Eurographics Conference on Visualization (EuroVis) 2012
S. Bruckner, S. Miksch, and H. Pfister
(Guest Editors)

Volume 31 (2012), Number 3

Illustrative Membrane Clipping

Å. Birkeland[1], S. Bruckner[2], A. Brambilla[1] and I. Viola[1,3]

[1]University of Bergen, Norway
[2]Vienna University of Technology, Austria
[3]Christian Michelsen Research, Norway

Abstract

Clipping is a fast, common technique for resolving occlusions. It only requires simple interaction, is easily understandable, and thus has been very popular for volume exploration. However, a drawback of clipping is that the technique indiscriminately cuts through features. Illustrators, for example, consider the structures in the vicinity of the cut when visualizing complex spatial data and make sure that smaller structures near the clipping plane are kept in the image and not cut into fragments. In this paper we present a new technique, which combines the simple clipping interaction with automated selective feature preservation using an elastic membrane. In order to prevent cutting objects near the clipping plane, the deformable membrane uses underlying data properties to adjust itself to salient structures. To achieve this behaviour, we translate data attributes into a potential field which acts on the membrane, thus moving the problem of deformation into the soft-body dynamics domain. This allows us to exploit existing GPU-based physics libraries which achieve interactive frame rates. For manual adjustment, the user can insert additional potential fields, as well as pinning the membrane to interesting areas. We demonstrate that our method can act as a flexible and non-invasive replacement of traditional clipping planes.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Volume Rendering—Manipulation and Deformation

1. Introduction

Clipping is an essential mechanism in visualization which resolves spatial occlusion in three- or more dimensions. This basic idea is used in visual depiction in many different variants, such as clipping into half-spaces, near clipping planes, section views, and cutaway-views which can all be considered variants of the concept.

In 3D data visualization, clipping becomes very powerful when making it adjustable by the user or adapting it to data properties. Clipping provides a simple mechanism for revealing otherwise invisible structures. Similarly, slicing uses the same interaction scheme, but shows only a single slice without any context outside of the intersecting plane.

The idea of removing occluding objects has been used in illustration craft for centuries. However, at closer inspection traditional illustrations rarely employ strict planar clipping but actually use hybrid representations where particular structures extend in front of and behind a hypothetical selected plane. Similarly, elongated structures do not appear or disappear from the rendering, as they would when cut by a plane. Instead, they are aligned with the cut plane.

One example, are dense cell illustrations where structures are clearly discernible and emerge out of the hypothetical clipping plane. It is then possible to follow elongated structures even if they spatially overlap. Figure 1 shows an illustration of a slice through a *Mycoplasma mycoides* cell. As one can see in this illustration, the cell appears to be cut open by a plane revealing the inner structures, yet the structures at the position of the cutting plane are not sliced.

This is practically impossible to achieve with dense 3D visualizations. Objects will likely be intersected by the clipping plane so that only partial structures are depicted. The remaining portion itself may not have a clear meaning to the viewer and may hinder spatial comprehension. The research question arises of how to formalize the illustrative clipping concept in an explorative 3D data visualization scenario with limited semantic information at hand. One way of interpret-

ing illustrative clipping is that the clipping geometry is not planar, but adapts to the structure of the object. The resulting clipping geometry will be a deformed 2D manifold instead of a plane. This manifold preserves the consistency in the structural depiction, provides subtle 3D shape cues and allows the following of elongated structures over the illustration. The resulting clipping membrane reveals otherwise occluded structures but at the same time communicates additional 3D information.

In our approach, the user positions the membrane in the scene and the scene elements act on the clipping geometry to cause the deformation. The membrane is utilized in order to reveal more of its structural properties, and to remove irrelevant information. To avoid cutting through an object when the focus is on its outer shape rather than its inner structures, the object repels the clipping membrane. Other data properties, such as object interfaces, can act attractively, so that the clipping membrane tends to be aligned with these interfaces. For this purpose, our clipping membrane concept is translated to a physics optimization problem. The scene elements generate a potential field where each data element's force is specified by a *potential field transfer function* (PFTF). The resulting potential field is an integral of (signed) forces which act on a deformable cloth object that constitutes the membrane. The membrane is not explicitly visible, but serves as a selection tool for direct volume rendering (DVR).

This idea has a wide range of potential applications. One scenario is the enhancement of traditional 2D slicing with selective depiction of 3D structures, but it also applies to adaptive clipping during mechanic movements of imaged structures in time-dependent data. To achieve interactive performance, the implementation of our clipping membrane concept makes use of NVIDIA's PhysX technology, which executes a pyhsically-based cloth simulation on the GPU.

2. Related Work

3D region extraction or clipping are common tools in medical workstations, for instance in ultrasound scanners with 3D acquisition capability [Kre99]. Manually defining the clipping geometry has been very popular for creating expressive visualizations [WEE02, NWKY05, FWH09]. Konrad-Verse et al. showed how manual sketching can define a deformable cutting plane, aiding in virtual resection [KVLP04]. Deformed planes are also used in slice-based blood-vessel visualization, reforming a plane to the center of the vessel [KFW*02]. However, manually defining complex geometries can be cumbersome. Automatic definition of 3D clipping regions is typically based on semantic information [VKG04].

Selective clipping adjustment has also been used in several visualization techniques. For preoperative planning, Beyer et al. demonstrated the use of clipping for combining the information of multiple imaging modalities [BHWB07]. Similarly, Burns et al. have developed a technique for combining 2D ultrasound and 3D CT in a fused visualization [BHW*07]. In their approach, the clipping geometry was defined by a 2D ultrasound plane. For additional context, certain features defined by segmentation were not clipped. Inspired by enhanced slice rendering [TMS*06] where contextual information of the surrounding areas is merged with the slice, we propose a method for using a deformed geometry to create expressive slice rendering without any required segmentation. Volume splitting [ISC07] can also be seen as an advanced form of clipping, but rather than removing parts of the volume, it translates the clipped parts in order to ensure direct line of sight. An example of this are exploded views [BG06], where the different parts of the volume are repelled from each other, as well as repelled from the line of sight.

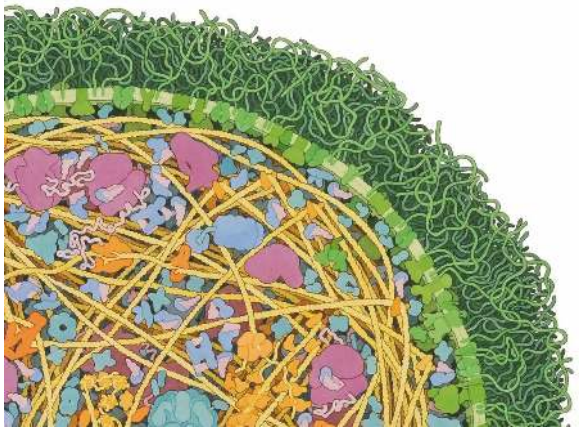


Figure 1: Illustration using clipping to reveal the macromolecular structures inside a *Mycoplasma mycoides* cell, by David S. Goodsell [goo12], the Scripps Research Institute.

Schultz et al. used deformable clipping geometry to simulate a familiar resection method for nerve tracts in the brain, called Klingler resection [SSA*08]. The technique known as Virtual Klingler uses the data intensities to aid in the deformation of the clipping geometry. Schultz's method was the first one to use forces to deform a clipping geometry. We see this method as a specific application of the general concept presented in this paper. While the Virtual Klingler method was highly focused on producing visualizations similar to Klingler resection, the technique had some limitations. The deformation of the plane was defined by creating a resistance field from MRI datasets. To achieve a deformation, the plane needed to be manually moved through the volume. We propose to use active forces, derived from the data, to deform an elastic geometry employed in volume visualization. In addition, we provide a controlling mechanism connecting data attributes to the strength of the forces and propose interaction schemes as well as visual mapping strategies.

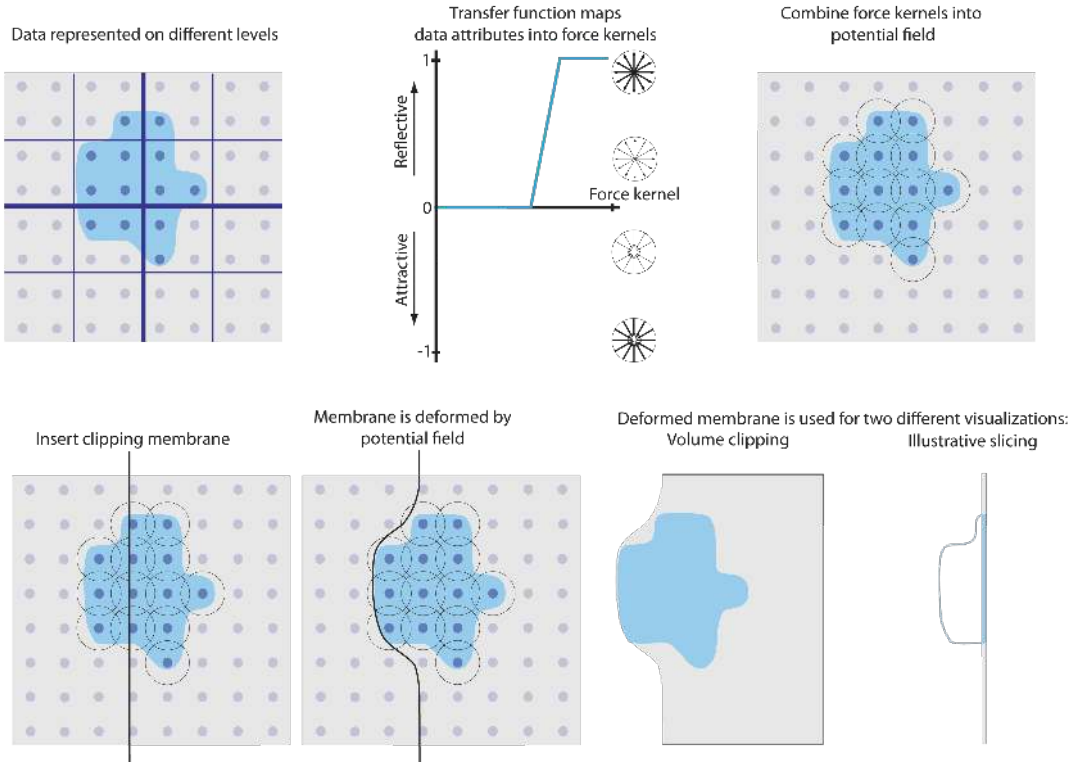


Figure 2: Elementary building blocks of the illustrative clipping pipeline and the two use-case scenarios. Data samples from a specific level are mapped to forces using a *potential field transfer function*. The forces constitutes a potential field, which interact with a flexible membrane. The deformed membrane is then applied in volume visualization.

Physically-based deformation of three-dimensional models is a very active research area, especially in the context of computer graphics. A detailed overview of the most recent approaches in this field was presented by Nealen et al. [NMK*06]. A notable amount of work has been dedicated to the physical simulation of cloths [HB00, MTCK*04]. In our work, we exploit the features of the NVIDIA PhysX library to simulate an elastic piece of cloth which acts as a clipping plane. Details about how the library handles this simulation can be found in Section 3.1 or in the paper by Müller et al. [MHHR07].

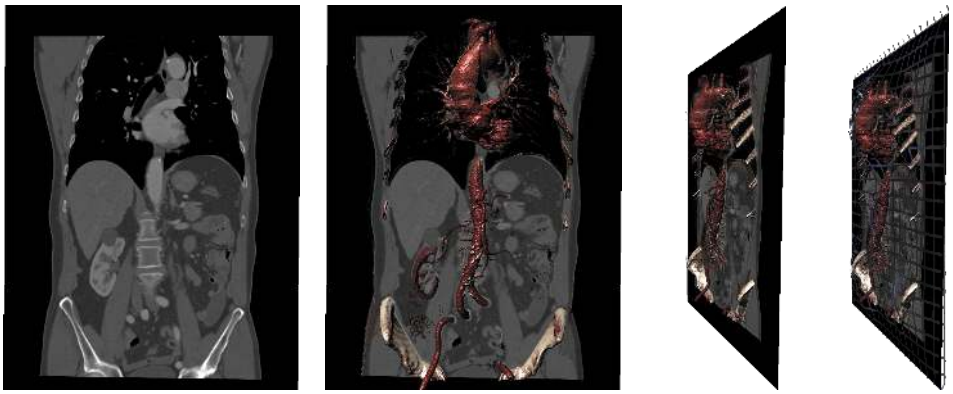
3. Methodology

The core idea of this paper, is to translate the illustrative clipping concept into an interaction between soft-body dynamics actors, as depicted in Figure 2. Firstly, the data attributes are translated into physical elements. This is done by creating an attractive or repelling force for each data element, called a *force kernel*. The combined forces from all the data elements construct a *potential field* depending on the data attributes. In addition, the user needs a controlling mechanism for linking data attributes to the strength of forces exerted by an element. We adapted the familiar transfer function concept, into a *potential field transfer function* (PFTF). Data attributes are then mapped by the PFTF to the strength of the force kernels.

Secondly, we need a geometry which can interact with the forces in the potential field. We therefore create a mesh which acts as an elastic membrane and then insert it into the potential field. By applying a soft-body physics simulation, each vertex in the membrane interacts with the nearby force kernels and conforms to the potential field created from the data attributes. As we aim to use the deformed geometry as a means to create expressive volume visualizations, we propose two different methods to exploit the deformable geometry in already existing rendering schemes. In volume clipping, we exchange the regular flat clipping plane with our deformed geometry. As a second utility example, we suggest the use of the deformed geometry for illustrative slice rendering, enhancing it with the local context defined by the deformed membrane.

3.1. Deformable Membrane

The clipping membrane is represented as a constrained elastic piece of cloth whose shape and position are automati-



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

Figure 3: Rendering of a medical CT dataset. Compared to the regular slice view (3a), adding local DVR provides interesting structures out of the plane (3b). Figure 3c shows a tilted view of the same plane and the grid in Figrue 3d shows the deformed membrane.

cally determined according to a set of physical properties. The physical simulation is based on *Position Based Dynamics* (PBD) [MHHR07], a physics-based approach to simulate soft bodies. Typical physical simulation algorithms are based on Newton's second law of motion, where every particle is associated with an ordinary differential equation (ODE) which includes the forces acting on the particle, the particle's velocity and acceleration, and possibly additional constraints. The system of ODEs is usually solved using an integration algorithm that takes into account all these variables. In the PBD approach, the positions of the particles are directly manipulated: in this way overshooting problems (caused by strong accelerations or high velocities) are avoided, and collisions and penetrations can be easily managed.

In PBD, the dynamical system is defined by a set of vertices and a set of constraints. Similarly to the common mass-spring models, every vertex $i \in \{1,\dots,N\}$ has a mass m_i, a position $\mathbf{x}_i$ and a velocity $\mathbf{v}_i$. A constraint $j \in \{1,\dots,M\}$ instead is described by either an equality $C_j(x_{i_1},\dots,x_{i_{n_j}}) = 0$ or an inequality $C_j(x_{i_1},\dots,x_{i_{n_j}}) \geq 0$, where C_j is a real valued function defined over the positions of a subset of the particles. A stiffness parameter $k_j \in [0,1]$ determines the *strength* of the constraint. Given a time step size Δt, the vertices are initially advected according to the explicit Euler's integration scheme [HNW93]:

$$\begin{cases} \tilde{\mathbf{v}}_i &= \mathbf{v}_i + \Delta t \frac{\mathbf{f}_{ext}(\mathbf{x}_i)}{m_i} \\ \tilde{\mathbf{x}}_i &= \mathbf{x}_i + \Delta t \tilde{\mathbf{v}}_i \end{cases} \tag{1}$$

where $\mathbf{f}_{ext}(\mathbf{x}_i)$ is the force the potential field exerts on particle i. At this point collisions are detected and, for each of them, an additional constraint is generated. Finally, the algorithm iterates through all the constraints and repeatedly modifies the temporary particle positions $\tilde{\mathbf{x}}_i$ trying to satisfy all the equalities and inequalities (taking into account their stiffness k_j). For more detailed information about the general approach, we refer to the article of Müller et al. [MHHR07].

In the case of cloth simulation, two types of constraints are used: the *stretching* and *bending*. The stretching constraint is defined over every edge of the cloth mesh. It aims at preserving the original length of the edge, therefore the associated stiffness parameter can be used to control how rigid the cloth is. The bending constraint instead preserves the angle between each pair of adjacent triangles, so its stiffness parameter determines the cloth's resistance to folding.

To maintain the simple clipping interaction the membrane must retain a certain shape. Without any constraints, the membrane would collapse. In our technique, the membrane is attached to a rigid frame. The rigid rectangular frame ensures that the membrane retains a topologically planar shape.

3.2. Potential field

The potential field is essentially a function $\mathbf{f} : \{1,\dots,N\} \to \mathbb{R}^3$ which associates a force vector to each of the N particles in the dynamical system. In particular, in the NVIDIA PhysX framework, the potential field is given by the sum of forces $\mathbf{f}_k$ exerted by a set of *kernels* $k \in \{1,\dots,K\}$ on the particles:

$$\mathbf{f}(i) = \sum_{k=1}^{K} \mathbf{f}_k(i) \tag{2}$$

The framework lets the user freely specify how the kernel force functions are computed. Specifically, in order to support datasets based on non-cartesian grids, we decided

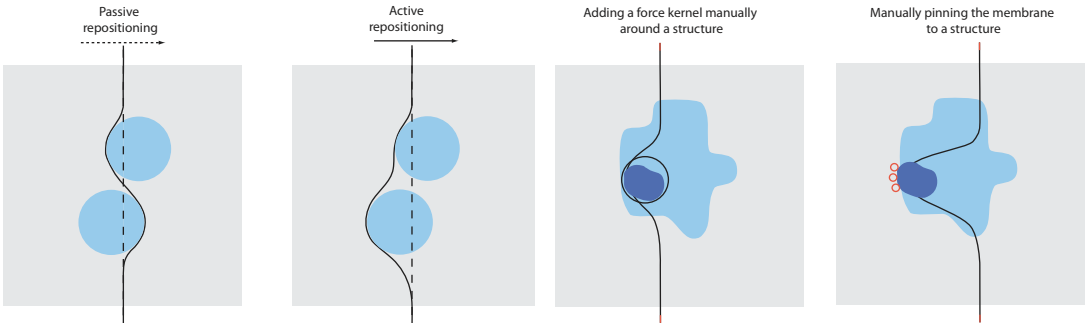


Figure 4: Two different methods for repositioning the membrane: Passive repositioning creates a new membrane at the position of the rigid frame. Active repositioning moves the frame and pulls the membrane through the potential field. We provide a means for interacting locally with the potential field by creating a force kernel with a desired radius to encapsulate interesting structures. In addition, the user can pin the membrane at the current position to prevent the membrane from moving through interesting features.

to treat each of the three components of $\mathbf{f}_k$ separately. The force a kernel k, with position $\mathbf{x}_k$, exerts on a particle i, with position $\mathbf{x}_i$, is given by:

$$f_k^j(i) = c(1 - \frac{|\mathbf{x}_k - \mathbf{x}_i|}{a_j}) \tag{3}$$

with $j \in \{0,1,2\}$. The parameter c is determined using the PFTF, while the parameter a_j defines the area of effect of the kernel and is proportional to the voxel size in the j^{th} direction.

At this point, building the potential field is straightforward: for every voxel the PFTF is evaluated, then a suitable kernel is generated and placed at the voxel's center. However, even with moderate size datasets, the resulting field would be so complex that the physical simulation would not run at interactive rates. To make the system more scalable, we adopt an octree structure to control kernels' placement: the desired octree depth is specified by the user, then, for every super-voxel at that depth, a kernel is generated. The kernel is placed at the center of the super-voxel and its strength is obtained averaging the PFTF values of all the included voxels. Eventually, the force exerted by the potential field on a certain particle is taken into account during the simulation as the f_{ext} term of Equation 1. The equilibrium state of the membrane is then determined through the PBD algorithm.

While the potential field push the membrane away from the original slice position, the only force back towards the slice comes from stretching the nodes. This results in the membrane not following the the cavities of the potential field. As a means to force the membrane back towards its original position, we add a global force towards the slice, resulting in a tighter fit around the structures.

3.3. Interaction

The main interaction with the potential field is done using a widget for creating a transfer function. Adjusting the transfer function to data with very similar data values can be very challenging. To simplify interaction, we allow the user to sketch directly on the original slice or clipping plane. From the sketch, we define a neighbourhood and create a histogram of the included samples which we use to adjust the potential field transfer function.

Positioning the membrane is essentially as simple as moving an ordinary clipping plane or slice through the volume. We propose two different schemes for positioning the membrane, passive insertion and active movement, as illustrated in Figure 4. During passive insertion, the membrane is not affected by the force field as it moves. In essence, a membrane is created inside the potential field during every reposition, and it is then deformed by the force kernels it intersects. In active mode, the membrane is moved across the potential field, and it is deformed in real-time during movement. As the membrane moves across the volume, it *collides* with the potential field. While the frame is moved further across the volume, the membrane moves through the kernels from one equilibrium state to another.

For certain structures, very similar data values makes it difficult defining a proper PFTF. A more direct interaction with the membrane can be useful to keep interesting features in view. We have created two methods for manually interacting with the physical model. One method, is where the user indirectly deforms the membrane by inserting additional repulsive or attractive force kernels at the location of the original slice. Then, using a simple click-and-drag motion, the user can add a kernel with the desired radius into the existing potential field at the depth specified by the slice position.

In addition, the user might desire to keep certain struc-

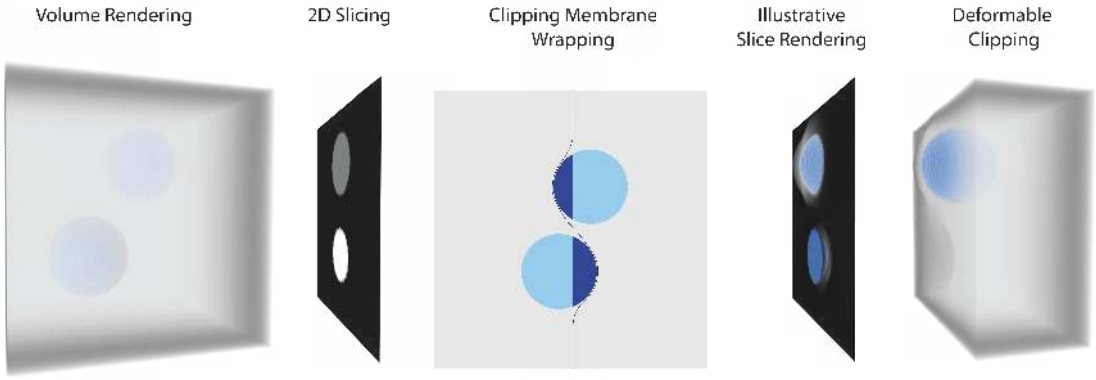


Figure 5: Traditional explorative tools compared to illustrative clipping applied to a synthetic dataset.

tures in view while exploring the rest of the volume. In this case, the membrane can be *pinned* to structures by simply clicking on the membrane. An immovable anchor is then attached to the membrane, keeping the attached elements from moving with the rest of the membrane. This allows the user to manually *fine tune* the clipping. Figure 4 shows examples of the two interaction schemes. To better illustrate the effect, we removed all other forces from scene.

3.4. Visual Mapping

We propose to apply the deformed clipping geometry in two different ways: membrane clipping in DVR and illustrative slicing with local DVR. Figure 5 illustrates the different rendering methods. In both cases, we use a bounding geometry, for calculating the entry and exit points for ray casting.

Regular clipping in ray casting can be achieved using a simple dot product test. With a deformed geometry, it becomes more challenging. Instead of testing which elements are in front or behind the plane, we start the ray casting at the surface of the cut. We then replace one face of the volume's bounding box with the clipping membrane. The new bounding geometry can be used to calculate new entry and exit points from the position of the membrane.

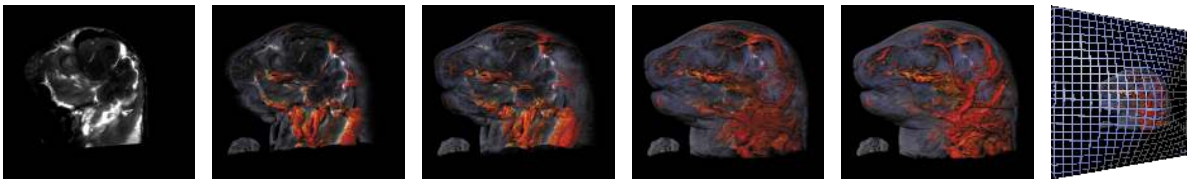
While clipping is a very common technique for volume exploration, domain experts, especially in radiology, tend to use slice rendering as their main visualization tool. Slice rendering provides highly detailed views, with no occlusion. The slice contains little or no context outside of the plane and a challenge rests on the user to comprehend the outer planar spatial extent of structures. Slab rendering can overcome this problem to some extent, by choosing a subset of the volume to be rendered. A slab is a subset of the volume defined by two parallel clipping planes. Similarly, with regular clipping, uninteresting objects may clutter the image and interesting features can be clipped.

To add 3D context to the slice of only those structures the user finds interesting, we propose using the elastic membrane for illustrative slicing, where DVR is only added in between the deformed membrane and the original slice. In this manner, we calculate the entry and exit points for ray casting from the proxy-geometry created by combining the slice and the membrane. The interesting structures are typically in the vicinity of the plane. We prevent clutter from distant structures by modulating the opacity of a certain sample, by the distance to the plane, as depicted in Figure 6. Blending between the slice and the local DVR, it is important to be able do differentiate between structures in front or behind the plane. We have created a technique for user-selected blending and emphasis. Using two sliders, the user selects the opacity and the emphasis between structures behind and in front of the plane. One slider controls the opacity of the slice from completely opaque to fully transparent. Emphasis on structures behind and in front of the slice can be changed by a slider, ranging from -1 to 1. At -1 the DVR behind the slice is emphasized and the opacity of the DVR in front is set to zero. At 0, both the sides of the slice have equal emphasis and at 1 only the DVR in front are displayed.

An example rendering of a synthetic dataset can be seen in Figure 5. On the left, regular volume rendering and slice rendering can be seen. Inserting the deformable plane in between the two spheres, it wraps around on either side. The illustrative slice rendering shows the slice blended with the local DVR. Volumetric clipping, on the other hand, completely removes one sphere while the other remains visible.

With regular clipping, understanding the shape of the cut is fairly easy, as the cut is flat. The shape of a deformed surface might be more difficult to interpret. Shading the surface of the cut can aid in understanding the shape of the membrane and aids in understanding how the structures in view spatially correlate.

Typically, shading in ray casting uses the gradients as surface normals. This does not work at the surface of the cut itself as the gradient might be parallel to the tangent of the



(a) 0.0 (b) 0.125 (c) 0.25 (d) 0.5 (e) 0.075 (f) 0.75 with the grid overlay.

Figure 6: A mouse imaged using ultramicroscopy [DLS*07]. Adjusting the transparency according to the distance to the original slice. The figures show the effect of adjusting the relative maximum distance for contributing to the image.

cut surface. To shade the surface of the cut, we use normals of the membrane in the first hit of the volume rendering, similar to Weiskopf et al. [WEE03]. We render the normals of the deformed membrane into a separate texture which is then used during the ray casting.

4. Implementation

The prototype was implemented on a system with an Intel Core i7 3 GHz CPU and a GeForce GTX 580 graphics card, running Windows 7. Our goal was to create a technique which can handle reasonable sized data sets and run both the physics simulation and rendering at interactive frame rates. The physical simulation was more computationally demanding than the CPU could process. Therefore, we chose to implement our technique using the high performance GPU-based physics library *NVIDIA PhysX Engine* [phy11]. The library provided us with a cloth data structure which we could use as the elastic membrane for our technique, as well as data types for defining force kernels and collision elements in a physics environment constituting our force field.

5. Results

Illustrators have the option of selecting which structures they wish to keep in the illustration. In visualization of 3D data without any semantic information, this selection becomes very difficult. Clipping allows the user to spatially select which elements should be included. The membrane can be used to selectively clip seismic datasets. Here, the horizons below the sea bed are of high interest to the geo-scientific domain. However, seismic data has a high presence of noise and is difficult to explore using standard DVR and clipping. Applying a clipping surface adapting to the data attributes reveals more of the horizon compared to a flat clipping plane, as shown in Figure 7. Here we used only the PFTF to create force kernels from high intensity samples (red). The membrane is then moved smoothly over the volume. Other types of datasets have structures completely surrounded by similar data values. In these situations, creating the membrane inside the data might provide a more suitable option. In Figure 8, the slice wraps around the nerve cells in the hippocampus of a mouse brain. The elongated structures are then preserved in the final image.

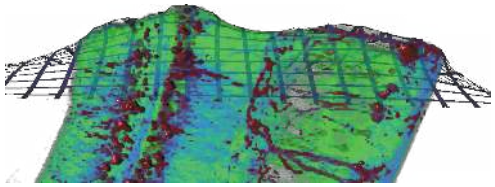


Figure 8: The hippocampus in a mouse brain imaged using ultramicroscopy [DLS*07]. Moving the slice plane through, the membrane is held back by the nerve cells (red), preserving it in the final image.

Basic slice viewing is the most common technique for exploring large 3D data. With a combination of a slice rendering and membrane defined local DVR, we can get a better view of the structure close to the slice, such as the blood vessels and bone in Figure 3. The additional information can provide a better contextual perception. These images where created by drawing a simple sketch on a slice of the main aorta, which then automatically generated the PFTF. 3D ultrasound data is difficult to interpret using DVR due to much noise and similar data values. Local DVR rendering combined with slice rendering, can be a powerful tool for selective display of data and can aid in spatial understanding of dense 3D ultrasound volumes. In Figure 9, the elongated blood vessel structures are preserved as 3D structures in the image. Capturing the blood vessels required some time to define the correct PFTF, due to many data samples with very similar values.

6. Discussion

Our illustrative clipping system is composed of two main elements, i.e., the potential field and the elastic membrane. Their behaviour and their interactions are determined by the physics-based simulator, and the whole process depends on a fair number of parameters which have to be set manually. The optimal configuration is admittedly highly dependent on

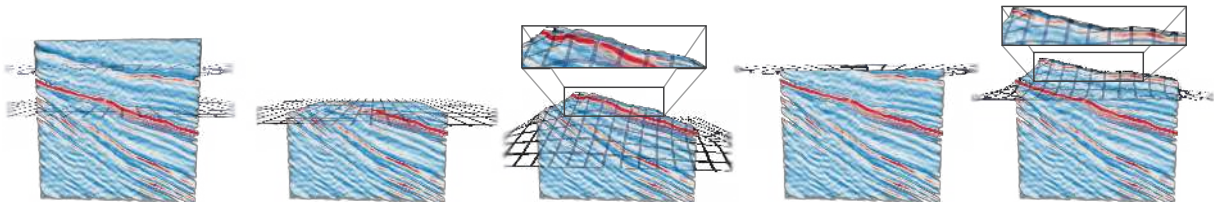
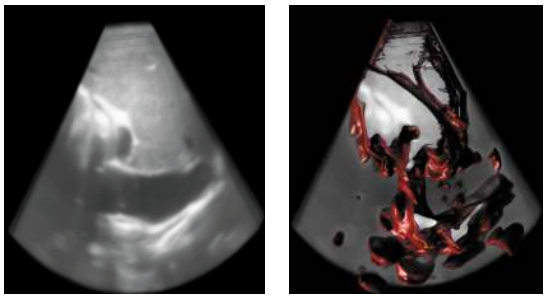


Figure 7: Revealing the structure of a seismic horizon in seismic amplitude data. Flat clipping compared to illustrative clipping.



(a) (b)

Figure 9: Certain structures are completely lost using basic slice viewing of 3D ultrasound (9a). Using the illustrative slicing technique, elongated structures such as the blood vessels are kept in the image (9b).

the purpose of the visualization scenario. We have explored the usage of the various parameters for both the potential field and the membrane, and here we give some guidelines to help the user to setup the system according to his/her needs.

One of the most relevant parameters is the scaling factor c, in Equation 3, which affects the strength of the force fields. Its value is determined through the PFTF, which has to be set according to scalar values in the dataset under consideration. In general, weak force fields make the membrane behave like a traditional clipping plane, while strong forces better emphasize inner structures, but may cause large oscillations of the membrane's position. All the examples in the paper have been generated with values of c up to 1000 and the membrane has never shown significant instability.

The global structure of the potential field is mainly affected by the positions and the areas of influence of every kernel. The placement is controlled by the octree level parameter: low level values produce dense kernel distributions, which, in turn, make the detection of the structures of interest highly accurate. The hierarchical ordering of voxels allows handling of even large datasets at interactive frame rates; however, in this case, generating a membrane which *wraps* around small structures can be difficult, as details may be lost due to averaging. In practice, the octree level can be used to control the tradeoff between speed and accuracy, and it has to be set, taking into account the dataset's size and resolution, and the available computational power. Figure 10 shows the effect of using two different levels.

The area of influence of the kernels depends on the parameter a_j . Since the membrane should not intersect structures of interest, this value has to be set so that adjacent force fields overlap. On the other hand, the larger a_j is, the less tightly the membrane will fit to the structures of interest. After extensive testing, we found out that the best results are achieved when adjacent kernels overlap for 50% of their volume, i.e. when $a_j = 2v_j$, with v_j given by the super-voxel size in the j^{th} direction.

Taking now into consideration the elastic membrane, the main concerns are the bending and stretching stiffness parameters, which determine the membrane's resistance to bending and stretching, respectively. In general, low values of these parameters make the clipping plane fit more tightly to the structures in the volume, while values close to 1 make the clipping plane smoother.

The final shape of the clipping plane is heavily affected by the structure of the mesh used to represent the membrane. It is important to set the mesh resolution high enough in order to capture the smallest structures of interest, but, as for the potential field resolution, too high values could compromise the interactivity of the approach. Moreover, increasing the resolution indirectly makes the membrane stiffer, therefore the kernel's strength c should be increased as well. We set an initial resolution of 50×50, but we let users freely modify this value in order to match the minimum feature size of specific datasets.

Lastly, the PBD algorithm assumes that every particle of the membrane has a certain mass. This value does not need to resemble the actual mass of the simulated piece of cloth, it is simply used to control the inertia of the particles. In the implementation of PBD provided by the NVIDIA PhysX framework, this parameter is optional, so we have not specified it. Its influence on the resulting visualization will be the subject of future investigation. The parameter settings for the examples in the paper are summarized in Table 1. Firstly, in all the cases, the default mesh resolution lead to fairly detailed results with no serious impact over the frame rate, so, unless there are very specific requirements, the average user won't need to modify it. The torso example can be used to

Data (res.)	Hierarchy	# of kernels	FPS w/physics	FPS display only	Figure
Torso ($256 \times 256 \times 556$)	3	2,3k	9	15	Figure 3 and 10b
Mouse ($424 \times 279 \times 190$)	2	15k	3	12	Figure 6
Seismic ($121 \times 121 \times 121$)	2	16k	5	13	Figure 7
Hippocampus ($212 \times 191 \times 44$)	1	16k	3	5	Figure 8
3D Ultrasound ($181 \times 245 \times 190$)	3	5k	10	14	Figure 9
Torso ($256 \times 256 \times 556$)	4	300	12	15	Figure 10a

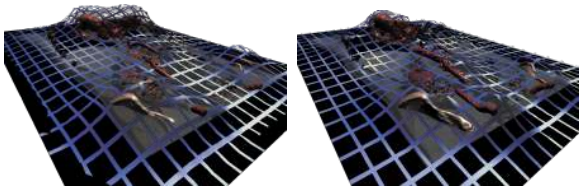
Table 1: Performance results from the technique. The numbers are taken from the images which are included in this paper. FPS w/physics shows the FPS while the physics simulation is running, and FPS display only shows when it is paused. Resolution of the membrane for all images was set to 50×50.

compare the effects of different octree levels: a higher value results in higher frame rates, with some minor differences in the rendered images (see the comparison in Figure 10).

The final results were shown to domain experts in the geological, medical and molecular biology domain. Geoscience experts described a recent work case where several horizons containing high-amplitude anomalies were dipping. The only way to display these surfaces were either to take a time slice (horizontal slice that does not consider the structural dip), or interpret a horizon and then create a surface that is used as a basis for making horizon slices. The second approach would be accurate, but also extremely time-consuming. Our method might be very useful to get a quick overview of high-amplitude anomalies and their extents, with an attractive trade-off between processing time and accuracy.

Medical experts stated that our deformable clipping approach could be interesting to combine with haptics for detecting cirrhosis, where dense *knots* form in the tissue and are otherwise difficult to detect in ordinary scanning. Overall, their comments stated that deformable clipping is promising for quickly gaining an overview of structures hidden in dense data.

Additionally, results from the adaptive seismic dataset clipping were shown to a molecular biologist and artist who is the author of the illustration in Figure 1. Currently, he is using automated methods for stylized rendering of entire molecular structures, but when it comes to revealing internal structures he needs to turn to a hand-craft. After we explained that the same approach can be utilized for adaptive clipping in molecular biology as well, he stated that our method could possibly automate the process of selective rendering, which is only possible nowadays by hand-drawing, and could aid in creating highly expressive images.



(a) (b)

Figure 10: Using a higher traversal depth in 10a for the data hierarchy results in a less close fit compared to 10b.

7. Summary and Conclusion

In this paper, we have presented an illustrative clipping technique for volume exploration using a deformable clipping geometry. By using a deformable geometry, we have demonstrated how we can utilize physically simulated soft bodies to create expressive visualizations of volumetric data. The data attributes are translated into a potential field through a simple transfer function, defining how the data interacts with an elastic membrane. In addition, we have shown how the elastic membrane can be utilized in volume rendering, by aiding in removing visual clutter and adding context in well established volume exploration tools. Using a hierarchical structure and the GPU based physics library, interactive frame rates are achieved for both solving the physics and visualizing the data.

Acknowledgements

This work has been carried out within the IllustraSound research project (# 193170), which is funded by the VERDIKT program of the Norwegian Research Council with support of the MedViz network in Bergen, Norway (PK1760-5897-Project 11). In addition, this work has been partially funded by the Austrian Science Fund (FWF) through the ViMaL project (grant no. P21695) and by the Petromaks program of the Norwegian Research Council through the Geoillustrator project (#200512). We would like to thank our medical partners, Odd Helge Gilja and Trygve Hausken, as well as geologist Sten-Andreas Grundvåg, for valuable input. Finally, we would like to thank Laura Nic Lochlainn for aiding in proof reading and improving the final manuscript, Statoil for providing the seismic dataset, and David S. Goodsell for the motivating molecular machines illustration.

References

[BG06] BRUCKNER S., GRÖLLER M. E.: Exploded Views for Volume Data. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (9 2006), 1077–1084. 2

[BHW*07] BURNS M., HAIDACHER M., WEIN W., VIOLA I., GRÖLLER E.: Feature Emphasis and Contextual Cutaways for Multimodal Medical Visualization. In *Proceedings of EUROGRAPHICS* (2007). 2

[BHWB07] BEYER J., HADWIGER M., WOLFSBERGER S., BUHLER K.: High-Quality Multimodal Volume Rendering for Preoperative Planning of Neurosurgical Interventions. *IEEE Transactions on Visualization and Computer Graphics* (2007). 2

[DLS*07] DODT H.-U., LEISCHNER U., SCHIERLOH A., MAUCH N. J. C. P., DEININGER K., DEUSSING J. M., EDER M., ZIEGLGÄNSBERGER W., AND K. B.: Ultramicroscopy: three-dimensional visualization of neuronal networks in the whole mouse brain. *Nature Methods 4*, 4 (March 2007), 331–336. 7

[FWH09] FUCHS R., WELKER V., HORNEGGER J.: Non-convex polyhedral volume of interest selection. *Journal of Computerized Medical Imaging and Graphics 34*, 2 (2009), 105–113. 2

[goo12] Illustration by David S. Goodsell, the Scripps Research Institute. http://mgl.scripps.edu/people/goodsell, 2012. 2

[HB00] HOUSE D. H., BREEN D. E. (Eds.): *Cloth modeling and animation*. 2000. 3

[HNW93] HAIRER E., NØRSETT S. P., WANNER G.: *Solving Ordinary Differential Equations I*, 2nd ed. Springer Series in Computational Mathematics. Springer, Berlin Heidelberg, 1993. 4

[ISC07] ISLAM S., SILVER D., CHEN M.: Volume Splitting and Its Applications. *IEEE Transactions on Visualization and Computer Graphics 13*, 2 (2007), 193–203. 2

[KFW*02] KANITSAR A., FLEISCHMANN D., WEGENKITTL R., FELKEL P., GRÖLLER M. E.: *CPR - Curved Planar Reformation*. Tech. rep., Institute of Computer Graphics and Algorithms, Vienna University of Technology, Favoritenstrasse 9-11/186, A-1040 Vienna, Austria, 2002. human contact: technical-report@cg.tuwien.ac.at. 2

[Kre99] KRETZTECHNIK AG: 3D Ultrasound: A Dedicated System. *European Radiology 9* (1999), S331–S333. 2

[KVLP04] KONRAD-VERSE O., LITTMANN A., PREIM B.: Virtual Resection with a Deformable Cutting Plane. In *SimVis* (2004), pp. 203–214. 2

[MHHR07] MÜLLER M., HEIDELBERGER B., HENNIX M., RATCLIFF J.: Position Based Dynamics. *Journal of Visual Communication and Image Representation 18*, 2 (2007), 109 – 118. 3, 4

[MTCK*04] MAGNENAT-THALMANN N., CORDIER F., KECKEISEN M., KIMMERLE S., KLEIN R., MESETH J.: Simulation of Clothes for Real-time Applications. In *Proceedings of EUROGRAPHICS* (2004). 3

[NMK*06] NEALEN A., MÜLLER M., KEISER R., BOXERMAN E., CARLSON M.: Physically Based Deformable Models in Computer Graphics. *Computer Graphics Forum 25*, 4 (2006), 809–836. 3

[NWKY05] NAKAO M., WATANABE T., KURODA T., YOSHIHARA H.: Interactive 3D Region Extraction of Volume Data Using Deformable Boundary Object. In *Medicine Meets Virtual Reality 13: The Magical Next Becomes the Medical Now*. IOS Press, 2005, pp. 349–359. 2

[phy11] Nvidia PhysX Engine. http://uk.geforce.com/hardware/technology/physx, 2011. 7

[SSA*08] SCHULTZ T., SAUBER N., ANWANDER A., THEISEL H., SEIDEL H.-P.: Virtual Klingler Dissection: Putting Fibers into Context. *Computer Graphics Forum 27*, 3 (May 2008), 1063–1070. 2

[TMS*06] TIETJEN C., MEYER B., SCHLECHTWEG S., PREIM B., HERTEL I., STRAUSS G.: Enhancing Slice-based Visualizations of Medical Volume Data. In *Proceedings of the Eurographics / IEEE VGTC Symposium on Visualization* (2006), pp. 123–130. 2

[VKG04] VIOLA I., KANITSAR A., GROLLER M. E.: Importance-Driven Volume Rendering. In *Proceedings of IEEE Visualization* (2004), pp. 139–146. 2

[WEE02] WEISKOPF D., ENGEL K., ERTL T.: Volume clipping via per-fragment operations in texture-based volume visualization. In *Proceedings of IEEE Visualization* (2002), pp. 93–100. 2

[WEE03] WEISKOPF D., ENGEL K., ERTL T.: Interactive Clipping techniques for Texture-Based Volume Visualization and Volume Shading. *IEEE Transactions on Visualization and Computer Graphics 9*, 3 (2003), 298 – 312. 7