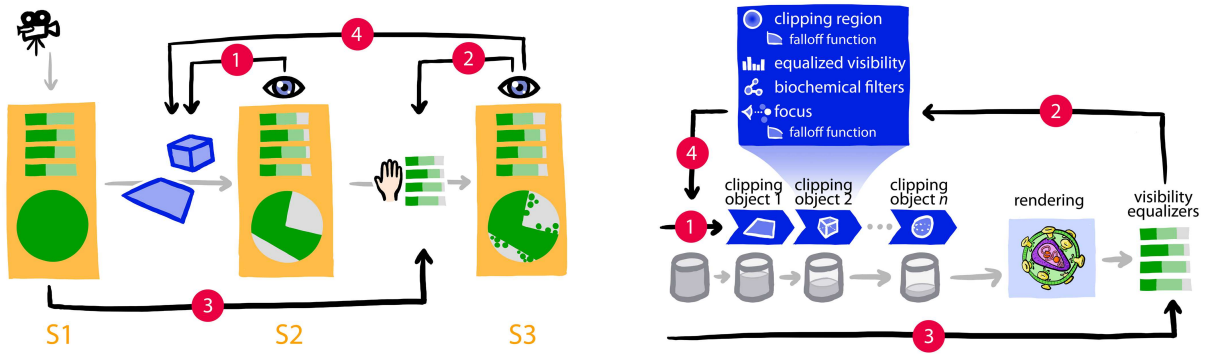
In the domain of dedicated large scale molecular visualization, our approach is the first to introduce illustrative occlusion management techniques.

3. Overview

In this work, we focus on visualization of 3D models of mesoscale molecular structures, such as cells or viruses. We utilize *cellView* [LAPV15] tool for both representation and rendering of our 3D scenes. These scenes can consist up to billions of individual atoms, each belonging to one of the molecular ingredients.

The two main components of our method are the *clipping objects* and the *visibility equalizer*. The visual encoding of the visibility equalizer is illustrated in Figure 3. It consists of a stacked histogram of the molecular ingredients, showing tree values per ingredient: a - the amount of visible instances of the given ingredient; b - the amount of instances of the ingredient which are occluded; c - the amount of instances which are clipped away by the clipping objects. The clipping objects are defined by distance functions, where the zero level set represents the clipping manifold. They can be arbitrarily positioned within the 3D scene. Additionally, the clipping objects contain several parameters, which modify the way in which they are clipping the molecules in the scene. In particular, each clipping object has a probability threshold v_1. v_1 specifies the probability of an instance inside the clipping manifold being clipped away. By default, $v_1 = 1$.

An important property of the clipping objects is that they always clip the scene per molecular instance. This means that the whole molecule is always either shown, or clipped away.



(a) (b)

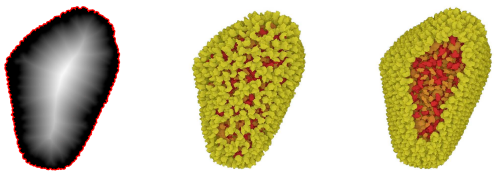
Figure 4: (a) The workflow of our method. The data can be displayed either without any clipping (*S1*), with deterministic clipping defined by the clipping objects (*S2*), or probabilistic clipping specified through the visibility equalizer (*S3*). (b) The technologies used in the workflow. The user can either use clipping objects to clip the data, or he can manipulate the visibility equalizers to modify clipping parameters of the clipping objects.

Figure 4 shows the workflow of our method in a form of a state-machine. In Figure 4a, three possible states are illustrated, denoted as *S1*, *S2*, and *S3*. In the state *S1*, the scene is displayed without any clipping. The visibility equalizer shows that no molecules are clipped, and provides information on the portions of visible instances of each ingredient.

Clipping objects can be used to transition to the state *S2*, where parts of the model are removed in a deterministic way. Here, the inside-outside test of the zero level sets of the clipping objects determine whether an instance is clipped away or not. The portions of the clipped instances are displayed by the visibility equalizer. The process of placing and manipulating clipping objects corresponds with the coarse visibility specification as introduced in Section 1.

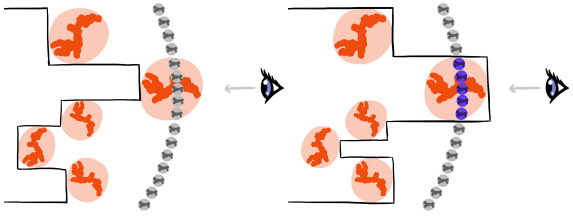
At any point, the user can interact with the visibility equalizer to either modify the amount of the visible instances in the scene by dragging any of the dark green bars in the histogram, or the amount of the clipped-away instances by dragging the light green bars of the histogram. When the bars in the histograms are dragged, probability thresholds of selected clipping objects are modified. This means the user is able to increase or decrease amounts of clipped away instances as well as the visible instances by dragging the visibility equalizer in a probabilistic manner. This situation is represented by the state *S3* and it corresponds with the fine-level visibility specification process.

Figure 4b represents the pipeline of the method. At the beginning of the pipeline, multiple clipping objects are placed into the 3D scene filtering the data. Each clipping object filters the output of the previous one. Clipped data are sent to the rendering stage, where the histograms of the visibility equalizer are calculated and displayed on each frame. In case the user interacts with the visibility equalizer, probability thresholds of the selected subset of the clipping object is modified, so that the 3D scene corresponds with how the user changed the values in the histograms. During the whole process of the visibility specification, the user is informed about the amounts of the visible and clipped-away instances in the scene through the visual encoding of the visibility equalizer.



(a) (b) (c)

Figure 6: View-dependent clipping.



(a) (b)

Figure 7: Islands.

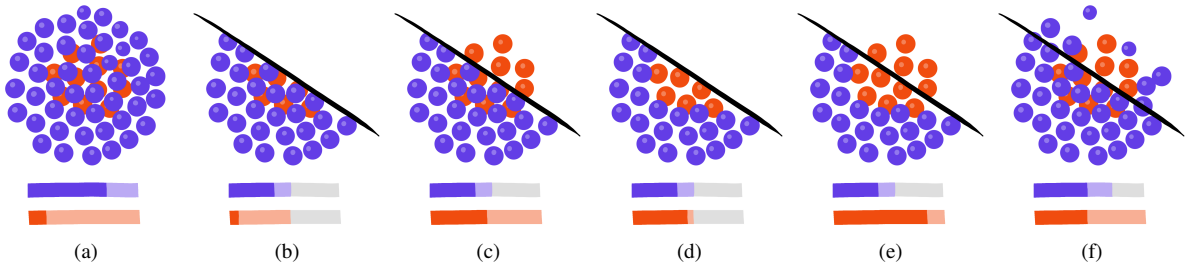


Figure 5: Visibility Equalizers.

4. Workflow

5. Object-Space Clipping

Clip objects define how many instances of a given ingredient type shall be displayed.

There are two non-exclusive ways how can a clip object influence the clipping, either in object-space or in view-space.

Using an object-space approach, the clip objects will discard instances independently from the view direction.

Clip objects are applied in serial as explained in Figure XX, This operation is recomputed every frame.

In this section we will explain in details how an individual clip object operate for the object-space clipping only.

The first step is the object-space clipping process is the localization of the clip sub-region.

Then, once the clip object is localized, instances will be clipped according to object-space clip parameters which we describe in the following subsections.

Moreover, we introduce advanced parametrization of the distance field falloff, for generating customizable gradient clipping effects.

5.1. Clip-region Localization

Clip objects can be associated with geometrical shapes to localize a sub-region of the domain which is influenced by the clipping.

Our system currently supports the following set of primitive shapes: plane, cube, sphere, cylinder and cone.

The first task of the object-space clipping is to determine whether an instance of a molecule is located in the sub-region defined by the clip-object geometry.

This operation is repeated every frame, for each instance of the scene.

To determine if an instance lies inside or outside the clip object region, we compute the signed distance between the instance bounding sphere and the closest point on the region surface.

Although supported shapes have a rather simple topology, it may still be computationally expensive, using a mesh-based representation, to compute a signed distance for a large number of instances.

Indeed, using a triangle-based discretization would imply computing the signed distance between the instances and every single triangle of the mesh.

To accelerate the computation we solve the problem analytically using a mathematical description of the 3D signed distance field (SDF).

Using such representation instead reduces the problem of evaluating the signed distance to solving trivial equations.

It is also possible to apply simple SDF operators to the distance field, such as translation, rotation and scaling.

The clipping region can also be reversed by inverting the result of the signed distance function, offering users flexibility.

For instance, using a spherical shape, the clip region would be set to the inside of the sphere by default, while in inverted mode it would correspond to the inside of the sphere.

It is worth mentioning that although the set of offered clip-shapes is yet limited it could easily be enriched by utilizing more complex SDF operators, such as union for instance, to merge several shapes together in one single distance field.

5.2. Clip Parameters

A clip-object comprise two basic parameters for each single ingredient that control the visibility of instances, based on their type.

It is worth mentioning that in case no shape is associated with the cull object, the clipping will be evaluated for the entire domain.

The first parameter is the percentage of visible elements of a given type.

We refer to this value as object-space clip probability.

This parameter allows us to control the degree of fuzziness of the clipping.

The other filtering parameters are related to biochemical properties and allows us to control the clipping based on the mass and/or quantity of given ingredient types.

These parameters can be interactively changed via the user interface.

****

Prior to the rendering, after localizing the clip region, each single instance is evaluated in order to determine if it shall be clipped.

First is applied filtering based on the clip probability.

For each instance, we compare a uniformly distributed random number with the clip probability of the instance ingredient type.

If the random number is higher than the probability, the instance is marked as culled, and will not be rendered.

The random number is initially set for each individual instance and remain the same for each re-evaluation of the clipping, in order to avoid getting different results each time.

Secondly, instances are filtered according to their biochemical properties, for each cut object and each ingredient type, the user defines range values for the both quantities and molecular weight.

Instances whose properties lie outside on these ranges are marked as culled and will not be rendered.

5.3. Falloff Function

We provide additional parameters to gradually remove instances given a geometrical shape, for illustration purposes.

TODO PMINDEK: Talk about gradient clipping here, motivation and parameters, maybe a figure too.

6. View-Dependent Clipping

While object-space clipping using primitive shapes allows for a great degree of flexibility, it requires cumbersome manual operations for complex set-ups, and is also limited in terms of shape diversity.

We additionally provide a functionality to specify a set of ingredient types as focus, and to selectively remove occluding instances.

We also provide a parameters to control the degree of fuzziness of the clipping.

Finally we introduce a falloff function inspired by our object-space falloff function that allows us to control the degree of aperture of the view-dependent clipping

6.1. Occlusion Queries

Due to the potentially large number of instances in our scenes, we accelerate the computation of occluding instances using an image-based approach on the GPU.

Modern graphics hardware already a fixed function called occlusion queries (OQ) and which allow to determine whether an instance is visible or hidden according to previously drawn geometries.

This approach, however, would require issuing one draw call per queries, which can seriously effect the frame rate when issuing hundred of thousands of queries, because of GPU driver overhead.

The rendering tool we are using already allows to render the entire scene in a single call to avoid latency due to the GPU driver.

Therefore, we extend the system using the programmable graphics pipeline instead, to implement custom occlusion queries without GPU driver overhead.

To determine the instances occluding, we priorly render an off-screen texture containing all the focus elements, which we will use as a depth-mask for the occlusion queries.

Instances are rendered using bounding sphere in order to lower to cost of the additional render pass.

Focus ingredients are priorly selected from the user interface.

There can be several ingredient types constituting the focus, however, only one focus mask can be generated per cut object.

Subsequently, we draw the bounding sphere of the remaining instances over the mask, fragments that will pass the depth test are therefore are guaranteed to belong to an occluding object.

From the fragment shader we then update the clip-state of the occluding instance using imageStore() functionality that allows us to write directly to the main video memory from the fragment shader.

The principle of depth queries is explained in Figure XX a.

6.2. Clip parameters

Similarly to object-space clipping we also provide an additional parameter to control the degree of fuzziness for the view-dependant clipping.

We dub this parameter view-dependant clip probability, and its functioning resemble the one of object-space clip probability.

This parameter is priorly set by the user for each protein type via the user interface.

Then, for each instance, after processing occlusion queries we would evaluate the clip probability with an uniformly distributed to determine the number of occluding instances that should remain visible.

A clip-probability of 0 would mean that all occluding instance of a given type shall be hidden, a probability of 0.5 that half of the instances shall be hidden, and a probability of 1 that all occluding instances should be visible, in other words, that no clipping should happen for that specific type.

6.3. Aperture Effect

Using the view-dependant clip probability value to discard an arbitrary number of occluding element will result in a uniform distribution of visible occluders over the focus.

By distributing occluders uniformly, however, we fragment the overall structure of the occluders compartment, which might not always be the best design choice to show focused ingredients nested in compartments.

Alternatively to uniform removal of occluders, we also propose a more artistic oriented option which we dub aperture effect.

We define an internal parameter, the aperture coefficient, which controls the distance from centre to edges of the mask, within which instances shall be removed.

A visual explanation of the aperture effect is shown in Figure X.

To enable this effect with compute the distance transform, in pixels, from the generated mask which we store in a separate texture.

We use the GPU Jump Flooding Algorithm by Rong & Tan [**?**] to interactively generate the distance transform of the mask.

After computation, the texture holds, for each pixel, the distance in pixels from the contours of the shape.

Then, while computing occlusion queries, we may discard instances in function of their distance to the contours of the mask by simply comparing this distance, with a user-defined threshold priorly set of each ingredient type.

7. Equalizing Visibility Histograms

To provide a clear overview of the scene properties, we display histograms for each ingredient type that indicate information about their visibility.

By default we chose to show three ranges for each histogram.

The first section of the histogram (dark green region) shows the percentage of instances that are currently visible on the screen.

The entire green section (dark & light green) represents the percentage of instances that are actually rendered.

In order to fill histograms with correct values, we perform book-keeping of both clipped and visible instances, which we recompute every frame.

Histograms are also interactive; the user can manipulate the visibility of the corresponding ingredient type by dragging the range handles, thus increasing or decreasing the the number of displayed elements.

7.1. Book-Keeping

The equalizing of the histogram requires to know for each single ingredient type, how many instances have been culled and how many instances are actually visible on the screen.

It is worth mentioning that our system leverages the power of the GPU to compute the clip-state of each instance every frame, thus offering a smooth and responsive user experience.

Therefore the current clip-state of every instance is stored in the GPU memory.

In order to avoid overhead due to data transfer between CPU and GPU, we perform book-keeping on the GPU using atomic operations.

Atomic operations are parallel programming feature which guarantee mutual exclusion when simultaneous thread wish to write in the same memory location.

We priorly declare a buffer to store the number of clipped and visible instances for each ingredient type.

To obtain the number of clipped instances, we simply increment the corresponding counter, each time an instance has been discarded using an atomic addition function.

For computing the number of visible instances, we first need information about actual visibility of each single instance.

We render an additional off-screen texture where each pixel contains the internal ID of the displayed instances.

We also declare an additional buffer to store a flag for each instance, which indicates if an instance is visible of hidden.

Then by simply browsing through each pixel of the aformentioned ID texture in an additional pass, we may simply update the visibility flag for the ID contained in each pixel.

Finally, similarly to the counting of clipped instances, we browse through each instance, and increment the corresponding counter accordingly to the visibility flag using an atomic addition function.

7.2. User Interaction

Upon manipulation of histogram ranges the system will either increase or decrease the number of clipped instances that correspond to the histogram ingredient type.

This is intended to optimise the way user interact with the system, by offering the user to directly manipulating quantities rather than abstract internal values such as the clip probability.

Additionally the user may also manipulate more advanced parameters in an additional UI panel.

We chose to map the motion of the histogram handles the clip probabilities.

The first handle manipulates the view-dependent clip probability, while the second handle controls the object-space one.

It is worth mentioning that the clip-probabilities that are manipulated by the user correspond to the currently selected clip-object.

The histogram view, however, remains the same when a different clip-object is selected.

*****

Quantities are relative by default, i.e, they represent a percentage of the total number of instances for a given ingredient.

However, thanks to our design choices, they can also be displayed as absolute quantities with limited additional effort.

For displaying absolute quantities we support logarithmic scaling to ensure ingredients present in low quantities to be visible in the histograms.

An logarithmic ruler is also provided to help the user understanding the displayed values

*****

It is also worth mentioning that histograms are displayed in a tree layout where nodes correspond to compartments and leaves to ingredients.

Additional histograms are therefore displayed for each compartment by averaging the values of the ingredients contained inside.

Upon manipulation of compartment histograms properties, all the children ingredients will be updated accordingly.

8. Enhancements

Additionally to standard clipping functionality we developed a few enhancements to improve the way we perceive the scene.

A crucial element when dealing with cluttered scenes is depth perception.

Depth cues can be implemented by imitating the way light interact in reality to help us understanding distances between objects.

However simulation illumination is rather expensive and prohibits a responsive user experience.

We implement simplistic depth cues instead with a rough approximation of light behaviour.

Additionally when provide option for context preserving view-dependent cut for a better understanding of the distance between objects in the view direction.

Finally we perform rendering of the clipped element as ghosts in order to better communicate the overall proportions of visible elements in the scene.

8.1. Depth Cues

Depth cues are essential in molecular visualization and there exists many reference in the literature on how improve depth perception for that specific case.

The first depth cue that we support is screen space ambient occlusion (SSAO) which allows to mimicate how global illumination works on the local scale, in image-space by evaluating for each pixel the depth of the surrounding pixels.

Similarly to [**?**], we support two levels of ambient occlusion with different search radius to provide illumination for different levels of magnification.

In addition to the limited range of the effect, SSAO does not account for directional light effects, which mean that shadows cannot be casted side-ways.

When performing cut away however we perform strong shape alteration of the dataset which might be hard to perceive without shadows.

Therefore, we additionally provide shadow mapping to better communicate the overall shape of the cuts.

A downside of this approach is that it require an additional draw pass for each light that casts shadows onto molecules.

Alternatively we also provide a cheaper method to provide additional cues about the shape of the clip objects.

TODO: PMINDEK, provide details about cheap depth cues

8.2. Depth Guidance

When observing still renders of a cut-away scene, it might still be challenging to perceive the depth of objects correctly, despite lighting-based depth cues.

We propose and additional method for depth guidance,

which modifying the results of the clipping to preserve elements, located in a close range to non-clipped elements, that would normally be clipped.

This way, assuming that the viewer is aware of what has been clipped, will intuitively understand where instances neighbouring the preserved instances are located.

This principle in shown in Figure XX, where we can observe bits of the green membrane preserved around channel molecules, and which indicate that the channel molecules are located on the surface of the object.

We are able to obtain a similar effect by modifying the fact we build the focus mask for occlusion queries.

A explanation of the depth guidance effect is shown in Figure XX.

By default the depth test use for rendering the mask is using the depth test "GREATER EQUAL" which will retain the deepest fragment only for the depth mask, as explained in Figure XX a.

Using a depth test condition "LESS EQUAL" allows us to keep the only closest fragments from the view when drawing the bounding sphere of of the focus elements.

Therefore, as the lipid instances of the membrane are always located beneath the bounding spheres of the channel proteins in view space, those instance are guaranteed to be preserved.

The principle of the inverted view mask is explained in figure XX b.

8.3. Clipping Ghosts

While histograms allow us to visually monitor the quantities of removed instances, it might still be interesting to convey this information in the 3D scene while preserving visibility settings.

We additionally provide the option to render ghosts of all the clipped instances on top the scene, using alpha blending.

We render all the ghosts in a separate off-screen texture which we later blend with the results of the scene rendering.

The blending of the scene with the ghost texture is designed to ensure that the depth of the ghosts texture will merge with the scene's depth, and thus, that occluded ghosts will not be visible in the final results.

We offer two rendering style for the ghosts, coloured filled or contour only.

For each ingredient type we offer the option to render ghosts or not via the user interface.

The resulting render of a scene with clipping ghosts can be seen in figure XX.

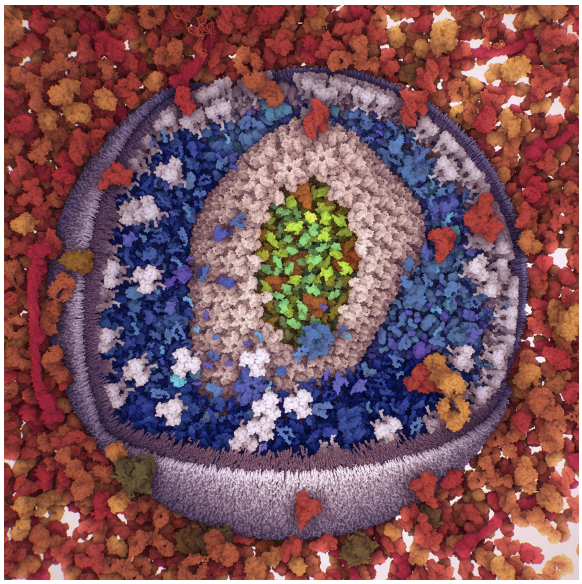


Figure 8: An illustration of the HIV virus in the blood serum utilizing cutaways created with our approach.

9. Results and Discussion

10. Evaluation

11. Conclusions

References

[BF08] BURNS M., FINKELSTEIN A.: Adaptive cutaways for comprehensible rendering of polygonal scenes. In *ACM SIGGRAPH Asia 2008 Papers* (New York, NY, USA, 2008), SIGGRAPH Asia '08, 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* (Aire-la-Ville, Switzerland, Switzerland, 2005), EUROVIS'05, Eurographics Association, 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* (2007), EUROVIS'07, Eurographics Association, pp. 275–282. 4

[BV09] BIRKELAND A., VIOLA I.: View-dependent peel-away visualization for volumetric data. In *Proceedings of the 25th Spring Conference on Computer Graphics* (New York, NY, USA, 2009), SCCG '09, ACM, pp. 121–128. 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. 4

[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

[ESH13] EICHELBAUM S., SCHEUERMANN G., HLAWITSCHKA M.: PointAO - Improved Ambient Occlusion for Point-based Visualization. In *EuroVis - Short Papers* (2013), Hlawitschka M., Weinkauf T., (Eds.), The Eurographics Association. 4

[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. 4

[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

[GKSE12] GROTTEL S., KRONE M., SCHARNOWSKI K., ERTL T.: Object-space ambient occlusion for molecular dynamics. In *Visualization Symposium (PacificVis), 2012 IEEE Pacific* (Feb 2012), pp. 209–216. 4

[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. 1

[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

[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* (Aire-la-Ville, Switzerland, Switzerland, 2005), EUROVIS'05, Eurographics Association, 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), B"uhler K., Linsen L., John N. W., (Eds.), EG Digital Library, The Eurographics Association, pp. 61–70. 4

[LBH12] LINDOW N., BAUM D., HEGE H.-C.: Interactive rendering of materials and biological structures on atomic and nanoscopic scale. *Computer Graphics Forum 31*, 3pt4 (2012), 1325–1334. 4

[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

[LRA*07] LI W., RITTER L., AGRAWALA M., CURLESS B., SALESIN D.: Interactive cutaway illustrations of complex 3d models. In *ACM SIGGRAPH 2007 Papers* (New York, NY, USA, 2007), SIGGRAPH '07, ACM. 3

[PJR*14] PARULEK J., JÖNSSON D., ROPINSKI T., BRUCKNER S., YNNERMAN A., VIOLA I.: Continuous levels-of-detail and visual abstraction for seamless molecular visualization. *Computer Graphics Forum 33*, 6 (2014), 276–287. 4

[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. 4

[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), Neumann L., Sbert M., Gooch B., Purgathofer W., (Eds.), The Eurographics Association. 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. *IEEE Transactions on Visualization and Computer Graphics 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